

Федеральное государственное учреждение «Федеральный научный центр
Научно-исследовательский институт системных исследований Российской академии наук»
(ФГУ ФНЦ НИИСИ РАН)

ТРУДЫ НИИСИ РАН

ТОМ 6 N 2

**МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ
СЛОЖНЫХ СИСТЕМ:**

ТЕОРЕТИЧЕСКИЕ И ПРИКЛАДНЫЕ АСПЕКТЫ

МОСКВА
2016

Редакционный совет ФГУ ФНЦ НИИСИ РАН:

В.Б. Бетелин (председатель),
Е.П. Велихов, В.А. Галатенко, В.Б. Демидович (отв. секретарь), Б.В. Крыжановский,
А.Г. Кушнirenко, А.Г. Мадера, М.В. Михайлюк, В.Я. Панченко, В.П. Платонов,
В.Н. Решетников

Главный редактор журнала:

В.Б. Бетелин

Научный редактор номера:

М.В. Михайлюк

Тематика номера:

Математическое моделирование и визуализация, моделирование в микроэлектронике,
вопросы программирования, математические исследования и вопросы численного анализа,
к истории науки и техники

Журнал публикует оригинальные статьи по следующим областям исследований:
математическое и компьютерное моделирование, обработка изображений, визуализация,
системный анализ, методы обработки сигналов, информационная безопасность,
информационные технологии, высокопроизводительные вычисления, оптико-нейронные
технологии, микро- и нанoeлектроника, вопросы численного анализа, история науки и
техники

The topic of the issue:

Mathematical modeling and visualization, modeling in microelectronics, problems in programming,
mathematical investigations and problems of numerical analysis, history of science and of technique

The Journal publishes novel articles on the following research arias: mathematical and computer
modeling, image processing, visualization, system analysis, signal processing, information security,
information technologies, high-performance computing, optical-neural technologies,
micro- and nanoelectronics, problems of numerical analysis, history of science and of technique

Заведующий редакцией: Ю.Н. Штейников

Издатель: ФГУ ФНЦ НИИСИ РАН,
117218, Москва, Нахимовский проспект 36, к. 1

СОДЕРЖАНИЕ

I. МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ВИЗУАЛИЗАЦИЯ

<i>Б.В. Крыжановский, Л.Б. Литинский.</i> Приближенное вычисление статистической суммы для модели Изинга на гиперкубе	5
<i>Б.В. Крыжановский, М.Ю. Мальсагов.</i> Улучшение скалярного нейросетевого дерева, используемого для поиска ближайшего соседа в бинарном пространстве большой размерности	11
<i>Р.О. Пономарев, Д.А. Кононов, Н.О. Пономарев.</i> Сценарное исследование сложных систем: уязвимость и чрезвычайные ситуации	21
<i>Р.О. Пономарев, Д.А. Кононов, И.М. Хасанов.</i> Сценарный анализ уязвимости сложной системы: вычислительный эксперимент	26
<i>Н.Н.Смирнов, В.Ф.Никитин, М.В. Михайлюк, П.Ю.Тимохин, В.В.Тюренкова, Л.И.Стамов.</i> Визуализация результатов моделирования неустойчивого вытеснения нефти из пористых сред	34
<i>М.В.Михайлюк, П.Ю.Тимохин.</i> Моделирование поверхности виртуального полинома	38
<i>А.М.Трушин, М.В.Михайлюк.</i> Определение коллизий аппроксимирующих капсул и сфер трехмерных виртуальных динамических объектов	42
<i>И.В.Афанаскин, Н.П.Ефимова, О.В.Ломакина, Ю.М. Штейнберг, Д.Т. Миронов.</i> Мониторинг заводнения нефтяных пластов с помощью суперэлементного моделирования трассировки закачиваемой воды	46
<i>М.Г. Фуругян.</i> Составление расписаний в многопроцессорной системе с учетом затрат на прерывания	57
<i>А.С. Осипов.</i> Об использовании элементов нечеткой логики в оценке алгоритмов идентификации лиц.....	62

II. МОДЕЛИРОВАНИЕ В МИКРОЭЛЕКТРОНИКЕ

<i>П.И. Кандалов.</i> Программный комплекс для теплового проектирования интервально стохастических температурных полей в электронных модулях STF-ElectronMod	70
<i>М.Ж. Акжолов.</i> Численное моделирование свободного конвективного теплообмена в корпусах электронных систем	74
<i>Н.В.Масальский.</i> Характеристики двух затворных КМОП нанотранзисторов с градиентно-легированной рабочей областью	77
<i>М.Ш.Сургуладзе.</i> Моделирование сбоев информационной безопасности в сложных технических системах	83
<i>В.Ю. Троицкий, М.В. Орешков, А.А. Захарченко, А.П. Трепалин, Т.В. Петлицкая, В.А. Филипеня.</i> Анализ причин неработоспособности кристаллов Ethernet контроллера 1990BG3T	88

III. ВОПРОСЫ ПРОГРАММИРОВАНИЯ

<i>Ю.А. Семенов, А.П. Овсянников, Т.В. Овсянникова.</i> Разработка банка алгоритмов и основ языка описания проблемы с целью минимизации числа программных ошибок	96
<i>А.А. Бурцев.</i> Внедрение в библиотеку FFTW собственных модулей, применяющих специализированный сопроцессор для ускорения операций преобразования Фурье	101

<i>Г.Л. Левченкова.</i> Способ интеграции табличных данных с целью оптимизации поиска	110
<i>А.Н.Годунов, В.А.Солдатов.</i> Повышение надёжности и масштабирование многопроцессорных систем в ОС РВ Багет	119

IV. МАТЕМАТИЧЕСКИЕ ИССЛЕДОВАНИЯ И ВОПРОСЫ ЧИСЛЕННОГО АНАЛИЗА

<i>И.Г. Царьков.</i> Особые множества и их связь с особенностями решений уравнения Эйконала	126
<i>В.Б. Демидович.</i> Об интерполяции функций обобщённо-полиномиальными сплайнами, построенными на базе гладких чебышёвских систем	129
<i>О.Ю. Сударева.</i> Быстрое вычисление значений многочлена на «мультимножестве» точек	143
<i>М.Л.Бахмутский.</i> Ускорение степенного метода нахождения максимальных собственных значений и их векторов для сверхбольших заполненных матриц	154

V. К ИСТОРИИ НАУКИ И ТЕХНИКИ

<i>В.Б.Демидович.</i> К 70-летию Владимира Борисовича Бетелина	160
--	-----

Приближенное вычисление статистической суммы для модели Изинга на гиперкубе

Б.В. Крыжановский¹, Л.Б. Литинский²

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»,
Центр оптико-нейронных технологий, Москва, Россия,

E-mail's: ¹ kryzhanov@mail.ru, ² litin@mail.ru

Аннотация: Ранее был разработан метод n -окрестностей для приближенного вычисления статистической суммы спиновой системы с произвольной матрицей связи. В настоящей работе эти результаты адаптируются для модели Изинга на гиперкубе произвольной размерности. Решается уравнение состояния, анализируется поведение свободной энергии. Для больших размерностей наша теория хорошо согласуется с результатами компьютерного моделирования. Для малых размерностей имеется заметное расхождение с точными результатами.

Ключевые слова: статистическая сумма, метод n -окрестностей, модель Изинга на гиперкубе.

1. Введение

В [1]-[3] был развит метод n -окрестностей для приближенного вычисления статистической суммы, который можно описать следующим образом. Пусть $N \gg 1$ - размерность задачи, $\mathbf{s}_0 \in \mathbf{R}^N$ - начальная конфигурация, а ее n -окрестность Ω_n есть множество всех конфигураций, отличающихся от $\mathbf{s}_0$ значениями n координат ($n = 0, 1, \dots, N$). Распределение энергий состояний из Ω_n хорошо аппроксимируется гауссовым распределением [3]. Средняя энергия E_n и дисперсия σ_n^2 могут быть точно выражены через параметры матрицы связи [1]-[2]. Тогда в выражении для статистической суммы суммирование по состояниям из Ω_n в асимптотическом пределе заменяется интегрированием $\exp(-\beta E)$ по гауссовой мере, а суммирование по всем n -окрестностям - интегрированием по параметру $x = n / N$ от 0 до 1.

В результате, статистическая сумма принимает вид интеграла $Z_N \approx \int dx \int dE \exp[-N \cdot F(x, E)]$, который вычисляется методом перевала. Функция двух переменных $F(x, E)$ зависит от таких параметров как обратная температура β , внешнее магнитное поле H (если имеется), след квадрата матрицы связи и еще несколько числовых характеристик: $F(x, E) \equiv F(x, E, \beta, H, \dots)$. В самом общем виде выражение для $F(x, E)$ получено в [2], [3]. Здесь мы адаптируем это выражение для матрицы связи, отвечающей модели Изинга на D -мерном гиперкубе, получаем уравнение состояния, строим и изучаем графики свободной энергии. Изложение по необходимости является очень сжатым. Более подробно материалы данной работы будут изложены в [4].

2. Основные выражения

Пусть $\mathbf{T} = (T_{ij})_1^N$ матрица связи, отвечающая D -мерной модели Изинга, $\mathbf{s}_0 = (1, 1, \dots, 1) \in \mathbf{R}^N$ - основное состояние спиновой системы, а Ω_x - x -окрестность основного состояния, где $x = n / N \in [0, 1]$. Энергию будем вычислять в расчете на один спин: если $\mathbf{s} \in \Omega_x$, и H - величина однородного магнитного поля, то:

$$E(\mathbf{s}, H) = -\frac{1}{2N} \sum_{ij} T_{ij} s_i s_j - \frac{H}{N} \sum_i s_i = E(\mathbf{s}) - H(1 - 2x).$$

Обозначим через E_0 энергию основного состояния: $E_0 = E(\mathbf{s}_0) = -\frac{1}{2N} \sum_{ij} T_{ij} = -D$. Тогда асимптотическое выражение для среднего значения энергии по Ω_x есть $E_x = E_0(1 - 2x)^2$, а

дисперсия равна $\sigma_x^2 = \frac{8 \sum_{ij} T_{ij}^2}{N^2} x^2 (1 - x)^2$. В [1]-[3]

показано, что асимптотическое выражение для статистической суммы приводится к виду

$$Z_N \sim \int_0^1 dx \int_{E_0}^{|E_0|} \exp[-N \cdot F(x, E)] dE, \text{ где } N \gg 1,$$

$$F(x, E) = L(x) + \beta [E - H \cdot (1 - 2x)] + \frac{1}{2} \left(\frac{E - E_x}{\sigma_x} \right)^2,$$

$$\text{а } L(x) = x \ln x + (1 - x) \ln(1 - x).$$

Для отыскания глобального минимума функции $F(x, E)$ необходимо решить систему уравнений:

$$\frac{\partial F}{\partial E} = \beta + \frac{E - E_x}{\sigma_x^2} = 0,$$

$$\frac{\partial F}{\partial x} = \ln \frac{x}{1-x} + \frac{E - E_x}{\sigma_x} \frac{\partial}{\partial x} \left(\frac{E - E_x}{\sigma_x} \right) + 2\beta H = 0.$$

Разрешая первое уравнение относительно β и подставляя результат во второе, получаем задачу минимизации функции одной переменной $f(x) = L(x) + \beta E_x - \frac{\beta^2 \sigma_x^2}{2} - \beta H(1-2x)$, при том условии, что для данного β выполняется неравенство: $\beta + \frac{E_0 - E_x}{\sigma_x^2} \leq 0$. Используем другую нормировку и более удобные обозначения:

$$b = \beta \frac{\sum_{ij} T_{ij}^2}{\sum_{ij} T_{ij}}, \quad \gamma = \frac{\left(\sum_{ij} T_{ij}\right)^2}{N \sum_{ij} T_{ij}^2}, \quad h = \frac{2NH}{\sum_{ij} T_{ij}}.$$

В окончательном виде задача принимает вид: для каждого значения «нормированной» обратной температуры b найти на интервале $(0, x_b)$ глобальный минимум функции

$$f(x) = L(x) - \frac{\gamma b}{2} \left[(1-2x)^2 + 8b(x(1-x))^2 + h(1-2x) \right], \quad (1)$$

где

$$x_b = \begin{cases} \frac{1}{2}, & \text{для } b \leq 1 \\ \frac{1 - \sqrt{1-1/b}}{2}, & \text{для } b \geq 1 \end{cases}. \quad (2)$$

Правая граница интервала, на котором ищется глобальный минимум $f(x)$, становится переменной, когда b превосходит 1.

Свободная энергия запишется в виде:

$$f(b) = \min_{x \in (0, x_b)} f(x).$$

Замечание. Для D -мерной модели Изинга с величиной связи J между ближайшими соседями имеем: $\sum_{ij} T_{ij} = DNJ$, $\sum_{ij} T_{ij}^2 = DNJ^2/2$, $b = \beta J$, $\gamma = 2D$, $h = H/(DJ)$. Характеристика γ равна числу ближайших соседей у каждого спина. Однако взаимодействие вдоль различных направлений D -мерной решетки может быть различным. В этом случае γ уже не будет целочисленным. Например, для двумерной модели Изинга с различными константами взаимодействия по вертикали (J) и по горизонтали (K) имеем: $\gamma = 2 \left(1 + \frac{2}{K/J + J/K} \right)$. Эта величина может принимать любое значение из интервала $2 < \gamma < 4$. Аналогично, для трехмерной модели Изинга с различными константами взаимодействия (J, K, L), параметр γ может принимать любое значение из интервала $2 < \gamma < 6$. В общем случае γ есть эффективное координационное число, которое

описывает взаимодействие спина с ближайшим окружением.

3. Анализ уравнения состояния при $H=0$

Исключим из рассмотрения внешнее поле, отложив общий случай до более подробной публикации [4]. Будем искать глобальный минимум функции $f(x)$ (1) при $h=0$. Необходимо решить уравнение состояния:

$$\frac{\partial f}{\partial x} = \ln \frac{x}{1-x} + 2\gamma b(1-2x)[1-4bx(1-x)] = 0. \quad (3)$$

Ясно, что $x_0 = 0.5$ является решением этого уравнения при любых значениях b и γ . Назовем решение x_0 тривиальным. Для очень малых значений b ($b \sim 0$), функция $f(x)$ при любом значении γ монотонно убывает от $f(0) = -\gamma b/2$ до $f(x_0) = -\ln 2 - \gamma b^2/4$ (рис. 1a). Следовательно, единственным решением уравнения (3) при малых b будет тривиальное решение x_0 , а свободная энергия имеет вид:

$$f(b) = \min_x f(x) = -\ln 2 - \frac{\gamma b^2}{4}, \quad b \sim 0. \quad (4)$$

Интервал значений $\gamma > 16/3$. Можно показать [4], что для таких γ точка x_0 остается единственным решением уравнения (3) до тех пор, пока b не сравняется с критическим значением

$$b_c = \frac{1 - \sqrt{1-4/\gamma}}{2}. \quad (5)$$

Пока $b < b_c$ свободная энергия имеет вид (4). Как только b превзойдет значение b_c , x_0 станет точкой максимума функции $f(x)$, а слева от x_0 возникнет точка минимума $x_1(b)$: $x_1(b) < x_0$ (рис. 1b). Свободная энергия вычисляется подстановкой $x_1(b)$ в выражение (1). При $b = b_c$ в системе происходит фазовый переход второго рода.

Дальнейшее увеличение параметра b сопровождается углублением минимума и неуклонным смещением точки минимума $x_1(b)$ в направлении 0. Когда b станет больше 1, правая граница интервала изменения x станет переменной - см. выражение (2) для x_b . При $b \rightarrow \infty$ интервал $(0, x_b)$ стягивается к началу координат, а глобальный минимум $x_1(b)$ неограниченно стремится к 0. Как и положено, когда температура стремится к 0, спиновая система стремится в основное состояние $\mathbf{s}_0 = (1, 1, \dots, 1)$.

Подставим в выражение (5) значения $\gamma = 6, 8, 10, 12, 14$, и вычислим критические значения

обратной температуры для 3D-, 4D-, 5D-, 6D- и 7D- модели Изинга соответственно. Полученные числа помещены в верхнюю строку табл.1. Во второй строке таблицы приведены значения b_c , полученные компьютерным моделированием – см. [5], [6] и [7]. Налицо хорошее согласие между нашими теоретическими оценками и результатами компьютерного эксперимента.

Таким образом, формула (5) дает разумную аналитическую оценку критической температуры для модели Изинга на гиперкубе размерности $D > 8/3$. Напомним, что параметр γ не обязан принимать только четные целочисленные значения – см. замечание в конце предыдущего пункта.

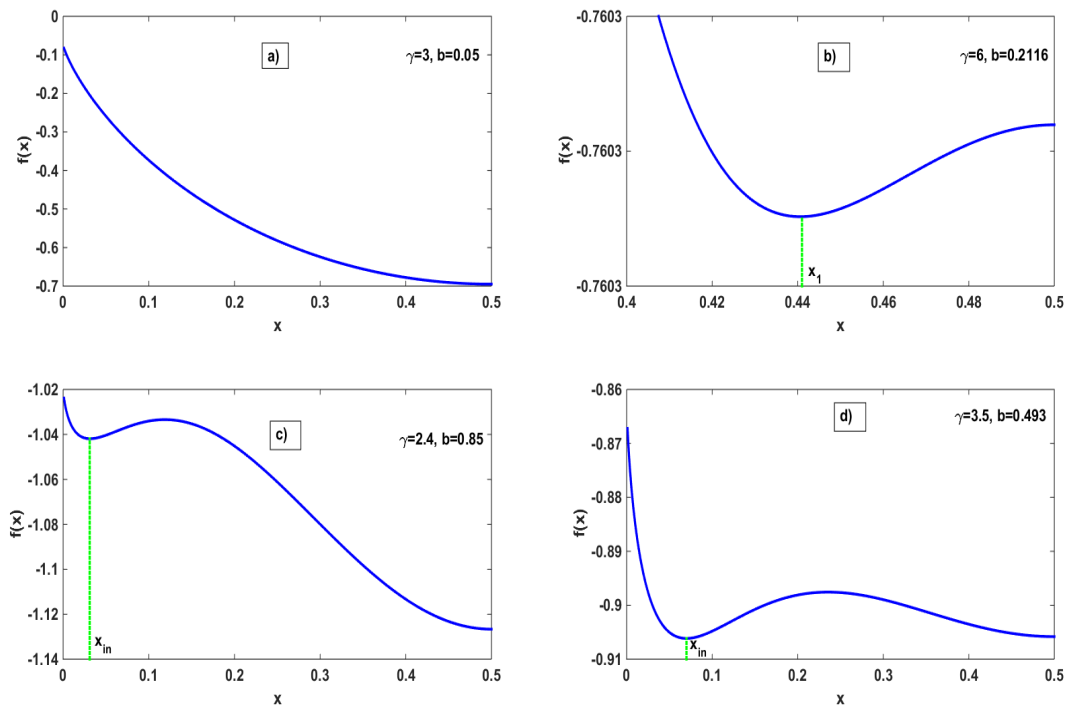


Рис.1. Общий вид функции $f(x)$ (1) в зависимости от значений γ и b (см. текст).

Табл.1. Критические значения обратной температуры b_c для D-мерной модели Изинга

Размерность решетки	3D	4D	5D	6D	7D
Наша теория: формула (5)	0.2113	0.1464	0.1127	0.0918	0.0774
Компьютерное моделирование	0.2216	0.1489	0.1139	0.0923	0.0777

Интервал значений $\gamma < 16/3$. Здесь функция $f(x)$ с ростом b трансформируется совершенно иначе. Главное отличие состоит в том, теперь у функции $f(x)$ непременно возникнет локальный минимум во внутренней точке $x_{in}(b)$ интервала $(0, x_0)$, и глубину этого

минимума необходимо сравнивать $f(x_0)$. Ситуация развивается по-разному в зависимости от того, меньше γ чем 2.75, или больше этого числа.

Интервал $\gamma < 2.75$ (рис.1с). Оказывается, в этом случае локальный минимум в точке x_{in}

никогда не станет глубже минимума в точке x_0 [4]. Следовательно, пока $b \leq 1$, глобальный минимум $f(x)$ находится в точке x_0 . А когда b превзойдет значение 1, и правая граница станет переменной x_b , глобальный минимум функции $f(x)$ автоматически переместится в граничную точку x_b :

$$f(b) = \begin{cases} -\ln 2 - \frac{\gamma b^2}{4}, & b \leq 1 \\ L(x_b) - \frac{\gamma}{4}(2b-1), & b \geq 1 \end{cases} \quad (6)$$

По мере дальнейшего возрастания b ($b \rightarrow \infty$) глобальный минимум либо будет находиться в точке x_b , либо в какой-то момент перескочит в текущее значение минимума $x_{in}(b)$; последнее произойдет, если при некотором значении b минимум $f(x_b)$ сравняется по величине с $f(x_{in})$. В любом случае, с неограниченным ростом b точка минимума стремится к 0, а система стремится в основное состояние.

При $b=1$ первая производная свободной энергии (6) имеет разрыв: $f'_b|_{b=1-0} = -\gamma/2$, $f'_b|_{b=1+0} = (1-\gamma)/2$, что свидетельствует о фазовом переходе первого рода. Это справедливо для любого значения $\gamma < 2.75$, в частности, и для $\gamma = 2$, которое отвечает одномерной модели Изинга. В то же время, точное решение одномерной модели Изинга [8] не дает фазовых переходов при конечных температурах. Приходится констатировать, что в области $\gamma < 2.75$ наш метод дает сомнительные результаты.

Интервал $2.75 < \gamma < 16/3$ (рис.1d). В этом случае «внутренний» локальный минимум $x_{in}(b)$ при некотором значении b_j , меньшем 1, непременно сравняется с минимумом в точке x_0 [4]. Соответствующее значение x обозначим x_j : $x_j = x_{in}(b_j)$. При $b = b_j$ глобальный минимум перескакивает из x_0 в x_j . Намагниченность состояния меняется при этом скачком: от $m_0 = 0$ до $m_j = 1 - 2x_j$, что свидетельствует о фазовом переходе первого рода. По мере дальнейшего увеличения b ($b \rightarrow \infty$) точка минимума $x_{in}(b)$ неуклонно стремится к 0, а система стремится в основное состояние.

Значение $\gamma = 4$, отвечающее 2D модели Изинга, принадлежит рассматриваемому интервалу $\gamma \in [2.75, 16/3]$. Наш подход предсказывает для 2D модели Изинга фазовый переход первого рода при

$b_c \approx 0.3912$. В то же время, восходящее к Л. Онсагеру точное решение 2D модели Изинга дает фазовый переход второго рода при заметно большем критическом значении $b_c \approx 0.4407$ [8]. Для этого интервала значений γ наш метод также дает сомнительные результаты.

4.Обсуждение и выводы

Проиллюстрируем сказанное несколькими расчетами. На рис.2. показана зависимость намагниченности $m = 1 - 2x$ от b для различных значений γ . Поначалу, пока значение b невелико, глобальный минимум находится в точке $x_0 = 0.5$, и намагниченность состояния равна 0. Затем при достижении параметром b критического значения, глобальный минимум уходит из точки x_0 , а намагниченность становится отличной от 0. Для $\gamma = 6$ это происходит при $b = b_c$ (5), причем намагниченность меняется без скачков. Для остальных значений γ это происходит при $b = b_j$, когда глобальный минимум перескакивает из x_0 в точку локального минимума x_j (см. выше). Для значений $\gamma < 16/3$ на графиках отчетливо виден скачок намагниченности $m = 1 - 2x_j$.

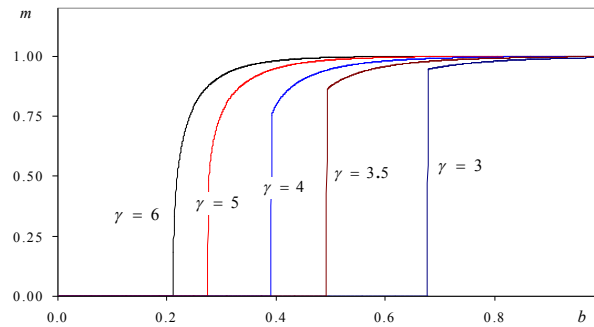


Рис. 2. Намагниченность m как функция b при различных значениях γ .

Аналогично ведет себя среднее значение энергии $\langle E \rangle = \sum_s E(s) \exp(-\beta \cdot E(s))$ - эту характеристику можно вычислить подобно тому, как вычислялась свободная энергия [4]. На рис.3 для различных значений γ показаны графики зависимости от b нормированной средней энергии $E = \langle E \rangle / E_0$. Как и положено, поначалу E зависит от b линейно, а для b , превосходящих критическое значение, поведение E меняется. Для $\gamma = 6$ функция $E(b)$, оставаясь непрерывной, плавно спадает до -1. Для $\gamma \in [2.75, 16/3]$ величина E меняется в критической точке скачком, после чего система стремится в основное состояние.

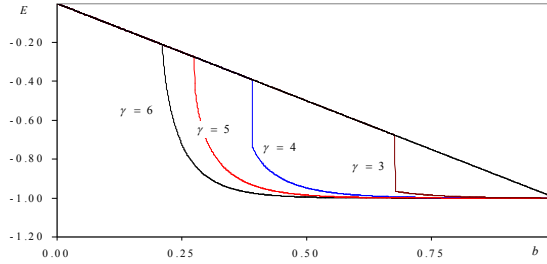


Рис. 3. Нормированная средняя энергия E как функция b при различных значениях γ .

Подобный скачок средней энергии свидетельствует о наличии щели в распределении энергий. Однако невозможно привести аргументы за то, почему при $\gamma = 6$ энергетическая щель отсутствует, а при $\gamma < 16/3$ - присутствует.

Наибольшим изъяном нашего подхода является скачок намагниченности для 2D модели Изинга. Эта ошибка является следствием тех искажений, которые обусловлены гауссовой аппроксимацией распределения энергий в n -окрестности. Не вдаваясь в детали, заметим, что из анализа уравнения состояния мы видим, что небольшая деформация хода кривых способна устранить этот дефект. Такая деформация кривых может явиться следствием уточнения нашей модели. Например, если аппроксимировать распределения энергий в n -окрестности не гауссовой плотностью, а более адекватным распределением. Это - предмет наших дальнейших исследований. Если не обращать внимания на скачок намагниченности, а интересоваться поведением свободной энергии, наш подход даже сейчас дает приемлемые результаты.

На рис. 4 для 2D модели Изинга показана зависимость критического значения обратной температуры β_c от отношения величин взаимодействия спинов по вертикали и горизонтали: $J = 1$, а K меняется от 0.2 до 1. Мы видим, что в целом наши результаты неплохо коррелируют с точным решением.

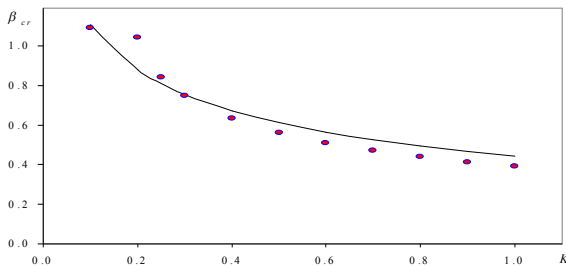


Рис. 4. Для 2D модели Изинга зависимость критической температуры от величины взаимодействия между спинами K : сплошная кривая - теория Онсагера, кружки - наша теория.

На рис. 5 приведены графики свободной энергии, построенные для различных значений K . Сплошной линией показана $f_{ex}(\beta)$, вычисленная по теории Онсагера, а кружочками - то, что получается в нашем подходе: $f_{app}(\beta)$. Видно, что отличие наших результатов от точной теории становится заметным только в районе критического значения обратной температуры β_c .

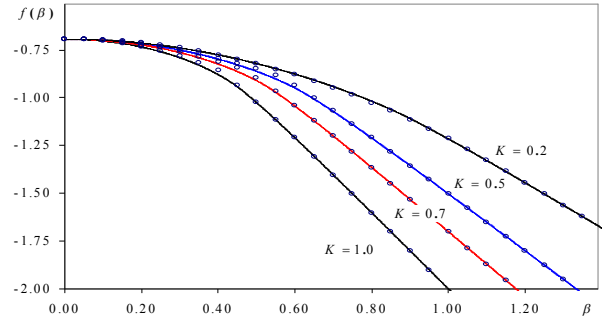


Рис. 5. Свободная энергия $f(\beta)$ для 2D модели Изинга при $K = 0.2, 0.5, 0.7, 1.0$: сплошные кривые - теория Онсагера, кружки - наша теория.

Чтобы сделать это утверждение более наглядным, на рис. 6 для различных значений K показаны графики относительной ошибки

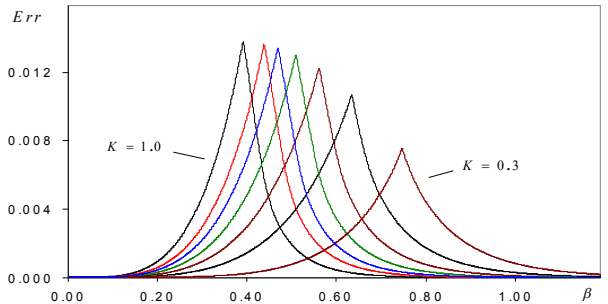


Рис. 6. Для 2D модели Изинга относительная ошибка вычисления свободной энергии для различных K . Слева направо: $K = 1, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3$.

Подведем итоги. Для кубических решеток с большим координационным числом - $\gamma > 16/3$ - наши результаты хорошо согласуются с общепринятыми. Компьютерное моделирование - так же, как и наш подход - предсказывает здесь фазовый переход второго рода. Значения критических температур, полученные компьютерным моделированием для целого ряда значений γ , хорошо описываются нашей формулой (5) - см. табл.1.

Когда $\gamma < 16/3$ наш подход работает плохо. Наибольшим дефектом является предсказание фазового перехода первого рода там, где либо вообще нет никакого перехода ($\gamma = 2$), либо где фазовый переход должен быть второго рода

($\gamma = 4$). Дальнейшие усилия будут направлены на устранение данного дефекта.

Работа выполнена при финансовой поддержке грантами РФФИ № 15-07-04861 и № 16-01-00626.

Approximate calculation of partition function for Ising model on hypercube

B.V. Kryzhanovsky, L.B. Litinskii

Abstract. Previously the n -vicinity method allowing us to calculate approximately the partition function of a spin system with an arbitrary connection matrix was developed. In the present publication, we use these results for the Ising model on a hypercube of an arbitrary dimension. The equation of state is solved and the behavior of the free energy is analyzed. In the case of large dimensions, our theory is in good agreement with the results of computer simulations. For small dimensions there are notable discrepancies with the exact results.

Key words: partition function, n -vicinities method, the Ising Model on hypercube.

Литература

1. B. Kryzhanovsky, L. Litinskii. [Approximate method of free energy calculation for spin system with arbitrary connection matrix](#) // Journal of Physics: Conference Series. - vol. 574, 2015. p. 012017.
2. Б.В. Крыжановский, Л.Б. Литинский. Обобщенное уравнение Брэгга-Вильямса для систем с произвольным дальним действием. ДАН, т.459 (2014), №6, с. 1-5.
3. Б.В. Крыжановский, Л.Б. Литинский Обобщенный подход к описанию распределения энергий спиновой системы. Труды НИИСИ. т.5 (2015), №2, стр. 5-21.
4. Б.В. Крыжановский, Л.Б. Литинский (готовится к печати, 2016).
5. H.W.J. Blote, L.N. Shchur, A.L. Talapov. The cluster processor: new results. Int. J. Mod. Phys. C10 (1999), p.1137.
6. P. H. Lundow, K. Markstrom. The critical behaviour of the Ising model on the 4-dimensional lattice. *Preprint* arXiv:1202.3031v1 (2004).
7. P. H. Lundow, K. Markstrom. The discontinuity of the specific heat for the 5D Ising model. Nuclear Physics B895 (2015), p.305.
8. Р. Бэкстер. Точно решаемые модели в статистической физике. М., Мир, 1985.

Улучшение скалярного нейросетевого дерева, используемого для поиска ближайшего соседа в бинарном пространстве большой размерности

Б.В. Крыжановский¹, М.Ю. Мальсагов²

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»,

Центр оптико-нейронных технологий, Москва, Россия,

E-mail's: ¹ kryzhanov@mail.ru, ² Magomed.Malsagov@gmail.com

Аннотация. Рассматривается задача поиска ближайшего соседа в бинарном пространстве большой размерности, где искомая точка является искаженной версией одной из эталонных точек. Ранние исследования показали, что единственными алгоритмами способными решить поставленную задачу являются полный перебор и нейросетевое дерево поиска. В рамках данной задачи алгоритм превосходит по скорости работы полный перебор в десятки раз. Более того, на больших размерностях нейросетевое дерево поиска можно считать точным алгоритмом, т.к. вероятность ошибки не поддается измерению. В настоящей работе предлагается модификация скалярного нейросетевого дерева, которая повышает превосходство алгоритма над полным перебором до сотен раз без потери надежности работы. Получены аналитические оценки вероятности ошибки дерева, а также оценка оптимального числа выходов в корне дерева.

Ключевые слова: Поиск ближайшего соседа, персептрон, дерево поиска, иерархический классификатор, мульти-классовый классификатор.

1. Введение

Задача поиска ближайшего соседа имеет широкую область применения. Например, распознавание образов [1-7], анализ данных [8], обработка изображений и видео [9-12], картография, компьютерные игры и т.д. Сформулировать кратко ее можно следующим образом. Имеется несколько эталонных точек в некотором пространстве, а так же запросная точка. Необходимо найти такую эталонную точку, которая окажется ближе всех остальных к запросной точке согласно некоторой метрике близости. В большинстве практических задач пространство евклидово и расстояние между точками выбирается евклидово (например, в картографии и компьютерных играх). Так же это может быть пространство признаков, например, яркость пикселей изображения, их координаты и т.д. Ввиду большой значимости задачи за десятилетия исследований было придумано множество алгоритмов для решения данной задачи. Некоторые алгоритмы точные, другие дают лишь приближенное решение. На сложность задачи влияет множество факторов: размерность пространства, тип пространства (бинарное, вещественное и т.д.), число эталонных точек, число запросных точек.

Среди множества алгоритмов для решения задачи поиска ближайшего соседа можно выделить несколько наиболее известных: диаграммы Вороного [13], k-d деревья (например, best-bin-first дерево) [14, 15], метрические деревья (например, spill дерево) [16, 17], LSH (Locality-Sensitive Hashing) [18, 19] и, конечно, полный перебор. Все эти алгоритмы показали свою эффективность в различных областях приложений. Однако такая «обширность» задачи накладывает определенные ограничения на алгоритмы – все они

специализированы по типу данных. Так, диаграммы Вороного применимы в одномерном и двухмерном пространствах, k-d деревья и метрические деревья хороши при размерностях меньше 100. Алгоритм LSH разрабатывался для больших размерностей и хорошо себя показал, но если данные распределены равномерно, то метод начинает очень часто ошибаться.

В настоящей работе предлагается метод решения задачи поиска ближайшего соседа в бинарном пространстве высокой размерности. Несмотря на свою простоту, бинарное пространство имеет свойства, которые делают невозможным применение распространенных методов. Во-первых, дискретность пространства, во-вторых, равномерность распределения точек в пространстве, а, в-третьих, все точки равно удалены друг от друга. Область применения так же сильно сужается: распознавание искаженных черно-белых изображений, анализ бинарных данных и т.д.

Ядром алгоритма являются векторные нейронные сети, а точнее векторный персептрон, хорошо исследованные в ряде работ [20, 21]. Наиболее известными из векторных сетей являются сети Поттса и Хопфилда [22, 23, 1]. Обучение этих сетей выполняется по правилу Хебба [24], вследствие чего процесс обучения является быстрым и детерминированным. Впервые описанный выше алгоритм был опубликован в [25]. Далее последовали исследования и усовершенствования алгоритма, была получена оценка вероятности ошибочного распознавания и другие оценки [26]. В настоящей статье сделан очередной шаг по усовершенствованию алгоритма с целью повышения быстродействия.

Работа имеет следующую структуру. После описания постановки задачи последует описание

простейшего варианта предлагаемого алгоритма. Затем будет продемонстрировано, как себя ведут известные алгоритмы в бинарном пространстве – их преимущества и недостатки. После экспериментального сравнения алгоритмов описывается некоторое улучшение предлагаемого алгоритма, результаты экспериментальных и аналитических исследований.

2. Постановка задачи

Работа предлагаемого алгоритма будет продемонстрирована на примере решения задачи идентификации образа. Суть задачи заключается в следующем. Имеется набор из M эталонных точек. Эталонная точка – это вектор размерности N , каждая компонента принимает равновероятно значения ± 1 . Эталонные точки генерируются случайным образом. Затем в одном из эталонов случайно изменяются несколько координат. Задача состоит в том, чтобы определить, какой из эталонов был искажен.

Очевидно, что при достаточно большом числе искаженных компонент, получившаяся точка может оказаться ближе к любому другому эталону, чем к исходному, или даже точно совпасть с другим эталоном. Поэтому уровень искажения эталона b будет ограничен: $b \in [0; 0.3]$.

Так как рассматриваемое пространство является бинарным, то для определения расстояния между двумя точками удобнее применять расстояние Хемминга – число совпадающих (несовпадающих) компонент у векторов. Определять это расстояние будем как скалярное произведение двух векторов.

Итак, эталон, чье скалярное произведение с запросной точкой максимально по абсолютной величине будет ответом задачи. Однако нельзя определить максимум, не проверив все остальные эталоны, т.е. необходимо сделать полный перебор. Поэтому следует ввести некоторый критерий остановки поиска. В предлагаемом алгоритме будет использоваться следующий критерий

$$|\mathbf{X}\mathbf{X}_\mu| \geq (1 - 2b_{\max})N, \mu = \overline{1, M}. \quad (1)$$

Здесь $b_{\max} \in (0; 0.5)$ – наперед заданная константа, которая определяет размер допустимой окрестности вблизи $\mathbf{X}$. Подробнее параметр $b_{\max}$ будет рассмотрен в описании алгоритма.

3. Описание алгоритма

Основой алгоритма является персептрон (рис. 1), точнее множество персептронов, обученных на своем множестве эталонов каждый и объединенных в древовидную структуру. Все персептроны, образующие дерево, одинаковы и различаются набором эталонов, на которых они обучены. Каждый персептрон состоит из N входных нейронов и двух выходных нейронов. Итак, персептрон, образующий корень дерева, обучается на всех эталонах. Связи, соединяющие входные нейроны с левым выходом, обучаются на первой половине эталонных точек, а

связи, идущие к правому выходу – на второй половине эталонов. Каждый из выходов персептрона указывает на следующий узел дерева. Эти узлы-потомки так же являются персептронами, идентичными корневому узлу. Однако левый персептрон-потомок обучается только на первой половине эталонов, которые соответствовали левому выходу корневого персептрона. Так последовательно дерево «растет» в ширину и глубину, а число эталонов, на которых обучается каждый последующий персептрон, уменьшается. Построение происходит до тех пор, когда на каждый выход персептрона приходится по одному эталону. Итак, каждый узел дерева обучается независимо друг от друга на своем подмножестве эталонных точек: корневой персептрон – на всех M эталонах; потомки корневого узла – на $M/2$ эталонов каждый; узлы на j -м уровне – на $M/2^{j-1}$ эталонах, где $j = 1, 2, \dots, k$, $k = \log_2 M$ – число уровней в дереве. Таким образом, лепесток дерева соответствует одной из эталонных точек.

Отдельно стоит сказать о способе обучения персептрона. В работе для обучения использовалось правило Хебба, которое в данном случае выглядит следующим образом

$$\mathbf{w}_i = \sum_{\mu=1}^2 \sum_{\mu=1+(i-1)\frac{M}{2}}^{\frac{i-1}{2}M} y_\mu \mathbf{X}_\mu. \quad (2)$$

Здесь W_{ij} – весовой коэффициент между j -ым входным нейроном и i -ым выходом, y_μ – случайным образом генерируемый коэффициент, соответствующий каждому эталону, равновероятно принимает значения 1 и -1.

Число операций на создание и обучение дерева составит

$$\Theta = 2MN \log_2 M. \quad (3)$$

Последовательное сужение области поиска – основная идея алгоритма. Прежде чем описывать алгоритм поиска в дереве, необходимо ввести несколько понятий и правил. Не смотря на то, что спуск по дереву происходит последовательно от узла-родителя к узлу-потомку, входной вектор (запросная точка) подается на каждый узел. Персептроны в узлах лишь отсеивают подмножества, в которых не должен находиться ответ. В процессе спуска используются два дополнительных «хранилища»: пул проигравших и пул ответов.

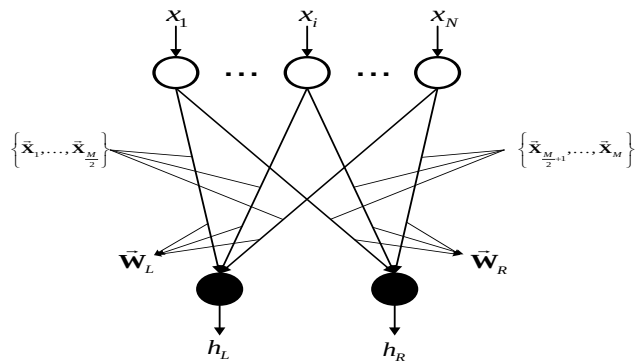


Рис. 1. Схема персептрона, расположенного в каждом узле дерева.

Пул проигравших. При подаче на персептрон входного вектора X на выходах формируется сигнал некоторой величины. Выход с максимальным по абсолютной величине сигналом считается победителем, все остальные – проигравшими. В пуле проигравших, соответственно, сохраняется величина сигнала проигравшего выхода, а также координаты узла, связанного с данным выходом.

Координаты узла – уникальный идентификатор, позволяющий однозначно определить узел дерева.

выходного нейрона. На узел-победитель подается запросная точка и уже на выходах персептрона этого узла определяется наибольший сигнал. Эта процедура – последовательный переход вниз по дереву по узлам-победителям – повторяется, пока не будет достигнут лепесток дерева. При этом на каждом этапе все проигравшие узлы сохраняются в пул проигравших. На самом деле в конце спуска получаются два ответа, так как предпоследний узел указывает на два лепестка, т.е. на два эталона. Поэтому оба эталона вместе со

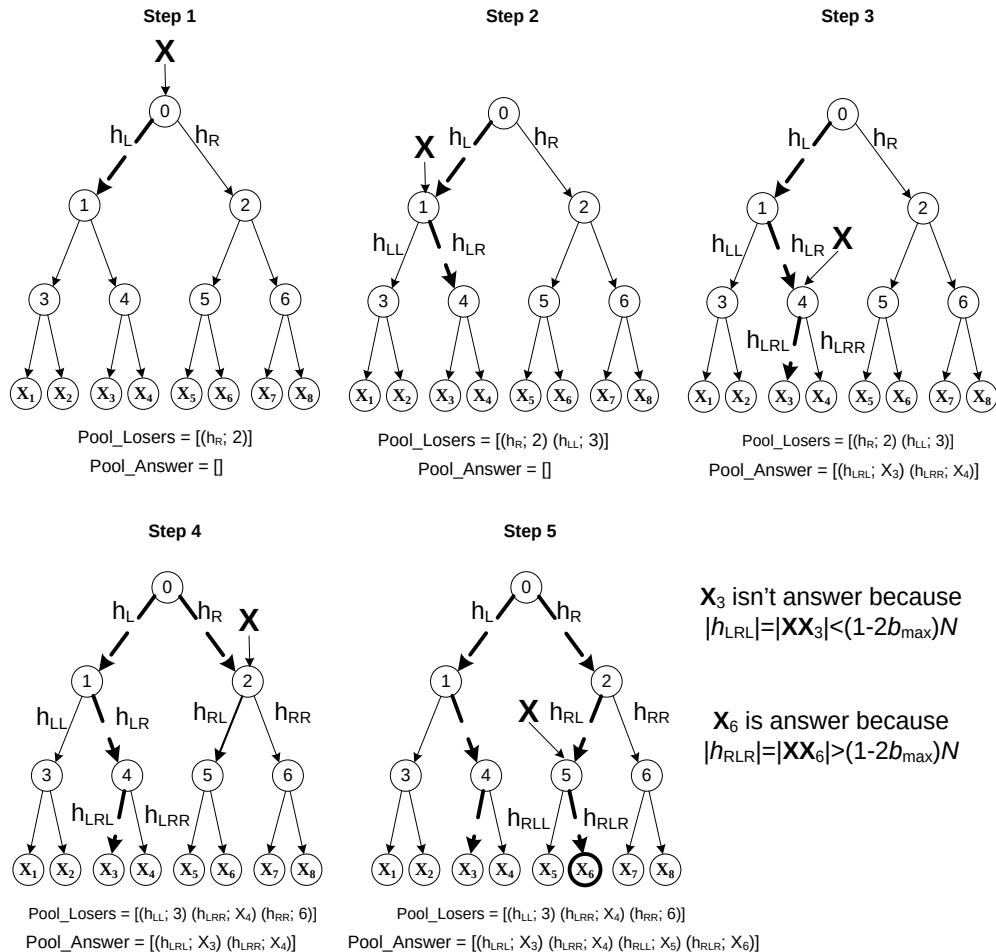


Рис. 2. Пошаговая работа алгоритма для дерева, построенного на восьми паттернах.

Пул ответов. После того, как алгоритм приходит к какому-либо решению (лепесток дерева), в пуле ответов сохраняются номер эталона, содержащегося в лепестке, и величина сигнала на выходе персептрона, соответствующего данному решению.

Таким образом, весь алгоритм поиска будет следующим (рис. 2). Начинается алгоритм с того, что на корневой узел подается запросная точка. Вычисляется величина сигнала на выходах корневого персептрона

$$h_i = \sum_{j=1}^N w_{ij} X_j \quad (4)$$

и определяется выход с наибольшим по модулю сигналом. Затем переходим к узлам, связанным с выходами корневого персептрона. Узел, соответствующий проигравшему выходу, сохраняется в пуле проигравших вместе с величиной сигнала

значения сигналов на выходах персептрона сохраняются в пул ответов. Стоит отметить, что величины сигналов в лепестках по абсолютной величине совпадают с величиной расстояния от запросной точки до соответствующих эталонов. Следовательно, остается проверить удовлетворяет ли какой-нибудь из полученных эталонов критерию остановки. Если критерий выполнен, то алгоритм останавливается, а данный эталон признается ответом задачи. В противном случае, возвращаемся к пулу проигравших, где выбираем узел с наибольшим сигналом. С этого узла начинается новый спуск по дереву до новых лепестков. Сам стартовый узел удаляется из пула проигравших. Эта процедура повторяется, пока не будет выполнен критерий или пул ответов не будет содержать все эталонные точки. В последнем случае просто выбирается ответ с наибольшим значением сигнала.

Стоит пояснить наличие в алгоритме «пула ответов». Могут быть случаи, когда ни один эталон не удовлетворяет критерию остановки. Тогда будет пройдено все дерево, а «пул ответов» будет содержать все эталоны. В результате, в качестве ответа системы, будет выбран наиболее подходящий эталон. Такой исход крайне редок, но возможен из-за сильного искажения входного вектора, либо неудачного выбора

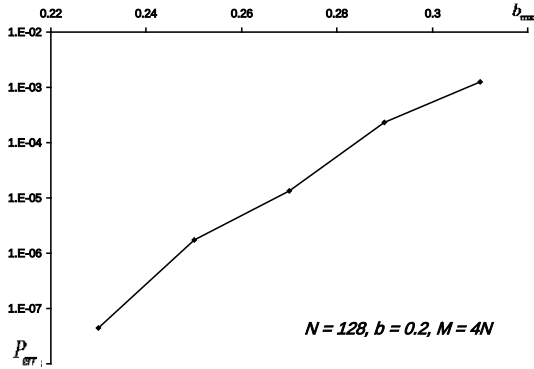


Рис. 3. Снижение вероятности ошибки нейросетевого бинарного дерева при стремлении $b_{\max}$ к b .

параметров в критерии остановки. С другой стороны, можно добавить условие, чтобы алгоритм остановился после нескольких срабатываний критерия, или просто после нескольких дополнительных спусков по дереву после срабатывания критерия. Такой подход имеет смысл, когда важна надежность поиска, а не скорость.

Остается вопрос выбора параметра $b_{\max}$ в критерии остановки. Этот параметр указывает, на каком максимальном расстоянии от запросной точки может находиться эталон, чтобы считаться правильным ответом. То есть при $b_{\max} = 0.5$ точка будет охватывать половину всего пространства вокруг себя, а при $b_{\max} = 0$ – необходимо точное совпадение. Параметр этот выбирается пользователем, однако, если он окажется меньше реального отклонения b запросной точки от правильного эталона, то система не найдет правильный ответ. В этом случае при поиске будет пройдено все дерево, что эквивалентно полному перебору, только вычислительных операций окажется в два раза больше. Выбор величины параметра $b_{\max}$ предоставляется пользователю, однако можно оценить его верхнюю границу исходя из параметров задачи:

$$b_{\max} < \frac{1}{2} \left(1 - \sqrt{\frac{2}{N} \ln \left[\frac{M-1}{P_{\max} \sqrt{2\pi N}} \right]} \right). \quad (5)$$

Здесь N – размерность пространства, M – число эталонов, сохраненных в дереве, $P_{\max}$ – максимально допустимая вероятность ошибки распознавания, обычно выбирается не больше чем $1/M$.

Тем не менее, стоит заметить, что свободный параметр $b_{\max}$ желательно выбирать как можно ближе к реальному уровню искажений b . Это приведет к

значительному снижению вероятности ошибки алгоритма. Так, например, если известно, что входные векторы содержат не более $b = 20\%$ искаженных бит, то следует задать $b_{\max} = 0.22$. На рисунке 3 показано снижение вероятности ошибки нейросетевого бинарного дерева при стремлении $b_{\max}$ к значению b . Видно, что незначительное снижение $b_{\max}$ с 0.31 до 0.29 приводит к понижению вероятности ошибки алгоритма более чем на порядок. Если же задать $b_{\max} = 0.22$, то не представляется возможности экспериментально измерить вероятность ошибки алгоритма (потребуется несколько месяцев машинного времени).

4. Экспериментальные результаты

Для тестирования алгоритмов бинарные эталонные вектора генерировались случайным образом. Нейросетевое дерево сравнивалось с алгоритмами LSH, spill дерево, k-d дерево и полный перебор (использовались реализации алгоритмов из библиотеки OpenCV 2.4.5).

Как и ожидалось, при $N > 100$ k-d дерево, а именно его модификация «best-bin-first», работает в десятки раз медленнее полного перебора, в некоторых случаях – в сотни раз. Поэтому его применение в данной формулировке задачи нецелесообразно.

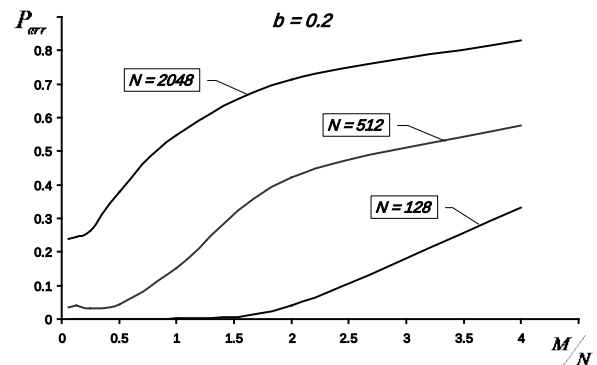


Рис. 4. Вероятность ошибки P_{err} распознавания алгоритма LSH. $b = 0.2$ – доля искаженных компонент в запросной точке, $P = 0.004$ – вероятность ошибки распознавания нейросетевого бинарного дерева при $N = 128$, для больших размерностей эта ошибка близка к нулю.

В случае, когда запросной точкой является один из эталонов, spill дерево показывает хорошие результаты – превосходит полный перебор по скорости работы (с ростом размерности и загрузки выигрыш возрастает, например, при $N = 2048$ и $M = 0.1N - 4N$ алгоритм работает быстрее в 1,5 – 30 раз). Тем не менее, уже небольшое отклонение запросной точки от эталона резко ухудшает работу алгоритма – он проигрывает полному перебору минимум в 1,5 – 2 раза.

Алгоритм LSH разрабатывался для работы с данными высокой размерности. Поэтому, в отличие от

k-d деревьев и spill дерева, рост размерности задачи не замедляет его работу. Однако вероятность ошибки распознавания алгоритма LSH (рис. 4) с ростом размерности и загрузки сети (числа сохраненных эталонных точек) быстро растет. Пока размерность и загрузка сети невелики, LSH вполне успешно решает поставленную задачу (ошибка алгоритма мала), но затрачивает при этом в 5 – 20 раз больше времени, чем полный перебор. При большей загрузке алгоритм LSH ищет решение в два-четыре раза быстрее полного

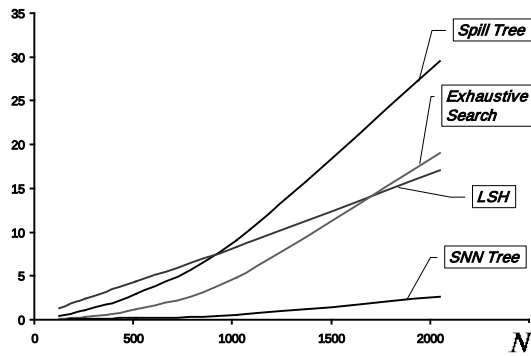


Рис. 5. Время работы алгоритмов при $M = N$, $b = 0.2$.

перебора, но тогда вероятность ошибки близка к 1. Такое поведение связано с тем, что LSH сильно зависит от структуры пространства – в бинарном пространстве эталоны распределены равномерно. Таким образом, поиск LSH осуществляет быстро, но величина вероятности ошибки делает его бесполезным для данной задачи.

На рисунке 5 показана зависимость времени поиска одной запросной точки от размерности задачи различными алгоритмами. Видно, время работы нейросетевого дерева на всем диапазоне в несколько раз меньше остальных алгоритмов и растет медленнее. Заметим, что у LSH линейная зависимость времени работы от размерности, следовательно, при каких-то параметрах он может оказаться быстрее нейросетевого дерева. Тем не менее, выше было показано, что вероятность ошибки при этом слишком высока и выигрыш в скорости бесполезен. Все перечисленные выше алгоритмы проигрывают полному перебору либо по скорости работы, либо по надежности. Поэтому предлагаемое в данной работе нейросетевое дерево будет сравниваться именно с полным перебором.

Рассмотрим теперь результаты скалярного нейросетевого дерева. Оценить, аналитически, вероятность ошибки предлагаемого алгоритма довольно трудно из-за большого числа зависимо действующих элементов и неупорядоченного обхода дерева (спуск может начаться с любого узла). Однако можно оценить верхнюю границу вероятности ошибки. Нейросетевое дерево может ошибиться в том случае, если в наборе паттернов присутствует более одного эталона удовлетворяющих критерию (1). Формально вероятность этого события можно выразить следующим образом:

$$P^* = 1 - \Pr \left[\bigcap_{m=1}^{M-1} |\mathbf{X}\mathbf{X}_m| < (1 - 2b_{\max})N \right]. \quad (6)$$

Наличие таких паттернов не всегда приводит к ошибке алгоритма. Поэтому вероятность (6) можно использовать как оценку сверху вероятности ошибки предлагаемого алгоритма.

Выражение (6) можно посчитать точно, используя методы комбинаторики:

$$P^* = 1 - \left\{ 1 - \sum_{k=0}^{b_{\max}N} \frac{C_N^k}{2^{N-1}} \right\}^{M-1}. \quad (7)$$

Однако использование формулы (7) при больших значениях $N > 200$ не представляется возможным. Для больших размерностей удобнее пользоваться приближенным выражением:

$$P^* < \frac{2M}{\sqrt{2\pi\tilde{N}}} \exp\left(-\frac{\tilde{N}}{2}\right), \quad \tilde{N} = N(1 - 2b_{\max})^2. \quad (8)$$

Из выражения (8) видно, что вероятность ошибки нейросетевого дерева экспоненциально уменьшается с ростом размерности задачи. Таким образом, при больших значениях размерности вероятность ошибки алгоритма измерить экспериментально невозможно. Если оценить вероятность ошибки при параметрах $N = M = 10000$ и $b_{\max} = 0.3$, то, согласно выражению (8), она составит $\sim 1E-270$. Поэтому становится понятно, почему в экспериментах при этих параметрах алгоритм никогда не ошибался. На рисунке 6 показана вероятность ошибки (в логарифмическом масштабе) от размерности задачи, вычисленная экспериментально и

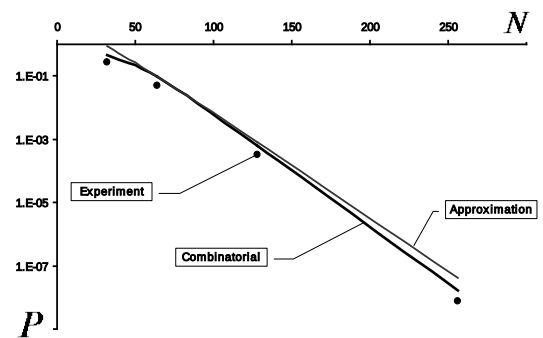


Рис. 6. Вероятность ошибки алгоритма.

по формулам (7), (8). Видно, что формулы достаточно хорошо согласуются друг с другом, а экспериментальные данные лежат ниже теоретических оценок. Таким образом, предложенный алгоритм нейросетевого дерева на больших размерностях можно считать точным.

Прежде чем рассматривать вычислительную сложность нейросетевого дерева и его превосходство над полным перебором заметим, что, как в полном переборе, так и в нейросетевом дереве основная доля вычислений приходится на подсчет скалярных произведений векторов, и это число операций зависит только от размерности и всегда постоянно. Поэтому сравнивать алгоритмы будем в количествах скалярных произведений. Чтобы определить ближайший эталон к запросной точке, его необходимо сравнить с каждым эталоном. Таким образом, вычислительная сложность полного перебора составит

$$\theta_{ES} = M. \quad (9)$$

Для скалярного нейросетевого дерева (СНСТ) получить единую оценку нельзя. В худшем случае может быть обчислено все дерево. Тогда максимальная вычислительная сложность составит

$$\theta_{SNNTree}^{\max} = 2M. \quad (10)$$

С другой стороны, в лучшем случае ответ можно получить после первого же спуска. Тогда минимальная вычислительная сложность составит

$$\begin{aligned} \theta_{SNNTree_1}^{\min} &= 2 \lfloor \log_2 M \rfloor \\ \theta_{SNNTree_2}^{\min} &= 2 \lceil \log_2 M \rceil. \end{aligned} \quad (11)$$

Здесь $\lfloor x \rfloor$ и $\lceil x \rceil$ – обозначают округление до

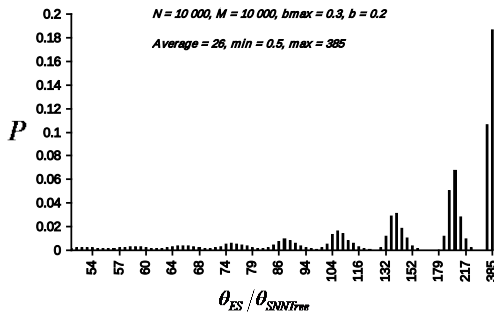


Рис. 7. Распределение отношения вычислительной сложности полного перебора θ_{ES} по обсчету одной запросной точки к вычислительной сложности нейросетевого дерева $\theta_{SNNTree}$.

ближайшего целого меньше x и округление до ближайшего целого больше x , соответственно. Два значения возникают, когда число эталонов не кратно

Выражения (9) – (11) позволяют определить максимальное ускорение, которое можно достичь, используя нейросетевое дерево вместо полного перебора. Так же стоит заметить, что предлагаемый алгоритм в худшем случае может оказаться лишь в 2 раза медленнее полного перебора. Однако эксперименты показали, что вероятность такого исхода составляет менее 0.005.

На рисунке 7 показано распределение отношения вычислительных сложностей полного перебора θ_{ES} к скалярному нейросетевому дереву $\theta_{SNNTree}$. Здесь представлена только крайняя правая часть распределения, так как хвост уходит влево к значению 0.5, а каждой точке соответствует вероятность 10^{-5} . Распределение это быстро затухающее, максимум приходится на крайние две точки, которые соответствуют первому спуску по дереву. Эти точки хорошо согласуются с выражением (11). Таким образом, с вероятностью 30% дерево выдаст ответ в 385 раз быстрее полного перебора. В данном примере среднее ускорение нейросетевого дерева составило 26 раз, а с вероятностью 86% ускорение окажется больше среднего. Конечно, были случаи, когда алгоритм оказался медленнее полного перебора – вероятность 0.005. В ходе тестирования алгоритм ни разу не ошибся.

5. Улучшение алгоритма

До сих пор рассматривалось нейросетевое дерево с минимальным числом настраиваемых параметров.

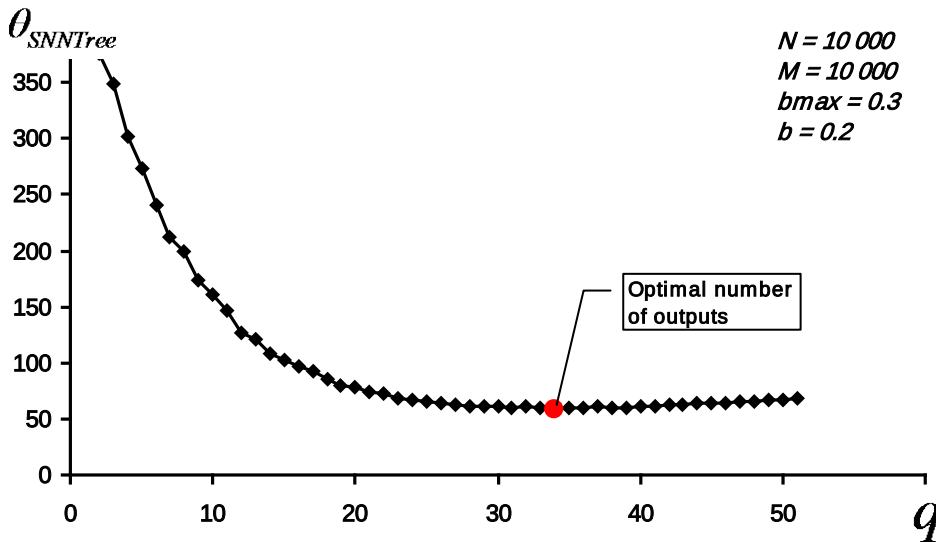


Рис. 8. Зависимость среднего значения вычислительной сложности $\theta_{SNNTree}$ от числа выходов корневого узла q .

степеням двойки, тогда некоторые ветки становятся длиннее других, в противном случае минимум один.

Однако можно работу предлагаемого алгоритма можно улучшить.

Далее будет предложен способ ускорения метода.
На корневой элемент дерева приходится

бесконечности, так как получится просто персептрон с одним выходом на каждый эталон, и весь алгоритм

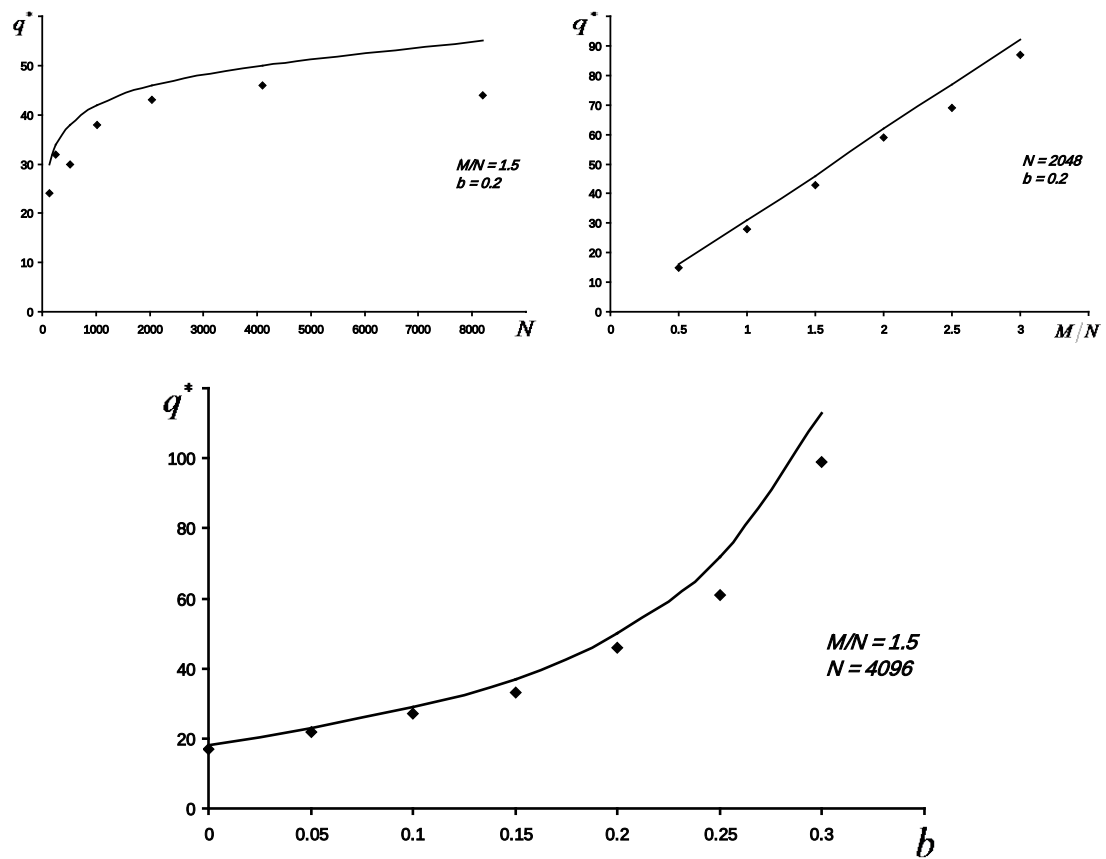


Рис. 9. Зависимости средней вычислительной сложности для значений q^* , полученных экспериментально (маркеры) и теоретически (сплошные линии). Вверху слева – зависимость от размерности при фиксированных уровне искажений и загрузке сети; вверху справа – зависимость от загрузки сети при фиксированных размерности и уровне искажений; внизу – зависимость от уровня искажений при фиксированных загрузке сети и размерности.

наибольшая нагрузка. В рассмотренных выше экспериментах нагрузка корневого персептрона существенно выше $0.14N$ – теоретически максимально возможной загрузки при использовании обучающего правила Хебба. Возникает вопрос, как он вообще работает? Очевидно, что плохо. С вероятностью 57% корневой элемент выбирает правильную ветку дерева, но даже столь незначительного перекоса в сторону правильной ветки достаточно, чтобы дерево работало эффективно. Надежность узлов экспоненциально растет в ходе спуска по дереву, что позволяет шаг за шагом уточнять ответ, а дополнительные итерации в случае ошибочного выбора ветки для спуска всегда скорректируют результат. Таким образом, вся система (дерево) работает надежно и быстро за счет исправления ошибок друг друга. Тем не менее, увеличить надежность корневого узла можно простым увеличением числа его выходов. Тогда нагрузка на каждый выход будет меньше и вероятность выбора правильного направления для спуска возрастет, уменьшив число спусков в целом. Однако нельзя увеличивать число выходов корневого узла до

сведется к полному перебору. Таким образом, возникает дополнительный параметр – оптимальное число выходов q корневого узла дерева.

На рисунке 8 показана зависимость среднего значения вычислительной сложности $\theta_{SNNTree}$ от числа выходов корневого узла q . График построен по экспериментальным данным. Для начала, заметим, что правильный выбор числа выходов на корне дерева позволяет в несколько раз увеличить скорость работы дерева без потери надежности его работы. Использование оптимального числа выходов дает выигрыш в скорости работы до 6-7 раз в зависимости от размерности задачи и числа эталонных точек.

Как и предполагалось, есть точка перегиба, после которой дальнейшее увеличение выходов корневого узла приводит к увеличению вычислительной сложности. Заметим также, что в области минимума зависимость меняется медленно. Поэтому ошибка в выборе оптимального значения q не окажет значительного влияния на скорость работы алгоритма.

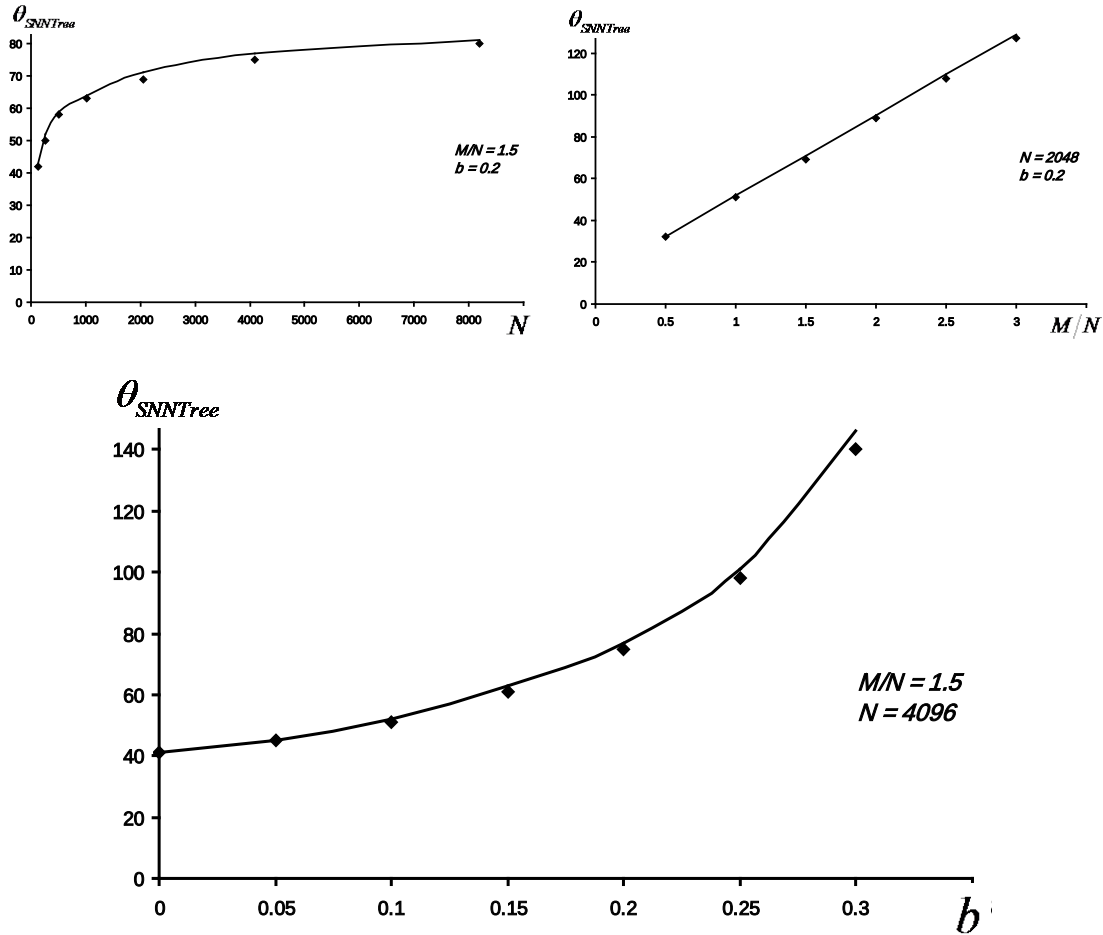


Рис. 10. Зависимости средней вычислительной сложности для значений q^* , полученных экспериментально (маркеры) и теоретически (сплошные линии). Вверху слева – зависимость от размерности при фиксированных уровне искажений и загрузке сети; вверху справа – зависимость от загрузки сети при фиксированных размерности и уровне искажений; внизу – зависимость от уровня искажений при фиксированных загрузке сети и размерности.

Используя экспериментальные результаты, удалось получить оценочное выражение для оптимального числа выходов корневого узла:

$$q^* = \frac{M \log_2 N}{N(1-2b)^2}. \quad (12)$$

На рисунке 9 продемонстрировано соотношение между оптимальным числом выходов q^* , полученным экспериментально – маркеры, и вычисленным по формуле (12) – сплошная линия. По горизонтальной оси отложена доля искаженных компонент во входном векторе. Видно, что выражение (12) неточно описывает экспериментальные наблюдения. Тем не менее, эта погрешность не оказывает большого значения на оценку вычислительной сложности алгоритма, что видно из рисунка 10.

На рисунке 10 представлены зависимости средней вычислительной сложности для значений q^* , полученных экспериментально и теоретически (выражение (12)). Таким образом, формулу (12) вполне можно использовать для выбора оптимального значения числа выходов корневого узла.

Также изменилось распределение отношения вычислительной сложности полного перебора θ_{ES} к скалярному нейросетевому дереву $\theta_{SNNTree}$. Уменьшилась величина максимального ускорения – при $q=2$ ускорение составляло 385 раз (Рис. 7), теперь же оно равно 193 (Рис. 11). При этом в 6.5 раз

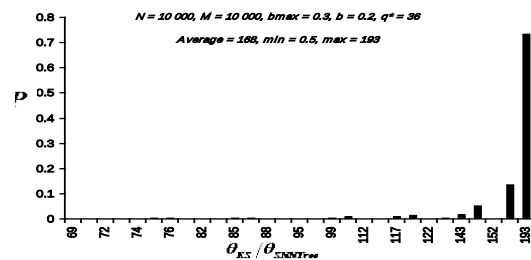


Рис. 11. Распределение отношения вычислительной сложности полного перебора θ_{ES} по обсчету одной запросной точки к вычислительной сложности нейросетевого дерева $\theta_{SNNTree}$.

возросло среднее ускорение с 26 до 168. С первого старта теперь дерево находит ответ с вероятностью 0.87. Вероятность того, что алгоритм будет работать медленнее полного перебора, составила 10^{-5} .

6. Заключение

Предложен древовидный алгоритм для решения задачи поиска ближайшего соседа в бинарном пространстве высокой размерности. Структура и размерность бинарного пространства не позволяют эффективно применять известные алгоритмы – LSH, k-d деревья и spill дерева.

Предлагаемый алгоритм нейросетевого дерева надежно решает поставленную задачу.

Удалось аналитически оценить верхнюю границу вероятности ошибки, которая снижается с ростом размерности задачи.

Проведены экспериментальные сравнения по скорости работы нейросетевого дерева с известными алгоритмами и полным перебором.

Предлагаемый алгоритм в несколько десятков раз работает быстрее всех других методов.

А использование модификации нейросетевого дерева позволяет увеличить ускорение еще до 6-7 раз в зависимости от условий задачи.

Работа выполнена при поддержке РФФИ, гранты № 15-07-04861 и № 16-01-00626.

Improving scalar neural network tree for nearest neighbor search in binary space of high dimension

V.M. Kryzhanovskiy, M.Yu. Malsagov

Abstract. In the binary space of large dimension we analyze the nearest neighbor search problem where the required point is a distorted version of one of the patterns. Previously it was shown that the only algorithms able to solve the set problem are the exhaustive search and the neural network search tree. For the given problem the speed of operation of the last algorithm is dozens of times larger comparing with the exhaustive search. Moreover, in the case of large dimensions the neural network tree can be regarded as an accurate algorithm since the probability of its error is so small that cannot be measured. In the present publication, we propose a modification of the scalar neural network tree allowing the speeding of the algorithm's operation up to hundred times without losses in its reliability. Analytical estimations of error probability of tree and optimal number of outputs in tree root are obtained.

Keywords: nearest neighbor search, perceptron, search tree, hierarchical classifier, multi-class classification.

Литература

1. J.J. Hopfield. Neural Networks and physical systems with emergent collective computational abilities. // Proc.Nat.Acad.Sci.USA. vol.79, pp.2554-2558 (1982).
2. T. Poggio, F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. // Science 247, pp.978-982 (1990).
3. G. Pinkas, R. Dechter. Improving Connectionist Energy Minimization. // Journal of Artificial Intelligence Research, vol.3 (1995), pp.23-48 (1995).
4. S. Mulder and D. Wunsch II. A Million City Traveling Salesman Problem Solution by Divide and Conquer Clustering and Adaptive Resonance Neural Networks. // Neural Networks vol.16, No.5-6, pp.827-832 (2003).
5. B.V. Kryzhanovsky, B.M. Magomedov, A.L. Mikaelyan. A Relation Between the Depth of a Local Minimum and the Probability of Its Detection in the Generalized Hopfield Model. Doklady Mathematics, vol.72, N3, pp. 986-990 (2005).
6. K.A. Smith. Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research. // INFORMS Journal on Computing v.11 (1), pp.15-34 (1999).
7. B.V. Kryzhanovsky, V.M. Kryzhanovsky, A.L. Mikaelian and A. Fonarev. Parametric dynamic neural network recognition power. Optical Memory&Neural Network, Vol. 10, №4, pp.211-218 (2001).
8. V. Koivune and S. Kassam. Nearest neighbor filters for multivariable data. IEEE Workshop on Nonlinear Signal and Image Processing, 1995.
9. C. Faloutsos, R. Barber, M. Flickner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. Journal of Intelligent Information Systems, 3(1994):231-262.
10. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: the QBIC system. IEEE Computer, 28(1995):23-32.

11. A. Pentkurd, R.W. Picard, and S. Sclaroff. Photobook: tools for content-based manipulation of image databases. In Proceedings of the SPIE Conference on Storage and Retrieval of Image and Video Databases II, 1994.
12. A.W.M. Smeulders and R. Jai, eds. Image Databases and Multi-media Search. Proceedings of the First International Workshop, IDB-AIMS '96, Amsterdam University Press, Amsterdam, 1996.
13. F.P. Preparata and M. Shamos. Computational Geometry. Springer-Verlag. 1985.
14. J.H. Friedman, J.L. Bentley and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software. vol. 3. pp. 209–226. 1977.
15. J.S. Beis and D.G. Lowe Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 1000-1006. 1997.
16. J.K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. Information Processing Letters. vol. 40. pp. 175–179. 1991.
17. Ting Liu, Andrew W. Moore, Alexander Gray and Ke Yang. An Investigation of Practical Approximate Nearest Neighbor Algorithms. Proceeding of Conference. Neural Information Processing Systems. 2004.
18. P. Indyk, R. Motwani, P. Raghavan and S. Vempala. Locality-preserving hashing in multidimensional spaces. In Proc. 29th STOC. pp. 618–625. 1997.
19. P. Indyk and R. Motwani Approximate nearest neighbors: Towards removing the curse of dimensionality. In Proc. 30th STOC. pp. 604–613. 1998.
20. B.V. Kryzhanovsky, L.B. Litinskii and A. Fonarev. Parametrical neural network based on the four-wave mixing process. Nuclear Instruments and Methods in Physics Research, A. vol 502, No.2-3, pp. 517 - 519 (2003).
21. B.V. Kryzhanovsky, A.L. Mikaelian and A.B. Fonarev. Vector neural net identifying many strongly distorted and correlated patterns. Int. conf on Information Optics and Photonics Technology, Photonics Asia-2004, Beijing-2004. Proc. of SPIE, vol. 5642, pp. 124-133.
22. I. Kanter. Potts-glass models of neural networks // Physical Review A.– 1988.– v.37 (7).– pp. 2739-2742.
23. F.Y. Wu. The Potts model // Review of Modern Physics.– 1982.– v.54.– 235-268.
24. D.O. Hebb. The Organization of Behavior // New York: Wiley, 1949.
25. V. Kryzhanovsky, M. Malsagov, J. A. C. Tomas. Hierarchical classifier: based on neural networks searching tree with iterative traversal and stop criterion. // Optical Memory and Neural Networks (Information Optics).– 2013.– Vol. 22, №4.– pp. 217–223.
26. Vladimir Kryzhanovsky, Magomed Malsagov, Juan Antonio Clares Tomas, Irina Zhelavskaya. On Failure Probability of Search in High-Dimensional Binary Space by Scalar Neural Network Tree // Proc. of 6th International Conference on Neural Computation Theory and Applications. Rome. 22-24 October 2014. P. 300-306.

Сценарное исследование сложных систем: уязвимость и чрезвычайные ситуации

Р.О. Пономарев¹, Д.А. Кононов², Н.О. Пономарев³

^{1,3} ФГБУН «Институт проблем управления им. В.А.Трапезникова РАН»,
Москва, Россия, E-mail's: ¹ RomaN3457@yandex.ru, ³ nik38958@rambler.ru,

² ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»,
Москва, Россия, E-mail: ² dmitrykon52@gmail.com

Аннотация. Рассмотрены возможности сценарного исследования уязвимости сложных систем и анализа чрезвычайных ситуаций. Предложена иерархия понятий, вербальные и формальные описания характеристик уязвимости. Формализованы постановки задач анализа уязвимости сложной системы в условиях распространения возмущений при реализации угроз.

Ключевые слова: сценарное исследование, уязвимость, чрезвычайная ситуация, сложные системы.

Введение

В работах [0, 0, 0, 0, 0] сформулированы базисные понятия сценарного исследования сложных систем (СС). Под сценарным исследованием понимается такой способ изучения СС, когда основным средством ее изучения является построение и анализ спектра сценариев в различных ее стратах, а целью исследования – синтез сценария с заданными свойствами. Особое внимание уделено задачам поиска условий их безопасного функционирования и развития [6]. Для решения задач безопасности предложено использовать средства сценарного анализа и сценарного синтеза на основе построения сценарной системы.

Сценарная система имеет следующие основные входные компоненты: основной метанабор, квазиинформационная гипотеза, формализованное описание предметной области исследования в виде экспертно-значимых событий (ЭЗС) [0].

Проведение сценарного исследования требует формализовать цели и задачи, в том числе построить определенную иерархию понятий, динамическую модель функционирования объекта, его сценарные характеристики, выделить свойства, экспертно-значимые состояния и т.п.

В работах [0, **Ошибка! Источник ссылки не найден.**, 0, 0, 0] были введены формальные модели стойкости, живучести и уязвимости сложных организационно-технических систем. В то же время в предыдущих работах не были исследованы проблемы управления безопасностью в чрезвычайных ситуациях (ЧС). В настоящей работе формализованы: понятие ЧС, превентивных способов их выявления на основе анализа характеристик сценариев

уязвимости, так же эффективных сценариев их ликвидации.

Для решения указанных задач следует:

- рассмотреть нормативные документы, которые формулируют понятие, определение и содержание чрезвычайной ситуации;
- построить иерархию формализованных понятий и сценарную модель ЧС;
- предложить методику анализа определения ЧС;
- предложить методы управления предупреждения и ликвидации ЧС.

Существующий в настоящее время аппарат системного анализа ориентирован на исследование проблем и динамики развития сложных крупномасштабных систем, обеспечивает рассмотрение множества альтернативных решений, каждое из которых описывается достаточно большим числом переменных, позволяет учитывать риски различного типа, вырабатывать эффективные решения в условиях ограниченного времени и ресурсов.

Широкое распространение в последние годы получили методы сценарного исследования СС.

Многогранность объекта исследования при формализованном подходе отображается в разнообразии его характеристик и свойств. Под характеристикой подразумевается формальная модель качественной стороны объекта исследования. Соответствующая выбранной шкале числовая функция f называется *числовой характеристикой*, или *показателем измерения объекта*; булевская характеристика объекта представляет собой модель *свойства объекта*. Возможны и другие модели характеристик. Ряд характеристик, используемых в прикладных системах различного назначения, в том числе технических, технологических, организаци-

онных, производственных, экономических и социальных, предложены в работах [0, 0, **Ошибка! Источник ссылки не найден.**, 0, 0].

Иерархия понятий при моделировании проблемы уязвимости и ЧС в сложных системах

Исходным пунктом исследования, предлагаемого в настоящей работе, является концепция безопасности сложной системы, наиболее важными свойствами которой, являются надежность, стойкость и живучесть. Каждое из этих свойств характеризует внутрисистемные связи и взаимодействие ее с окружающей средой с различных сторон. Исследование перечисленных свойств позволяет уменьшить угрозу возникновения нештатных ситуаций (НШС) и чрезвычайных ситуаций (ЧС), приводящих к выходу из целевого режима функционирования (ЦРФ), авариям и катастрофам в СС.

Надежность – системный параметр, свойство системы сохранять в течение заданного промежутка времени значение параметров, характеризующих проектный ЦРФ системы. Это свойство характеризует эффективность проектирования системы.

Надёжность сложной системы позволяет обеспечить минимальное количество уязвимых мест в системе, реализация угроз в которых может приводить к критической ситуации.

Стойкость – системный параметр, характеризующий способность противостоять возмущениям, в том числе воздействиям и функционировать в штатном режиме в условиях возникновения НШС. Основной характеристикой стойкости системы служит время достижения системой предельного состояния работоспособности. Увеличение этого промежутка времени, как правило, способствует уменьшению опасности развития НШС в системе.

Живучесть – системный параметр, свойство системы, характеризующее ее способность функционировать под влиянием воздействий, возникающих в процессе ее функционирования с учетом возможности восстановления функционирования.

Живучесть сложной системы учитывает возможность самовосстановления при реализации угрозы или ремонтоспособности повреждённых элементов в системе во время кризисной ситуации.

Кризисная ситуация – ситуация, при которой без надлежащего управления следует выход за пределы безопасности, при этом методами поиска обратного управления может быть найдено несколько вариантов управления, которое сохраняет функционирование системы в пределах безопасности.

Критическая ситуация – ситуация, т.е. конфигурация значений существенных параметров сис-

темы, при которой без надлежащего управления следует выход за пределы безопасности, при этом методами поиска обратного управления может быть найдено единственный вариант управления, которое сохраняет функционирование системы в пределах безопасности.

Живучесть системы не допускает перехода критической ситуации в ЧС, если обеспечено надлежащее управление. Надёжность, стойкость и живучесть характеризуются значениями и структурой показателей её компонент, а также динамикой изменения системных элементов окружающей среды.

Промежуточной является *неконтролируемая ситуация* – это такая ситуация, когда при обнаружении выхода за пределы допустимости, невозможно найти управленческие решения с целью избежать чрезвычайной ситуации.

Чрезвычайная ситуация – это неблагоприятное сочетание факторов и обстоятельств, нарушающих условия их нормальной жизнедеятельности, препятствующих производственной, хозяйственной, бытовой и другим видам деятельности. Чрезвычайная ситуация характеризуется выходом целевых параметров за границу допустимости.

К чрезвычайным ситуациям невоенного характера можно отнести стихийные бедствия, катастрофы, аварии, в частности АЭС, гидротехнических сооружений, химических производств и т.п.

ЧС классифицируют по различным параметрам.

По масштабам выделяют:

- межгосударственные;
- общегосударственные;
- региональные;
- местные;
- объектовые;
- технологические ЧС.

По времени, динамике развития и ликвидации последствий выделяют:

- стратегические ЧС, которые очень быстро приводят к катастрофическим последствиям большого масштаба;
- медленно развивающиеся ЧС;
- ЧС оперативного плана (аварии), последствия которых обычно носят локальный характер.

По виду нанесенного ущерба различают:

- ЧС с прямым и косвенным ущербом;
- ЧС, представляющие угрозу жизни людей;
- ЧС, приводящие к нарушению экологического равновесия;
- ЧС, приводящие уничтожению материальных ресурсов и т. д.

ЧС подразделяются также по

- причинам возникновения;
- последствиям;
- стратегиям их ликвидации;
- возможностям использования противоаварийного упреждающего планирования и т.д.

Существующая тенденция к возрастанию масштабов ЧС и тяжести их последствий обуславливает необходимость своевременного их прогнозирования и управленческих структур – систем управления в условиях ЧС (АСУ ЧС) [0].

Важнейшей компонентой АСУ ЧС является система поддержки принятия решений на всем цикле прогнозирования и ликвидации ЧС, исходным пунктом которой является построение модели ЧС.

Для моделирования ЧС необходимо определить и формализовать свойства, характеристики, динамику изменения обстановки, в том числе модель превентивных мероприятий, снижающих возможность появления непредвиденных ситуаций. Превентивные мероприятия улучшают эти системные параметры. Превентивные мероприятия включают:

- защиту системы;
- профилактику возникновения источника ЧС;
- мониторинг состояния системы;
- мероприятия по подготовке к ЧС;
- своевременное восстановление элементов.

Уязвимость – системный параметр, характеризующий возможность нанесения описываемой системе повреждений любой природы, нарушающих ЦРФ. Априори характер, степень и возможности устранения указанных повреждений зависят от созданной конструкции (структуры системы), условий функционирования (эксплуатации), а также методов и средств воздействия. Уязвимость непосредственно связана с надежностью, стойкостью и живучестью ОТС. Концептуальная модель уязвимости СС предложена в [Ошибка! Источник ссылки не найден.].

Данные ранее вербальные описания понятий теории уязвимости являются первоначальным этапом моделирования. Для построения методик эффективного управления в выделенных ситуациях необходимо формальное описание. Для изучаемой проблематики введем следующие определения.

Соответствие процесса функционирования СС ее назначению, а также безопасности, определим областью целевого назначения и безопасности (ОЦНБ) $Q(\mathbf{z}(t), t)$, зависящей от расширенного состояния системы $\mathbf{z}(t)$.

Здесь $\mathbf{z}(t) = (\mathbf{x}(t), \mathbf{y}(t))$, $\mathbf{x}(t) \in X$ – внутреннее состояние системы, $\mathbf{y}(t) \in Y$ – состояние внешней среды системы, $Z = X \times Y$.

Определение 1. Сценарий $\mathfrak{R}(\Delta)$ на временном горизонте Δ назовем сценарием безопасности, если выполнено условие

$$\mathbf{z}(t) \in Q(\mathbf{z}(t), t) \text{ для всех } t \in \Delta,$$

т.е. все экспертно-значимые события $\mathbf{z}(t)$ сценария $\mathfrak{R}(\Delta)$ содержится внутри множества условно допустимых состояний $Q(\mathbf{z}(t), t)$.

Сценарная стойкость системы (компоненты системы) – сохранять сценарий безопасности при реализации угроз.

Определение 2. Чрезвычайная ситуация $\mathbf{z}(t)$ в сложной системе – это такое состояние, при котором сценарий поведения системы отклоняется от сценария безопасности $\mathfrak{R}(\Delta)$, или при некотором t выполнено:

$$\mathbf{z}(t) \notin Q(\mathbf{z}(t), t).$$

ЧС возникает в результате возникновения источника ЧС – угрозы $\gamma \in \Gamma^{(Y, \alpha)}$, где $\Gamma^{(Y, \alpha)}$ – множество угроз, при этом наносится ущерб $W(\mathfrak{R}(\Delta, \gamma), Q)$ элементам системы S , в том числе расширенному системному элементу $E^{(Z)}(B, r, p)$, где $B \subseteq Z$, $r \in R^{(Z)}$, $p \in P^{(Z)}$. Таким образом, нарушается стойкость системного элемента.

Опасность возникновения ЧС в СС – это возможность реализации угрозы, приводящей к ЧС, которая может быть обнаружена при превентивном исследовании ее поведения. С точки зрения сценарного анализа для определения характеристики опасности введено понятие T -шаговой угрозы [0].

Защищенность от ЧС по отношению к заданной компоненте S_α , $\alpha \in \hat{A}$ СС – это способность системы частично или полностью восстанавливать ЦРФ при реализации угрозы по отношению компоненты S_α .

Мероприятия при ЧС – это совокупность управляющих воздействий (планируемых и реализуемых), направленных на решение задач по предупреждению и ликвидации ЧС.

По характеру повреждений при наступлении ЧС выделим:

- элемента системы $x_i \in X$;
- конструктивного решения (структуры системы) $R(X)$;
- концепта системы, т.е. заданных (проектируемых) свойств, в том числе ЦРФ $P(X)$;
- системных элементов, в том числе подсистем рассматриваемой системы $M_E^{(X)}(B, r, p)$;
- внутреннего состояния системы, т.е. набора системных элементов и связей между ними;
- комплексные повреждения.

По степени повреждений выделим:

- сильные повреждения;
- средние повреждения;
- незначительные повреждения.

Характеристикой уязвимости является коэффициент $C_\alpha^{(y)}(\gamma, \Delta)$ уязвимости компоненты S_α СС при применении угрозы $\gamma \in \Gamma^{(Y, \alpha)}$ на временном отрезке Δ , которая представляет собой величину ущерба $W(\mathfrak{R}(\Delta, \gamma), Q)$.

Таким образом могут быть исчислены

- минимальный коэффициент уязвимости для множества угроз $\Gamma^{(Y, \alpha)}$

$$C_\alpha^{(z, \min)}(\gamma, \Delta) = \min_{\gamma \in \Gamma^{(Y, \alpha)}} C_\alpha^{(z)}(\gamma, \Delta);$$

– максимальный коэффициент уязвимости для множества угроз $\Gamma^{(Y, \alpha)}$

$$C_{\alpha}^{(z, \max)}(\gamma, \Delta) = \max_{\gamma \in \Gamma^{(Y, \alpha)}} C_{\alpha}^{(z)}(\gamma, \Delta);$$

– минимальный коэффициент уязвимости для элементов системы

$$C_{\alpha}^{(z, \min)}(\gamma, \Delta) = \min_{\alpha \in A} C_{\alpha}^{(z, \min)}(\gamma, \Delta);$$

В соответствии с таким подходом сильными повреждениями назовём повреждения, при которых стойкость компоненты системы нарушается на величину равную элементу вектора поля изменения умноженного на максимальный коэффициент уязвимости, при реализации угроз $\Gamma^{(Y, \alpha)}$.

$$\Psi_{\alpha}^{(z, \max)} \geq C_{\alpha}^{(z, \max)}(\gamma, \Delta) \delta a(t),$$

где $\Psi_{\alpha}^{(z, \max)} = d_E^{\max}(\mathfrak{N}, a, \Delta)$, a – целевой режим функционирования, $\delta a(t)$ – евклидово рассогласование между a и $z(t)$ в момент времени t .

Средними повреждениями назовём такие повреждения, при которых стойкость компоненты нарушается в пределах от $\Psi_{\alpha}^{(z, \min)}$ до $\Psi_{\alpha}^{(z, \max)}$.

Несущественными повреждениями назовём такие повреждения, при которых стойкость компоненты сложной системы нарушается на величину равную элементу вектора поля изменений умноженного на минимальный коэффициент уязвимости при реализации угроз $\Gamma^{(Y, \alpha)}$.

Чрезвычайные ситуации могут различаться по длительности времени:

- мгновенные (соответствующие одному шагу сценария);
- средней продолжительности (в пределах горизонта сценария);
- длительные (превышающие горизонт сценария).

По направленности ЧС:

- ЧС, увеличивающая силу воздействия на систему (величина ущерба увеличивается со временем);
- ЧС с ослабевающей силой воздействия на систему (величина ущерба уменьшается со временем);
- ЧС постоянной силы (величина ущерба во времени не изменяется или изменяются незначительно на выбранном промежутке сценария).

– максимальный коэффициент уязвимости для элементов системы

$$C^{(z, \max)}(\gamma, \Delta) = \max_{\alpha \in A} C_{\alpha}^{(z)}(\gamma, \Delta)$$

На основе этих характеристик компоненту S_{α} сложной системы S назовем системно уязвимой, если найдется угроза $\gamma \in \Gamma^{(Y, \alpha)}$, реализация которой нарушает ее стойкость системы S .

По воздействию ЧС на элементы системы:

- влияющие напрямую (реализация ЧС в элементе является источником ЧС смежного элемента);
- влияющие косвенно (реализация ЧС в элементе является источником ЧС для элементов, находящихся в пределах достижимости для источника ЧС на заданном горизонте сценария);
- не влияющие (реализация ЧС в элементе при распространении возмущений по системе не реализует ЧС на заданном горизонте сценария для экспертно-значимых элементов).

Заключение

Укажем направления дальнейших исследований.

1. Поиск областей воздействия для заданных конфигураций структуры (системных элементов).
2. Определение характера воздействия для элементов базисной структуры системы на основе расчета базисных сценариев функционирования системы.
3. Выявление узлов уязвимости структуры системы.
4. Введение и обоснование применения новых средств сценарного исчисления [0, **Ошибка! Источник ссылки не найден.**].
5. Формальный анализ причин возникновения ЧС, ее продолжительности, наносимого ущерба.
6. Разработка эффективных превентивных мероприятий по предотвращению ЧС и оптимальные сценарии ликвидаций последствий.

Scenario research of complex systems: vulnerability and emergencies

R.O. Ponomarev, D.A. Kononov, N.O. Ponomarev

Abstract. Scenario research of complex systems' vulnerability and the analysis of emergencies are considered. The hierarchy of concepts, the verbal and the formal descriptions of vulnerability characteristics are proposed. Tasks of complex systems' vulnerability analysis under disturbance propagation in the implementation of threats are formalized.

Keywords: scenario research, vulnerability, emergencies, complex systems.

Литература

1. Д.А. Кононов. Модели и методы анализа и синтеза сценариев развития социально-экономических систем. Диссертация на соискание ученой степени доктора технических наук. – М.: ИПУ РАН, 2010, 432 с.
2. Д.А. Кононов, С.А. Косяченко, В.В. Кульба. Модели и методы анализа сценариев развития социально-экономических систем в АСУ ЧС //АиТ. № 9. 1999. с. 122-136.
3. В.В. Кульба, Д.А. Кононов, С.А. Косяченко, А.Н. Шубин. Методы формирования сценариев развития социально-экономических систем /Серия «Системы и проблемы управления». – М.: СИНТЕГ, 2004, 296 с.
4. Модели и методы анализа и синтеза сценариев развития социально-экономических систем. Кн. 1 /Под редакцией чл.-корр. РАН В.Л.Шульца, д.т.н., проф. В.В.Кульбы – М.: Наука, 2012. (коллективная монография). 307 с.
5. Модели и методы анализа и синтеза сценариев развития социально-экономических систем. Кн. 2 /Под редакцией чл.-корр. РАН В.Л.Шульца, д.т.н., проф. В.В.Кульбы – М., Наука, 2012. (коллективная монография). 358 с.
6. Д.А. Кононов. Эффективные стратегии формирования сценариев в АСУ ЧС //АиТ. № 2. 2001. с. 170-181.
7. Д.А. Кононов, А.А. Кочкаров, Н.О. Пономарев. Стойкость сложных технических систем: сценарный взгляд на проблему /Труды НИИСИ РАН. – М.: НИИСИ РАН, 2012, с. 80-86.
8. Кононов Д.А., Пономарев Н.О. Сценарный анализ эффективности применения методов группового управления на железнодорожном транспорте //Интеллектуальные системы управления на железнодорожном транспорте. Первая научно-техническая конференция. Материалы конференции. – М.: ОАО «НИАС», 2012.
9. Д.А. Кононов, Н.О. Пономарев. Сценарные характеристики стойкости и живучести сложных организационно-технических систем //«Проблемы управления безопасностью сложных систем». Материалы XX международной научной конференции. Москва, декабрь 2012 г. РГГУ. – М.: 2012. с. 396-400.
10. Д.А. Кононов, Н.О. Пономарев. Управление безопасностью объектов инфраструктуры железнодорожного транспорта /Управление развитием крупномасштабных систем (MLSD'2013): Материалы Седьмой международной конференции, 30 сентября – 2 октября 2013г., Москва: в 2 т., с. 96-100.
11. Д.А. Кононов, Н.О. Пономарев, Р.О. Пономарев, М.П. Барбашев. Региональные системы: методические основы определения уязвимости. – XVIII Международная конференция «Управление развитием крупномасштабных систем» MLSD-2015, Том 2, с. 147-150.
12. Н.О. Пономарев, Д.А. Кононов, Д.А. Швецов, Р.О. Пономарев. Сценарное исследование уязвимости сложных организационно-технических систем /Труды НИИСИ РАН «Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты» – М.: НИИСИ РАН, Т. 4, № 1, 2014, с. 64-68.
13. Д.А. Кононов. Основы исчисления сценариев поведения сложных систем в АСУ ЧС //АиТ. № 9. 2002. с. 142-152.
14. В.В. Кульба, Д.А. Кононов, С.А. Косяченко. Сценарное исчисление как методология анализа сложных систем //Тр. ИПУ РАН. Т. IX. – М.: ИПУ РАН. 2000. с. 16-34.

Сценарный анализ уязвимости сложной системы: вычислительный эксперимент

Р.О. Пономарев¹, Д.А. Кононов², И.М. Хасанов³

¹ ФГБУН «Институт проблем управления им. В.А.Трапезникова РАН»,
Москва, Россия, E-mail: RomaN3457@yandex.ru

² ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»,
Москва, Россия, E-mail: dmitrykon52@gmail.com

³ ФГБОУВО «Московский технологический университет»,
Московский государственный технический университет радиотехники, электроники
и автоматики, Москва, Россия, E-mail: kit2010-10@mail.ru

Аннотация: Предложена методика сценарного анализа уязвимости сложной системы. Рассмотрен пример сценарного анализа уязвимости сложной системы «городская инфраструктура». Разработаны вычислительные алгоритмы реализации предложенной методики. Приведены результаты расчетов. Проведена классификация опасности возникновения чрезвычайных ситуаций.

Ключевые слова: сценарный анализ, уязвимость, городская инфраструктура, сложная система, вычислительный эксперимент.

Введение

Общетеоретическая модель сценарного исследования уязвимости и поведения сложной системы в условиях ЧС для прикладного применения требует подробного описания основных элементов проблемной ситуации. Для реализации этой задачи следует провести комплекс работ:

- сформулировать проблему исследования;
- построить модель объекта;
- определить и формализовать экспертно-значимые события;
- определить и формализовать перечень угроз;
- формализовать сценарные характеристики объекта исследования;
- разработать программу сценарного анализа (вычислительного эксперимента);
- разработать алгоритмы и программы расчета характеристик;
- провести необходимые расчеты;
- сформулировать выводы анализа.

Рассмотрим возможности применения общей методологии уязвимости для модели поведения сложной системы «операторные графы» [1, 2].

В предложенной графовой модели каждая вершина может являться носителем потенциальной угрозы. В статье рассматривается задача определения наиболее уязвимых элементов системы от источника угроз на основе характеристики «Т-шаговая угроза». Моделирование поведения объекта, связанного с возникновением и развитием новой ситуации, позволяет прогнозировать момент времени (шаг сценария) первого выхода значения в вершине за границы целевого режима функционирования (ЦРФ) в отсутствие управляющих воздействий.

1. Методика сценарного анализа уязвимости сложной системы на графовой модели

Для моделирования процесса функционирования СС на орграфе оперирующая сторона располагает средствами регистрации значений мгновенного изменения значений $\delta I_j(\tau)$ параметров в j -й вершине графа, которые представляют собой возмущения или управления, реализованные в j -й вершине в момент времени τ , т.е. регистрировать k -шаговый импульсный процесс возмущений (ИПВ)

$$\delta \mathbf{Im}(t, k) = (\delta I_j(\tau) \quad 1 \leq j \leq n; \quad t \leq \tau \leq t+k).$$

В соответствии со схемой преобразования состояния в операторных графах динамику системы определяют соотношения:

$$\mathbf{z}(t) = \mathbf{z}(t-1) + \mathbf{Im}(t) \quad \text{при } t=1, 2, \dots$$

или

$$\mathbf{Im}(t) = A(t-1)\mathbf{Im}(t-1) + \delta \mathbf{Im}(t) \quad \text{при } t=1, 2, \dots$$

Здесь $\mathbf{z}(0) = \mathbf{z}^{(0)}$ – начальное, $\mathbf{z}(t)$ – текущее состояние системы; $\mathbf{z}(t-1)$ – расширенное состояние системы в момент времени $t-1$; $A(t-1)$ – матрица смежности в момент времени $t-1$; $\mathbf{Im}(t-1)$ – импульс, накопленный к моменту времени $(t-1)$, $\mathbf{Im}(0) = \delta \mathbf{Im}(0)$ – начальный импульс; $\delta \mathbf{Im}(t)$ – импульс в моменты времени $t=1, 2, \dots$

В рамках модели «Угроза-НШС» [1] границы допустимости зададим для вектора целевых элементов $\delta \mathbf{a} = \{\delta_i, i \in N_a \subseteq N\}$ из множества целевых элементов N_a . Рассогласование между ЦРФ $\mathbf{a}$ и текущим расширенным состоянием ОТС $\mathbf{z}(t)$ задается

в виде условий:

$$\rho(\mathbf{z}(t) - \mathbf{a}) \leq \delta \text{ при } t=1, 2, \dots,$$

где ρ – заданная метрика. В частности, при задании границ допустимости

$$\delta(t) = \{\delta_i(t), i \in N\} \text{ при } t=1, 2, \dots,$$

по каждому элементу системы временные границы стойкости системы по отношению к импульсным возмущениям должны удовлетворять условиям:

$$|z_i(t) - a_i| \leq \delta_i(t) \text{ при } t=1, 2, \dots, i \in N.$$

При этом границы стойкости существенно зависят от ρ , $\mathbf{a}$ и δ .

Выход величины элемента $z_j(t)$ за рамки условия $\mathbf{z}(t) \in Q(\mathbf{z}(t), t)$ при $t \in \Delta$ характеризует i -элементную стойкость СС. При определении характеристики $T_{ij}^{(y)}(t_0)$ – «первый момент выхода значения параметра x_j за границу стойкости при реализации угрозы в i -м элементе» может быть построена матрица T -шаговых угроз, т.е. времени отказа элемента $x_j \in X$ при реализации угрозы $\gamma \in \Gamma^{(y, \alpha)}$ в i -элементе в момент времени t_0 .

Потенциально-опасный объект – это вершина графа (целевая вершина), в которой значение параметра выходит за допустимые границы целевого режима функционирования, в результате чего она является источником угрозы для всей системы.

Источник ЧС для сложной системы моделирует импульс $\delta \mathbf{Im}(\tau)$, реализованный в момент времени τ .

Построение графовой модели СС состоит из ряда процедур:

I. Анализ ситуации (погружение в проблему, идентификация проблемы).

I.1. Формулировка задачи и цели исследования.

I.2. Изучение текущей ситуации или процесса с позиции поставленной цели.

I.3. Сбор, систематизация, анализ существующей статистической и качественной информации по проблемам; источники информации: статистические отчеты, документы, эксперты, средства массовой информации, собственные источники и др.

I.4. Выделение основных характеристических признаков изучаемого процесса (исследуемой ситуации) и выявление взаимосвязей между ними; определение действий основных объективных законов развития исследуемой ситуации, что позволит выделить объективные зависимости и тенденции в процессах, происходящих в ситуациях.

I.5. Определение присущих исследуемой ситуации требований, условий, ограничений.

I.6. Выделение основных субъектов, связанных с ситуацией, определения их субъективных интересов в развитии данной ситуации, что позволит определить возможные изменения в объективном развитии ситуации, выделить факторы, на которые реально могут влиять субъекты ситуации.

I.7. Определение путей, механизмов действия, реализации интересов основных субъектов, что

позволит в дальнейшем определить стратегии поведения и предотвращения нежелательных последствий развития ситуации.

II. Построение графовой модели проблемной ситуации в СС.

II.1 Выделение факторов, по мнению экспертов характеризующих проблемную ситуацию.

II.1.1. Выделение базисных факторов, описывающих суть проблемы.

II.1.2. Выделение в совокупности базисных факторов целевых факторов.

II.1.3. Выделение в совокупности базисных факторов факторов-угроз, которые в модели являются потенциально возможными рычагами воздействия на ситуацию.

II.1.4. Определение факторов-индикаторов, отражающих развитие процессов в проблемной ситуации и их влияние на систему.

II.2. Группировка факторов по блокам. Объединяются в один блок факторы, характеризующие данную сферу проблемы и определяющие процессы в этой сфере. Возможны варианты в зависимости от специфики проблемы, целей анализа, количества субъектов ситуации и т.д. (например, геополитический, макро- и микроэкономический блок, социальные, демографические, отраслевые, федеральные и региональные блоки).

II.2.1. Выделение в блоке группы интегральных показателей (факторов), по изменению которых можно судить об общих тенденциях в данной сфере.

II.2.2. Выделение в блоке показателей (факторов), характеризующих тенденции и процессы в данной стране.

II.3. Определение связей между факторами.

II.3.1. Определение связей и взаимосвязей между блоками факторов, что позволит определить основные направления влияния факторов разных блоков друг на друга.

II.3.2. Определение непосредственных связей факторов внутри блока:

II.3.2.1. Определение направления влияний и взаимовлияний между факторами, т.е. выявление цепочки: «причина-следствие».

II.3.2.2. Определение позитивности, влияния (положительное «+», отрицательное «-»), и степени влияния («сильно», «слабо»).

II.3.3. Определение связей между факторами различных блоков.

II.4. Построение графа ситуации.

II.5. Уточнение целевых факторов, выделенных экспертами.

II.6. Составление уравнений модели ситуации.

II.7. Проверка адекватности модели, т.е. сопоставление полученных результатов с характеристиками системы, которые при тех же исходных условиях были в прошлом; если

результаты сравнения неудовлетворительные – возвращаются к пункту 1 и т.д.

III. Моделирование – генерация сценариев развития СС.

Сценарий может моделироваться по трем основным направлениям:

- прогноз развития ситуации без воздействия на процессы: синергия развития ситуации;
- прогноз развития ситуации с выбранным комплексом мероприятий-управлений (прямая задача);
- синтез комплекса мероприятий для достижения необходимого изменения состояния ситуации (обратная задача).

Исследования, связанные с процессом моделирования также проводятся поэтапно.

III.1. Определение начальных условий, тенденций, характеризующих развитие ситуаций на данном этапе и необходимых для обеспечения адекватности модельного сценария реальной ситуации.

III.2. Задание целевых желаемых направлений (увеличение, уменьшение) и силы (слабо, сильно) изменения тенденций процессов в ситуации.

III.3. Выбор комплекса мероприятий (совокупности управляющих факторов), определение их возможной и желаемой силы и направленности воздействий на ситуацию.

III.4. Выбор наблюдаемых факторов (индикаторов), характеризующих развитие и соответствие ситуации желаемому результату; осуществляется в зависимости от целей анализа и желания пользователя [2].

2. Вычислительный эксперимент сценарного анализа уязвимости городской инфраструктуры

Рассмотрим применение предлагаемой методики сценарного анализа уязвимости для городской инфраструктуры.

Целью сценарного анализа является изучение функционирования СС под воздействием возмущений (угроз) путем построения сценариев поведения системы и последующего их анализа по характеристике «Т-шаговая угроза».

Модель городской инфраструктуры имеет следующую структуру. Вершинами являются взаимосвязанные объекты: 1. Населённый пункт. 2. Лесной массив. 3. Промышленное предприятие. 4. Экология. 5. Транспортный узел. 6. Склад. 7. Энергосеть. 8. Водохранилище.

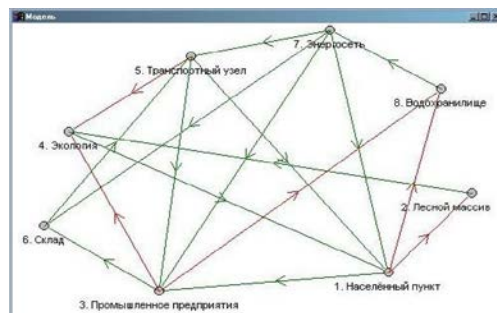


Рис 1. Граф городской инфраструктуры.

По графу строим матрицу смежности.

Таблица 1. Матрица смежности.

Вер.	1	2	3	4	5	6	7	8
1	0	-1	1	0	0	0	0	-1
2	0	0	0	1	0	0	0	0
3	0	0	0	-1	0	1	0	-1
4	1	0	0	0	0	0	0	0
5	1	0	1	-1	0	0	0	0
6	0	0	0	0	1	0	0	0
7	1	0	1	0	1	1	0	0
8	0	0	0	0	0	0	1	0

Как показано в [1], результат возмущения $F(\delta \text{Im}(t,k), t)$ процесса реализации угроз $\delta \text{Im}(t,k)$ момент времени t может быть рассчитан в соответствии с теоремой 1. Однако, расчёт времени выхода системы за границы допустимости при возмущении представляется проблематичным. В этих условиях следует провести вычислительный эксперимент. Для реализации сценарного анализа уязвимости системы проведём вычислительный эксперимент, который состоит в следующем.

Рассмотрим 3 типа реализации угроз.

Сосредоточенные угрозы:

- исследуем сценарии поведения сложной системы при реализации единичной угрозы в каждой вершине графа. По восьми вершинам графа построены восемь сценариев C1–1... C8–1. Вычисляем в Matlab матрицу T -шаговых угроз MT-1 (см. табл. 3);
- исследуем сценарии поведения сложной системы при реализации удвоенной угрозы в каждой вершине графа. По восьми вершинам графа построены восемь сценариев C1–2... C8–2. Вычисляем в Matlab матрицу T -шаговых угроз MT-2 (см. табл. 4).

Распределённые угрозы в восьми замкнутых контурах графа:

- единичная угроза реализуется последовательно в каждой вершине контура. По восьми контурам графа построены восемь сценариев C1K1... C8K1. Вычисляем в Matlab матрицу T -шаговых угроз MT-3 (табл. 5).
- удвоенная угроза реализуется последовательно в каждой вершине контура. По восьми контурам графа построены восемь сценариев C1K2... C8K2. Вычисляем в Matlab матрицу T -шаговых угроз MT-4 (табл. 6).

Периодические угрозы в контурах графа:

– в вершинах контура последовательно реализованные единичные угрозы являются периодическими, т.е. к исходным угрозам в каждой вершине контура на следующем шаге сценария последовательно добавляются аналогичные угрозы. По восьми контурам графа построены восемь сценариев С1К1п... С8К1п. Вычисляем в Matlab матрицу T -шаговых угроз МТ-5 (табл. 7).

– в вершинах контура последовательно реализованные удвоенные угрозы являются периодическими. По восьми контурам графа построены восемь сценариев С1К2п... С8К2п. Вычисляем в Matlab матрицу T -шаговых угроз МТ-6 (табл. 8).

Цель эксперимента – анализ матриц T -шаговых угроз, для каждой вершины определены шаги сценария выхода за ЦРФ и выявлены уязвимые места системы.

Алгоритмическое обеспечение исследования составляют следующие алгоритмы.

Алгоритмы построения характеристик и матрицы T -шаговых угроз.

1. Алгоритм построения матрицы характеристик сценариев.

Этап 1. Состоит из ввода данных пользователем:
– матрицы смежности A , построенной по заданному графу;
– начального состояния системы, которое представляет собой вектор $z(0)$;
– вектора импульса возмущений $\delta \text{Im}(\tau)$, который состоит из вектора столбца, соответствующего угрозам, реализованным в вершинах в момент времени τ ;
– заданного горизонта сценария;
– целевого вектора a , выбранного пользователем, как желательный вектор состояния эффективного функционирования системы;
– заданной границы допустимости δ .

Этап 2. Вычисление сценария возмущения.

Этап 3. Вычисление сценария поведения системы.

Этап 4. Формирование базы данных сценариев (БДС). База данных является трёхмерным массивом сценариев, различающихся импульсами возмущения.

Этап 5. Построение матрицы характеристик. Характеристики отражают отклонение сценариев от ЦРФ в относительной и абсолютной шкале.

Этап 6. Формирование базы данных, которая представляет собой трёхмерный массив, состоящий из матриц характеристик сценариев в зависимости от импульсов возмущений.

Этап 7. Разработка цикла в Matlab для построения сравнительных характеристик: нахождение максимального и минимального отклонений в абсолютной и относительной шкале, нахождение максимального и минимального отклонений от целево-

го направления для экспертно-значимых элементов и для всей системы в целом.

Этап 8. Визуализация выводных данных в виде таблиц или диаграмм.

2. Алгоритм построения матрицы T -шаговых угроз.

Учитываются только те элементы, значения которых выходят за допустимые пределы на заданном горизонте сценария.

Этап 1. На данном этапе осуществляется выбор матрицы из БД матриц характеристик сценариев, которая соответствует определённой угрозе.

Этап 2. Разработка цикла в Matlab для поиска первого выхода за границы ЦРФ (конкретное значение каждого экспертно-значимого элемента сравнивается с предельно допустимым значением для ЦРФ).

Этап 3. Формирование столбцов матрицы T -шаговых угроз, каждый из которых соответствует первому выходу значений в вершинах графа за предельно допустимые значения при реализации угрозы. Формирование всех полученных столбцов в общую матрицу T -шаговых угроз.

Этап 4. Вывод данных – матрица T -шаговых угроз в виде таблицы.

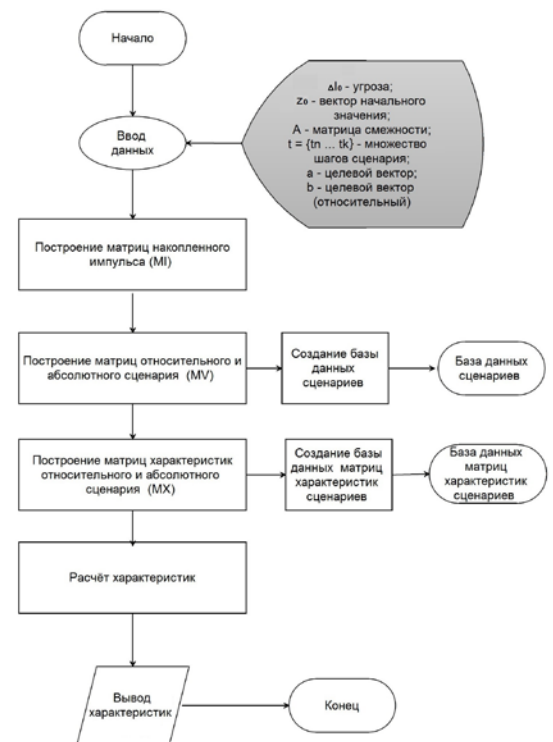
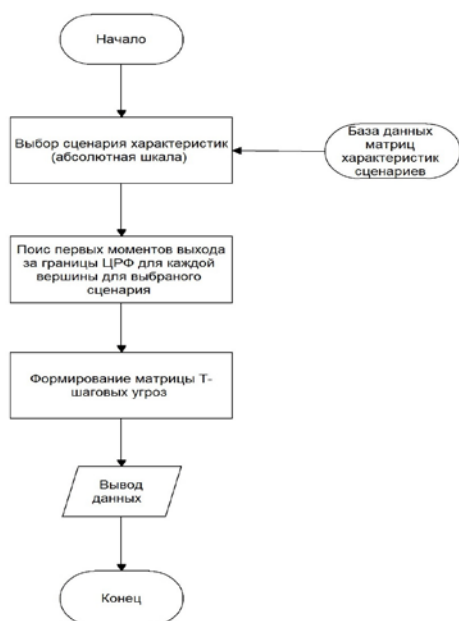


Рис 2. Формирование характеристик и БД

Рис 3. Алгоритм построения матрицы T -шаговых угроз

```

Command Window
>> MX =cat(8,MX1,MX2,MX3,MX4,MX5,MX6,MX7,MX8)
for k=1:8
MX0=MX(:, :, k);
for t=1:8
x=MX0(t, :)
ab=abs(x);
f=find(ab>abs(Vd(t, :)))
mi=min(f)
T(t, :)=mi
end
MT(:, k)=T
end

```

Рис 4. Вариант программы в Matlab для построения матрицы T -шаговых угроз

3. Результаты эксперимента

1. Исходные данные:

В соответствии с алгоритмом для расчёта характеристик предлагаются следующие параметры: целевой вектор желательного состояния системы и допустимые отклонения от целевого вектора.

Таблица 2. Целевой вектор и допустимые отклонения

№ вер.	Целевой вектор $\mathbf{a}$	Допустимое отклонение δ (15%)	Границы целевого вектора	Единица измерения
1	26000	3900	29900	чел.
2	200	30	230	Га
3	500000	75000	575000	Руб.
4	24,2	3,6	27,8	мг/м ³
5	2000	300	2300	т/день
6	10000	1500	11500	м ²
7	28000	4200	32200	кВт/ч
8	7500	1125	8625	м ³

2. Исследуем влияние на граф городская инфраструктура сосредоточенных угроз.

Для построения матрицы T -шаговых угроз начальный импульс $\mathbf{Im}(0)$, представляющий угрозу, реализуется последовательно в каждой вершине. Сценарий C1-1 соответствует угрозе $Im_1(0)=1$, реализованной в вершине «Населённый пункт».

Аналогично построены сценарии C2-1...C8-1. Результат – матрица MT-1 T -шаговых угроз.

Таблица 3. Матрица T -шаговых угроз (MT-1)

Вер.	C1	C2	C3	C4	C5	C6	C7	C8
1	-1	-1	-1	-1	-1	-1	-1	-1
1	19	22	19	23	25	20	23	21
2	13	17	14	12	14	14	13	11
3	30	29	31	29	32	32	30	28
4	7	9	6	10	7	7	11	9
5	18	16	19	17	19	19	17	15
6	19	23	20	18	21	21	19	17
7	19	23	20	18	21	20	19	17
8	20	18	21	18	21	21	19	17

Начальная угроза, реализованная в каждой из вершин, изменяет состояние этого элемента на нескольких шагах сценария.

По матрице T -шаговых угроз определяем, что реализация угрозы «Промышленное предприятие» наносит максимальный ущерб «Экологии» на шестом шаге сценария C3-1 (Сценарий от реализации единичной угрозы в третьей вершине).

Для построения матрицы T -шаговых угроз начальный импульс $\mathbf{Im}(0)$, представляющий угрозу, реализуется последовательно в каждой вершине. Сценарий C1-2 соответствует угрозе $Im_1(0)=2$, реализованной в вершине «Населённый пункт».

Аналогично построены сценарии C2-2...C8-2. Результатом является матрица MT-2 T -шаговых угроз.

Таблица 4. Матрица T -шаговых угроз (MT-2)

Вер.	C1	C2	C3	C4	C5	C6	C7	C8
1	-2	-2	-2	-2	-2	-2	-2	-2
1	18	22	19	17	20	19	23	22
2	13	11	14	11	14	14	12	11
3	25	28	25	29	32	26	30	29
4	5	9	6	4	6	6	5	4
5	18	16	13	16	19	14	17	16
6	19	23	20	18	20	20	19	18
7	19	23	20	18	20	20	18	17
8	20	18	20	18	21	21	19	18

По матрице T -шаговых угроз определяем, что угроза в «Экологию» и в «Водохранилище» наносит максимальный ущерб «Экологии» на четвёртом шаге сценария C4-2 и сценария C8-2.

3. Исследуем влияние на функционирование городской инфраструктуры распределённых угроз по контуру, рассредоточенных по времени.

Для построения матрицы T -шаговых угроз (MT-3) системный элемент модели, представляет

собой замкнутый контур попарных связей. Построим матрицу T -шаговых угроз при реализации угрозы в каждом элементе контура.

В графе «Городская инфраструктура» выбраны восемь контуров:

- 1К (вершины «Населённый пункт» – «Лесной массив» – «Экология»);
- 2К (вершины «Населённый пункт» – «Промышленное предприятие» – «Экология»);
- 3К (вершины «Промышленное предприятие» – «Склад» – «Транспортный узел» – «Населённый пункт»);
- 4К (вершины «Промышленное предприятие» – «Водохранилище» – «Энергосеть»);
- 5К (вершины «Населённый пункт» – «Промышленное предприятие» – «Водохранилище» – «Энергосеть» – «Транспортный узел» – «Экология»);
- 6К (вершины «Населённый пункт» – «Водохранилище» – «Энергосеть» – «Транспортный узел»);
- 7К (вершины «Промышленное предприятие» – «Склад» – «Транспортный узел»);
- 8К (вершины «Населённый пункт» – «Водохранилище» – «Энергосеть»).

3.1. Построим сценарии, когда угрозы в вершинах контура реализуются последовательно.

Для контура 1К строится сценарий С1К1 при реализации единичной угрозы в вершине «Населённый пункт» на первом шаге ($t_1=1$), в вершине «Лесной массив» на втором шаге ($t_2=2$), в вершине «Экология» на третьем шаге ($t_3=3$). Аналогично построены сценарии С2К1...С8К1. Результатом является матрица МТ-3 T -шаговых угроз.

Таблица 5. Матрица T -шаговых угроз (МТ-3)

В. р.	С1 К1	С2 К1	С3 К1	С4 К1	С5 К1	С6 К1	С7 К1	С8 К1
1	19	20	21	20	20	19	20	19
2	14	14	15	14	14	14	15	14
3	31	27	27	32	27	31	27	31
4	7	7	8	7	7	6	7	6
5	19	19	15	19	19	18	15	19
6	20	21	22	21	21	20	21	20
7	20	21	21	20	21	20	21	20
8	21	21	22	21	21	20	22	20

Построим сценарии, когда в вершинах выбранного контура реализуются удвоенные угрозы.

Для контура 1К строится сценарий С1К2 при реализации удвоенной угрозы в вершине «Населённый пункт» на первом шаге ($t_1=1$), в вершине «Лесной массив» на втором шаге ($t_2=2$), в вершине «Экология» на третьем шаге ($t_3=3$). Аналогично построены сценарии С2К2...С8К2.

По полученным расчётам строится матрица T -шаговых угроз МТ-4.

Таблица 6. Матрица T -шаговых угроз (МТ-4)

В. р.	С1 К2	С2 К2	С3 К2	С4 К2	С5 К2	С6 К2	С7 К2	С8 К2
1	19	19	20	19	19	19	20	19
2	14	14	10	14	14	13	15	13
3	26	26	27	26	26	25	27	25
4	6	7	7	6	6	6	7	6
5	13	14	14	14	14	13	14	13
6	20	20	21	20	20	20	21	20
7	20	20	21	20	20	20	21	20
8	20	16	16	16	16	20	16	20

3.2. Построим сценарии, когда угрозы в вершинах контура реализуются периодически.

Для контура 1К строится сценарий С1К1п при реализации периодической единичной угрозы в вершине «Населённый пункт» на первом шаге ($t_1=1$), в вершине «Лесной массив» на втором шаге ($t_2=2$), в вершине «Экология» на третьем шаге ($t_3=3$), в вершине «Населённый пункт» на четвёртом шаге ($t_4=4$), в вершине «Лесной массив» на пятом шаге ($t_5=5$), в вершине «Экология» на шестом шаге ($t_6=6$). Аналогично построены сценарии С2К1п...С8К1п.

По полученным расчётам строится матрица T -шаговых угроз МТ-5.

Таблица 7. Матрица T -шаговых угроз (МТ-5)

В. р.	С1 К1 п	С2 К1 п	С3 К1 п	С4 К1 п	С5 К1 п	С6 К1 п	С7 К1 п	С8 К1 п
1	20	20	21	20	20	19	21	20
2	14	15	15	14	14	13	15	14
3	31	27	27	32	27	31	28	31
4	13	8	8	7	7	6	8	7
5	19	19	15	19	19	18	20	19
6	20	21	22	21	21	20	22	20
7	20	21	21	21	21	20	21	20
8	21	21	22	21	21	20	22	21

Для контура 1К строится сценарий С1К1п при реализации периодической удвоенной угрозы в вершине «Населённый пункт» на первом шаге ($t_1=1$), в вершине «Лесной массив» на втором шаге ($t_2=2$), в вершине «Экология» на третьем шаге ($t_3=3$), в вершине «Населённый пункт» на четвёртом шаге ($t_4=4$), в вершине «Лесной массив» на пятом шаге ($t_5=5$), в вершине «Экология» на шестом шаге ($t_6=6$). Аналогично построены сценарии С2К2п...С8К2п.

По полученным расчётам строится матрица T -шаговых угроз МТ-6.

Таблица 8. Матрица T-шаговых угроз (MT-6).

Вер	C1 К1 п	C2 К1 п	C3 К1 п	C4 К1 п	C5 К1 п	C6 К1 п	C7 К1 п	C8 К1 п
1	19	20	20	19	19	19	20	19
2	14	14	15	14	14	13	15	14
3	26	26	27	26	26	26	27	26
4	7	7	7	6	6	6	7	6
5	19	14	14	14	14	18	15	19
6	20	21	21	20	20	20	21	20
7	20	20	21	20	20	20	21	20
8	21	16	16	21	16	20	17	20

По полученным результатам предлагается классифицировать опасность наступления ЧС на пять классов уязвимости элементов сложной системы:

- максимальная уязвимость элемента: отклонение значения в вершине от ЦРФ на 3-8 шаге сценария;
- критическая уязвимость элемента: отклонение значения в вершине от ЦРФ на 9-14 шаге сценария;
- средняя уязвимость элемента: отклонение значения в вершине от ЦРФ на 15-20 шаге сценария;
- незначительная уязвимость элемента: отклонение значения в вершине от ЦРФ на 21-26 шаге сценария;
- минимальная уязвимость элемента: отклонение значения в вершине от ЦРФ на 27-32 шаге сценария.

Выводы эксперимента

Вершины можно разделить на группы:

- максимально уязвимым элементом системы является «Экология»;
- критически уязвимые элементы – «Экология», «Лесной массив», «Транспортный узел»;
- элементы средней уязвимости – «Население», «Энергосеть», «Склад», «Водохранилище», «Транспортный узел», «Лесной массив»;
- незначительно уязвимые элементы – «Население», «Промышленное предприятие», «Склад», «Электросеть», «Водохранилище»;
- минимально уязвимый элемент – «Промышленное предприятие».

По классам уязвимости системы можно определить, в каких элементах системы реализация угроз может привести к критической ситуации.

В частности, в графе городская инфраструктура критическая ситуация возникает для «Экологии» на пятом шаге сценария, построенном от единичной угрозы, реализованной в третьей вершине

«Промышленное предприятие», при условии, что шестой шаг соответствует первому выходу за границы ЦРФ.

Заключение

Результаты вычислительного эксперимента:

- нет прямо пропорциональной зависимости для времени выхода за пределы допустимых значений вектора состояния системы при реализации единичной и удвоенной угрозы;
- минимально уязвимой вершиной к угрозам в графе является вершина «Промышленное предприятие»;
- максимально уязвимой вершиной к угрозам в графе является вершина «Экология»;
- при единичных угрозах, последовательно реализованных в вершинах контура, вершина «Водохранилище» – незначительно уязвима.
- при удвоенных угрозах, последовательно реализованных в вершинах контура, вершина «Водохранилище» – средне уязвима.
- при реализации единичных и удвоенных угроз в вершинах контура, шаги сценария, при которых значение вершин «Склад», «Энергосеть» отклоняются от ЦРФ, отличаются незначительно;
- система наиболее устойчива к угрозам, реализованным в первом контуре;
- система наименее устойчива к угрозам, реализованным в пятом контуре.

Scenario analysis of complex systems vulnerability: simulation experiment

R.O. Ponomarev, D.A. Kononov, I.M. Khasanov

Abstract. The technique of the scenario analysis of complex systems' vulnerability is proposed. The example of researching of «urban infrastructure» model is considered. Computing algorithms for realization of the technique are developed. The results of the calculations are presented. The classification of emergency occurrence danger is conducted.

Keywords: scenario analysis, vulnerability, urban infrastructure, complex systems, simulation experiment.

Литература

1. Д.А. Кононов, А.А. Кочкаров, Н.О. Пономарев. Стойкость сложных технических систем: сценарный взгляд на проблему / Труды НИИСИ РАН, т. 2 (2012), № 2 , 80-86.
2. Модели и методы анализа и синтеза сценариев развития социально-экономических систем. Кн. 2 /Под редакцией чл.-корр. РАН В.Л.Шульца, д.т.н., проф. В.В.Кульбы – М., Наука, 2012. (коллективная монография). 358 с.

Визуализация результатов моделирования неустойчивого вытеснения нефти из пористых сред

Н.Н.Смирнов¹, В.Ф.Никитин², М.В. Михайлюк³, П.Ю.Тимохин⁴, В.В.Тюренкова⁵,
Л.И.Стамов⁶

^{1,5} ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН» -
ФГБОУ ВО «Московский государственный университет имени М.В.Ломоносова»,
Механико-математический факультет, Москва, Россия,
E-mail's: ¹mech.math.msu@inbox.ru, ⁵turenkova.v.v@rambler.ru,

² ФГБОУ ВО «Московский государственный университет имени М.В.Ломоносова»,
Механико-математический факультет, Москва, Россия,
E-mail: vfnikster@gmail.com,

^{3,4,6} ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»,
E-mail's: ³mix@niisi.ras.ru, ⁴webpismo@yahoo.de, ⁶lyubens@mail.ru

Аннотация: В работе предлагается интерфейс для моделирования и визуального анализа результатов моделирования неустойчивого вытеснения нефти из пористых сред. Для визуализации поверхности уровня насыщенности вытесняющей жидкости предлагается использовать метод рейкастинга.

Ключевые слова: пористая среда, вытеснение нефти, моделирование, визуализация.

Введение

Одной из важнейших народно-хозяйственных задач является увеличение отдачи нефтяных месторождений. Для этого используются многие методы, среди которых достойное место занимает метод вытеснения нефти путем закачивания в месторождение вытесняющей жидкости. Моделирование этого процесса является важной и актуальной фундаментальной задачей, представляющей как научный, так и практический интерес.

Среда, с которой приходится иметь дело, является пористой и существует ряд исследований, посвященных моделированию движения жидкостей в таких средах [1, 2, 13 и др.].

Предлагаемые некоторыми авторами [4] многофазные модели используют большое число коэффициентов, что может быть избыточно для процесса вытеснения с неустойчивым фронтом.

Односкоростные модели с поверхностным натяжением [5] являются экономичными с практической точки зрения, однако они не могут описать неустойчивость фронта вытеснения.

Для выполнения экспертной оценки и подстройки модели процесса вытеснения необходима визуализация результатов моделирования в наглядной, интуитивно понятной форме.

Классическим подходом является расчет на CPU координат вершин сетки и полигональной модели поверхности (набора треугольников, образующих замкнутую внешнюю оболочку расчетной сетки) [6].

В исследовании [7] представлена классификация стратегий параллельной визуализации на CPU, основанная на порядке выполнения стадии сортировки по видимости.

Бурный рост производительности графических процессоров (GPU) и активное внедрение в их архитектуру программируемых (шейдерных) стадий открыло путь разработчикам к созданию новых эффективных систем визуализации, работающих на отдельных рабочих станциях.

В данной статье рассматривается технология визуализации, разрабатываемая в рамках инновационного инициативного проекта по моделированию неустойчивого вытеснения нефти из пористых сред.

Целью проекта является создание цифровой модели неустойчивого вытеснения с учетом физико-химических превращений, снабженной визуализатором результатов моделирования.

Проект не ставит своей задачей повторение всех опций, заложенных в существующих коммерческих симуляторах, такие как ECLIPSE, STARS, tNavigator и др.

Проект предполагает создать ориентированный на решение конкретной задачи продукт с ограниченным количеством опций, но превосходящий существующие моделирующие комплексы по качеству используемых моделей, глубине разрешения протекающих процессов и качеству представления результатов.

В качестве основы для разрабатываемой в проекте системы визуализации результатов моделирования в настоящей работе предлагается метод визуализации поверхности уровня насыщенности вытесняющей жидкости, основанный на рейкастинге.

1. Основные решаемые задачи

В настоящей работе используется модель вытеснения нефти из пористых сред, предложенная в работах [8-13]. Моделирование в общем случае проводится внутри прямоугольного параллелепипеда, причем число ячеек моделирования (как и размер ячейки) по каждому измерению может быть различным. Начало системы координат располагается в центре (или в углу) одной из граней параллелепипеда (начальной грани), оси Y и Z расположены в плоскости этой грани, а ось X направлена вдоль третьей стороны (см. рис. 1). Предполагается, что внутри этого параллелепипеда находится пористая среда, заполненная жидкостью, которую необходимо вытеснить. Это осуществляется с помощью вытесняющей жидкости, которая закачивается под давлением в плоскость начальной грани (т.е. через сечение $x=0$). Модель характеризуется рядом параметров, таких, как уровень капиллярного давления, отношение вязкостей вытесняемой и вытесняющей жидкостей, амплитуда пертурбации и др. Результатами моделирования являются массивы значений величин насыщенности вытесняющей жидкости в ячейках для шагов моделирования $i = 0, \dots, k$. Насыщенность вытесняющей жидкости в каждой ячейке C определяется соотношением

$$s = \frac{V_{1,C}}{V_{1,C} + V_{2,C}}, \text{ где } V_{1,C} - \text{объем вытесняющей}$$

жидкости в ячейке, $V_{2,C}$ - объем вытесняемой жидкости в этой же ячейке. Множество значений $s(x, y, z, t)$ насыщенности вытесняющей жидкости для фиксированного момента времени t можно представить в виде 3D текстуры. Для анализа этого показателя удобно визуализировать «поверхность уровня насыщенности s^* », т.е. поверхность, с одной стороны которой $s \geq s^*$, а с другой - $s < s^*$. Однако такая постановка требует уточнения. Дело в том, что

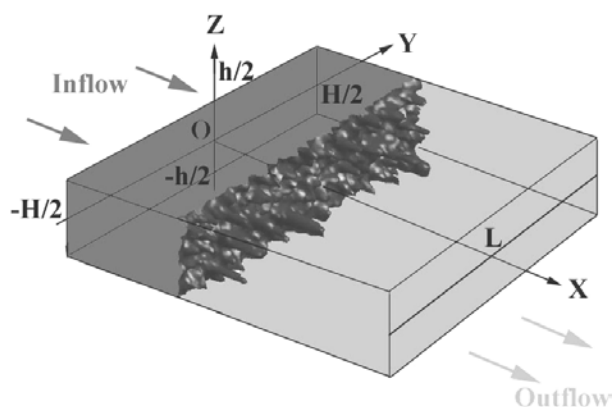


Рис. 1. Объем моделирования

насыщенность s не растет монотонно, поэтому по любому направлению значения s^* могут чередоваться как с большими, так и с меньшими значениями насыщенности. В данной работе предлагается строить «видимую поверхность уровня s^* », т.е. по каждой

линии взгляда, начиная от наблюдателя, выбирается первая расчетная ячейка, в которой $s \geq s^*$. На рис. 2 на линии взгляда ОВ выбранная точка В означает, что на отрезке АВ $s < s^*$, в точке В $s \geq s^*$, а на отрезке ВС s может быть произвольным.

Также для анализа результатов моделирования вычисляются некоторые интегральные показатели. Рассмотрим, для примера, следующие показатели.

Процентное содержание вытесняющей жидкости в объеме пор образца

$$p(t) = \frac{V_1(t)}{V_p} \cdot 100\%, \text{ где } V_1(t) - \text{объем всей}$$

имеющейся в образце вытесняющей жидкости в момент времени t , V_p - объем пор образца.

Перепад давления между входом и выходом $d(t) = P_{IN}(t) - P_{OUT}(t)$, где $P_{IN}(t)$ - давление на входе, $P_{OUT}(t)$ - давление на выходе в момент времени t .

Расход $V_{IN}(t)$ вытесняющей жидкости на входе.

Отношение объема поданной вытесняющей жидкости к объему пор

$$d(t) = \frac{V_1(t)}{V_p}, \text{ где } V_1(t) - \text{объем закачанной}$$

вытесняющей жидкости к моменту времени t , а V_p - объем пор во всем образце.

Данные показатели могут быть представлены в виде двумерных графиков, в которых абсциссой является время t , а ординатой - значения соответствующего показателя.

2. Интерфейс системы моделирования

Интерфейс системы моделирования представляет

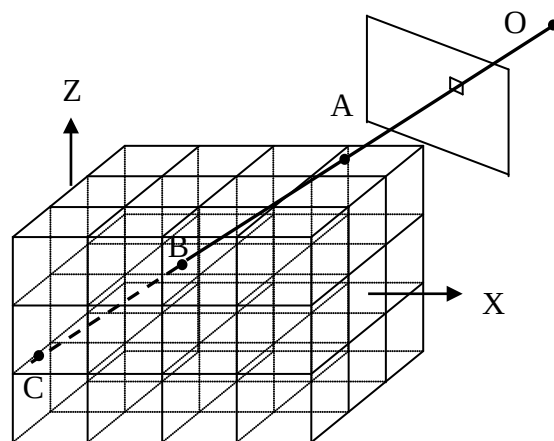


Рис. 2. Регулярная сетка сцены.

собой окно windows приложения, включающее меню, панель инструментов (toolbars), панель задания параметров моделирования, область визуализации поверхности уровня насыщенности и область вывода графиков интегральных показателей (см. рис. 3).

Меню приложения позволяет создать новый файл (проект моделирования) или загрузить сохраненный ранее файл с параметрами моделирования, а также

сохранить новый или измененный файл и вывести на печать графики и поверхность уровня насыщенности. Можно также спрятать области вывода параметров и графиков с тем, чтобы расширить центральную область, в которой визуализируется поверхность уровня. С другой стороны, окно каждого интегрального показателя можно переместить в любое место основного окна, а также изменить его размер. Наряду с графиками полученных интегральных показателей в эти окна можно вывести графики этих же показателей, полученных в некоторых заранее известных модельных экспериментах. Это удобно для сравнения полученных результатов с некоторыми эталонными. Справка в меню позволяет получить справочную информацию по модели, окну и интерфейсу приложения.

Панель инструментов частично повторяет операции меню (создание нового файла, открытие имеющегося файла и сохранение файла). Кроме этого в панели имеется ряд кнопок для проигрывания сохраненных результатов моделирования. Проигрывание возможно как в непрерывном режиме с разной скоростью, так и пошагово. В любой момент можно поставить воспроизведение на паузу, перейти к предыдущему кадру, к следующему, на начало или на конец; можно также прекратить воспроизведение.

В панели задания параметров моделирования написаны имена параметров, единицы их измерения, типы (целый или вещественный) и диапазоны возможных значений, а также значения по умолчанию. При вводе неправильного значения система выдаст сообщение об ошибке.

В центральном окне производится синтез изображения видимой поверхности уровня для заданного значения s^* . Синтез такого изображения предлагается выполнять на основе метода рейкастинга (ray casting, см. [14, 15]). Его суть заключается в следующем. Из точки О расположения наблюдателя (виртуальной камеры) через центр каждого пиксела области вывода 3D-поверхности испускается луч.

Вдоль этого луча ищется ближайшая ячейка моделирования, в которой значение $s \geq s^*$ (см. рис. 2). Пиксел закрашивается оттенком выбранного цвета, зависящим от расстояния от наблюдателя до ячейки моделирования или зависящим от нормали в этой точке поверхности.

С помощью компьютерной мыши виртуальную камеру пользователь может перемещать и поворачивать относительно осей ее системы координат.

Программная реализация предложенного интерфейса выполнялась с использованием открытой версии библиотеки QT, а визуализация результатов моделирования – с помощью графической библиотеки OpenGL.

Отдельной задачей являлось обеспечение переносимости проекта на различные платформы Windows. Так как Microsoft прекратила сопровождение Windows XP, то для обеспечения запуска проекта на этой операционной системе необходимо было скомпилировать проект с опцией «Platform Toolset», установленной в значение «Visual Studio 2013 - Windows XP (v120_xp)».

3. Заключение

В работе предлагается интерфейс для моделирования неустойчивого вытеснения нефти из пористых сред, а также метод рейкастинга для визуализации поверхности уровня насыщенности вытесняющей жидкости в таких средах.

Поверхность можно исследовать, рассматривая с разных сторон путем перемещения и поворота виртуальной камеры вокруг ее осей координат. В дальнейшем предполагается выполнять эту визуализацию в стерео режиме [16].

Работа выполнена при поддержке РФФИ, грант № 16-29-15099-офи_м.

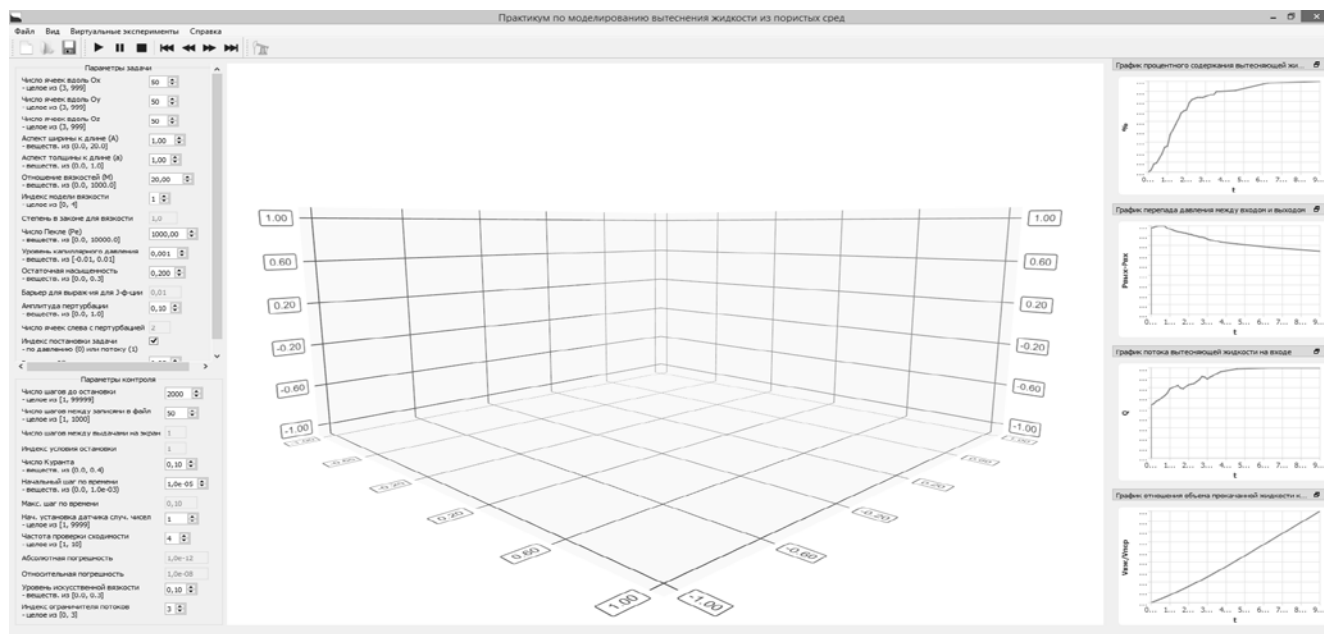


Рис.3. Окно приложения для моделирования вытеснения жидкости из пористой среды

Visualization of simulation results of unstable oil displacement from porous media

N.N.Smirnov, V.F.Nikitin, M.V. Mikhaylyuk, P.Yu.Timokhin, V.V.Tjurenkova, L.I.Stamov

Abstract. The paper proposes an interface for modeling and visual analysis of the simulation results of unstable oil displacement from porous media. To visualize the level surface of the displacing fluid saturation, it is proposed to use ray-casting method.

Keywords: porous medium, oil displacement, simulation, visualization.

Литература

1. Л.С. Лейбензон. Движение природных жидкостей и газов в пористой среде. ОГИЗ Государственное Издательство технико-теоретической литературы, М.-Л., 1947.
2. K.S.Sorbie, N.R.Zhang, N.B. Tsibuklis. Linear viscous fingering: new experimental results, direct simulation and the evaluation of averaged models. Chemical Engineering Science. Vol. 50, No. 4, pp. 601-616, 1995.
3. N.N.Smirnov, V.F.Nikitin, O.E.Ivashnyov, A.Maximenko, M.Thiercelin, A.Vedernikov, B.Scheid, J.C.Legros. Microgravity investigations of instability and mixing flux in frontal displacement of fluids. Microgravity sci. technol. 2004, XV/3, pp. 3-28.
4. М.М.Хасанов, Г.Т. Булгакова. Нелинейные и неравновесные эффекты в реологически сложных средах. Издательство Института Компьютерных Исследований, Москва - Ижевск, 2003.
5. E.Meiburg, G.M.Homsy. Nonlinear unstable viscous fingers in Hele-Shaw flows. II. Numerical simulation. Phys. Fluids 1988, 31(3).
6. F.Abraham, W.Celes. Distributed Visualization of Complex Black Oil Reservoir Models // Eurographics Symposium on Parallel Graphics and Visualization (2009), EGPGV 2009, Munich, Germany P. 87-94.
7. S.Molnar, M.Cox, D.Ellsworth, H.Fuchs. A Sorting Classification of Parallel Rendering. // IEEE Computer Graphics & Applications 14, 4 (1994), 23-32.
8. N.N.Smirnov, V.F.Nikitin, A.Maximenko, M.Thiercelin, J.C.Legros. Instability and mixing flux in frontal displacement of viscous fluids from porous media. Physics of Fluids, 2005, vol.17, 084102.
9. В.Б.Бетелин, В.Ф.Никитин, Н.Н.Смирнов, Е.В.Михальченко, Е.И.Скрылева, Л.И.Стамов, В.В.Тюренкова. Компьютерный керносимулятор — подходы и методы. Вестник кибернетики. № 4(20), 2015, стр. 33-44.
10. N.N.Smirnov, V.F.Nikitin, O.E.Ivashnyov, A.Maximenko, M.Thiercelin, A.Vedernikov, B.Scheid, J.C.Legros. Microgravity investigations of instability and mixing flux in frontal displacement of fluids (2004) Microgravity Science and Technology, 15 (2), pp. 35-51.
11. V.R.Dushin, V.F.Nikitin, J.C.Legros, M.V.Silnikov. Mathematical modeling of flows in porous medium. WSEAS Transactions on Fluid Mechanics. (2014) Vol. 9, 116-130.
12. N.Smirnov, V.Dushin, V.Nikitin, O.Logvinov, E.Skryleva. Investigations of porous media permeability and fluid displacement instability. ELGRA News – Bulletin of the European Low Gravity Research Association, vol. 29, Sept. 2015, p. 149.
13. О.А.Логвинов, Е.И.Скрылева. Вытеснение вязкой жидкости из кольцеобразной ячейки Хеле-Шоу со стоком. Вестник Московского университета. Серия 1: Математика. Механика. 2016, № 4, с. 39-43.
14. А.В.Мальцев, М.В.Михайлюк. Регулярные сетки для высоко реалистичной визуализации 3D сцен в реальном времени // «Информационные технологии и вычислительные системы», 2012, №4, с. 49-58
15. М.В.Михайлюк, М.А.Торгашев. Моделирование и визуализация объемных динамических эффектов для имитационно-тренажерных комплексов. Информационные технологии и вычислительные системы, № 4, 2014, стр. 44-50.
16. М.А.Торгашев. Реализация стереорежима в реальном масштабе времени для различных устройств отображения. // Программные продукты и системы, № 2, 2010, с. 23-29.

Моделирование поверхности виртуального полигона

М.В. Михайлюк¹, П.Ю. Тимохин²

ФГУ «ФНИЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail's : ¹ mix@niisi.ras.ru, ² webpismo@yahoo.de

Аннотация: Рассматривается технология моделирования поверхности (земного рельефа) виртуального полигона для колесных и гусеничных робототехнических средств, включающая создание геометрической модели рельефа и его текстурирование путем смешивания нескольких текстур.

Ключевые слова: виртуальный полигон, робототехнические средства, сетка высот, мульти-текстурирование.

Введение

Для выполнения различных работ в сложной окружающей обстановке в настоящее время широко используются робототехнические средства. Роботы применяются в опасной для человека среде (космическое пространство, участки местности с повышенной радиацией, очаги пожаров и т.д.). Управляют такими роботами специально обученные операторы. Обучение операторов управлению сложными и дорогостоящими роботами на реальных устройствах является рискованной и затратной задачей. Альтернативой является обучение на виртуальных трехмерных моделях робототехнических устройств и окружающей обстановки [1, 2]. В этой связи возникает идея создания «универсального» полигона – «роботодрома», в котором был бы предусмотрен широкий круг объектов, возможных событий и сценариев, который позволял бы смоделировать различные проблемные ситуации для отработки операторами методов их разрешения с помощью робототехнического средства. Например, агентство передовых оборонных исследовательских проектов DARPA создало виртуальных роботов и виртуальный полигон — целый мир, в котором эти роботы могут действовать. Весь этот комплекс является чем-то вроде компьютерной игры, в которой смоделированы основные физические параметры робота и разных типов местности. В данной работе рассматривается технология моделирования поверхности (земного рельефа) таких виртуальных полигонов.

1. Создание рельефа полигона

Рельеф полигона создается дизайнером в системе моделирования, например 3DS MAX. В общем случае результатом является полигональная модель, т.е. поверхность, представляющая собой множество связанных треугольников. Эти треугольники могут быть разной величины и формы (см. рис. 1). Для задачи визуализации такая модель вполне подходит, но для системы динамики (для расчета движения робота по поверхности) более эффективной представляется модель, основанная на равномерной сетке высот (см. рис. 2). Рассмотрим алгоритм построения сетки высот (и, со-

ответственно, ее триангуляции (разбиения на треугольники) из полигональной модели. Пусть в аффинном пространстве выбрана некоторая система координат, в которой задана полигональная модель, состоящая из множества треугольников T_i , а каждый треугольник определен тремя вершинами, хранимыми в списке. Прямоугольная сетка высот задана в плоскости XY, имеет k на m ячеек, каждая из которых размера ω на ω . Идея перевода полигональной модели в сетку высот состоит в том, что для каждого треугольника модели определяются попавшие на его поверхность узлы сетки и для них вычисляются высоты на основе интерполяции высот в вершинах треугольника. Возь-

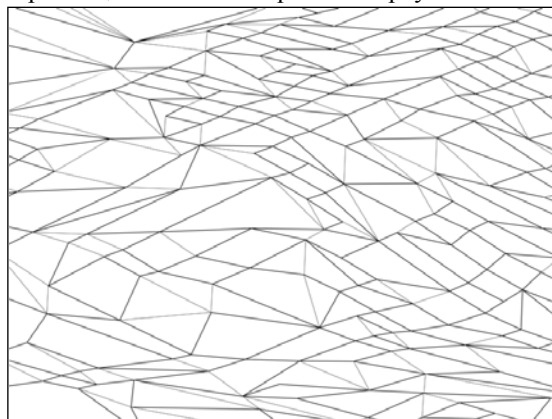


Рис. 1. Полигональная модель рельефа

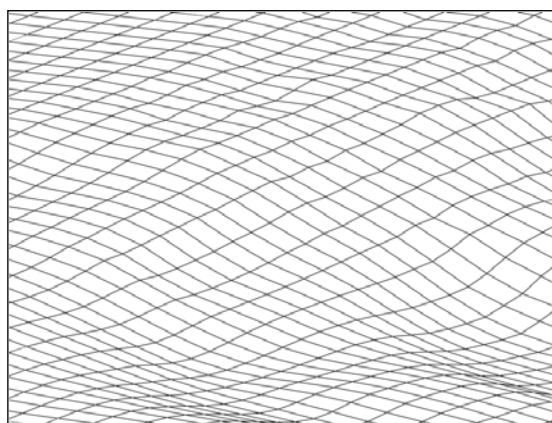


Рис. 2. Модель рельефа, заданная сеткой высот

мом некоторый треугольник T и рассмотрим для него отдельные шаги алгоритма более подробно.

1. Определение вершин треугольника T с минимальным и максимальным значением координаты y . Это легко сделать с помощью нескольких сравнений, а изменяя ссылки на вершины треугольника, добиться того, что это будут первые по списку верши-

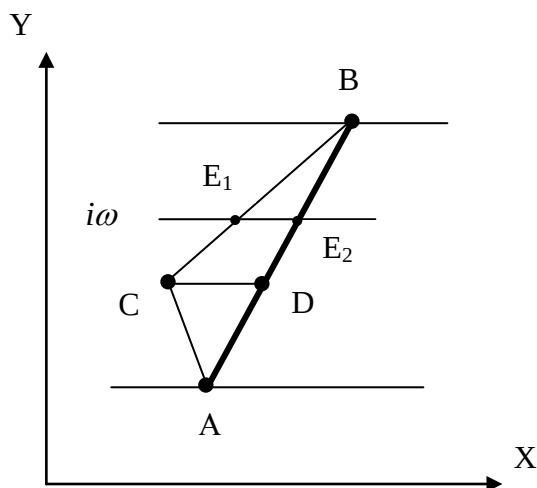


Рис. 3. Треугольник полигональной модели

ны. Обозначим их через A и B , а третью вершину – через C (см. рис. 3.). Вершина C может находиться как слева так и справа от отрезка AB .

2. Разбиение на два треугольника. Проведя через C горизонтальный отрезок до пересечения со стороной AB , разобьем T на два треугольника – CBD и ACD . Рассмотрим первый из них. Независимо от позиции C перенумерацией вершин этого треугольника можно добиться того, что C будет находиться левее отрезка BD , как изображено на рис. 3.

3. Выбор горизонтальных линий сетки, пересекающих треугольник CBD . Эти линии будут иметь y -координаты $i\omega$ для i , принадлежащих отрезку $\left[\lceil C_y / \omega \rceil, \lfloor B_y / \omega \rfloor \right]$ (здесь $\lceil \cdot \rceil$ и $\lfloor \cdot \rfloor$ означают целую часть сверху и снизу соответственно).

4. Вычисление высот в точках E_1 и E_2 пересечения линии $y = i\omega$ со сторонами CB и DB . Отрезок CB задается параметрически в виде $C + \lambda(B - C)$, где $\lambda \in [0, 1]$. Точке E_1 соответствует $\lambda = (i\omega - C_y) / (B_y - C_y)$. Обозначив через $h(C)$ и $h(B)$ высоты в точках C и B , получим

$$h(E_1) = h(C) + \lambda(h(B) - h(C)).$$

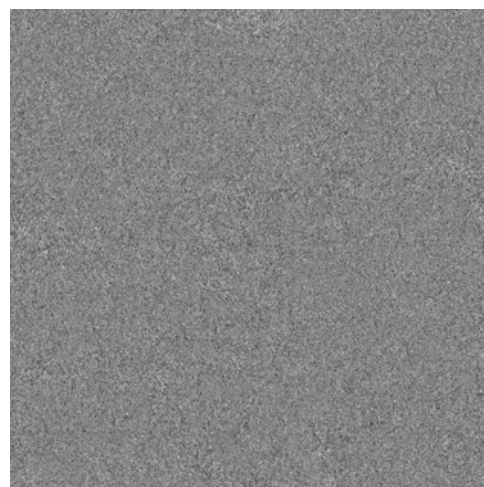
Аналогично вычисляется высота в точке E_2 .

5. Определение точек сетки на линии $y = i\omega$, лежащих внутри треугольника CDB . Координаты x этих точек будут равны $j\omega$ для j из отрезка $\left[\lceil E_{1,x} / \omega \rceil, \lfloor E_{2,x} / \omega \rfloor \right]$.

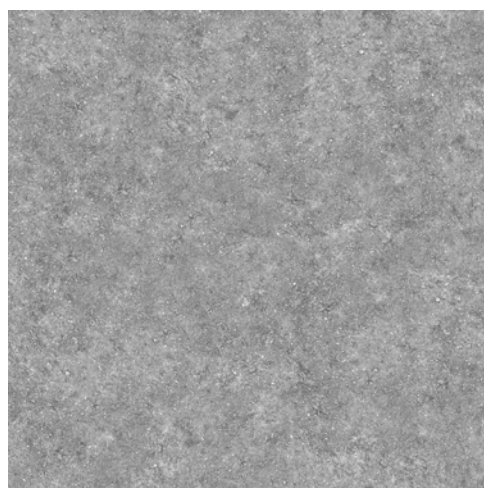
6. Вычисление высот в точках сетки на линии $y = i\omega$, лежащих внутри треугольника CDB . Аналогично п. 4 отрезок E_1E_2 можно записать в параметрическом виде как $E_1 + \lambda_i(E_2 - E_1)$. Тогда высота в точке $(i\omega, j\omega)$ вычисляется по формуле

$$h(E_1) + \lambda_i(h(E_2) - h(E_1)).$$

Таким образом, перебрав все треугольники полигональной модели рельефа, строится равномерная сетка высот этого рельефа. Разбивая каждый квадрат на



а) карта травяного покрова



б) карта участков песка



в) маска

Рис. 4. Текстуры карты

два треугольника, получаем равномерную триангуляцию. Для получения высоко реалистичной модели рельефа необходимо дополнить ее текстурами.

2. Текстурирование полигона

Полигон представляет собой участок местности размером 250 на 250 метров. Его поверхность в основном состоит из земляного покрова, на котором растет трава. Покрыть такую большую площадь одной высоко реалистичной текстурой не представляется возможным, так как графическая карта имеет ограничения на объем загружаемой текстурной карты. Наложение небольшого кусочка текстуры с повторением его в обоих направлениях порождает изображение с ярко выраженным трафаретом, которое не является реалистичным. Здесь предлагается использовать технологию смешивания (Blend) нескольких текстур небольшого размера с маской большого размера.

На рис. 4 показаны используемые текстуры и маска.

Текстуры травяного покрова и песка имеют размер 128 x 128 пикселей, а маска – размер 4096 x 4096. Эти текстуры накладываются на поверхность полигональной модели с различным числом повторений. То есть текстурные координаты для первых двух текстур намного больше, чем текстурные координаты маски. При наложении текстур производится смешивание по формуле

$$C_f = \alpha_m C_g + (1 - \alpha_m) C_s,$$

где C_f - результирующий цвет, C_g - цвет тексела текстуры травы, C_s - цвет тексела текстуры песка, α_m - цвет тексела текстуры маски. Такое смешивание дает четкую картинку, лишенную заметных повторений трафарета. Кроме того, чтобы избежать размазывания текстуры на наклонных участках рельефа, используется инструмент Relax модификатора Upwgar системы 3DS MAX, который позволяет накладывать текстуру перпендикулярно к поверхности. На рис. 5 показан результат описанного наложения текстур.

3. Перемещение робота по полигону



Рис. 5. Поверхность полигона

Движение колесного робота по поверхности виртуального полигона основано на обработке коллизий (столкновений) колес с этой поверхностью. Для этого на центральной линии (обода) колеса выбирается три контрольные точки: самая нижняя и две боковые точки, отстоящие от нее на угол φ (см. рис. 6.). На каждом шаге моделирования вычисляется расстояние по

нормали от поверхности полигона до нижней и одной из боковых (в зависимости от направления движения колеса) контрольных точек. Если расстояние отрицательное (т.е. колесо проваливается под поверхность),

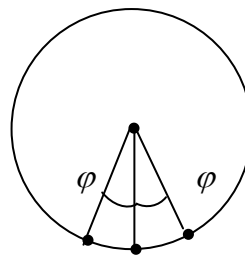


Рис. 6. Контрольные точки на колесе

то вычисляется сила реакции опоры, действующая на колесо с целью перемещения колеса на поверхность. Если расстояние положительное, то сила тяжести опускает колесо в нужное место. Эти силы действуют сначала на подвеску колеса, а через нее – на кузов робототехнического средства. Движение гусеничного робота по поверхности выполняется как движение колесного робота, а затем колеса обтягиваются гусеницей. На рис. 7 показан гусеничный робот МФ-4 на виртуальном полигоне.

Заключение



Рис. 7. Гусеничный робот МФ-4 на полигоне

Программная реализация виртуального полигона работает на персональных компьютерах под управлением операционной системы Windows. Апробация разработанного полигона показала его пригодность и адекватность для решения поставленных задач.

Работа выполнена при поддержке гранта РФФИ № 16-07-00760.

Virtual Polygon Surface Modeling

M.V.Mikhaylyuk, P.Y.Timokhin

Abstract: The technology of virtual surface (land relief) modeling for wheeled and tracked robots is proposed. It includes geometric relief model creation and texturing by blending some textures.

Keywords: virtual polygon, wheeled robot, height grid, multitexturing.

Литература

1. М.В.Михайлюк, Б.В.Бурдин, И.Г.Сохин, М.А.Торгашев. Использование виртуальных 3D-моделей для экспериментальной отработки бортовых полетных операций, выполняемых с помощью антропоморфных роботов. Робототехника и техническая кибернетика, № 1, 2013, с. 42 - 46.
2. М.В.Михайлюк. Тренировочный комплекс операторов робототехнических средств по ликвидации последствий чрезвычайных ситуаций. Сб. статей "Фундаментальные проблемы системной безопасности", вып. 2, Вычислительный центр РАН им. А.А.Дородницына, издательство "Вузовская книга", М., 2010, с. 441- 444.

Определение коллизий аппроксимирующих капсул и сфер трехмерных виртуальных динамических объектов

А.М. Трушин¹, М.В. Михайлюк²

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail's: ¹ leha_trushin@mail.ru, ² mix@niisi.ras.ru

Аннотация: Трехмерные виртуальные динамические объекты в системах компьютерного моделирования могут сталкиваться друг с другом. Определение таких столкновений (коллизий) является важной и неотъемлемой частью моделирования динамики виртуальных объектов. В настоящей работе предлагаются алгоритмы определения коллизий аппроксимирующих капсул между собой, а также капсул со сферами.

Ключевые слова: определение коллизий, аппроксимирующие контейнеры, моделирование динамики.

Введение

Одной из важнейших задач при моделировании трехмерных виртуальных динамических объектов (тел) является определение их столкновений (коллизий) [1]. Определение коллизий объектов произвольных форм является трудной и ресурсоёмкой задачей [2]. Поэтому один из распространенных методов определения коллизий основывается на том, что каждый виртуальный динамический объект окружается (аппроксимируется) одним или несколькими геометрическими контейнерами (прямоугольными параллелепипедами, сферами, капсулами и т.д.) [3]. В этом случае определение коллизии двух тел состоит из определения параметров (точки, глубины и нормали) столкновения их аппроксимирующих контейнеров.

Алгоритмы определения коллизий разделяются на два типа: априорные и апостериорные [4]. Априорные алгоритмы предсказывают столкновения тел до их фактического пересечения. Апостериорные алгоритмы определяют коллизии тел уже по факту их пересечения. В общем случае априорные алгоритмы с большей точностью определяют параметры коллизии. Но такие алгоритмы обладают большим объемом входных данных (положения и ориентации объектов, а также их физические характеристики - скорости, силы, моменты и т.д.). Поэтому эти алгоритмы имеют большую вычислительную сложность и позволяют моделировать динамику в режиме реального времени (не менее 25 шагов моделирования в секунду) только небольшого количества виртуальных объектов. Апостериорные алгоритмы определения коллизий используют в качестве входных данных лишь положения и ориентации объектов.

В настоящей работе предлагаются алгоритмы определения коллизий капсул между собой и капсул со сферами. Коллизии сфер и боксов между собой рассмотрены в [5-6]. При этом мы предполагаем, что скорости объектов невелики, так, что за один шаг моделирования они не могут сильно проникнуть друг в друга.

1. Постановка задачи

В настоящей работе будем использовать

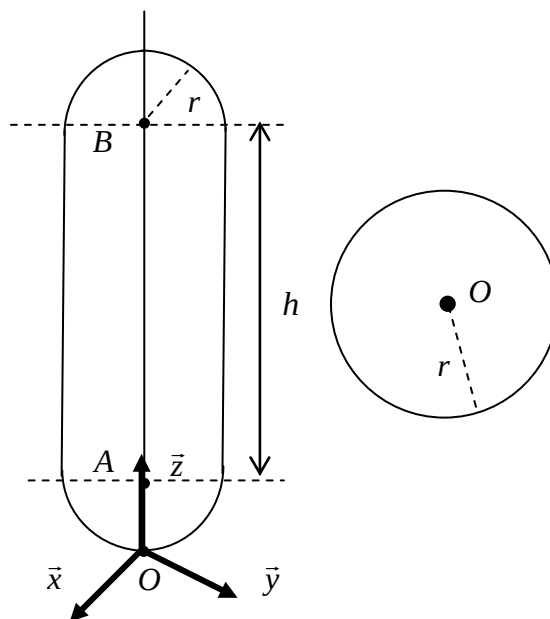


Рис. 1. Капсула $C(O, h, r)$ и сфера $S(O, r)$

ортонормированные правосторонние системы координат. Каждый аппроксимирующий контейнер имеет матрицу перехода из его локальной системы координат (л.с.к.) в мировую (м.с.к.). Все вычисления будут производиться в мировой системе координат.

Начало O л.с.к. капсулы находится в нижней точке одной из полусфер, ось $\bar{z}$ направлена вдоль оси капсулы (см. рис. 1). Центры полусфер капсулы будем обозначать через A и B , их радиусы - через r , расстояние между ними - h , а саму капсулу - через $C(O, h, r)$. Отрезок AB в дальнейшем будем называть осью капсулы. Начало л.с.к. сферы находится в ее

между осями соответствует расстоянию между точками P_1 и P_2 этих осей. Покажем, что хотя бы одна из этих точек должна совпадать с концом оси. Действительно, если обе точки лежат внутри осей, то можно выбрать такие малые числа Δ_1 и Δ_2 , что точки $P'_1 = P_1 + \bar{z}_1 \Delta_1$ и $P'_2 = P_2 + \bar{z}_2 \Delta_2$ будут расположены ближе друг к другу, чем точки P_1 и P_2 (см. рис 4). Противоречие показывает, что хотя бы одна из точек P_1 или P_2 совпадает с концом оси капсулы и задача сводится к нахождению минимального расстояния концов одной капсулы до оси другой. Рассмотрим

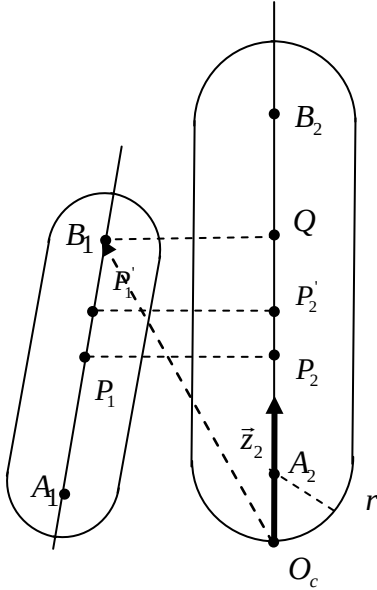


Рис. 4. Оси капсул компланарны, но не параллельны

случай, когда P_1 совпадает с концом B_1 оси первой капсулы. Длина проекции отрезка $O_c B_1$ на линию оси второй капсулы равна $\lambda = (B_1 - O_c, \bar{z}_2)$, поэтому основание Q перпендикуляра, опущенного из B_1 на ось $\bar{z}_2$ равно

$$Q = O_c + \lambda \bar{z}_2.$$

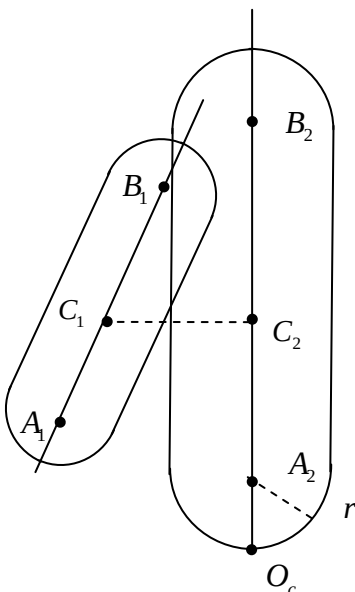


Рис. 5. Оси капсул не компланарны

Если $0 \leq \lambda \leq 1$, то Q лежит на оси $A_2 B_2$ второй капсулы и отрезок $Q B_1$ будет определять минимальное расстояние между точкой B_1 и отрезком $A_2 B_2$. В противном случае минимальное расстояние будет совпадать с одним из расстояний от B_1 до концов оси второй капсулы, т.е. $\min(\rho(B_1, A_2), \rho(B_1, B_2))$.

Таким образом, весь алгоритм сводится к вычислению минимального расстояния всех концов оси одной капсулы до оси другой (всего 4 случая) и выбора из них минимального значения. Здесь также используется Алгоритм_1 пересечения сфер, который вычисляет точки контакта, нормали и глубины проникновения.

3. Пусть оси двух капсул не компланарны, т.е. соответствуют скрещивающимся прямым (см. рис. 5). В этом случае необходимо определить две такие точки C_1 и C_2 их осей, что отрезок $C_1 C_2$ задает кратчайшее расстояние между осями капсул. Тогда определение коллизии двух капсул сводится к определению коллизии двух сфер, центры которых совпадают с найденными точками, а радиусы – с радиусами капсул.

Зададим обе оси в параметрическом виде:

$$A(s) = A_1 + s(B_1 - A_1), \quad 0 \leq s \leq 1,$$

$$B(t) = A_2 + t(B_2 - A_2), \quad 0 \leq t \leq 1.$$

Пусть $\bar{u} = B_1 - A_1$, $\bar{v} = B_2 - A_2$ и $\bar{w}_0 = A_1 - A_2$. Тогда произвольный вектор, соединяющий точки этих осей можно записать в виде:

$$\bar{w}(s, t) = A(s) - B(t) = \bar{w}_0 + s\bar{u} - t\bar{v}.$$

Так как оси скрещиваются, то кратчайшим расстоянием между ними является длина общего перпендикуляра, а концы этого перпендикуляра – ближайшие точки на этих прямых. Пусть $\bar{w}_c = \bar{w}(s_c, t_c)$ – общий перпендикуляр к двум осям. Тогда

$$\begin{cases} (\bar{u}, \bar{w}_c) = 0 \\ (\bar{v}, \bar{w}_c) = 0 \end{cases}.$$

Подставим в эту систему выражение для $\bar{w}_c$:

$$\begin{cases} (\bar{u}, \bar{u})s_c - (\bar{u}, \bar{v})t_c = -(\bar{u}, \bar{w}_0) \\ (\bar{v}, \bar{u})s_c - (\bar{v}, \bar{v})t_c = -(\bar{v}, \bar{w}_0) \end{cases}.$$

Сделаем замены $a = (\bar{u}, \bar{u})$, $b = (\bar{u}, \bar{v})$, $c = (\bar{v}, \bar{v})$, $d = (\bar{u}, \bar{w}_0)$ и $e = (\bar{v}, \bar{w}_0)$. Тогда получим

$$\begin{cases} s_c = \frac{be - cd}{ac - b^2} \\ t_c = \frac{ae - bd}{ac - b^2} \end{cases}.$$

Знаменатель

$ac - b^2 = |\bar{u}|^2 |\bar{v}|^2 - (|\bar{u}| |\bar{v}| \cos \alpha)^2 = (|\bar{u}| |\bar{v}| \sin \alpha)^2 \geq 0$, где α – угол между осями (он не равен нулю, т.к. прямые не параллельны). Подставив s_c и t_c в исходные уравнения прямых, получим точки концов общего перпендикуляра для этих осей. Если $0 \leq s_c \leq 1$ и $0 \leq t_c \leq 1$, то эти точки будут ближайшими на осях капсул. Если неравенства нарушаются, то это означает, что полученные точки лежат вне осей, и ближайшими точками будут являться ближайшие концы этих осей. Таким образом, получаем следующий алгоритм:

Начало.

$$s_c = \frac{be - cd}{ac - b^2}.$$

$$t_c = \frac{ae - bd}{ac - b^2}.$$

Если $s_c > 1$, то $s_c = 1$,

иначе, если $s_c < 0$, то $s_c = 0$.

Если $t_c > 1$, то $t_c = 1$,

иначе, если $t_c < 0$, то $t_c = 0$.

$$C_1 = A_1 + s_c \vec{u}$$

$$C_2 = A_2 + t_c \vec{v}.$$

Конец.

Далее с помощью Алгоритма_1 вычисляются точки контакта, нормали и глубины проникновения.

Заключение

Описанные алгоритмы определения коллизий капсул между собой и капсул со сферами позволяют рассчитывать информацию о коллизиях с небольшими вычислительными затратами.

Совместно с алгоритмами обработки коллизий, описанными в [7], это позволяет моделировать Совместно с алгоритмами обработки коллизий, описанными в [7], это позволяет моделировать динамику большого (несколько тысяч) количества виртуальных объектов в режиме реального времени.

Рассмотренные алгоритмы были реализованы в рамках разработанной в ФГУ ФНЦ НИИСИ РАН системы моделирования динамики виртуальных объектов. Реализованные алгоритмы были протестированы на большом наборе тестовых сцен, имитирующих разнообразные ситуации физического взаимодействия твердых тел (упругие удары, покой множества тел и т.д.). Апробация алгоритмов в рамках вышеуказанной подсистемы динамики показала реалистичность поведения виртуальных объектов и приемлемое время расчетов. Даже в сложных сценах (множество тел, лежащих друг на друге) все расчеты производились в режиме реального времени «с запасом» (времена суммарных расчетов библиотеки динамики не превышали 5 мс на один кадр моделирования).

Работа выполнена при поддержке РФФИ, грант № 16-37-00218.

Collision detection for bounding capsules and spheres for 3D virtual dynamic objects

A.M. Trushin, M.V. Mikhaylyuk

Abstract. Virtual dynamic 3D objects in computer simulation systems may collide with each other. Collision detection is important and integral part of virtual object's dynamics simulation. In this paper we propose collision detection's algorithms for bounding capsules between themselves and capsules with spheres.

Keywords: collision detection, bounding volume, capsule, dynamics modeling.

Литература

1. M. Moore, J. Wilhelms. Collision Detection and Response for Computer Animation. Computer Graphics, 1998, Vol. 22, no. 4, pp. 289–298. Available from: <http://www.cs.princeton.edu/courses/archive/spring01/cs598b/papers/moore88.pdf>
2. C. Lin Ming, S. Gottschalk. Collision detection between geometric models: a survey. Proc. of IMA conference on mathematics of surfaces. 1998, Vol. 1, pp. 602–608. Available from: <https://users.soe.ucsc.edu/~pang/161/w06/notes/cms98.pdf>
3. Bounding volume. Wikipedia. The Free Encyclopedia. Available from: https://en.wikipedia.org/wiki/Bounding_volume
4. C. Ericson. Real-Time Collision Detection. CRC Press, 2004, 594 p. Available from: <https://books.google.ru/books?id=WGpL6Sk9qNAC>
5. М.В. Михайлюк, А.М. Трушин. Расчет коллизий прямоугольных параллелепипедов в задачах динамики. Труды НИИСИ РАН. Том 2, №2. Обработка изображений, моделирование и визуализация: теоретические и прикладные аспекты. М.: НИИСИ РАН, 2012. С. 51–59.
6. А.М. Трушин. Определение коллизий аппроксимирующих сфер и прямоугольных параллелепипедов в системах трехмерного моделирования. Программные продукты и системы – 2015. – №4. – С. 105–109
7. А.М. Трушин. Обработка коллизий виртуальных объектов с помощью метода последовательных импульсов. Труды НИИСИ РАН. Том 4, №2. Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты. М.: НИИСИ РАН, 2014. С. 95–105.

Мониторинг заводнения нефтяных пластов с помощью суперэлементного моделирования трассировки закачиваемой воды

И.В.Афанаскин¹, Н.П.Ефимова², О.В.Ломакина³, Ю.М. Штейнберг⁴, Д.Т. Миронов⁵

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,

E-mail's : ¹ ivan@afanaskin.ru , ² efinatka@gmail.com, ³ ovllom@mail.ru, ⁴ yurash22@gmail.com,

⁵ dmitry.mironov@tns-simcor.com

Аннотация: Рассмотрена суррогатная модель прослеживания закачиваемой в нефтяной пласт воды на базе концепции суперэлементов. Для аппроксимации системы уравнений использована полностью явная численная схема. Модель может быть использована для решения задач контроля заводнения нефтяных месторождений и оперативного управления разработкой.

Ключевые слова: прослеживание закачиваемой воды, заводнение, экспресс-моделирование, суррогатная модель, суперэлементная модель, сетка Вороного.

Введение

Основным методом разработки нефтяных месторождений в России является заводнение. При этом доля воды в продукции большинства добывающих скважин превышает 80 %. Поэтому актуальной задачей является определение источников обводнения скважин и регулирование разработки (в первую очередь – изменение режима работы нагнетательных и добывающих скважин) таким образом, чтобы уменьшить обводненность продукции. Для решения этой задачи можно использовать суррогатную модель на базе концепции суперэлементов. Модель должна учитывать двухфазный характер фильтрации и позволять осуществлять трассировку (прослеживание) закачиваемой в пласт воды, т.е. определять, какой процент воды в продукции добывающей скважины I пришел из нагнетательной скважины J . Одной из таких моделей посвящена данная работа. В виду больших размеров расчетных ячеек (из-за использования концепции суперэлементов) для упрощения вычислений предлагается использовать полностью явную численную схему.

1. Математическая модель двухфазной фильтрации нефти и воды с трассировкой нейтральной примеси

Система уравнений, описывающая двухфазную фильтрацию упругих жидкостей, состоит из двух уравнений сохранения количества основных компонентов (объемов нефти и воды в стандартных условиях) и обобщенного закона Дарси,

капиллярными и гравитационными силами пренебрегается [1, 3, 4]:

уравнения сохранения объемов нефти и воды

$$\frac{\partial}{\partial t} \left(\frac{mS_o}{B_o} \right) + \operatorname{div} \left(\frac{\vec{W}_o}{B_o} \right) = -\bar{q}_o, \quad (1)$$

$$\frac{\partial}{\partial t} \left(\frac{mS_w}{B_w} \right) + \operatorname{div} \left(\frac{\vec{W}_w}{B_w} \right) = -\bar{q}_w + \bar{q}_a, \quad (2)$$

обобщенный закон Дарси

$$\vec{W}_o = -\frac{kk_{ro}}{\mu_o} \operatorname{grad}(P), \quad (3)$$

$$\vec{W}_w = -\frac{kk_{rw}}{\mu_w} \operatorname{grad}(P), \quad (4)$$

где m - пористость; S_o и S_w , B_o и B_w , $\vec{W}_o$ и $\vec{W}_w$, $\bar{q}_o$ и $\bar{q}_w$, $\bar{q}_a$, k_{ro} и k_{rw} , μ_o и μ_w - насыщенность, объемный коэффициент, вектор скорости фильтрации, плотность источника (стока), учитывающего работу скважин, плотность источника (стока), учитывающего работу законтурной водоносной области, относительная фазовая проницаемость, вязкость нефти и воды соответственно; k - абсолютная проницаемость; P - пластовое давление.

Для трассировки воды используется уравнение сохранения объема k -ой нейтральной примеси:

$$\frac{\partial}{\partial t} \left(\frac{mS_w C_k}{B_w} \right) + \operatorname{div} \left(\frac{\vec{W}_w C_k}{B_w} \right) = -\bar{q}_w C_{kw} + \bar{q}_a C_{ka}, \quad (5)$$

где C_k - концентрация k -ой примеси в водной фазе в пласте, C_{kw} - концентрация k -ой примеси в скважине, C_{ka} - концентрация k -ой примеси в законтурной области. Для трассировки воды предлагается считать в качестве k -ой примеси воду из k -ого источника. В качестве источников воды могут выступать нагнетательные скважины, законтурная область, заколонные перетоки.

Систему уравнений (1)-(4) необходимо дополнить замыкающими соотношениями:

$$S_o + S_w = 1, \quad (6)$$

$$\sum_{k=1}^{N_k} C_k = 1, \quad (7)$$

$$m = m_0 [1 + C_r (P - P_0)], \quad (8)$$

$$B_o = B_{o0} [1 - C_o (P - P_0)], \quad (9)$$

$$B_w = B_{w0} [1 - C_w (P - P_0)], \quad (10)$$

где N_k - количество источников воды, m_0 - пористость при начальном пластовом давлении; B_{o0} и B_{w0} - объемный коэффициент нефти и воды при начальном пластовом давлении; C_r , C_o и C_w - коэффициенты сжимаемости пласта, нефти и воды; P_0 - начальное пластовое давление. Соотношения (8)-(10) справедливы в предположении слабой сжимаемости породы и флюидов в рабочем диапазоне давлений выше давления насыщения нефти газом.

Систему уравнений (1)-(4) с учетом соотношений (6), (8)-(10) можно преобразовать к следующей системе дифференциальных уравнений для водонасыщенности $S \equiv S_w$ и давления P [4]:

$$m_0 [C_r + (C_w - C_o)S + C_o] \frac{\partial P}{\partial t} = -B_{o0} \left[\bar{q}_o + \operatorname{div} \left(\frac{\vec{W}_o}{B_o} \right) \right] - \quad (11)$$

$$-B_{w0} \left[\bar{q}_w - \bar{q}_a + \operatorname{div} \left(\frac{\vec{W}_w}{B_w} \right) \right] m_0 \left[\frac{\partial S}{\partial t} + (C_r + C_w)S \frac{\partial P}{\partial t} \right] = -B_{w0} \left[\bar{q}_w - \bar{q}_a + \operatorname{div} \left(\frac{\vec{W}_w}{B_w} \right) \right] \quad (12)$$

Аналогично преобразуем уравнение (5):

$$m_0 \left[\frac{\partial (SC_k)}{\partial t} + (C_r + C_w)SC_k \frac{\partial P}{\partial t} \right] = -B_{w0} \left[\bar{q}_w C_{kw} - \bar{q}_a C_{ka} + \operatorname{div} \left(\frac{\vec{W}_w C_k}{B_w} \right) \right] \quad (13)$$

Уравнения (11)-(13) дополняются начальными условиями:

$$P = P(x, y, z, t = 0), \quad (14)$$

$$S = S(x, y, z, t = 0), \quad (15)$$

$$C_k = C_k(x, y, z, t = 0), \quad k = 1..N_k \quad (16)$$

и граничными условиями непротекания на внешних границах. Для моделирования законтурной водоносной области предлагается использовать источники слагаемые в уравнениях (11)-(13), записанные специальным образом [1]. Аналогично можно моделировать заколонные перетоки.

2. Численная схема

Рассмотрим расчетную сетку, составленную из так называемых суперэлементов [5] – ячеек с размерами в плане, сопоставимыми с расстоянием между скважинами (300-500 м). Тогда количество ячеек в модели будет примерно равно количеству скважин. Скважины используются как центры ячеек. В неразбуренной части объекта для построения сетки могут быть введены фиктивные скважины. Использование такой сетки (по сравнению с обычно применяемыми при моделировании) позволяет сократить вычислительные затраты в тысячи раз [5].

Задача построения суперэлементной сетки в плане (2D сетки) сводится к задаче построения для каждой скважины зон дренирования. Это понятие было введено А.П. Крыловым. Эта задача в свою очередь сводится к построению областей Вороного [6]:

1. Нанесение на карту скважин (пусть залежь вскрывается N скважинами) и границ залежи, рис. 1.а.
2. Построение выпуклой (условно выпуклой) оболочки по N скважинам путем соединения внешних скважин залежи прямыми линиями, рис. 1.б.
3. Построение триангуляции на множестве скважин залежи (участка). Триангуляцией на множестве точек N называется такое разделение залежи на участки, когда все точки соединены непересекающимися прямолинейными отрезками так, что любая грань, лежащая внутри выпуклой оболочки N , является треугольником. Построение триангуляции Делоне (триангуляция Делоне – триангуляция для заданного множества точек N на плоскости, при которой для любого треугольника все точки из N за исключением

точек, являющихся его вершинами, лежат вне окружности, описанной вокруг треугольника), рис. 1.в.

4. Построение областей Вороного для скважин, включенных в триангуляцию. Область Вороного для каждой точки (скважины) может рассматриваться как совокупность точек плоскости, ближайших к скважине. Каждое

ребро многоугольника Вороного строится как перпендикуляр к середине ребра каждого треугольника из полученной триангуляции. Каждая вершина области Вороного будет находиться как точка пересечения трех ребер многоугольников Вороного, рис. 1.г.

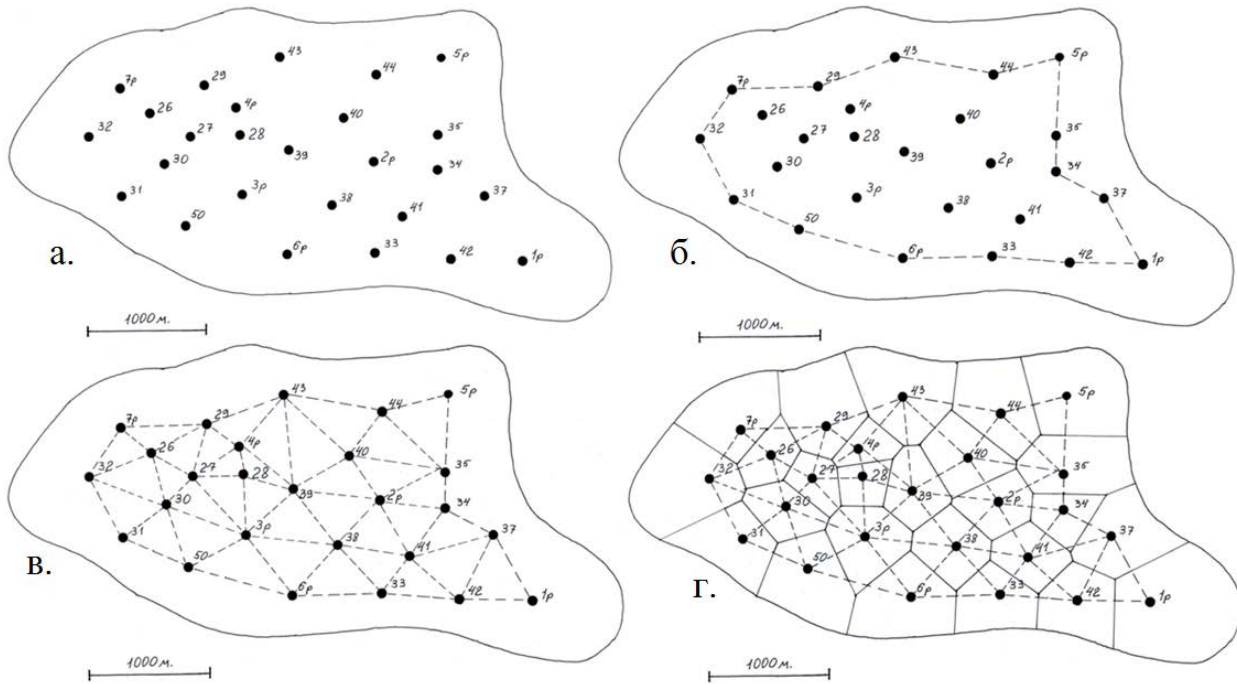


Рис. 1. Этапы построения в плане сетки суперэлементов

- а. – нанесение на карту скважин и границ, б. – построение условно выпуклой оболочки по внешним скважинам, в. – построение триангуляции Делоне, г. – построение областей Вороного

Для получения 3D суперэлементной сетки, построенную в плане сетку копируют для разных слоев, изменяя при этом глубину залегания кровли ячеек и их толщину, рис. 2.

Большой размер ячеек позволяет использовать полностью явную схему. Рассмотрим такую схему для уравнений (11)-(13) на сетке суперэлементов в плане (2D – плоская модель):

$$P_i^{n+1} = P_i^n - \Delta t^{n+1} \left\{ \frac{B_{o0} \left[(q_o)_i^n + \sum_{j=1}^{N_i} (q_o)_{ij}^n \right]}{F_i h_i(m_0)_i [C_r + (C_w - C_o) S_i^n + C_o]} + \frac{B_{w0} \left[(q_w)_i^n - (q_a)_i^n + \sum_{j=1}^{N_i} (q_w)_{ij}^n \right]}{F_i h_i(m_0)_i [C_r + (C_w - C_o) S_i^n + C_o]} \right\} \quad (17)$$

$$S_i^{n+1} = S_i^n - \Delta t^{n+1} \frac{B_{w0}}{F_i h_i(m_0)_i} \left[(q_w)_i^n - (q_a)_i^n + \sum_{j=1}^{N_i} (q_w)_{ij}^n \right] - (C_r + C_w) S_i^n (P_i^{n+1} - P_i^n) \quad (18)$$

$$(C_k)_i^{n+1} = \frac{1}{S_i^{n+1}} \left\{ S_i^n (C_k)_i^n - \Delta t^{n+1} \frac{B_{w0}}{F_i h_i(m_0)_i} \times \left[(q_w)_i^n (C_{kw})_i^n - (q_a)_i^n (C_{ka})_i^n + \sum_{j=1}^{N_i} \left\{ (C_k)_{ij}^n (q_w)_{ij}^n \right\} \right] - (C_r + C_w) S_i^n (C_k)_i^n (P_i^{n+1} - P_i^n) \right\} \quad (19)$$

где n - номер шага по времени; i - номер ячейки; Δt^{n+1} - шаг по времени; F_i , h_i и $(m_0)_i$ - площадь, толщина и пористость при давлении P_0 ячейки i ; $(q_\alpha)_i^n$ - дебит фазы $\alpha = o, w$ скважины (в стандартных условиях) в ячейке i на n -ом шаге по

времени; $(q_\alpha)_{ij}^n$ - поток фазы $\alpha = o, w$ (в стандартных условиях) между ячейками i и j на n -ом шаге по времени; $(q_a)_i^n$ - поток воды из законтурной области в ячейку i на n -ом шаге по времени; N_i - количество соседних ячеек для ячейки i .

При расчетах концентрации по формуле (19) предполагается, что закачиваемая вода может замещать связанную воду. При этом связанная вода становится подвижной, а часть закачиваемой воды наоборот связывается. Если считать, что такого эффекта не происходит, то концентрацию,

рассчитанную по формуле (19), нужно корректировать следующим образом:

$$(\bar{C}_k)_i^{n+1} = \begin{cases} (C_k)_i^{n+1}, & \sum_{k=1}^{N_k-1} (C_k)_i^{n+1} < 1 - \frac{(S_{wcr})_i}{S_i^{n+1}} \\ (\bar{C}_k)_i^n, & \sum_{k=1}^{N_k-1} (C_k)_i^{n+1} \geq 1 - \frac{(S_{wcr})_i}{S_i^{n+1}} \end{cases} \quad (20)$$

где $k = N_k$ - примесь, отвечающая за связанную воду, $(S_{wcr})_i$ - насыщенность связанной водой в ячейке i .

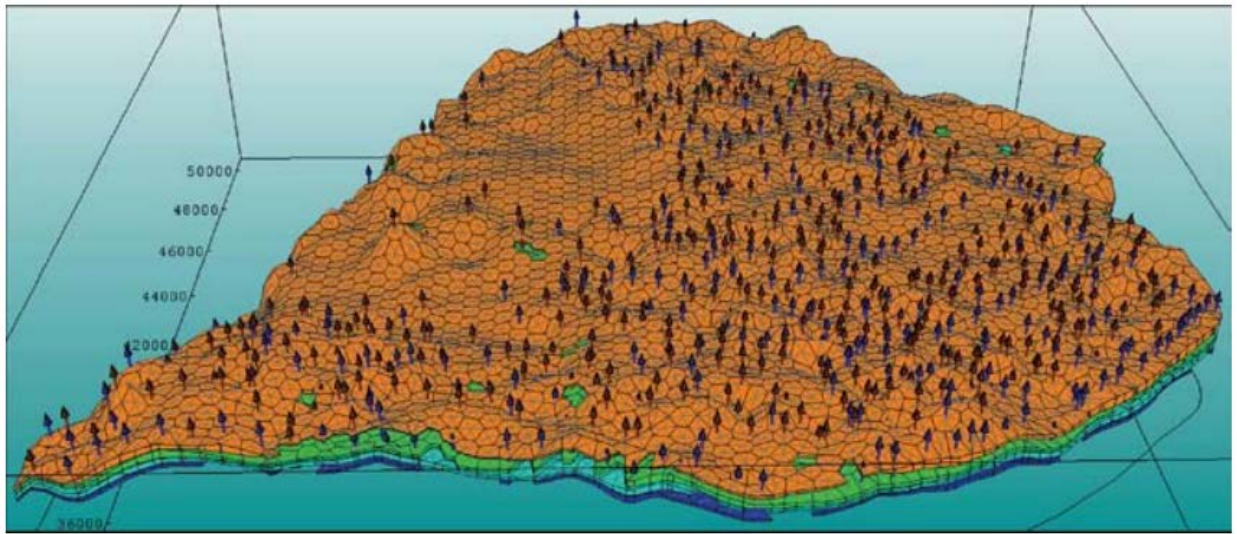


Рис. 2. Пример 3D сетки суперэлементов для реального месторождения [2]

Поток между ячейками i и j можно представить в виде:

$$(q_\alpha)_{ij}^n = \frac{k_{ij} (k_{ra})_{ij}^n}{\mu_\alpha (B_\alpha)_{ij}^n} \cdot \frac{P_i^n - P_j^n}{L_{ij}} A_{ij} h_{ij}, \quad \alpha = o, w, \quad (21)$$

где A_{ij} - длина совместного ребра ячеек i и j , рис. 3., L_{ij} - расстояние между центрами ячеек (скважинами) i и j , рис. 3.,

$$k_{ij} = \frac{2k_i k_j}{k_i + k_j}, \quad (22)$$

$$(k_{ra})_{ij}^n = \begin{cases} (k_{ra})_i^n, & P_i^n > P_j^n \\ (k_{ra})_j^n, & P_i^n < P_j^n \end{cases} \quad (23)$$

$$(B_\alpha)_{ij}^n = \frac{1}{2} [(B_\alpha)_i^n + (B_\alpha)_j^n], \quad (24)$$

$$h_{ij} = \frac{1}{2} [h_i + h_j]. \quad (25)$$

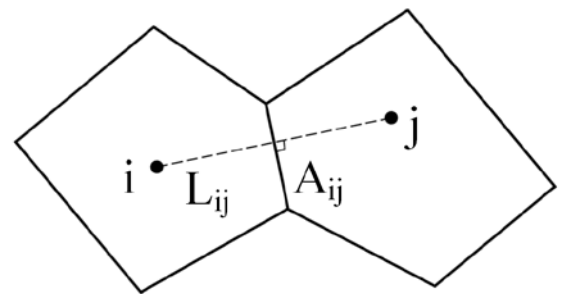


Рис. 3. Схема для расчета потоков между ячейками

3. Моделирование скважин

Дебиты скважин по жидкости q_l , нефти q_o , воде q_w , обводненность W и забойное давление P_w можно определить из следующих соотношений:

$$(q_l)_i^n = P I_i^n (P_i^n - P_w^n), \quad (26)$$

$$(q_o)_i^n = (q_l)_i^n (1 - W_i^n), \quad (27)$$

$$(q_w)_i^n = (q_l)_i^n W_i^n, \quad (28)$$

$$W_i^n = \frac{(k_{rw})_i^n}{(k_{rw})_i^n + (k_{ro})_i^n \frac{\mu_w (B_w)_i^n}{\mu_o (B_o)_i^n}}, \quad (29)$$

где PI_i^n - коэффициент продуктивности скважины i по жидкости, который определяется как:

$$PI_i^n = \frac{2\pi k_i h_i}{\ln \left[\frac{(R_c)_i}{(r_w)_i} \right] + Skin_i^n} \cdot \left[\frac{(k_{ro})_i^n}{\mu_o (B_o)_i^n} + \frac{(k_{rw})_i^n}{\mu_w (B_w)_i^n} \right] \quad (30)$$

где $(r_w)_i$ - радиус скважины по долоту, $Skin_i^n$ - скин-фактор скважины, $(R_c)_i$ - эквивалентный радиус блока, определяемый как:

$$(R_c)_i = 0,14\sqrt{2F_i}. \quad (31)$$

Для добывающих скважин необходимо задать дебит жидкости или забойное давление, для нагнетательных - расход воды или забойное давление.

Нетрудно заметить, что в такой модели вода появится в продукции добывающей скважины сразу же после превышения текущей водонасыщенности ячейки значения насыщенности связанной водой. Это, конечно, неверно. Вода необходимо время для того, чтобы пройти расстояние от нагнетательной скважины до добывающей. Для воссоздания этого эффекта используется ступенчатое задание функций относительной фазовой проницаемости:

$$(k_{rw})_i^n = \begin{cases} 0, S_i^n < (S_c)_i \\ A_i \left[\frac{S_i^n - (S_{wcr})_i}{1 - (S_{wcr})_i} \right]^{\alpha_i}, S_i^n \geq (S_c)_i \end{cases} \quad (32)$$

$$(k_{ro})_i^n = \begin{cases} B_i, S_i^n < (S_c)_i \\ B_i \left[\frac{1 - (S_{owcr})_i - S_i^n}{1 - (S_{owcr})_i - (S_{wcr})_i} \right]^{\beta_i}, S_i^n \geq (S_c)_i \end{cases} \quad (33)$$

где $(S_c)_i$ - водонасыщенность на фронте вытеснения; $(S_{wcr})_i$ - насыщенность связанной водой; $(S_{owcr})_i$ - насыщенность остаточной нефтью; коэффициенты A_i , B_i , α_i , β_i - определяются по результатам интерпретации исследований ядра и могут зависеть от пористости или проницаемости.

Водонасыщенность на фронте вытеснения $(S_c)_i$ определяется из соотношения:

$$f'((S_c)_i) = \frac{f((S_c)_i) - f((S_{wcr})_i)}{(S_c)_i - (S_{wcr})_i}, \quad (34)$$

где $f(S_i)$ - функция Бакли-Лeverетта:

$$f(S_i) = \frac{(k_{rw})_i}{(k_{rw})_i + (k_{ro})_i \frac{\mu_w}{\mu_o}}. \quad (35)$$

Доля воды из k -ого источника в продукции i -ой скважины на n -ом временном шаге может быть определена, как:

$$(\bar{C}_{kw})_i^n = \begin{cases} \text{если } \sum_{k=1}^{N_k-1} (C_k)_i^n = 0, \text{ то } 0 \\ \text{если } \sum_{k=1}^{N_k-1} (C_k)_i^n > 0, \text{ то } \begin{cases} \text{если } W_i^n = 0, \text{ то } 0 \\ \text{если } W_i^n > 0, \text{ то } \frac{(C_k)_i^n}{\sum_{k=1}^{N_k-1} (C_k)_i^n} \end{cases} \end{cases} \quad (36)$$

Предложенная 2D расчетная схема легко обобщается на случай 3D. При этом дебит каждого слоя определяется согласно его продуктивности при известном дебите скважины по жидкости или забойном давлении.

4. Моделирование законтурной области

Законтурной областью называют связанную с нефтяной залежью область пласта, заполненную водой и находящуюся за пределами внешнего контура нефтеносности.

Поток воды из законтурной области в ячейку i на n -ом шаге по времени можно задать, пользуясь методом последовательной смены стационарных состояний:

$$(q_{wa})_i^n = (AQ_{mult})_i PI_a (P_a^n - P_i^n), \quad (37)$$

где PI_a - «коэффициент продуктивности» законтурной области; $(AQ_{mult})_i$ - множитель, характеризующий долю потока из законтурной области в ячейку i в общем потоке из законтурной области в пласт; P_a^n - давление в законтурной области на n -ом шаге по времени.

По аналогии с формулой для давления в ячейке P_i^{n+1} (17) можно получить выражение для давления в законтурной области:

$$P_a^{n+1} = P_a^n - \Delta t^{n+1} \frac{B_{w0} \sum_{i=1}^{N_a} (q_{wa})_i^n}{V_{pa} (C_{ra} + C_w)}, \quad (38)$$

где N_a - количество ячеек модели нефтяной залежи, связанных с законтурной областью; V_{pa} - поровый объем законтурной области; C_{ra} - сжимаемость пласта в законтурной области.

5. Определение шага по времени

Первый шаг по времени Δt^1 задается в качества исходных данных.

Все последующие шаги по времени определяются как:

$$\Delta t^{n+1} = \min\{\Delta t_p^n, \Delta t_s^n\}, \quad (39)$$

где

$$\Delta t_p^n = \Delta t^n \frac{\varepsilon_p}{\Delta P^n}, \quad (40)$$

$$\Delta t_s^n = \Delta t^n \frac{\varepsilon_s}{\Delta S^n}, \quad (41)$$

где ε_s и ε_p - задаваемые максимальные изменения насыщенности и давления за один шаг по времени,

$$\Delta P^n = \max_i \{|P_i^n - P_i^{n-1}|\}, \quad (42)$$

$$\Delta S^n = \max_i \{|S_i^n - S_i^{n-1}|\}. \quad (43)$$

В конце шага по времени $n+1$ проверяются условия:

$$\max_i \{|P_i^{n+1} - P_i^n|\} \leq \varepsilon_p, \quad (44)$$

$$\max_i \{|S_i^{n+1} - S_i^n|\} \leq \varepsilon_s. \quad (45)$$

Если условия (44) и (45) выполняются – переходят к новому шагу по времени $n+2$, иначе определяют новое значение Δt^{n+1} и пересчитывают P_i^{n+1} и S_i^{n+1} :

$$\Delta t^* = C_\varepsilon \Delta t^{n+1}, \quad (46)$$

$$\Delta t^{n+1} = \Delta t^*, \quad (47)$$

где $C_\varepsilon \leq 1$ - задаваемая константа.

Кроме того, на шаг по времени Δt^{n+1} накладываются следующие ограничения:

$$\Delta t^{n+1} \geq \Delta t_{\min}, \quad (48)$$

где минимальный шаг по времени $\Delta t_{\min}$ определяется из соображений скорости счета,

$$\Delta t^{n+1} \leq \Delta t_{\max}, \quad (49)$$

где максимальный шаг по времени $\Delta t_{\max}$ определяется исходя из требуемых интервалов выдачи результатов.

И наконец, после удовлетворения условиям (44), (45) и (48), (49) необходимо проверять корректность применения модели двухфазной фильтрации (1)-(4):

$$\min_i \{P_i^{n+1}\} \geq P_s, \quad (50)$$

где P_s - давление насыщения нефти газом.

После этого рассчитывается погрешность материального баланса по фазам:

$$\delta_\alpha^{n+1} = \sum_i [(V_\alpha)_i^0 + (Q_{\alpha, inj})_i^{n+1} - (V_\alpha)_i^{n+1} - (Q_{\alpha, prod})_i^{n+1}] \rightarrow 0, \quad (51)$$

$$\alpha = o, w,$$

где $(V_\alpha)_i^0$ - начальные запасы фазы α в ячейке i в поверхностных условиях

$$(V_\alpha)_i^0 = F_i \cdot h_i \cdot m_i(P_i^0) \cdot (S_\alpha)_i^0 \cdot \frac{1}{B_\alpha(P_i^0)} \quad (52)$$

и $(V_\alpha)_i^{n+1}$ - текущие запасы фазы α в ячейке i в поверхностных условиях

$$(V_\alpha)_i^{n+1} = F_i \cdot h_i \cdot m_i(P_i^{n+1}) \cdot (S_\alpha)_i^{n+1} \cdot \frac{1}{B_\alpha(P_i^{n+1})} \quad (53)$$

$(Q_{\alpha, inj})_i^{n+1}$ - накопленная закачка фазы α в ячейку i в поверхностных условиях на текущий момент времени, $(Q_{\alpha, prod})_i^{n+1}$ - накопленная добыча фазы α в ячейку i в поверхностных условиях на текущий момент времени.

Нормальной погрешностью материального баланса для задач моделирования разработки нефтяных месторождений считается 5-7 %.

6. Проверка математической модели

Для проверки предложенной модели в гидродинамическом симуляторе Dz10 (авторы программы Р.М. Кац, Е.Р. Волгин, программа Dz10 успешно прошла сравнительное тестирование с коммерческим гидродинамическим симулятором Eclipse Schlumberger [4]) была создана секторная модель с 4 скважинами – 2 нагнетательными и 2 добывающими. Параметры модели следующие:

1. Размеры модели в плане 1500*1500 м.
2. Тип расчетной сетки – прямоугольная блочноцентрированная.
3. Количество ячеек – 3600. Размеры ячеек в плоскости ХУ – 25х25 м. Толщина – переменная, табл. 1.
4. Абсолютная проницаемость – переменная, табл. 1.
5. Пористость – переменная, табл. 1.
6. Начальное пластовое давление – 339 атм.
7. Начальная водонасыщенность – 0,2 д.ед.
8. Объемный коэффициент воды – 1,02 м³/м³. Вязкость воды – 0,36 сПз. Сжимаемость воды – 4,7*10⁻⁵ 1/атм.
9. Объемный коэффициент нефти – 1,55 м³/м³. Вязкость нефти – 0,397 сПз. Сжимаемость нефти – 1,01*10⁻⁴ 1/атм. Растворимость газа в нефти – 203,5 м³/м³. Давление насыщения нефти газом – 18,8 атм.
10. Сжимаемость породы – 4,7*10⁻⁵ 1/атм.
11. Капиллярным давлением пренебрегается.

12. Функции относительных фазовых проницаемостей (ОФП) - приведены на рис. 4.
13. Дебит жидкости добывающих скважин – табл. 1.
1. Минимальное забойное давление в добывающих скважинах - 18,8 атм.
14. Закачка воды в нагнетательную скважину – не ограничивалась. Забойное давление в нагнетательной скважине – табл. 1.
15. Скин-фактор для всех скважин – 0 ед.
16. Радиус скважин – 0,1 м.
17. Срок разработки - 35 лет.
18. Притока воды из-за контура нет.

При расчетах в Dz10 для задания свойств по площади был использован метод линейной интерполяции.

При расчетах на модели суперэлементов было установлено, что решение неустойчиво при задании

постоянного шага по времени приемлемой с вычислительной точки зрения длительности. При использовании переменного шага по времени неустойчивости решения удастся избежать, при этом средний шаг по времени может быть достаточно большим. В процессе расчетов для автоматизации процесса компенсации отбора закачкой добывающие скважины управлялись дебитом, а нагнетательные – забойным давлением.

Расстановка скважин в плане и конфигурация суперэлементов приведена на рис. 5.

Длина совместного ребра соседних ячеек для суперэлементной модели и расстояние между центрами соседних ячеек (скважинами) приведены в табл. 2.

Табл. 1

Параметры модели						
Скважина (суперэлементная ячейка)	Площадь, тыс. м ²	Толщина, м.	Пористость, д.ед.	Абсолютная проницаемость, мД	Дебит, м ³ /сут	Забойное давление, атм
1 – нагн.	514	25	0,20	70	-	372
2 – доб.	628	22	0,18	53	236	18,8
3 – доб.	603	19	0,16	44	168	18,8
4 – нагн.	504	15	0,15	38	-	372

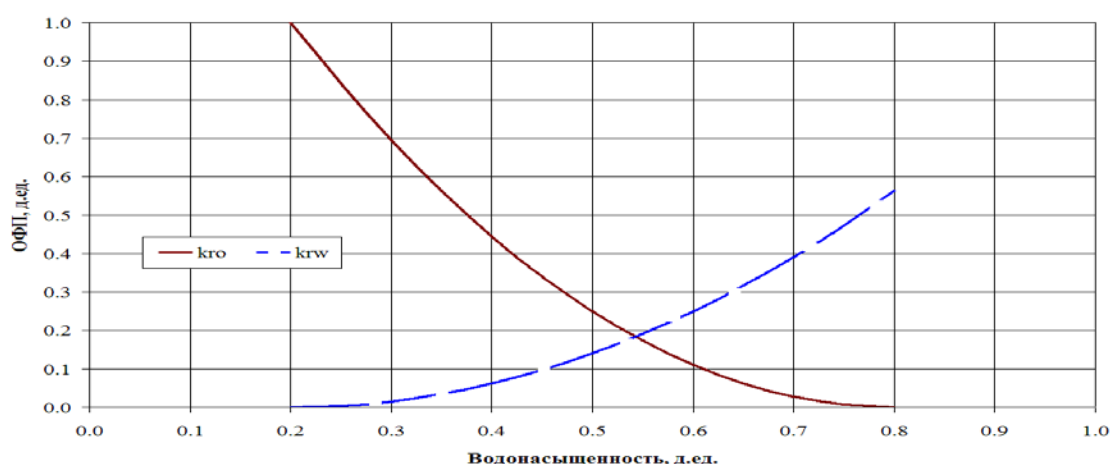


Рис. 4. Относительные фазовые проницаемости (ОФП) для нефти (kro) и воды (krw)

В ходе расчетов было установлено, что для обеспечения устойчивости счета на суперэлементной модели необходимо использовать следующие значения максимальных изменений расчетных параметров за один шаг по времени: $\varepsilon_s = 0,005$ д.ед. и $\varepsilon_p = 0,01$ атм. При этом были приняты следующие значения максимального и минимального временных шагов $\Delta t_{\max} = 10$ сут., $\Delta t_{\min} = 1$ сек. Изменение временного шага в процессе решения приведено на рис. 6.



● Добывающая скважина
▲ Нагнетательная скважина
Рис. 5. Расстановка скважин в плане и конфигурация суперэлементов

Расчеты с помощью Dz10 в течение первых 25 лет приняты в качестве «истории» при адаптации суперэлементной модели. Затем сделан прогноз разработки на 10 лет. На рис. 7 и 8 приведены результаты адаптации суперэлементной модели. Видно, что получено хорошее совмещение «истории» с расчетами на суперэлементной модели. Также видно хорошее совпадение прогнозных данных суперэлементной модели с данными Dz10. Установлено, что использование в качестве адаптивных параметров коэффициентов функций

ОФП (A_i , B_i , α_i , β_i) при использовании суперэлементной модели может привести к потере устойчивости решения. Поэтому в качестве адаптивного параметра использовалась насыщенность на фронте вытеснения S_c . Ее варьирование приводит к изменению времени обводнения добывающих скважин и изменению величины скачка обводненности в момент прорыва воды в скважину.

Табл. 2

Геометрические параметры суперэлементной модели

Пары ячеек	1-2	1-3	2-3	2-4	3-4
Длина совместного ребра ячеек, м	650	538	425	650	663
Расстояние между центрами соседних ячеек, м.	638	889	925	1013	813

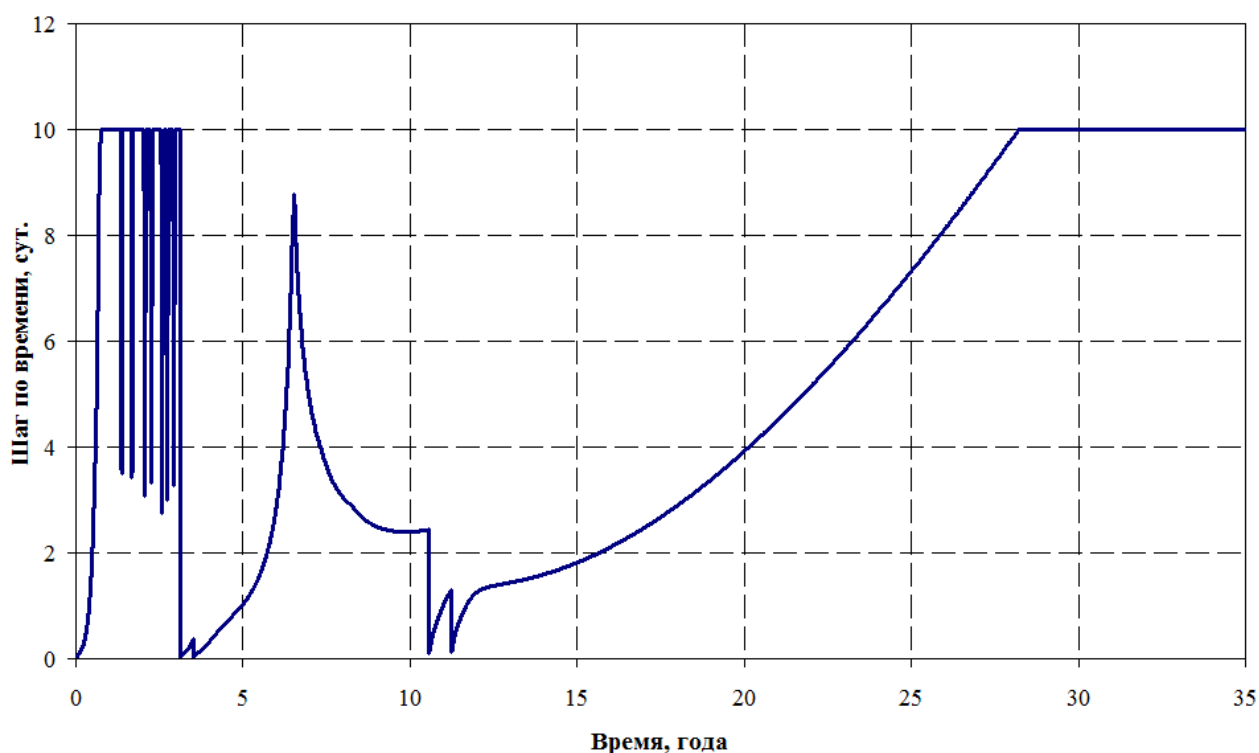


Рис. 6. Изменение временного шага

Вода, закачиваемая в скважину 1, принята в качестве примеси 1, соответственно ее концентрация в водной фазе обозначена C_1 . Вода, закачиваемая в скважину 4, принята в качестве примеси 2, соответственно ее концентрация в водной фазе обозначена C_2 .

Концентрации примесей 1 и 2 (воды, закачиваемой в скважины 1 и 4) в водной фазе для ячеек 1-4 при замещении пластовой связанной воды закачиваемой водой, т.е. концентрации, рассчитанные по формуле (19), приведены на рис. 9. Видно, что в этом случае при бесконечной промывке концентрации закачиваемой воды (примеси 1 и 2) в водной фазе в ячейках, содержащих нагнетательные скважины, соответственно 1 и 4, стремятся к 1.0.

Концентрации примесей 1 и 2 (воды, закачиваемой в скважины 1 и 4) в водной фазе без замещения пластовой связанной воды закачиваемой водой, т.е. концентрации, рассчитанные по формуле (20), приведены на рис. 10. В этом случае при бесконечной промывке концентрация закачиваемой воды (примеси 1 и 2) в водной фазе в ячейках, содержащих нагнетательные скважины, соответственно 1 и 4, стремится к 0.75, что соответствует единица минус концентрация связанной воды в водной фазе при 100 % водонасыщенности.

На рис. 11 приведены концентрации примесей 1 и 2 (воды, закачиваемой в скважины 1 и 4) в добываемой воде в скважинах 2 и 3. Видно, что закачиваемая вода от обоих нагнетательных скважин сначала прорывается в добывающую

скважину 2, а потом в скважину 3. После прорыва закачиваемой воды в добывающие скважины доля воды от нагнетательной скважины 1 в продукции

добывающих скважин медленно падает, а доля воды от нагнетательной скважины 4 растет.

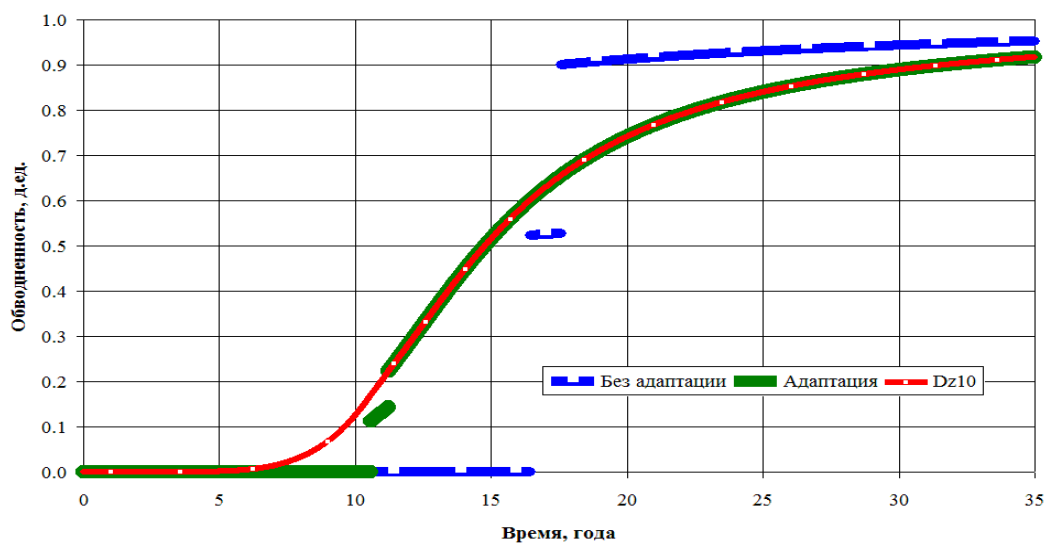


Рис. 7. Расчетная обводненность

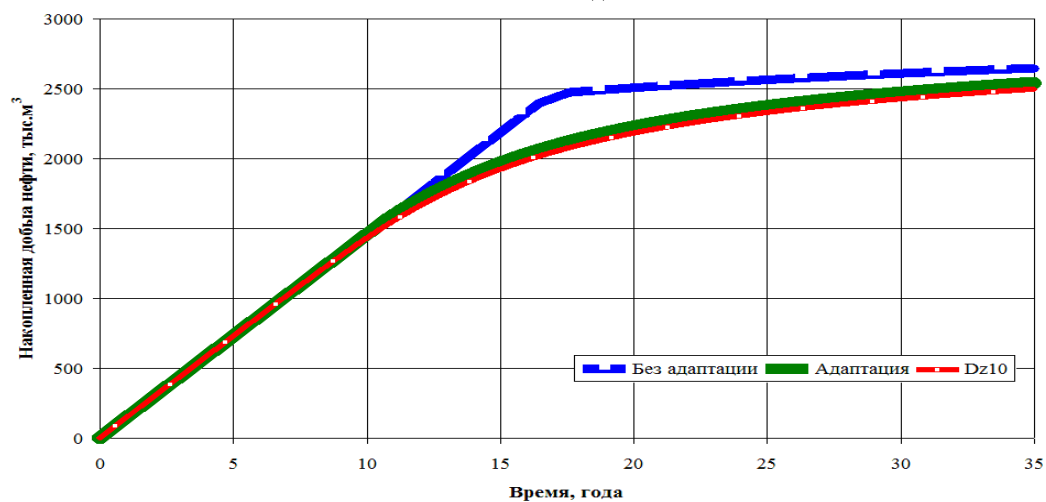


Рис. 8. Расчетная накопленная добыча нефти

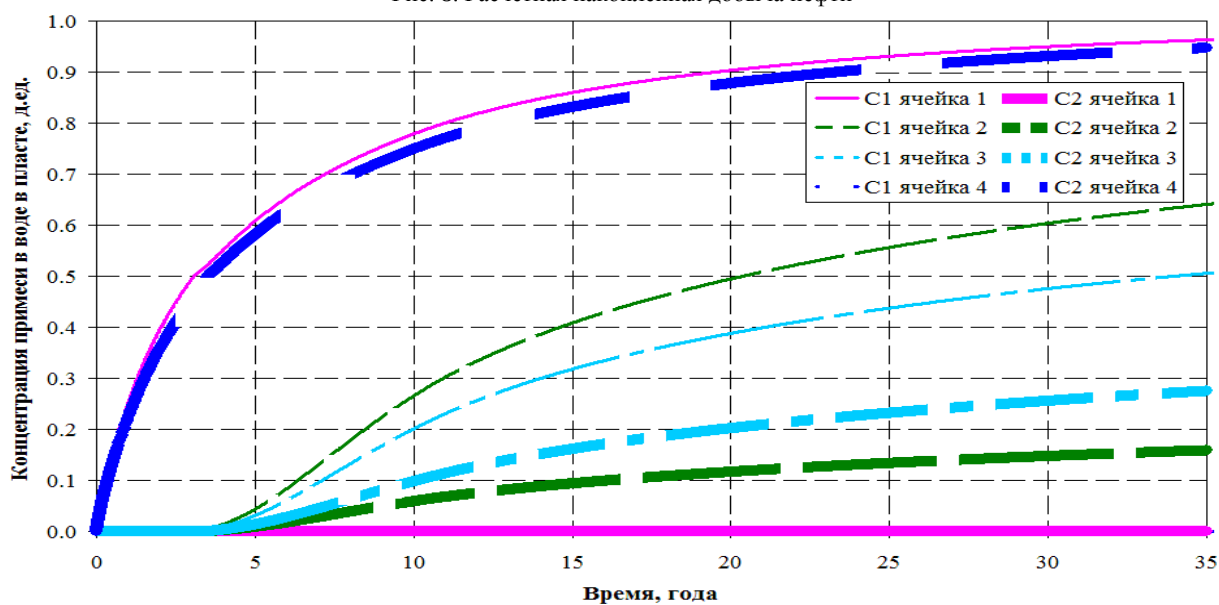


Рис. 9. Концентрация примеси (закачиваемой воды) в водной фазе при замещении пластовой связанной воды закачиваемой водой

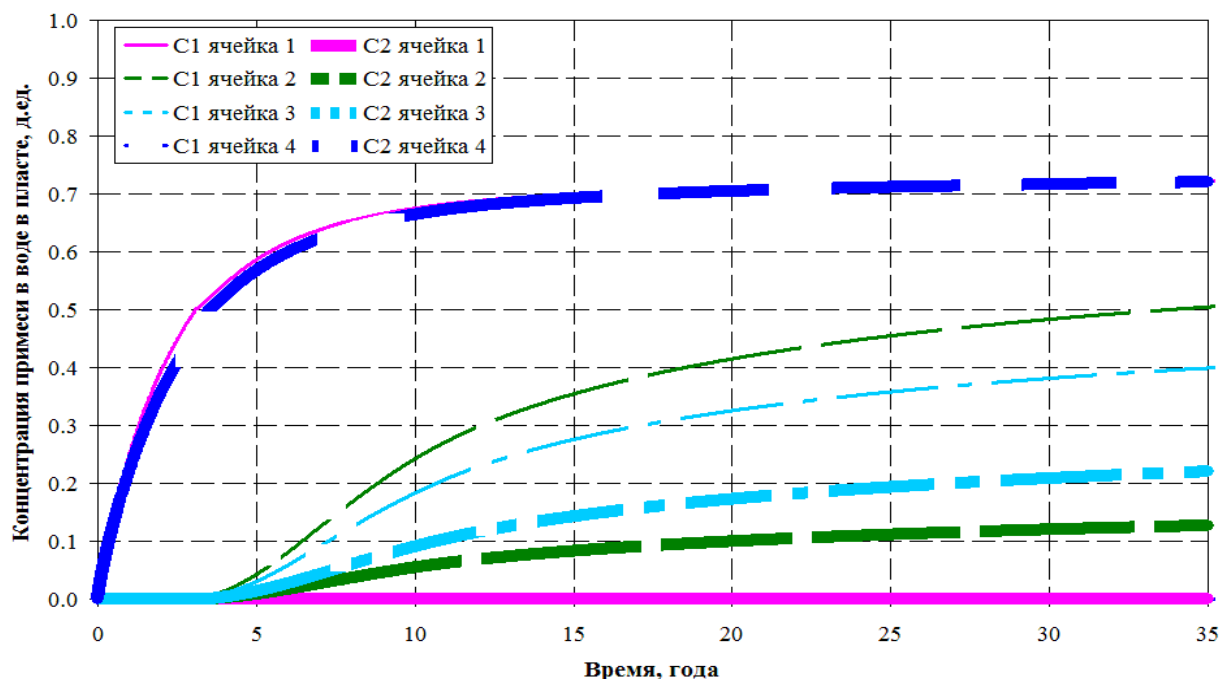


Рис. 10. Концентрация примеси (закачиваемой воды) в водной фазе без замещения пластовой связанной воды закачиваемой водой

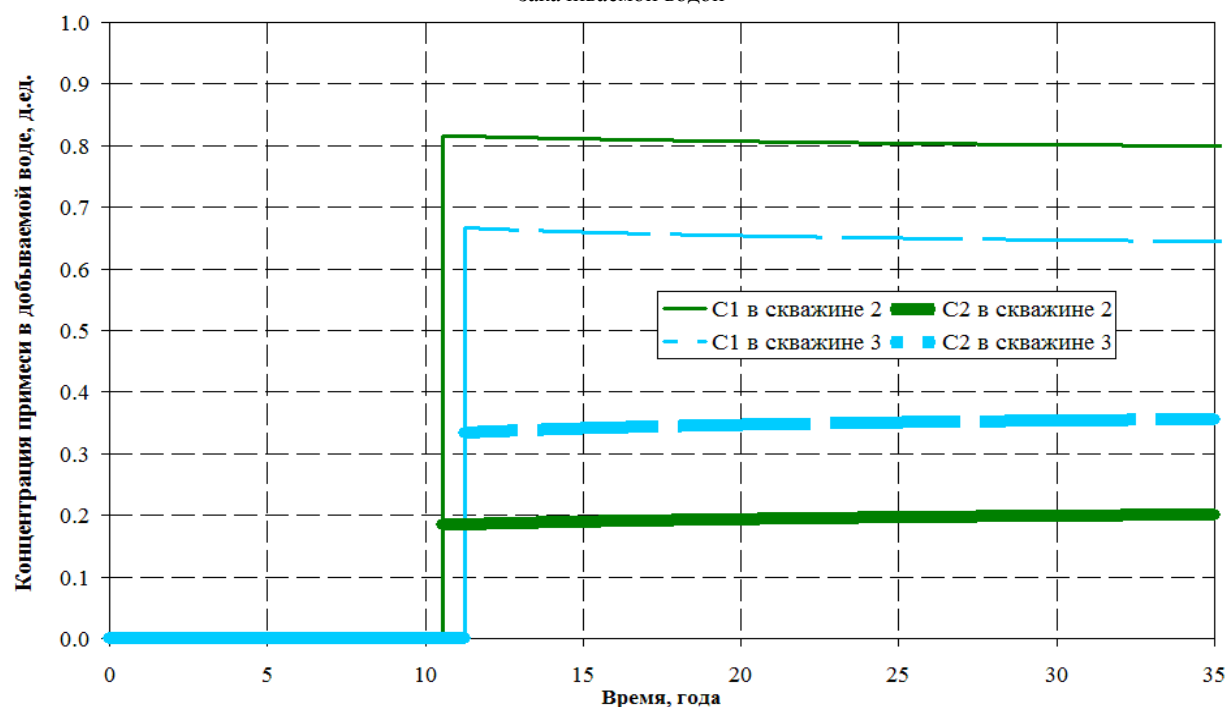


Рис. 11. Концентрация примеси (закачиваемой воды) в добываемой воде для скважин 2 и 3

Заключение

1. В работе предложена суррогатная модель работы группы нагнетательных и добывающих скважин в предположении слабой сжимаемости пластовых флюидов (в условиях двухфазной фильтрации нефти и воды).
2. В основе модели лежит полностью явная численная схема, использующая концепцию суперэлементов.
3. Результаты расчетов на тестовой модели близки к результатам расчетов с помощью программы Dz10 (авторы Р.М. Кац, Е.Р. Волгин), которая успешно прошла сравнительное тестирование с коммерческим гидродинамическим симулятором Eclipse Schlumberger.

4. Для обеспечения устойчивости счета рекомендуется управлять добывающими скважинами с помощью дебита жидкости, а нагнетательными – с помощью забойного давления и использовать переменный шаг по времени.
5. Оценен порядок максимально допустимого изменения расчетных параметров за один шаг по времени: насыщенность - 0,005 д.ед., давление - 0,01 атм.
6. Для обеспечения устойчивости в качестве параметров для адаптации модели рекомендовано использование насыщенности на фронте вытеснения.
7. Для мониторинга заводнения, контроля распространения воды из разных источников по пласту и оценки степени влияния нагнетательных скважин на добывающие

предложено использовать уравнение переноса нейтральной примеси. Для трассировки воды предлагается считать в качестве k -ой примеси воду из k -ого источника. В качестве источников воды могут выступать нагнетательные скважины, законтурная область, заколонные перетоки.

8. Модель может быть использована для решения задач оперативного управления разработкой нефтяных месторождений, контроля разработки и построения карт изменяющихся параметров пласта (давление, насыщенность, проводимость, гидропроводность, пьезопроводность, плотность запасов и пр.).

Работа выполнена при поддержке гранта РФФИ № 16–29–15135 офи_м

Oil reservoirs waterflooding monitoring by application of super elements modelling for injected water tracing

I.V. Afanaskin, N.P. Efimova, O.V. Lomakina, Yu.M. Shteynberg, D.T. Mironov

Abstract: Article describes quasi-model of injected water tracing, which is based on super-elements concept. System of equations approximation conducted by fully-explicit numerical scheme. Model could be applied for cases of oil fields water flooding control and real time reservoir development management.

Key words: injected water tracing, waterflooding, express-simulation, quasi-model, super-elements model, Voronogo Grid.

Литература

1. Х.Азиз, Э.Сеттари. Математическое моделирование пластовых систем. М.-Ижевск: Институт компьютерных исследований, 2004. – 416 с.
2. Д.Б.Булыгин, А.Б.Мазо, К.А.Поташев, Е.И.Калинин. Геолого-технические аспекты суперэлементной фильтрационной модели нефтяных месторождений // Научно-технический журнал «Георесурсы», 2013, № 3(53). - С. 31-35.
3. Р.Д.Каневская. Математическое моделирование гидродинамических процессов разработки месторождений углеводородов. М.-Ижевск: Институт компьютерных исследований, 2002. – 140 с.
4. Р.М.Кац, Е.Р.Волгин, И.В.Афанаскин. Численное моделирование двухфазной фильтрации нефти и воды // Труды НИИСИ РАН, том 4, № 2, 2014. – с. 141-148.
5. А.Б.Мазо, Д.В.Булыгин. Суперэлементы. Новый подход к моделированию разработки нефтяных месторождений // Научно-технический журнал «Нефть. Газ. Новации», 2011, № 11. – С. 6-8.
6. Н.И.Хисамутдинов, М.М.Хасанов, А.Г.Телин. и др. Разработка нефтяных месторождений. Издание в 4 т. – Т. 1. Разработка нефтяных месторождений на поздней стадии. М.: ВНИИОЭНГ, 1994. – 240 с.

Составление расписаний в многопроцессорной системе с учетом затрат на прерывания

М.Г. Фуругян

Вычислительный центр имени А.А.Дородницыны Федерального исследовательского центра
«Информатика и управление» РАН, Москва, Россия, E-mail : rts@ccas.ru

Аннотация: Рассматривается задача планирования вычислений в многопроцессорной системе с одинаковыми директивными интервалами с учетом затрат на обработку прерываний и переключений. Разработан алгоритм построения допустимого расписания, который при фиксированном числе процессоров является псевдополиномиальным, а от числа работ имеет линейную оценку сложности.

Ключевые слова: планирование вычислений, многопроцессорная система, оценка сложности.

1. Введение

Проблема поиска допустимого расписания в многопроцессорных системах без учета затрат на обработку прерываний и переключений при заданных директивных сроках выполнения работ исследовалась в [1] (случай идентичных процессоров), [2, 3] (случай произвольных процессоров), [4] и других работах. В [5] подобная задача исследовалась с учетом затрат на прерывания и переключения. В этой работе дополнительно предполагается, что работа процессоров представляется как набор последовательных тактов, а прерывания и переключения при выполнении работ могут осуществляться соответственно в конце и в начале тактов. В настоящей работе рассматривается задача составления допустимого расписания в многопроцессорной системе с идентичными процессорами с учетом затрат на обработку прерываний и переключений для случая, когда директивные интервалы всех работ совпадают, а прерывания допускаются в произвольные моменты времени.

2. Постановка задачи

Имеется множество работ $N = \{1, 2, \dots, n\}$, выполняемых с помощью m идентичных процессоров. Для каждой работы заданы ее длительность t_i и директивный интервал $[0, F]$. Предполагается, что

$$t_i \leq F \quad (1)$$

при всех $i \in N$. В фиксированный момент времени каждая работа может выполняться только одним процессором, а каждый процессор может выполнять не более одной работы. При выполнении работ допускаются прерывания и переключения с одного процессора на другой. При прерывании (или переключении) некоторой работы $i \in N$ процессором, на котором она выполнялась, в течение интервала времени длиной τ_1 , начиная с момента прерывания, выполняется специальная работа w_1 и в этом интервале не может выполняться никакая другая работа. Процессором, на котором возобновляется

выполнение прерванной работы, непосредственно перед ее возобновлением в течение интервала времени длиной τ_2 выполняется специальная работа w_2 и в этом интервале не может выполняться никакая другая работа. Предполагается, что

$$\tau_1 + \tau_2 \leq F, \quad (2)$$

т.е. в каждом директивном интервале можно выполнить обработку хотя бы одного прерывания.

Например, в многопроцессорных системах работы w_1 и w_2 — это выполнение специальных программ, осуществляющих запись данных о прерванной программе из памяти одного процессора (на котором она выполнялась до прерывания) в память другого процессора (на котором она будет выполняться после прерывания). Будем предполагать, что работы w_1 и w_2 — непрерываемые и могут выполняться только последовательно: работа w_2 может быть начата только после окончания выполнения работы w_1 .

Требуется определить, существует ли допустимое расписание выполнения работ N с учетом затрат на прерывания и переключения (т.е. такое расписание, при котором каждая работа выполняется в директивном интервале $[0, F]$), и найти его, если оно существует.

3. NP-полнота задачи

Покажем, что при $m \geq 2$ рассматриваемая задача является NP-трудной. Действительно, рассмотрим задачу о разбиении [6, 7], в которой задано множество натуральных чисел $a_1, a_2, \dots, a_n$

($B = \sum_{i=1}^n a_i$ — четное число) и требуется определить,

существует ли множество $N' \subset \{1, 2, \dots, n\}$, такое, что $\sum_{i \in N'} a_i = \sum_{i \in N \setminus N'} a_i$. Эта задача является NP-полной [6, 7] и полиномиально сводится к рассматриваемой задаче. Для этого достаточно положить

$m = 2$, $N = \{1, 2, \dots, n\}$, $t_i = a_i, i \in N$, $F = B/2$ (в этом случае выполнение хотя бы одного прерывания делает невозможным существование допустимого расписания и получается задача о разбиении).

4. Вспомогательные алгоритмы

Для описания алгоритма решения поставленной задачи нам понадобятся три вспомогательных алгоритма (разд. 4.1 – 4.3).

4.1. Модифицированный алгоритм упаковки

4.1.1. Описание алгоритма

Данный алгоритм $A_1(N_0, m_0)$ является модификацией алгоритма упаковки [1] для случая, когда учитываются временные затраты на обработку прерываний и переключений. В результате работы алгоритма $A_1(N_0, m_0)$ строится допустимое расписание выполнения множества работ $N_0 \subseteq N$ на m_0 процессорах ($m_0 \leq m$) при условии, что выполнено неравенство

$$\sum_{i \in N_0} t_i + (m_0 - 1)(\tau_1 + \tau_2) \leq m_0 F. \quad (3)$$

Условие (3) гарантирует существование допустимого расписания.

Алгоритм работает следующим образом. При $m_0 = 1$ все работы выполняются поочередно на имеющемся одном процессоре. Это возможно в силу (3).

Пусть теперь $m_0 \geq 2$. Занумеруем процессоры от 1 до m_0 . Пусть $N_{01} = \{i \in N_0 : t_i + \tau_1 + \tau_2 > F\}$, $N_{02} = N_0 \setminus N_{01} = \{i \in N_0 : t_i + \tau_1 + \tau_2 \leq F\}$. Таким образом, при всех $i \in N_{02}$

$$t_i + \tau_1 + \tau_2 \leq F. \quad (4)$$

Рассмотрим сначала случай, когда $|N_{01}| \geq m_0$. Пусть подмножество $\bar{N}_{01} \subseteq N_{01}$ такое, что $|\bar{N}_{01}| = m_0$. В разд. 4.1.2 будет показано, что в этом случае для любого $i_0 \in \bar{N}_{01}$ выполнено неравенство

$$\sum_{i \in N_0 \setminus \bar{N}_{01} \cup \{i_0\}} t_i \leq F. \quad (5)$$

Поэтому в данном случае для $(m_0 - 1)$ работ из множества $\bar{N}_{01} \setminus \{i_0\}$ выделяем $(m_0 - 1)$ процессоров: на каждую работу по одному процессору. Каждую работу $i \in \bar{N}_{01} \setminus \{i_0\}$ выполняем без прерываний, начиная с момента времени $t = 0$. Для всех оставшихся работ, т.е. работ из множества $N_0 \setminus \bar{N}_{01} \cup \{i_0\}$, выделяем последний, m_0 -й процессор и выполняем их последовательно без прерываний, начиная с момента времени $t = 0$.

Теперь рассмотрим случай, когда

$|\bar{N}_{01}| = k < m_0$. В разд. 4.1.2 будет показано, что в этом случае выполняется неравенство

$$\sum_{i \in N_{02}} t_i + (m_0 - k - 1)(\tau_1 + \tau_2) \leq (m_0 - k)F, \quad (6)$$

т.е. для множества работ $i \in N_{02}$ выполняется условие (3) с $(m_0 - k)$ процессорами. В этом случае каждой работе $i \in N_{01}$ выделяем по одному из первых k процессоров и выполняем эти работы без прерываний, начиная с момента времени $t = 0$. На оставшееся множество работ выделяем остальные $(m_0 - k)$ процессоров и выполняем эти работы следующим образом.

Занумеруем работы из множества N_{02} . Первую работу полностью выполняем на $(k + 1)$ -м процессоре без прерываний, начиная с момента времени $t = 0$. Это возможно в силу (1). Пусть уже назначено $(l - 1)$ работ ($l \geq 2$) из множества N_{02} . Пусть $(l - 1)$ -я работа завершилась в момент времени $t = T$ и непосредственно перед завершением она выполнялась на $(k + r)$ -м процессоре. Выберем очередную работу $l \in N_{02}$. Если $T + t_l \leq F$, то выполняем ее на $(k + r)$ -м процессоре без прерываний, начиная с момента времени $t = T$. Если $T + t_l > F$ и $T + \tau_2 > F$, то работу l полностью выполняем без прерываний на $(k + r + 1)$ -м процессоре, начиная с момента времени $t = 0$. Если же

$$T + t_l > F, \quad T + \tau_2 \leq F, \quad (7)$$

то работу l сначала выполняем на $(k + r + 1)$ -м процессоре, начиная с момента времени $t = 0$, в течение промежутка времени $t_l - (F - T - \tau_2)$. Затем она прерывается и на этом процессоре выполняется работа w_1 в течение промежутка времени τ_1 . На $(k + r)$ -м процессоре, начиная с момента времени $t = T$, выполняется работа w_2 в течение промежутка времени τ_2 , а затем в течение промежутка времени $(F - T - \tau_2)$ вновь выполняется прерванная работа l .

Описанную процедуру выполняем поочередно для всех работ из множества N_{02} . Корректность данной процедуры следует из неравенства (6). Вычислительная сложность алгоритма $A_1(N_0, m_0)$ составляет $O(|N_0|)$.

4.1.2. Обоснование алгоритма

Покажем сначала, что при обработке каждого прерывания работы w_1 и w_2 выполняются последовательно – сначала w_1 , а затем w_2 . Пусть x_1 – момент прерывания работы $l \in N_{02}$, а x_2 – момент возобновления ее выполнения, σ_{11}, σ_{12} – соответственно моменты начала и завершения выполнения работы w_1 ; σ_{21}, σ_{22} – соответственно

моменты начала и завершения выполнения работы w_2 .

$$\begin{aligned} \text{Тогда } x_1 &= t_l - (F - T - \tau_2), \quad x_2 = T + \tau_2, \\ \sigma_{11} &= t_l - (F - T - \tau_2), \quad \sigma_{12} = t_l - (F - T - \tau_2) + \tau_1, \\ \sigma_{21} &= T, \quad \sigma_{22} = T + \tau_2. \end{aligned}$$

$$\begin{aligned} \text{В силу (4) } \sigma_{12} &\leq \sigma_{21}. \text{ Кроме того, в силу (2),} \\ (7) \quad \sigma_{12} &= t_l - (F - T) + \tau_1 + \tau_2 = \\ &= (t_l + \tau_1 + \tau_2) - F - T \leq T < F, \quad \sigma_{12} \leq F, \end{aligned}$$

т.е. работы w_1 и w_2 выполняются в директивном интервале $[0, F]$.

Поскольку в случае, когда $|\bar{N}_{01}| = k < m_0$, число прерываний не более $(m_0 - k - 1)$, то неравенство (6) гарантирует, что все работы $l \in N_{02}$ могут быть выполнены $(m_0 - k)$ процессорами.

Докажем справедливость неравенства (5). Действительно, в противном случае выполнялись бы соотношения

$$\begin{aligned} \sum_{i \in N_0} t_i &= \sum_{i \in \bar{N}_{01} \setminus \{i_0\}} t_i + \sum_{i \in N_0 \setminus \bar{N}_{01} \cup \{i_0\}} t_i > \\ &\sum_{i \in \bar{N}_{01} \setminus \{i_0\}} [F - (\tau_1 + \tau_2)] + F = \\ &= (m_0 - 1)(F - (\tau_1 + \tau_2)) + F = \\ &= m_0 F - (m_0 - 1)(\tau_1 + \tau_2), \end{aligned}$$

которые противоречат неравенству (3).

Докажем справедливость неравенства (6). Действительно, в силу (3)

$$\begin{aligned} \sum_{i \in N_{02}} t_i &= \sum_{i \in N_0} t_i - \sum_{i \in N_{01}} t_i \leq m_0 F - (m_0 - 1)(\tau_1 + \tau_2) - \\ &\sum_{i \in N_{01}} [F - (\tau_1 + \tau_2)] = \\ &= m_0 F - (m_0 - 1)(\tau_1 + \tau_2) - k(F - (\tau_1 + \tau_2)) = \\ &= (m_0 - k)F - (m_0 - k - 1)(\tau_1 + \tau_2). \end{aligned}$$

4.2. Алгоритм разбиения множества процессоров

Алгоритм $A_2(k)$ строит всевозможные разбиения множества процессоров $P = \{1, 2, \dots, m\}$ на группы, число которых не менее k . Каждое разбиение множества P представляет собой совокупность множеств $P_1, P_2, \dots, P_r$ ($r \geq k$), таких, что

$$P_i \cap P_j = \emptyset \text{ при } i \neq j, \quad \bigcup_{i=1}^r P_i = P.$$

Поскольку все процессоры идентичные, то данный алгоритм заключается в перечислении всех разбиений натурального числа m на не менее чем k натуральных чисел $m_1, m_2, \dots, m_r$ ($r \geq k$, $m_1 + m_2 + \dots + m_r = M$, $m_1 \leq m_2 \leq \dots \leq m_r$). Для этого можно воспользоваться алгоритмом перечисления всех разбиений числа m [8] и брать только те из них, которые состоят не менее чем из k частей. Можно также воспользоваться алгоритмом

построения всех разбиений числа m ровно с r частями [8] и применять его для всех $r = k, k+1, \dots, m$. Как следует из [8], количество всех разбиений числа m составляет $O((\exp \sqrt{m})/m)$, а при переходе от одного разбиения к другому требуется выполнить $O(m)$ операций. Таким образом, вычислительная сложность алгоритма $A_2(k)$ составляет $O(\exp \sqrt{m})$.

4.3. Алгоритм разбиения множества работ

Алгоритм $A_3(a_1, \dots, a_k)$ определяет, существует ли разбиение множества работ N на k непересекающихся подмножеств $N_1, N_2, \dots, N_k$

$$\left(\bigcup_{j=1}^k N_j = N, \quad N_i \cap N_j = \emptyset \text{ при } i \neq j \right), \text{ таких, что}$$

$$\sum_{i \in N_j} t_i \leq a_j \text{ при } j = 1, 2, \dots, k. \text{ Если такое разбиение}$$

существует, то алгоритм строит все такие разбиения.

Определим $(k+1)$ -мерный массив $C(i_1, \dots, i_k, i_{k+1})$, элементы которого принимают значения 0 или 1 и, кроме того, $0 \leq i_r \leq a_r$, $r = 1, \dots, k$, $0 \leq i_{k+1} \leq n$.

$C(i_1, \dots, i_k, l) = 1$, если существует разбиение множества работ $N^l = \{1, 2, \dots, l\}$ на k непересекающихся подмножеств $(N_i^l \cap N_j^l = \emptyset \text{ при } i \neq j$,

$$\bigcup_{i=1}^k N_i^l = N^l), \text{ таких, что } \sum_{i \in N_j^l} t_i = i_j \text{ при}$$

всех $j = 1, 2, \dots, k$. В противном случае $C(i_1, \dots, i_k, l) = 0$. Элементы массива будем вычислять рекуррентно для $l = 0, 1, \dots, n$ следующим образом.

Полагаем $C(0, \dots, 0, 0) = 1$, $C(i_1, \dots, i_k, 0) = 0$ при всех остальных значениях $i_1, \dots, i_k$. Предположим, что уже вычислены значения элементов $C(i_1, \dots, i_k, l-1)$ при $0 \leq l < n$ и всех значениях $i_1, \dots, i_k$. Вычислим значения элементов $C(i_1, \dots, i_k, l)$. Для всех $i_1, \dots, i_k$ и всех $j = 1, 2, \dots, k$, при которых $C(i_1, \dots, i_k, l-1) = 1$ и $i_j + t_l \leq a_j$, полагаем $C(i_1, \dots, i_j + t_l, \dots, i_k, l) = 1$. Для всех остальных значений $i_1, \dots, i_k$ полагаем $C(i_1, \dots, i_k, l) = 0$. В результате данной процедуры будут вычислены все элементы массива C .

Если найдутся значения $i_1^0, \dots, i_k^0$, при которых $C(i_1^0, \dots, i_k^0, n) = 1$, то искомое разбиение существует и может быть построено следующим образом. Определим $N_1 = N_2 = \dots = N_k = \emptyset$. Для $l = n, n-1, \dots, 1$ выполним последовательно сле-

дующую процедуру. Пусть $C(i_1^0, \dots, i_k^0, l) = 1$. Выберем такое j ($1 \leq j \leq k$), что $C(i_1^0, \dots, i_{j-1}^0, i_j^0 - t_l, i_{j+1}^0, \dots, i_k^0, l-1) = 1$. В силу построения массива C такое j существует. Полагаем $N_j = N_j \cup \{l\}$. В результате выполнения описанной процедуры можно построить все искомые разбиения. Вычислительная сложность построения массива C и всех разбиений составляет $O\left[nk \prod_{r=1}^k a_r\right]$.

5. Алгоритм решения исходной задачи

Перейдем к описанию алгоритма решения исходной задачи.

1. Вычислить величину

$$\alpha = \sum_{i \in N_0} t_i + (m-1)(\tau_1 + \tau_2).$$

2. Если $\alpha \leq mF$ (т.е. выполнено условие (3) для $N_0 = N, m_0 = m$), то применить алгоритм $A_1(N, m)$. На этом алгоритм завершает работу.

3. Если $\alpha > mF$ (т.е. условие (3) для $N_0 = N, m_0 = m$ не выполнено), то число прерываний следует сократить по меньшей мере до $m-k$, где $k = \lceil (\alpha - mF) / (\tau_1 + \tau_2) \rceil + 1$.

Для каждого разбиения множества процессоров P на подмножества $P_1, P_2, \dots, P_r$ (которое соответствует разбиению числа m : $m = m_1 + m_2 + \dots + m_r$, $m_j = |P_j|$, $|P_j|$ – число процессоров в P_j , $j = 1, \dots, r$), построенного с помощью алгоритма $A_2(k)$, применить алгоритм $A_3(a_1, \dots, a_r)$ при $\alpha_j = m_j F - (m_j - 1)(\tau_1 + \tau_2)$, $j = 1, \dots, r$.

4. Если в результате работы алгоритма A_3 , примененного для некоторого разбиения множества P на подмножества $P_1, P_2, \dots, P_r$, будет построено разбиение множества N на подмножества $N_1, N_2, \dots, N_r$ (т.е. для каждого N_j и m_j , $j = 1, \dots, r$, выполнено условие (3)), то применить алгоритм $A_1(N_j, m_j)$ для каждого $j = 1, \dots, r$.

Тем самым будет построено расписание выполнения работ для каждого из множеств N_j на m_j процессорах, $j = 1, \dots, r$. Эти расписания определяют допустимое расписание выполнения множества работ N на m процессорах. Если для каждого разбиения множества P соответствующего разбиения множества N не существует (т.е. алгоритм A_3 не находит его), то допустимого расписания в исходной задаче не существует.

Зная оценку сложности алгоритмов A_1, A_2, A_3 , получаем, что вычислительная сложность алгоритма решения исходной задачи составляет

$$O\left[(\exp \sqrt{m})nr \prod_{r=1}^k a_r + n\right] \text{ или}$$

$$O\left[(\exp \sqrt{m})nm \prod_{j=1}^m (Fm_j - (m_j - 1)(\tau_1 + \tau_2))\right] \text{ или}$$

$$O\left[(\exp \sqrt{m})nm \prod_{j=1}^m (Fm - (m - 1)(\tau_1 + \tau_2))\right] \text{ или}$$

$$O[(\exp \sqrt{m})nm^{m+1}(F - (\tau_1 + \tau_2))^m].$$

В случае, когда число процессоров фиксировано или ограничено сверху, алгоритм является псевдополиномиальным, а от числа работ имеет линейную оценку сложности.

Calculation planning in multiprocessor system with cost accounting for interrupts

M.G. Furugyan

Abstract: The problem of deadline scheduling in multiprocessor system with identical directive intervals and with cost accounting for interrupts is studied. The algorithm of construction of feasible schedule was elaborated. This algorithm is pseudopolynomial and has linear complexity depending on number of jobs when number of processors is fixed.

Keywords: calculation planning, multiprocessor system, estimation of complexity

Литература

1. *В.С. Танаев, В.С. Гордон., Я.М. Шафранский.* Теория расписаний. Одностадийные системы. – М.: Наука, 1984.
2. *T. Gonzales., S. Sahni* Preemptive scheduling of uniform processor systems//Journal of the Association for Computing Machinery. – 1978, 25, 1. Pp. 92 – 101.
3. *A. Federgruen, H. Groenevelt.* “Preemptive Scheduling of Uniform Machines by Ordinary Network Flow Technique”. //Management Sci. – 1986, V. 32, No.3, March . PP. 812 – 829.
4. *P. Bruker* Scheduling Algorithms. – Springer, 2007.
5. *Д.С. Гуз, М.Г. Фуругян.* Планирование вычислений в многопроцессорных АСУ реального времени с ограничениями на память процессоров.// Автоматика и телемеханика, 2005, №2, с. 138-147.
6. *М. Гэри, Д. Джонсон.* Вычислительные машины и труднорешаемые задачи. – М.: Мир, 1982.
7. *Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн.* Алгоритмы. Построение и анализ. М.: Вильямс, 2005.
8. *Г. Эндрюс.* Теория разбиений. М.: Наука, 1982.

Об использовании элементов нечёткой логики в оценке алгоритмов идентификации лиц

А.С. Осипов

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: osipa@niisi.ras.ru

Аннотация: В статье рассматривается ряд вопросов, связанных с исследованием производительности алгоритмов распознавания лиц, в рамках разработанного в НИИСИ РАН эмпирического подхода к оценке алгоритмов компьютерного зрения. В частности, изучается возможность применения эталонных изображений и мер оценки качества, созданных с использованием элементов теории нечётких множеств. При этом одному тестовому изображению может соответствовать несколько нечётких эталонов, разработанных с целью тестирования различных свойств алгоритма. Приводятся примеры, демонстрирующие содержательность данного подхода.

Ключевые слова: оценка производительности, анализ изображений, распознавание лиц, ground truth образы, теория нечётких множеств.

Введение

В последние десятилетия, распознавание и идентификация лиц представляет собой одну из основных задач компьютерного зрения. Развитие систем видеонаблюдения, систем персонального доступа, биометрических технологий (в настоящее время системы на базе технологии распознавания лица - на втором месте по доле рынка после дактилоскопических считывателей [1]), социальных сетей способствует поддержанию ее актуальности. Здесь под задачей идентификации лиц (face recognition) мы будем понимать следующее: дана база изображений (фотографий или видеок кадров) конкретных людей, состоящая из конечного набора классов. Каждый класс изображений представляет собой изображения одного человека. По предъявлении входного изображения человеческого лица требуется определить его принадлежность одному из данных классов (или установить отсутствие такой принадлежности).

К настоящему времени предложены разнообразные алгоритмы решения данной задачи, составляющие основу конкурирующих между собой программных продуктов в области видео-идентификации. При этом стоимость таких продуктов может быть весьма высока. Например, система распознавания лиц ForensicaGPS [2], основанная на преобразовании двумерной фотографии или видеок кадра в 3D образ, была приобретена спецслужбами Саудовской Аравии и, по некоторым данным, используется в создании программного обеспечения «высокотехнологичной ограды» на границе с Ираком; при этом общая стоимость создания такого ограждения оценивается в 3.4 млрд. евро. Вместе с тем, зачастую надёжность предлагаемых программных продуктов для решения конкретной практической задачи недостаточно высока. Это делает проблему сравнительной оценки данных программ с целью выявления лучших (для конкретного практического применения) весьма острой [3].

Оценка качества работы различных компьютерных программ, решающих некоторую практическую задачу, представляет собой процесс, не имеющий единой методики. Основные отличия методик, применяемых в сравнительных исследованиях различных программ, заключаются в следующем:

- тип критерия оценки качества (количественный или качественный, использующий эталоны или нет);
- тип эталонов и тестовых материалов (реальные или синтезированные), их параметры, количество, источники (оригинальные или общедоступные) и т.п.

Используемая в НИИСИ РАН методика строит оценку качества на основе вычисления некоторой количественной меры различия, оценивающей результат работы программы на некотором наборе тестовых изображений, для которых эталонное решение, так называемое ground truth, известно априори. Такой подход к оценке качества программных продуктов в англоязычной литературе называется discrepancy method, что определило выбор названия нашей методики: EDEM (Empirical Discrepancy Evaluation Method) [3]. Характерной чертой методики является внесение в тестовые изображения контролируемых искажений: (с известными параметрами) и различных преобразований объектов на изображении. Данная методика находится в процессе совершенствования, как в плане выбора и построения тестовых материалов и соответствующих эталонов, так и в плане анализа и сочетания разных количественных мер различия. Инструментальной средой поддержки методики EDEM является программная система PICASSO (основные особенности ее последней версии приведены в [3]). Отметим здесь, что предназначение PICASSO - предоставить возможность пользователю создавать, хранить и редактировать тестовые и эталонные изображения, вносить контролируемые возмущения в тестовый материал, использовать различные меры

близости, и проводить сравнительное тестирование различных программных средств в интерактивном режиме и отображать результаты тестирования.

В настоящее время разработано большое число алгоритмов обработки и анализа изображений, в той или иной степени использующих элементы теории нечётких множеств [4]. Во многих ситуациях использование таких алгоритмов представляется вполне оправданным (применительно к алгоритмам обработки изображений, данный вопрос рассматривался в [5]). Что касается задачи распознавания лиц, то, например, большая вариабельность распознаваемых объектов свидетельствует в пользу применения подобных алгоритмов. Современные методы сравнительного исследования алгоритмов анализа изображений должны как уметь работать с подобными алгоритмами, так и включать в себя элементы нечёткой логики.

1. Об алгоритмах распознавания лиц

Как отмечалось выше, актуальность задачи распознавания лиц привела к появлению разнообразных алгоритмов ее решения. Здесь мы остановимся лишь на некоторых из них.

Как правило, образы базы изображений представляют собой обучающую выборку, при помощи которой определяются настроечные параметры алгоритма. Затем, в качестве первичной верификации, к распознаванию предъявляется набор изображений тех же людей, чьи изображения содержатся в базе. Этот набор изображений составляет тестовую выборку.

Алгоритмы распознавания лиц обычно реализованы по следующей схеме. На первом этапе решения данной задачи, для каждого изображения обучающей выборки составляется набор (вектор) его характерных признаков. На следующем этапе, используя векторы характерных признаков, с помощью алгоритмов машинного обучения строится модель - классификатор, эффективно разделяющая между собой наборы признаков, соответствующие изображениям разных лиц. Этот этап завершает первичную настройку алгоритма. Далее, для тестового изображения составляется вектор характерных признаков, который подаётся на вход классификатора, сравнивается с его содержимым, после чего делается заключение о принадлежности входного вектора одному из классов обучающей выборки.

Пожалуй, наиболее простым методом классификации изображений является метод поиска ближайшего соседа в пространстве образов. Каждый образ тестового набора представляет собой элемент в пространстве образов и ищется ближайшее расстояние между ним и элементами обучающего набора. Элемент тестового набора относят к тому же классу, что и элемент обучающего набора, на котором достигается это расстояние. Если все изображения (значения яркости их пикселей) нормализованы с нулевым средним и единичной дисперсией, то тогда данная процедура эквивалентна нахождению образа из обучающей выборки, наиболее коррелирующего с

рассматриваемым тестовым образом, поэтому данный метод называют также методом корреляции [6]. Поскольку данные нормализованы, результат не зависит от освещённости изображения и настроек видеокamеры. У данного метода есть ряд известных недостатков. Во-первых, если освещённость объектов тестовой выборки и обучающей выборки сильно меняются, то соответствующие точки в пространстве образов расположены недостаточно плотно (для эффективного распознавания требуется, чтобы образы, соответствующие одному классу, располагались плотно, в кластерах, и при этом чтобы кластеры, соответствующие разным классам, были разделены между собой). Так что для удовлетворительной работы метода в условиях различной освещённости объектов требуется большое количество тестовых образов для каждого уровня освещённости. Это, в частности, требует создания и хранения большой базы изображений. Кроме того, сама по себе процедура вычисления наилучшей корреляции является вычислительно затратной.

Одним из естественных путей уменьшения вычислительных затрат является сокращение размерности у векторов характерных признаков изображений лиц. Здесь оказалось продуктивным использование метода главных компонент (англ. PCA), поскольку основной целью этого метода, первоначально использовавшегося в статистических вычислениях, было уменьшить размерность данных, потеряв при этом наименьшее количество информации. Это можно сформулировать как проблему поиска подпространств меньшей размерности, в ортогональной проекции на которые разброс данных (то есть среднеквадратичное отклонение от среднего значения) максимален.

Применительно к задаче распознавания лиц, здесь возможна следующая реализация. Пусть имеется набор S векторов-столбцов обучающей выборки: $S = [x_1, \dots, x_p]$. Каждый вектор имеет длину n и соответствует одному из образов базы изображений (изначально такой вектор строился из значений яркости всех пикселей данного образа, в результате получались вектора большой длины, при этом исчезала часть информации о структуре исходного двумерного изображения). Каждый вектор этой выборки относится к одному из классов: $C_1, \dots, C_N$. Вычисляется матрица разброса данных обучающей выборки (эмпирическая ковариационная матрица):

$$\Sigma_S = \sum_{i=1}^p (x_i - \mu_S) * (x_i - \mu_S)^T,$$

где μ_S - среднее выборки (вектор), а T означает транспонирование. В результате получается матрица $n \times n$. Далее, при $m < n$ ищется матрица W_S размера $n \times m$, реализующая ортогональное проектирование на подпространство размерности m , где разброс данных максимален:

$$W_S = \max_W \det(W^T \Sigma_S W), \quad W \in \mathbb{R}^{n \times m},$$

$$W W^T = E \in \mathbb{R}^{n \times m}.$$

где E - единичная матрица. Легко видеть, что решением этой задачи будет матрица $W_S = [w_1, \dots, w_m]$, составленная из собственных векторов, отвечающих m наибольшим собственным значениям (расположенным в порядке убывания). Отметим, что размерность векторов-столбцов матрицы равна n и совпадает с размерностью векторов обучающей выборки S , представляющие элементы базы изображения лиц. По этой причине, в англоязычной литературе, по аналогии с собственными векторами (eigenvectors) данные векторы стали называть eigenfaces [7], давшие название всему методу распознавания лиц: Eigenface.

Далее элементы выборки S преобразуются в векторы $[y_1, \dots, y_p]$ m - мерного признакового пространства посредством матрицы W_S :

$$y_i = W_S^T x_i, \quad i = 1, \dots, p.$$

В дальнейшем тестовые n - мерные изображения преобразуются в элементы m - мерного признакового пространства посредством той же матрицы W_S , где в этом пространстве находятся ближайшие к ним соседи (в терминах некоторой метрики) из выборки $[y_1, \dots, y_p]$. В настоящее время существует ряд модификаций метода Eigenface. В частности, вместо преобразования изображения в одномерный вектор (некоторые недостатки которого приведены выше) используется непосредственно двумерная матрица изображений (в некоторых источниках эту модификацию называют двумерным PCA методом [8]). Это даёт выигрыш в вычислении матрицы разброса, но накладывает дополнительные ограничения на размерность признакового пространства.

Недостаток данного подхода заключается в том, что при реализации максимального разброса данных $[y_1, \dots, y_p]$ обучающей выборки данные, отвечающие одному классу, могут располагаться недостаточно плотно (т. е. кластеризация этих данных может отсутствовать). Для решения этой проблемы был разработан более тонкий метод построения матрицы W_S , реализующей проектирование исходных данных на признаковое пространство. Именно, по обучающей выборке $S = [x_1, \dots, x_p]$ с элементами из классов $C_1, \dots, C_N$ строится матрица разброса данных между классами:

$$\Sigma_S = \sum_{i=1}^N n_i (\mu_i - \mu) * (\mu_i - \mu)^T,$$

где μ_i - среднее i - го класса, n_i - число элементов в нём, μ - среднее всей выборки. Данная матрица размера $n \times n$ будет иметь ранг не превосходящий $N-1$. Строится также матрица разброса внутри классов:

$$\Sigma_I = \sum_{i=1}^N \sum_{x_j \in C_i} (x_j - \mu_i) * (x_j - \mu_i)^T.$$

Искомая матрица W_S ищется методами линейного дискриминантного анализа как максимум линейного дискриминанта Фишера:

$$W_S = \max_W \frac{\det(W^T \Sigma_S W)}{\det(W^T \Sigma_I W)}, \quad W \in \mathbb{R}^{n \times m},$$

$$W W^T = E \in \mathbb{R}^{n \times n}.$$

Числитель в правой части данной формулы «отвечает» за максимизацию разброса данных между выборками, а знаменатель - за минимизацию разброса внутри выборок. Решением данной задачи будет матрица, $W_S = [w_1, \dots, w_m]$, составленная из обобщённых собственных векторов:

$$\Sigma_S w_i = \lambda_i \Sigma_I w_i, \quad i = 1, \dots, m.$$

отвечающим наибольшим обобщённым собственным значениям λ_i , при этом в силу того, что ранг Σ_S не превосходит $N-1$, существует максимум $N-1$ таких векторов. Таким образом, в данном методе размерность m не превосходит $N-1$, т. е. она всегда меньше числа классов. Отметим, что матрица Σ_I также сингулярна (ее ранг не превосходит $p-N$). Для преодоления этой трудности в [9] вначале данные входной выборки преобразуются в признаковое пространство размерности $p-N$ посредством метода главных компонент (как в методе Eigenface), а затем в этом пространстве ищется матрица, реализующая максимум дискриминанта Фишера в соответствии с приведённой выше формулой. Данный метод получил название - метод фишеровских лиц (Fisherface method). Различные вариации этого метода используются и в настоящее время [10].

Отметим, что во всех рассмотренных выше методах расположение в признаковом пространстве элементов обучающей выборки оказывает основное влияние на качество дальнейшего распознавания. При съёмках объектов при различном освещении разброс элементов выборки может быть высок. В этой связи, в [9] рассматривалась возможность сокращения размерности m признакового пространства путём отбрасывания у его элементов трёх первых компонент (предполагалось, что первые три компоненты более других зависят от изменений яркости образа). Однако это предположение было признано нецелесообразным (при отбрасывании могла быть потеряна часть информации, существенной для идентификации объекта).

Что касается классификации тестовых объектов, весьма распространённым является метод k ближайших соседей (k -nearest neighbors algorithm) в соответствии с которым объект относят тому классу, который является наиболее распространённым среди соседей данного элемента. В качестве определения расстояния между объектами может использоваться Евклидова метрика. В задаче распознавания лиц используется и нечёткая версия данного алгоритма. В данной версии вначале для всех элементов обучающей выборки определяется степень их принадлежности классам базы изображений (чем больше из k ближайших соседей принадлежат одному классу, тем степень принадлежности этому классу для данного элемента выше). Затем для элемента, соответствующего тестовому изображению

вычисляется степень его принадлежности к каждому из классов. При этих вычислениях используются степени принадлежности k ближайших соседей из обучающей выборки и расстояния до них [11-12]. Эксперименты, проведенные с элементами нескольких известных баз изображений лиц, показали более высокий процент правильного распознавания нечёткой версии метода k ближайших соседей по сравнению с классической версией данного метода [12].

Что касается вектора признаков, соответствующих изображениям, в настоящее время как правило используются признаки, полученные в результате применения к изображению различных алгоритмов обработки. Например, в качестве таких признаков могут выступать признаки, полученные при обработке изображения фильтрами Габора (Gabor phase congruency feature vector [10]). Также в распознавании лиц в качестве набора признаков используется гистограмма направленных градиентов (англ. Histogram of Oriented Gradients, HOG), характеризующая особые точки изображения. Она может быть произведена путем разделения изображения на маленькие связные области, именуемые ячейками, и расчетом для каждой ячейки гистограммы направлений градиентов или направлений краев для пикселей, находящихся внутри ячейки. Комбинация этих гистограмм и является искомым дескриптором изображения. Достоинствами этой гистограммы являются ее инвариантность относительно геометрических и фотометрических преобразований (поскольку элементы HOG строятся локально). Пример изображения лица и соответствующей гистограммы ориентированных градиентов, (построенной средствами MATLAB Computer Vision System Toolbox Version 7.0) изображен на Рис. 1.

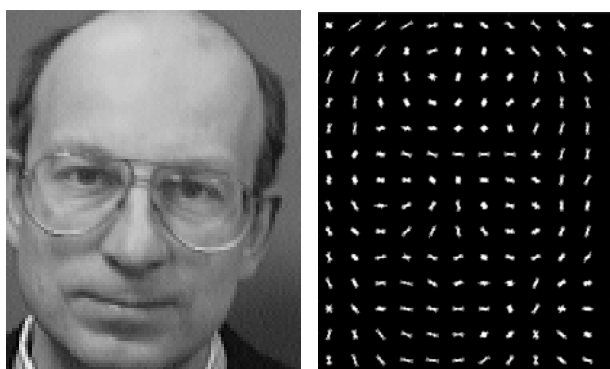


Рис.1 - Изображение лица и его гистограмма направленных градиентов.

В качестве признаков используются также коэффициенты преобразования Фурье матрицы изображения (их «значимая» часть), коэффициенты дискретных косинус и вейвлет - преобразований.

Следует отметить и нейросетевые алгоритмы, используемые в решении задачи распознавание лиц (как и в других задачах классификации). Например, в работе [13] используется признаковый набор, составляющий часть коэффициентов косинус - преобразования исходного изображения обучающей

выборки. По данной выборке происходит настройка нейронной сети методом обратного распространения ошибки. При этом сравнительно высокая производительность алгоритма обеспечивается при небольшом числе весовых коэффициентов сети. Вместе с тем, до настоящего времени активному распространению нейросетевых алгоритмов для распознавания лиц препятствовали их известные недостатки: эвристичность подходов для проектирования нейронных сетей, трудности в подготовке репрезентативной обучающей выборки, возможность тупиковых ситуаций в обучении нейронных сетей и т. п.

2. О нечётких методах оценки

Первоначально использование нечёткой логики использовалось нами при оценке алгоритмов обработки изображений. Так, в [14] было предложено применить нечёткие ground truth эталоны и нечёткие меры различия к оценке производительности детекторов границ. Введение нечётких ground truth эталонов преследовало цель не только использовать их для тестирования нечётких детекторов границ (которые могут присваивать каждому обработанному пикселю значения степени принадлежности нескольким классам: классам граничных пикселей, фону и шуму). Преследовалась также цель сделать процесс тестирования обычных «чётких» детекторов границ более глубоким. Именно, характерная особенность предложенного подхода заключалась в том, что одному тестовому изображению могло соответствовать несколько нечётких ground truth эталонов, при этом каждый из них предназначен для выявления некоторой особенности тестируемого алгоритма. Например, один нечёткий ground truth образ лучше определяет способность детектора границ выявлять слабоконтрастные края, в то время как другой образ более приспособлен для проверки детектора выделять непрерывные границы. В [3] рассматривалась возможность использования данного подхода при сравнительном исследовании алгоритмов сегментации изображений.

Применительно к оценке производительности алгоритмов распознавания лиц, в настоящее время используются стандартные статистические метрики: общий процент правильно распознанных изображений, проценты неправильно отклонённых («ложная тревога») и неправильно принятых («ложное срабатывание») применительно к конкретным классам, производные от этих метрик и т. п. (см., например, [10]).

Остановимся подробнее на этих концепциях нечёткой логики применительно к рассматриваемой в данной статье задаче оценки качества алгоритмов распознавания лиц. Прежде всего, напомним несколько основных понятий из теории нечётких множеств.

Именно, пусть X есть непустое множество (в нашем случае это множество изображений лиц некоторой выборки). Нечёткое множество C на X есть пара $\langle X, f_C \rangle$, где f_C есть отображение X на $[0, 1]$.

Значение $f_C(x)$ для элемента (изображения) $x \in X$ называется степенью принадлежности x множеству C , (в нашем случае это степень принадлежности некоторому классу C базы изображений) а функция f_C называется функцией принадлежности нечёткого множества.

Обозначим множество всех нечётких множеств на X за $[0, 1]^X$. Для A и B из $[0, 1]^X$ нечёткое отношение включения

$A \subset B$ означает что $f_A(x) \leq f_B(x)$ для всех $x \in X$.

Нечёткая мера сходства есть отображение

$s: [0, 1]^X \times [0, 1]^X \rightarrow [0, 1]$, сопоставляющее множествам $A, B \in [0, 1]^X$ степень сходства

$s(A, B) \in [0, 1]$, удовлетворяющее условиям:

- $s(A, A) = 1$ для любого нечёткого A
- $s(A, B) = s(B, A)$ для всех нечётких A и B (коммутативность)
- $s(A, C) \leq s(A, B) \wedge s(B, C)$ при $A \subset B \subset C$,

где $p \wedge q$ обозначает минимум из p и q ; максимум из p и q обозначается как $p \vee q$. Важными примерами данных мер являются:

$$S_2(A, B) = \frac{\sum_{x \in X} 2(f_A(x) \cdot f_B(x))}{\sum_{x \in X} f_A(x) + f_B(x)}.$$

Заметим, что обычные (чёткие) подмножества M из X включаются в данный подход, если мы будем рассматривать их как стандартные характеристические функции $1_M: X \rightarrow [0, 1]$. То есть, если у нас имеется изображение x , относящееся к некоторому классу C , то в этом случае $f_C(x) = 1$ и $f_{C'}(x) = 0$ для всех классов C' отличных от C . Как показано в [15], для чётких A или B отождествлённых со своими характеристическими функциями, s_1 и s_2 совпадают с мерами оценки качества классификации Шорта и Хеллдена соответственно (Short's and Hellden's classification accuracy measures). Изначально эти меры определялись только для чётких множеств, так что можно рассматривать меры s_1 и s_2 как их обобщение.

В нашей ситуации каждое изображение x может принадлежать любому из N классов (фактически, мы имеем N - мерную функцию принадлежности). Пусть у нас имеются две нечётких классификации, соответствующие одному и тому же набору классов:

$$F^1 = \langle X, f_{C_1}^1, \dots, f_{C_N}^1 \rangle, F^2 = \langle X, f_{C_1}^2, \dots, f_{C_N}^2 \rangle.$$

Для этого случая в [15] введены итоговые меры сходства (overall accuracy measures), сходные с S_1 и S_2 :

$$OA_1(F^1, F^2) = \frac{\sum_i \sum_{x \in X} f_{C_i}^1(x) f_{C_i}^2(x)}{\sum_i \sum_{x \in X} f_{C_i}^1(x) + f_{C_i}^2(x)},$$

$$OA_2(F^1, F^2) = \frac{\sum_i \sum_{x \in X} 2(f_{C_i}^1(x) f_{C_i}^2(x))}{\sum_i \sum_{x \in X} f_{C_i}^1(x) + f_{C_i}^2(x)}.$$

Перечисленные выше нечёткие метрики реализованы в текущей версии программной системы

PICASSO. Ранее они использовались в сравнительном тестировании детекторов границ [14].

Под нечёткими ground truth образами здесь мы будем понимать совокупность F^1 , где $f_{C_1}^1, \dots, f_{C_N}^1$ - определённые для каждого $x \in X$ функции принадлежности классам $C_1, \dots, C_N$. Например, в качестве F^1 здесь рассматриваются априори определённые функции принадлежности классам $C_1, \dots, C_N$ для тестовой выборки, а под F^2 - результат распознавания, когда для x $f_{C_i}^2(x) = 1$, если по результатам распознавания x отнесли к классу C_i и $f_{C_j}^2(x) = 0$ для j отличных от i .

Примеры задания таких образов мы рассмотрим в следующем разделе.

3. Эксперименты

Для экспериментов нами была выбрана известная база ORL, созданная лабораторией AT&T в период между 1992 и 1994 гг. Она включает 400 образов, содержащих изображения лиц 40 людей (по 10 изображений каждого человека) [16]. Все изображения в полутоновом (grayscale) режиме размера 112×92 пикселей. Изображения сделаны при незначительной вариации освещённости и отличаются выражением лица, поворотами головы и другими деталями (например, в очках и без очков).

Также нами был создан ряд новых изображений на основе образов базы ORL. Для этого использовалась свободно распространяемая программа морфинга изображений Squirrelz Morph версии 2.1. Она позволяет по двум изображениям лица осуществлять преобразование начального изображения лица в конечное с сохранением промежуточных результатов. При этом результаты сохраняются как видеофайл, анимация, или набор отдельных видеок кадров. Тем самым, удалось существенно расширить набор экспериментальных данных.

В качестве тестируемых алгоритмов распознавания лиц в основном тестировались демонстрационные версии программ, доступные на сайте <http://www.advancedsourcecode.com/> посвящённом программному обеспечению в области компьютерного зрения и биометрии. Все алгоритмы реализованы в среде MATLAB, тестирование проводилось с использованием упоминавшейся выше программной системы PICASSO.

Перед началом распознавания каждому элементу распознаваемой выборки (куда могли входить и элементы обучающей выборки алгоритма) предписывались степени принадлежности каждому из распознаваемых классов (т. е. для каждого $x \in X$ определялись величины $f_{C_1}^1(x), \dots, f_{C_N}^1(x)$). Далее по завершении тестирования вычислялись меры OA_1 и OA_2 по приведённым выше формулам, а также доля правильно распознанных изображений (обычная статистическая мера). При этом был получен ряд содержательных результатов.

В качестве примера приведём сравнение между собой программных реализаций двух алгоритмов: упоминавшегося выше алгоритма Eigenface и

алгоритма, где в качестве признаков образа используются локальные бинарные шаблоны (англ. local binary patterns) называемым для краткости lbp – алгоритмом. Использовалась выборка, частично состоящая из исходных изображений базы ORL и частично из изображений, полученных из анфасных изображений лиц из ORL путём поворота головы вправо и влево на различные углы. Данные повернутые изображения были созданы при помощи Sqirlz Morph, пример приведён на Рис. 2.



Рис. 2 – Исходное изображение базы ORL, его повороты вправо на 15°, 10° и влево на 25°.

В качестве обучающей выборки было взято 30 изображений из ORL, представляющих 10 человек (по 3 обучающих образа на человека). В качестве тестовой выборки были взяты изображения одного человека с Рис. 2 : 8 восемь поворотов головы вправо и влево от 15° до 30° с интервалом в 5°, и 7 оставшихся исходных изображений из ORL, итого 15 изображений. После процедуры тестирования доля правильно распознанных изображений составила 0.66 для алгоритма Eigenface и 0.73 для lbp-алгоритма (достаточно адекватные величины, учитывая небольшой объём обучающей выборки). То есть общая производительность lbp-алгоритма оказалась выше. Одновременно с этим, элементам тестовой выборки были заданы степени принадлежности своему классу, равные 1 для поворотов головы, 0.8 - для остальных образов, а степени принадлежности другим классам были равны нулю. При этом были получены следующие результаты вычисления нечётких мер OA_1 и OA_2 : для алгоритма Eigenface $OA_1 = 0.66$, $OA_2 = 0.8$ и для lbp - алгоритма $OA_1 = 0.64$ и $OA_2 = 0.78$. Таким образом, здесь предпочтение было отдано алгоритму Eigenface. Непосредственная проверка показала, что алгоритм Eigenface правильно распознал 7 изображений поворота головы из 8, в то время как lbp - алгоритм - только 4. Тем самым, на основании всего тестирования выработки можно сделать вывод, что несмотря на несколько более высокую производительность lbp - алгоритма, алгоритм Eigenface проявляет большую устойчивость распознавания при небольших поворотах головы распознаваемого объекта. Тестирование на выборках большего объёма показало примерно одинаковую производительность этих алгоритмов, при этом алгоритм Eigenface продемонстрировал большую устойчивость распознавания при поворотах головы до 30°.

По аналогичной схеме тестировалась способность алгоритмов распознавать изображения лица в очках и без очков. Некоторые классы изображений ORL (в том числе класс, соответствующий Рис. 2), содержат изображения обоих видов. Использовалась та же

обучающая выборка, что и в предыдущем примере – 30 изображений, по 3 обучающих изображения для каждого класса. Здесь результаты алгоритма Fisherface оказались выше, чем у Eigenface и lbp - алгоритма (хотя общая производительность последнего в ряде тестов была выше), что согласуется с результатами [9].

Как отмечалось выше, для эффективного функционирования алгоритма распознавания лиц большое значение имеет расположение элементов обучающей выборки в признаковом пространстве, в частности, чтобы представители разных классов были разделены между собой. Для проверки этого свойства, при помощи Sqirlz Morph из пар изображений, представляющих разные классы, были созданы наборы гибридных изображений. Здесь для каждого гибридного изображения естественно задавать его степени принадлежности обоим родительским классам: чем ближе изображение к одному из этих классов, тем степень принадлежности к этому классу выше, и наоборот, тем ниже степень принадлежности ко второму родительскому классу. Примеры родительских и гибридных изображений приведены на Рис. 3.

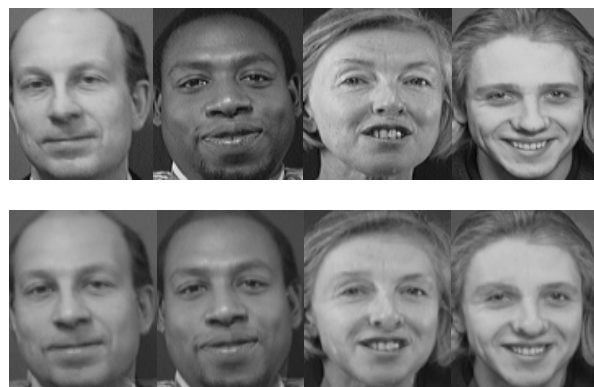


Рис. 3: Верхний ряд - исходные изображения из базы ORL, Нижний ряд - гибридные изображения. Степени принадлежности родительским классам для гибридных изображений - (0.75,0.25) - для первого и третьего гибридного изображения, (0.25,0.75) - для второго и четвертого.

В наших экспериментах мы использовали ту же обучающую выборку, что и в предыдущем примере (30 изображений, 10 классов). Затем из некоторых пар обучающей выборки делались наборы гибридных изображений, из которых выбирались для тестирования по 72 изображений со степенями принадлежности родительским классам (0.9,0.1),..., (0.55,0.45), (0.45,0.55),..., (0.1,0.9). Степень принадлежности остальным 8 классам для каждого набора полагалась равной нулю. В процессе тестирования вычислялись значения мер OA_1 и OA_2 . В частности, для алгоритма Fisherface, применённого к набору гибридных изображений, соответствующих правой родительской паре на Рис. 3, они составили: $OA_1 = 0.57$, $OA_2 = 0.73$, в то время как для lbp – алгоритма они составили, соответственно, 0.44 и 0.64. При непосредственной проверке оказалось, что алгоритм Fisherface отнёс каждое изображение из набора к тому родительскому классу, степень

принадлежности к которому у изображения выше. Таким образом, разделение гибридного набора было выполнено корректно. Иная ситуация оказалась у lbp – алгоритма: из 72 изображений лишь 48 были отнесены к родительскому классу с преобладающей степенью принадлежности, 4 изображения – к родительскому классу с меньшей степенью принадлежности, а остальные 20 – к другим классам. По результатам тестирования всех наборов можно сделать вывод, что при лучшем разделении гибридного набора значения мер OA_1 и OA_2 были выше, при этом, для качественного тестирования взаимного расположения элементов обучающей выборки в признаковом пространстве потребуется разработка новых нечётких мер сходства.

5. Выводы

Проведённые эксперименты показали, что нечёткие меры сходства, применённые в сочетании со стандартными статистическими оценками производительности, позволяют получить дополнительную информацию о тестируемых алгоритмах распознавания лиц. Сочетание мер разных классов соответствует нашему эмпирическому подходу к оценке алгоритмов компьютерного зрения [3], [5]. В частности, при тестировании детекторов границ нами использовались статистические меры, оценивающие общую производительность алгоритма, в сочетании с мерами оценки качества локализации, позволяющими проводить качественную оценку алгоритма [5]. Применительно к задаче распознавания лиц нечёткие меры могут играть роль средств качественной оценки. При этом различия в оценке лучшего алгоритма в терминах статистических и нечётких мер сходства (как в приведённом в предыдущем разделе примере) играют взаимно дополняющую роль.

Сравнивая алгоритмы распознавания лиц, использующие элементы нечёткой логики с нечёткими алгоритмами выделения границ можно заметить, что последние могут выдавать для каждого пикселя

обрабатываемого изображения несколько значений – степеней его принадлежности граничным классам, фону и шуму. В отличие от нечетких детекторов границ, выдача у алгоритмов распознавания лиц всегда однозначна. Представляется перспективным использование предложенного подхода к исследованию нечётких алгоритмов распознавания лиц на стадии разработки. Обычно однозначной выдаче нечёткого алгоритма (определённой по пороговым значениям) предшествует определение степеней принадлежности тестового изображения классам, из которых состоит база лиц. На этой стадии, возможно использование предлагаемого подхода (путём сравнения априорных степеней принадлежности со степенями принадлежности, определёнными алгоритмом) с целью улучшения качества работы алгоритма.

Что касается нечётких ground truth образов, в настоящее время не существует детально разработанной процедуры их создания. В наших экспериментах, при определении для тестовых изображений априорных степеней принадлежности классам базы образов мы руководствовались эвристическими интуитивно понятными соображениями. В то же время, развитие методологии распознавания лиц позволяет, на наш взгляд, приступить к формализации процесса разработки таких образов. В частности, в задаче распознавания лиц есть важная подзадача – задача распознавания черт лица (face features recognition), и здесь имеются предпосылки к созданию такой формализации. Разработка процедуры создания нечётких ground truth образов в сочетании с созданием нечетких мер сходства, чувствительных к той или иной особенности тестируемых алгоритмов составляют основные направления наших дальнейших исследований.

Работа выполнена в рамках НИР № 0065-2014-0012 («Создание системы, позволяющей получить объективную оценку качества работы разнородных алгоритмов решения прикладных задач»). Частично поддержана РФФИ, проект № 14-07-00502.

On the use of fuzzy logic in evaluation of the face detection algorithms

A.S. Osipov

Abstract: In the paper some issues related to the performance evaluation of the face detection algorithms within an empirical approach to evaluation of computer vision algorithms developed by SRISA RAS, is considered. In particular, the possibility to use for this purpose the ground truth images and performance measures designed with the use of fuzzy set theory is studied. Under this approach, several ground truth images designed to test various properties of the algorithm been studied, may correspond to a single test image. Some practical examples which demonstrate the utility of this approach are given.

Keywords: performance evaluation, image analyses, face recognition, ground truth images, fuzzy sets.

Литература

1. О.Е. Федосеева. Биометрия: взгляд в будущее. Системы безопасности, 2008, № 6 с. 126-128.
2. Facial Recognition Evolves to 3D, ForensicaGPS Unveiled. Доступно по адресу: <http://docslide.us/documents/face-recognition-55844ae19e4c4.html>
3. А.В. Захаров, П.П. Кольцов, Н.В. Котович, А.С. Куцаев, А.А. Кравченко, А.С. Осипов. О технологии сравнительного анализа программно-технических решений в области обработки визуальной информации в интересах ОАО РЖД. Сборник трудов конференции ИСУЖТ-2014, Т. 2. ОАО «НИИАС», 2015, с. 187-192.
4. Bezdek, J.C., Keller, J., Krisnapuram, R., Pal, N. Fuzzy models and algorithms for pattern recognition and image processing. The handbooks of fuzzy set series. vol. 4, 2006, Springer Science & Business Media.
5. А.В. Захаров, П.П. Кольцов, Н.В. Котович, А.А. Кравченко, А.С. Куцаев, А.С. Осипов, Некоторые методы сравнительного исследования детекторов границ. Труды НИИСИ РАН, 2012, том 2, № 1, стр.4-14.
6. R. Brunelli, T. Poggio. Face recognition: Features vs. Templates. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1993, vol. 15, No. 10, pp. 1042-1053.
7. M. Turk, A. Pentland. Eigenfaces for Recognition. Journal of Cognitive Neuroscience, 1991, vol. 3, no. 1, pp. 71 - 86.
8. J. Yang, D. Zhang, A. F. Frangi, J. -Y. Yang. Two - dimensional PCA: a new approach to appearance-based face representation and recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, vol. 26, no. 1, pp. 131-137.
9. P. N. Belhumeur , J. P. Hespanha , D. J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. IEEE Transactions on Pattern recognition and Machine Intelligence, 1997, vol. 19, no. 7, pp. 711-720.
10. V. Štruc, N. Pavešić. The complete Gabor-Fisher Classifier for robust face recognition. EURASIP Journal on Advances in Signal Processing, 2010, vol. 2010, Article ID 847680, 26 pages.
11. J.-H. Zhai, C.-Y. Bai, S.-F. Zhang. Face recognition based on 2DPCA and fuzzy -rough technique. Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC 2010), July 2010, pp. 725-729.
12. X. Li. Face recognition method based on fuzzy 2DPCA. Journal of Electrical and Computer engineering, 2014, vol. 2014, Article ID 919041, 7 pages.
13. Z. Pan, H. Bolouri. High speed face recognition basedd on discrete cosine transforms and neural networks. Technical report. 1999 University of Hertfordshire.
14. A. Osipov. A fuzzy approach to performance evaluation of edge detectors. Lecture Notes in Signal Science, Internet and Education, 2007, WSEAS Press, pp. 94-99.
15. G. Jäger, U Benz. Measures of classification accuracy based on fuzzy similarity. IEEE Transactions on Geoscience and Remote Sensing, 2000, vol. 38, no. 3 pp. 1462-1467.
16. База ORL изображений лиц лаборатории AT&T. Доступна по адресу: <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>

Программный комплекс для теплового проектирования интервально стохастических температурных полей в электронных модулях STF-ElectronMod

П.И. Кандалов

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: petrki87@gmail.ru

Аннотация: Рассматривается программный комплекс "STF-ElectronMod" для теплового проектирования стационарных интервальных трехмерных температурных полей в электронных модулях. Описывается структура программного комплекса, приводятся результаты численного моделирования по программному комплексу "STF-ElectronMod".

Ключевые слова: программный комплекс, математическое моделирование, температурные поля

Математические модели, положенные в основу современных программных комплексов теплового проектирования (Beta Soft, TGM, Thermal Analysis, Thermal Solution, ANSYS и др.), исходят из допущения, согласно которому факторы, определяющие тепловые режимы и, как следствие, температурные поля в технических системах, носят детерминированный характер. Иначе говоря, все исходные данные, определяющие характер протекания теплового процесса в системе, являются однозначно определенными и полностью известными. В действительности факторы, которые определяют тепловые процессы в технических системах, в частности электронных системах, будучи интервальными, представляют собой случайные величины, принимающие значения внутри своих интервалов изменения, подчиняющиеся некоторым усеченным (ограниченными границами интервалов) законам распределения вероятностей. К факторам, обуславливающим интервальную неопределенность тепловых процессов относятся:

- статистические технологические разбросы при изготовлении технической системы и ее элементов;
- интервально-стохастические параметры окружающей среды;
- стохастические факторы, возникающие в процессе функционирования технической системы.

Поэтому для построения высокоэффективного программного комплекса компьютерного моделирования необходимо, чтобы математические модели и последующие алгоритмы положенные в основание программы, учитывали случайный характер факторов, определяющих тепловые режимы технической системы.

Для решения поставленной задачи были разработаны стохастическая математическая модель анализа интервально стохастических температурных распределений «СТР-модель» [1-3] и алгоритм компьютерного расчета распределений статистических мер «КПСМ-алгоритм» стационарных интервально стохастических температурных распределений. Полу-

ченные математическая модель и алгоритм реализованы в программном комплексе «STF-ElectronMod» (Simulation of Temperature Fields of Electronic Modules) [4].

Разработанный программный комплекс STF-ElectronMod предназначен для компьютерного моделирования детерминированных и интервально стохастических трехмерных температурных полей в электронных модулях, содержащих многослойную печатную плату (МПП), установленные на ее поверхности интегральные микросхемы (МС) и электро- радиоэлементы (ЭРЭ), конструктивные элементы теплоотвода, охлаждения и крепления, с учетом особенностей конструкции, реальных условий эксплуатации и монтажа МС и ЭРЭ, многослойности МПП, конструкций корпусов МС, при различных условиях теплоотвода и охлаждения.

Программный комплекс реализован с применением объектно-ориентированного программирования (ООП) на языке высокого уровня Pascal с использованием программной среды Embarcadero RAD Studio для Microsoft Windows. Использование ООП обеспечивает комплексу высокую степень модульности благодаря таким свойствам, как инкапсуляция, полиморфизм и позднее связывание. Модульность архитектуры соответствует основному базовому принципу открытости программного комплекса, что обеспечивает создание надежного и эффективного комплекса моделирования, который позволяет в случае необходимости наращивать и модернизировать отдельные модули системы, при этом производимые действия оказывают минимальное влияние на работоспособность всей системы в целом.

Структура программного комплекса состоит из следующих элементов:

- подсистема GUI (Graphic User Interface);
- подсистема математического моделирования;
- подсистема STF Kernel;
- подсистема информационного обеспечения.

Подсистема GUI предоставляет разветвленную и удобную пользовательскую оболочку в функции которой входят предоставление пользователю гибкие и удобные механизмы:

- задания исходных данных;
- графического отображения конструкции электронного модуля в соответствии с заданными исходными данными;
- визуализации рассчитанных статистических мер (математических ожиданий (МО), дисперсий (Д), среднеквадратических отклонений (СКО)) интервально стохастических трехмерных температурных полей электронных модулей в виде цветных изотерм;
- визуализацию рассчитанных детерминированных трехмерных температурных полей электронных модулей в виде цветных изотерм;
- сохранение и загрузку исходных и рассчитанных данных в едином формате хранения модельных данных;
- интерактивное отображение рассчитанных

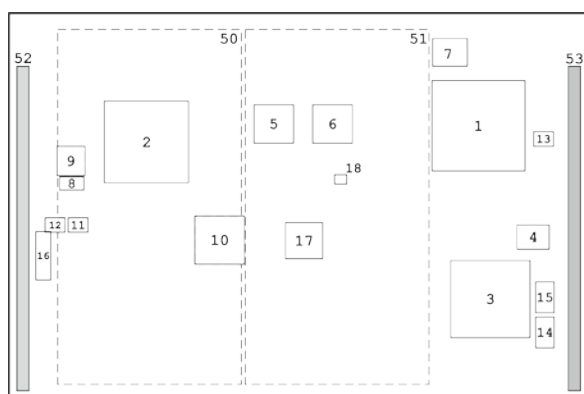


Рис. 1. Тепловая модель TOP поверхности МПП

данных в графическом и цифровом виде;

- представление конструкций МС, ЭРЭ и теплопроводов системы охлаждения с помощью многополюсного направленного графа тепловой схемы;
- формирование отчета и протокола результатов моделирования, представленным в удобном для пользователя виде.

Подсистема математического моделирования

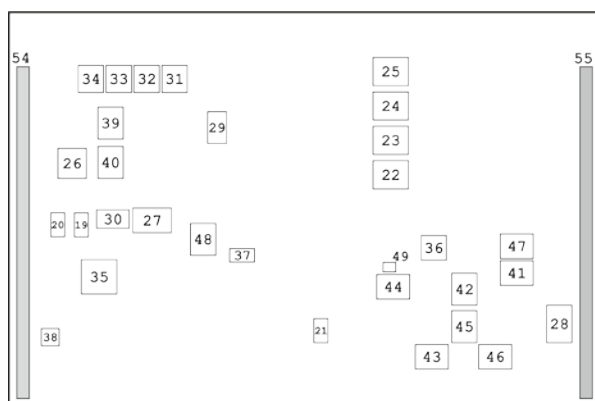


Рис. 2. Тепловая модель BOTTOM поверхности МПП

реализует программный модуль с алгоритмами математических моделей. На данный момент в данной

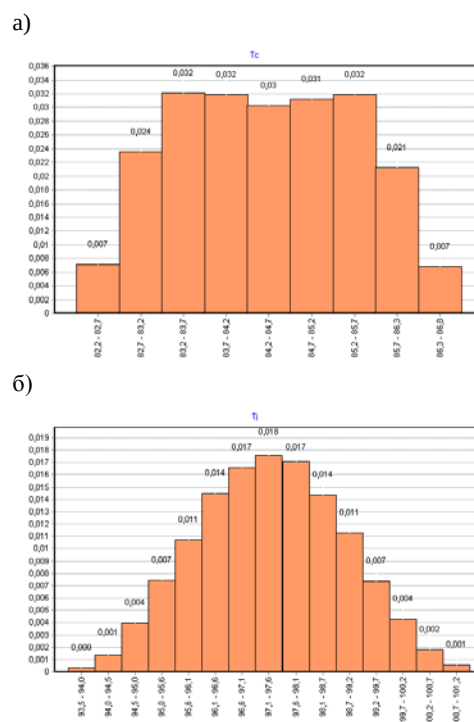


Рис. 3. Гистограммы, рассчитанные для интервальных распределений температур корпуса (а) и кристалла (б) для микросхемы МС 1

подсистеме реализованы ряд алгоритмов, обеспечивающих расчеты:

- интервально-стохастических трехмерных температурных полей в электронных модулях[1];
- детерминированных температурных полей в электронных модулях [5];
- статистических мер (МО, Д, СКО) интервально-стохастических трехмерных температурных полей электронных модулей [1, 2]; границ интервалов изменения реальных значений стохастических температур в различных точках электронного модуля [1];

- конструкций МС, ЭРЭ и теплопроводов системы охлаждения с помощью многополюсного направленного графа тепловой схемы [5, 6];

В основу подсистемы расчетного ядра STF Kernel программного комплекса «STF-ElectronMod» положена детерминированная стационарная модель уравнений теплопроводности [7, 8]. STF Kernel предоставляет общий Application Programming Interface (API) для подсистемы математического моделирования. На основе функции, классов и методов, предоставляемых API, реализуются все алгоритмы в подсистеме математического моделирования программного комплекса. API имеет унифицированный интерфейс и может быть предоставлен внешним приложением либо по средством DLL (Dynamic Link Library), либо в качестве COM объекта (Component Object Model), что делает подсистему STF Kernel универсальной по отношению к сторонним приложениям.

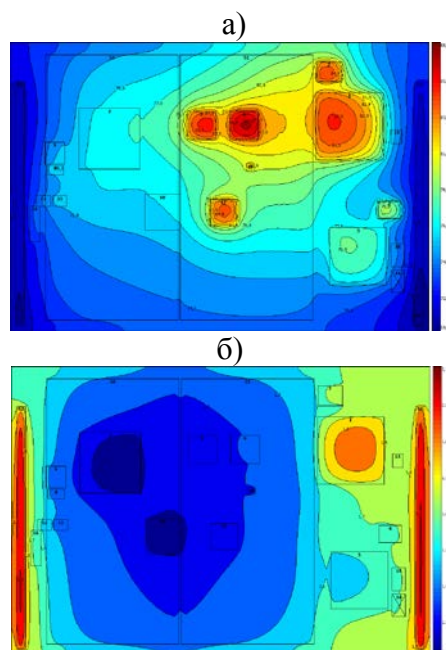


Рис. 4. Распределение температурного поля МО (а) и СКО (б) на TOP поверхности МПП ЭМ

Подсистема информационного обеспечения программного комплекса состоит из двух файловых баз данных (БД):

- БД материалов;
- БД тепловых схем.

БД материалов хранится единым файлом в формате XML (eXtensible Markup Language) и содержит знания о теплопроводности материалов и его названии, которые используются при задании исходных данных теплопроводностей слоев МПП, зазоров между элементами и МПП, выводов и т.д.

БД тепловых схем представляет собой совокупность файлов формата XML, каждый из которых содержит описание шаблона тепловой схемы одного элемента в виде направленного графа [6].

Хранение информации о материалах и шаблонов тепловых схем в виде файлов формата XML обеспечивает легкий доступ к операциям добавления/редактирования/удаления и не требует дополнительных манипуляций на машине пользователя связанных с настройками полноценных СУБД. Помимо этого, файлы БД могут быть перенесены с одной машины на другую путем простого копирования на внешние носители информации.

Рассмотрим пример применения «СТР-модели» для численного моделирования электронной системы, содержащей девять однотипных электронных модулей, образующих восемь вертикальных каналов, через которые нагнетается воздух. Электронные модули зажимаются по краям специальными фиксирующими устройствами.

Статистический разброс мощностей источников теплоты (МС, установленных в электронных модулях) составляет $\pm 10^\circ\text{C}$ от номинального значения для МС (рис. 1, рис. 2) с номерами 1-7,10, 14-17, 22-25, 27, 28, 31, 36, 38-48 и $\pm 15^\circ\text{C}$ для остальных элементов. Математическое ожидание температуры воздуха в каналах между электронными модулями и ее статистический разброс, составляет $71,3^\circ\text{C} \pm 3\%$. На крышке корпуса МС 1 (рис. 1) через эластичную теплопроводную прокладку установлен штыревой радиатор, выполненный из меди. Многослойная структура МПП электронного модуля состоит из 31 слоя содержащих сигнальные слои, слои полиимида, слои заземления и распределения питания. В конструкцию электронного модуля входят также два дополнительных мезонинных электронных модуля, электрические разъемы и металлические планки, которые прижимаются и фиксируются в направляющих системы.

Моделирование по разработанному программному комплексу показало, что реальные значения температуры корпусов ЭМ не являются детерминированными, а представляют собой интервально стохастические температуры, изменяющиеся в пределах своих интервалов. Гистограммы, рассчитанные для интервальных распределений температуры кристаллов и корпусов микросхемы МС 1, приведены на рис. 3. Максимальному разогреву подвержена МС 36: МО температуры корпуса составляет $105,98^\circ\text{C}$, СКО = $2,83^\circ\text{C}$, интервал изменения $99,33 - 112,74^\circ\text{C}$. Таким образом, разброс реальных температуры МС № 36 может достигать $\pm 7\%$ относительно ее МО (= $105,98^\circ\text{C}$).

Результаты моделирования статистических мер (МО и СКО) температурных полей на плате ЭМ в условиях конвективного теплообмена приведены для TOP поверхности платы на рис. 4. Минимальное и максимальное значение температуры для распределения МО температурного поля ЭМ составляет $69,96^\circ\text{C}$ и $88,13^\circ\text{C}$ соответственно.

Разработанная математическая «СТР-модель» и алгоритм компьютерного расчета распределений статистических мер «КРСМ-алгоритм» стационарных интервально стохастических температурных распределений температуры реализованы в программном комплексе «STF-ElectronMod» и применяются для теплового проектирования сложных электронных систем. Особенностью программного комплекса является наличие развитых подсистемы GUI, упрощающего оценивание мест наибольшего нагрева конструкции визуализацией результатов моделирования в виде полей МО и СКО, а также построением гистограмм интервальных распределений температуры. Практика моделирования ЭМ показывает, что положенные в основу программного комплекса методы, позволяют моделировать трехмерные температурные поля ЭМ с высокой эффективностью.

Software Application for Thermal Design of Interval Stochastic Temperature Fields in Electronic Modules

STF-ElectronMod

P. I. Kandalov

Abstract. The article consider the software application "STF-ElectronMod" to design a steady-state thermal interval 3D temperature fields in electronic modules. The simplified structure of software is described. The results of computer simulation of the electronic module is present.

Keywords: software application, mathematical modeling, temperature fields

Литература

1. А.Г. Мадера, П.И. Кандалов Анализ интервально стохастических температурных полей технических систем // Программные продукты и системы. – 2014. – №4. – С. 41 – 45.
2. П.И. Кандалов, А.Г. Мадера Моделирование температурных полей технических систем в условиях интервально стохастической неопределенности // Шестая Российская национальная конференция по теплообмену. – М.: Изд-во МЭИ. – 2006. – Т.1. – С. 191 – 192.
3. P.I. Kandalov, A.G. Madera Simulation of temperature fields of technical systems in interval indeterminacy conditions [электронный ресурс] // Materials of the V International Research and Practice Conference. European Science and Technology, 3-4 October 2013. – Munich, Germany. – V. 1. – P. 404 – 411. – Режим доступа: <http://www.sciencic.com/archive.php>.
4. П.И. Кандалов, А. Г. Мадера Свид. 2015619835 Российская Федерация. Свидетельство об официальной регистрации программы для ЭВМ. Simulation of Temperature Fields of Electronic Modules 2014 (STF-ElectronMod 2014); заявитель и правообладатель ФГУ ФНЦ НИИСИ РАН (RU). – №2015613488; заявл. 28.04.2015; опубл. 15.09.2015, Реестр программ для ЭВМ.
5. А.Г. Мадера, П.И. Кандалов Матрично-топологический метод математического и компьютерного моделирования температурных полей в электронных модулях: программный комплекс STF-ElectronMod // Программные продукты и системы. – 2012. – №4. – С. 160 – 164
6. А. Г. Мадера Моделирование теплообмена в технических системах – М.: Науч. Фонд «Первая исслед. лаб. им. ак. В.А. Мельникова», 2005. – 208 с.
7. П.И. Кандалов, А.Г. Мадера Моделирование температурных полей в многослойных структурах // Программные продукты и системы. – 2008. – № 4. – С. 46 – 49.
8. А.Г. Мадера, П.И. Кандалов Моделирование трехмерных температурных полей в электронных модулях // Программные продукты и системы. – 2010. – №2. – С. 29 – 33

Численное моделирование свободного конвективного теплообмена в корпусах электронных систем

М.Ж. Акжолов

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: agmprof@mail.ru

Аннотация: Приведены результаты численного моделирования конвективного теплообмена тепловыделяющих элементов и воздушной среды в электронных системах при различных граничных условиях.

Ключевые слова: электронные системы, тепловыделяющие элементы, конвективный теплообмен.

Задачи по изучению процессов теплообмена в электронных системах (ЭС) между тепловыделяющими элементами (ТВЭ) и окружающей средой (воздух) рассматривались в [1 – 3, 5 – 9]. При моделировании принималось, что газовая среда идеальная, вязкость отсутствует, теплопроводность в газе пренебрежимо мала по сравнению с конвективным теплопереносом. Математическая модель газодинамических процессов, протекающих при теплообмене ТВЭ и средой (система уравнений газовой динамики Эйлера) приведена в [1]. В работе [1] были изложены результаты расчетов сформулированной задачи, когда на ТВЭ представляли собой граничные условия, соответствующие заданным значениям температуры. Геометрия расчетной области приведена на рисунке (Рис.1.). Указанные граничные условия ТВЭ в математическом алгоритме реализовывались согласно соотношению $E(x_{гр}, y_{гр}, t) = T_{гр} \cdot C_p$, где E – полная энергия, C_p – теплоемкость воздуха. Там же приведены динамика развития тепловых процессов в зависимости от времени и сделаны анализ и оценка адекватности выбранной математической модели.

На практике при тепловом проектировании ЭС большой интерес представляет охлаждение ТВЭ при различных граничных условиях. В данной работе проводилась исследования газодинамические процессы вблизи ТВЭ для трех случаев:

I. Неадиабатические условия по периметру прямоугольной расчетной области;

II. С открытыми жалюзями в верхней части

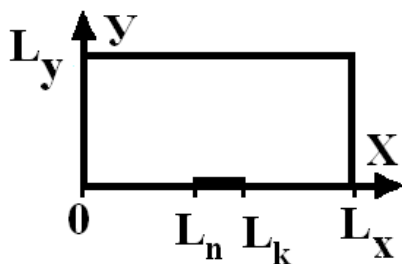


Рис. 1. Геометрия расчетной области

расчетной области;

III. С открытыми жалюзями на правой и левой вертикальных сторонах области.

Во всех трех случаях размеры расчетной области и мощности источника тепла были одинаковые и

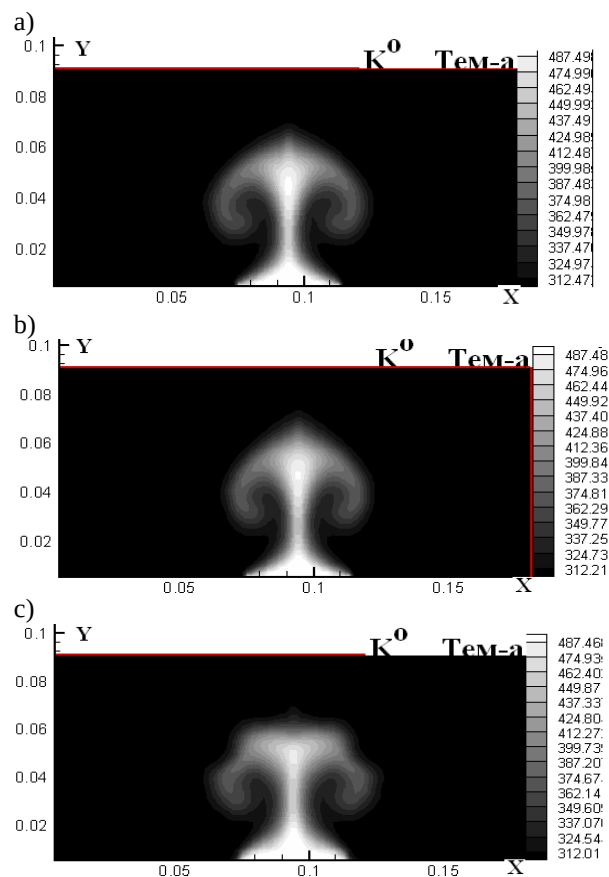


Рис. 2. Температурное поле воздуха в момент времени $t = 0.05$ сек. а) по I-му варианту; б) по II-му варианту; в) по III-му варианту.

составляли $L_x = 180$ мм, $L_y = 90$ мм (рис.1). На нижней границе участка в интервале $[L_n, L_k]$ задан источник тепла (ТВЭ) длиной 40 мм. Расчетная область

разбивалась на 90x45 квадратных ячеек со сторонами $d_x = d_y = 2$ мм, шаг по времени равен $dt = 10^{-8}$ сек.

Начальные и граничные условия:

Для случая I

– начальные условия (задавался невозмущенный воздух с нулевыми скоростями и плотностью воздуха при температуре 300⁰K):

$$U(x,y,0) = 0.0 \text{ м/с}; V(x,y,0) = 0.0 \text{ м/с}; \rho(x,y,0) = \rho_0; \rho_0 = 1.225 \text{ кг/м}^3; E(x,y,0) = E_0; E_0 = T_0 * C_p; T_0 = 300 \text{ K}^0; C_p = 1004.800 \text{ Дж/(K*кг)},$$

– граничные условия (задавались по всему периметру прямоугольной расчетной области условия невозмущенного потока всего расчетного периода $t > 0$ (условия неадиабатические)):

$$U_{\text{гр}}(t) = 0.0 \text{ м/с}; V_{\text{гр}}(t) = 0.0 \text{ м/с}; \rho_{\text{гр}}(t) = \rho_0; E_{\text{гр}}(t) = E_0.$$

Тепловыделяющий источник ТВЭ задавался на участке $[L_n, L_k]$ (рис.1) по следующему соотношению:

$$E_{\text{гр}}(t) = T_{\text{гр}} * C_p, \text{ где } T_{\text{гр}} = 500^0\text{K}.$$

Для случая II начальные и граничные условия сохраняются и дополнительно вводится на верхней границе условия свободного вытекания из жалюзи с шириной отверстия 40 мм.

Для III случая, все условия сохраняются, но вводится условие свободного вытекания из жалюзи на правой и левой вертикальной границах, расположенных в их верхней части симметрично, с шириной отверстий по 10 мм.

Результаты компьютерных расчетов приведены на рис. 2 – 4.

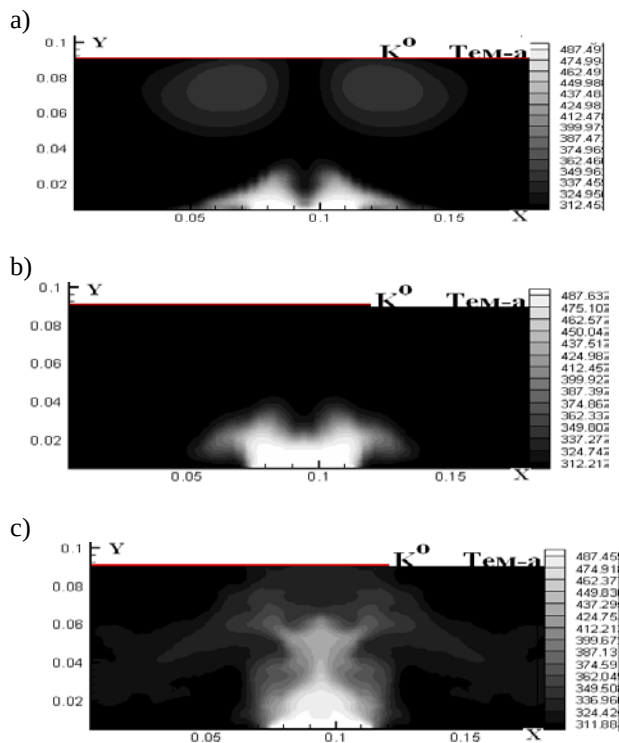


Рис. 3. Температурное поле воздуха в момент времени $t = 0.35$ сек. а) по I -му варианту; б) по II -му варианту; в) по III -му варианту.

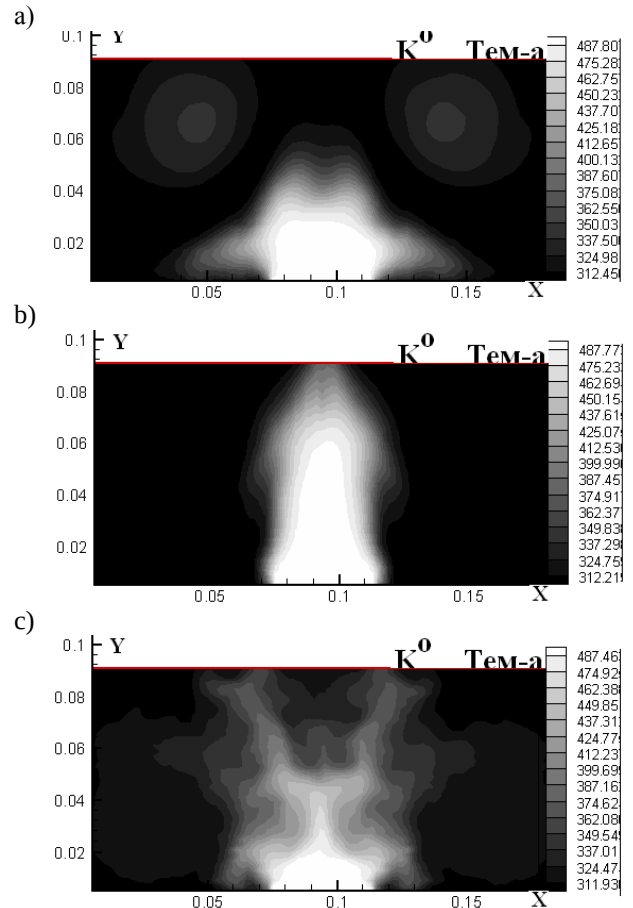


Рис. 4. Температурное поле воздуха в момент времени $t = 0.60$ сек. а) по I -му варианту; б) по II -му варианту; в) по III -му варианту.

Результаты и анализ

На рис. 2 приведены расчетные значения температурного поля воздуха в области с источником тепла в момент времени 0.05 сек, рассчитанные по трем вариантам. На рис. 3 – в момент времени 0.35 сек.

На рис. 4 – в момент времени 0.60 сек.

Анализ полученных результатов показывает следующее. При нагревании области, занятой ТВЭ, над ним развивается конвективный процесс, который происходит за счет действия сил давления и гравитации (максимальная скорость $\leq \sqrt{2g_0H}$, где $H = 90$ мм – высота расчетной области). Полученное изображение развития конвективных процессов в воздухе над ТВЭ достаточно хорошо отражает моделируемое физическое явление. Тепловые потоки, возникшие за счет нагревания ТВЭ развиваются плавно (монотонно) начиная с приповерхностного слоя, а дальнейшее развитие процесса во времени показывает расширение области охвата конвекцией все большего объема воздуха. По результатам моделирования хорошо видно, что в один и тот же момент времени размеры и формы охватываемой тепловым возмущением области изменяется по

разному в зависимости от внешних граничных условий т.е. без жалюзей и с жалюзями на верхней границе на правых и левых боковых вертикальных границах.

Вышеизложенные математические модели позволяют проводить вычислительные эксперименты

для решения задач охлаждения ТВЭ вычислительных систем и оптимизировать размеры корпусов ЭС, размеры и расположение жалюзей, а также мощностей источников теплоты.

Simulation of convective heat transform capsulated electronic systems

M.J. Akzholov

Abstract. The results of numerical modeling of convective heat transfer of fuel elements and air in electronic systems for different boundary conditions, are considered

Key words: electric systems, heat source, convection, source power (электронные системы, тепловыделяющие элементы, конвективный теплообмен).

Литература

1. М.Ж. Акжолов. Численное моделирование конвективного теплообмена в электронной системе при различных мощностях источника тепла. // Труды НИИСИ РАН, Том 6 №1, 2016. С. 62 – 63.
2. М.Ж. Акжолов. Математическое моделирование конвективного теплообмена тепловыделяющего элемента с воздушной средой в электронной системе. // Труды НИИСИ РАН, Том 5 №2, 2015. С. 103 – 105.
3. М.Ж. Акжолов, П.И. Кандалов, И.Г. Лебо, А.Г. Мадера. Компьютерное моделирование конвективных процессов в воздушной среде вблизи электронных устройств. // Труды НИИСИ РАН, Т. 1, №2, 2011. С. 44 – 46.
4. Мадера А.Г. Моделирование теплообмена в технических системах. – М.: НФ им. ак. В.А. Мельникова, 2005.
5. А.Г. Мадера Концепция математического и компьютерного моделирования тепловых процессов в электронных системах // Программные продукты и системы. 2015. № 4. С.79 – 86
6. П.И. Кандалов, А.Г. Мадера Моделирование температурных полей в многослойных структурах // Программные продукты и системы. 2008. № 4. С. 11
7. А.Г. Мадера, П.И. Кандалов Моделирование трехмерных температурных полей в электронных модулях // Программные продукты и системы. 2010. №2. С. 36
8. A.G. Madera Simulation of stochastic heat conduction processes // International Journal of Heat and Mass Transfer. 1994. Т. 37. № 16. С. 2571 – 2577.

Характеристики двух затворных КМОП нанотранзисторов с градиентно-легированной рабочей областью

Н.В.Масальский

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: volkov@niisi.ras.ru

Аннотация: При помощи численного моделирования исследуются электро-физические характеристики двух затворных транзисторных КНИ наноструктур с неравномерно-легированной рабочей областью. Предложенная математическая модель имплементирует подход численного решения 2D уравнения Пуассона. На основании полученного распределения потенциала вычисляются пороговое напряжение, подпороговый наклон и вольт-амперные характеристики суб-50 нм структур.

Ключевые слова: симметричный двух затворный КНИ КМОП нанотранзистор, 2D уравнение Пуассона, градиентно-легированная рабочая область, вольт-амперные характеристики

Введение

Двух затворный полевой транзистор со структурой «кремний на изоляторе» - основной претендент для реализации задач масштабирования микросхем для перспективных технологических норм. Кроме того, транзисторы с двумя независимыми затворами предоставляют большую функциональную гибкость [1-4]. Одна из главных проблем – неуклонное снижение концентрации носителей при неуклонном масштабировании топологии транзистора, которое влияет на производительность устройства [3-7]. Традиционные приемы решения данной проблемы натапливаются на паразитные эффекты, которые особенно проявляются в немасштабируемых свойствах транзисторных структур (пороговое напряжение, подпороговый ток и т.д.) [8, 9].

Еще одна негативная тенденция, связанная с ростом степени интеграции транзисторов, которая влечет уменьшение характерных размеров и снижением напряжения питания, становится главной причиной для обеспечения надежности устройств из-за сравнительно высоких абсолютных значений напряженностей электрических полей [6, 7]. Одна из главных проблем таких устройств – повышение чувствительности к изменению входного сигнала, которое влияет на предельную производительность микросхемы [10].

В качестве варианта решения сформулированной выше задачи в работе исследуются потенциальные возможности двух затворной транзисторной архитектуры с продольно градиентно-легированной рабочей областью. Такие структуры для объемных МОП транзисторов, которые еще называют «продольный

асимметричный канал» или «градуированный канал» представлены в работах [11–14].

В настоящей работе анализируются два варианта градуируемого канала (высоко-низкое и низко-высокое). Они предназначены для того, чтобы преодолеть проблемы, такие как деградация горячих носителей, roll-off порогового напряжения и паразитные биполярные эффекты, сопутствующие для однородно легированных транзисторов [3, 5, 6, 14, 15]. Главная особенность в результате предложенных модификаций рабочей области транзистора - меньше управляемости низко легированной области по сравнению с высоко легированной областью, т. к. известно, что, все отрицательные эффекты, например, эффекты горячих носителей, возникают у стока [16-18]. Таким образом, низко легированная область рядом с стоком может обеспечить эффективное подавление отмеченных выше отрицательных эффектов.

В данной работе были проанализированы такие структуры при помощи численного моделирования. Модель имплементирует подход численного решения 2D уравнения Пуассона. На основании распределения потенциала вычисляются электрическое поле, пороговое напряжение, подпороговый наклон и вольт-амперные характеристики структур в области латеральных дека нано размеров.

1. Прототип устройства

В модельной конфигурации двух затворного симметричного КНИ КМОП нанотранзистора, структура которого схематично представлена на рис. 1, рабочая область состоит из двух частей с разными концентрациями так, что

$$\begin{aligned}
L_g &= L_{g1} + L_{g2} \\
L_{g1} &= L_{g2} \\
N_{A_i} &= \begin{cases} N_{A1}, 0 \leq y \leq L_{g1} \\ N_{A2}, L_{g2} < y \leq L_g \end{cases}
\end{aligned} \quad (1)$$

где L_g - длина рабочей области (длина затвора), $i = 1$ область с концентрацией легирования N_{A1} и $i = 2$ область с концентрацией легирования N_{A2} .
Толщина рабочей области t_s , толщина подзатворного диэлектрика t_g

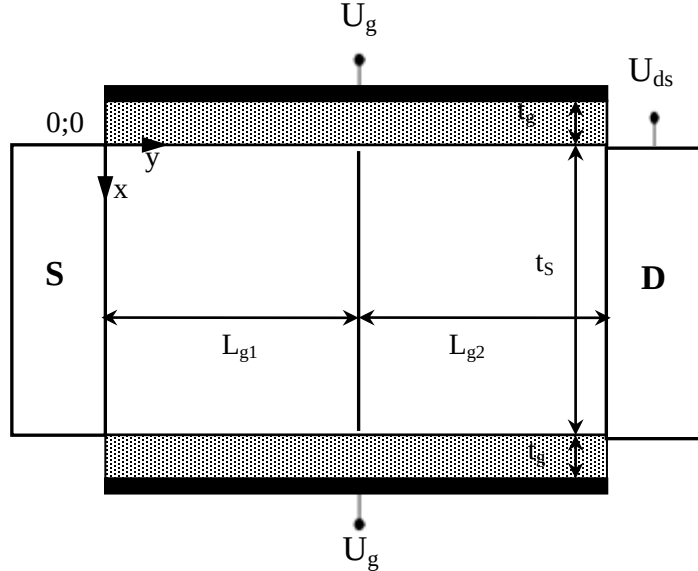


Рис. 1.

В работе проанализированы несколько типов транзисторных структур. В качестве базовой используется однородно легированная структура, где $N_{A1} = N_{A2}$. Также рассматриваются структуры с высокой и низкой концентрацией легирования ($N_{A1} > N_{A2}$), с низкой и высокой концентрацией ($N_{A1} < N_{A2}$)

2. Аналитическое выражение для распределения потенциала в рабочей области транзистора

Аналитическое выражение для распределения потенциала рабочей области транзистора вытекает из решения классического 2D уравнения Пуассона. Оно, как известно, широко используется для определения данной характеристики:

$$\frac{\partial^2 \varphi(x, y)}{\partial x^2} + \frac{\partial^2 \varphi(x, y)}{\partial y^2} = \frac{q N_{A_i}(y)}{\varepsilon_s}, \quad (2)$$

где q - заряд электрона, $\varphi(x, y)$ - потенциал в рабочей области транзистора, ε_s - диэлектрическая проницаемость рабочей области, $N_{A_i}(y)$ - концентрация легирования рабочей области, задаваемое соотношением (1).

Для решения (2) используются классические граничные условия в виде

$$\varphi(x, y)|_{x=0} = \varphi_{fi}(y)$$

$$\frac{\varepsilon_g}{t_g} (U_g - U_{FB_i} - \varphi_{fi}(y)) = -\varepsilon_s \frac{\partial \varphi_i(x, y)}{\partial x} \Big|_{x=0}$$

$$\frac{\varepsilon_g}{t_g} (U_g - U_{FB_i} - \varphi_{bi}(y)) = -\varepsilon_s \frac{\partial \varphi_i(x, y)}{\partial x} \Big|_{x=t_s}$$

$$\varphi_i(x, 0) = U_{bi}$$

$$\varphi_i(x, L_g) = U_{bi} + U_{ds}$$

где $\varphi_{fi}(y)$ - фронтальный поверхностный потенциал, $\varphi_{bi}(y)$ - обратный поверхностный потенциал, ε_g - диэлектрическая проницаемость подзатворного окисла, U_g - напряжение на затворах, U_{FB_i} - напряжение плоских зон, U_{bi} - встроенная разность потенциалов, U_{ds} - напряжение сток-исток.

Распределение потенциала на верхней (фронтальной) и нижней (обратной) поверхностях вытекают из решения уравнения Пуассона полученного для параметрического представления потенциала, которое получено в приближении разделения переменных [9]. В данном случае выражения для потенциалов можно представить в виде:

$$\begin{aligned} \phi_{b_i}(y) = & U_{1i} \exp(\sqrt{A_{0i}} y) \\ & + U_{2i} \exp(-\sqrt{A_{0i}} y) - \frac{A_{1i}}{A_{0i}}, \end{aligned} \quad (3)$$

$$\begin{aligned} \phi_{f_i}(y) = & U_{3i} \exp(\sqrt{A_{0i}} y) + \\ & U_{4i} \exp(-\sqrt{A_{0i}} y) - D_i \end{aligned} \quad (4)$$

где

$$A_{0i} = 2 \frac{\kappa_s (2t_g + \varepsilon_r t_s)}{t_g t_s^2 (1 + 2t_g / \varepsilon_r t_s)}$$

$$A_{1i} = \kappa_s \left[\frac{qN_{Ai}}{\varepsilon_s} - \frac{2((2t_g + \varepsilon_r t_s)U_g + (t_g + \varepsilon_r t_s)\phi_{f2}) + 2\phi_{f1}}{t_f t_s^2 (1 + 2t_f / \varepsilon_r t_s)} \right]$$

$$D_i = \frac{\sigma_1 (1 + 2 / \varepsilon_r t_s) \phi_{b_i}(y) + \phi_{f2} - \phi_{f1}}{(1 + 2 \frac{t_g}{\varepsilon_r t_s})}$$

$$\varepsilon_r = \frac{\varepsilon_g}{\varepsilon_s}$$

где коэффициенты U_{1i} , U_{2i} , U_{3i} и U_{4i} определяются из граничных условий. κ_s – подгоночный параметр, который связывает производную продольного электрического поля на любой глубине рабочей области с производной продольного электрического поля на фронтальной поверхности Si-SiO₂, полагая, что рабочая область достаточно тонкая ($t_s \approx 10$ нм) соотношение для κ_s можно представить в виде

$$\frac{\partial^2 \phi_i(y)}{\partial y^2} \cong \frac{1}{\kappa_s} \frac{\partial^2 \phi_{fi}(y)}{\partial y^2}.$$

2. Результаты моделирования

Моделирование характеристик рассматриваемого устройства осуществлялось в два этапа. Первоначально численно решалось уравнение Пуассона с расширенными граничными условиями. Кроме того, чтобы проанализировать эффект градуируемой структуры рабочей области, предполагалось наличие однородности потенциального и электрического полей на границах. Соответственно вычислялось распределение потенциала рабочей области транзистора. В результате получены – поверхностный потенциал, электрическое поле, пороговое напряжение, подпороговый наклон, которые сопоставлялись с данными моделирования программы ATLAS.

На втором этапе при помощи потенциальной модели [9] определялись вольт-амперные

характеристики (ВАХ) устройства во всей области функционирования. Вольт-амперные характеристики рассчитывались с учетом полевой и концентрационной зависимости подвижности.

В качестве базового устройства выбран двухзатворный симметричный равномерно легированный КНИ нанотранзистор с параметрами $L_g=45$ нм, $t_s=10$ нм, $t_g=1.8$ нм, $N_A=5.5 \times 10^{17}$ см⁻³. ВАХ базового устройства при различных напряжениях сток-исток U_{ds} приведены на рис. 2.

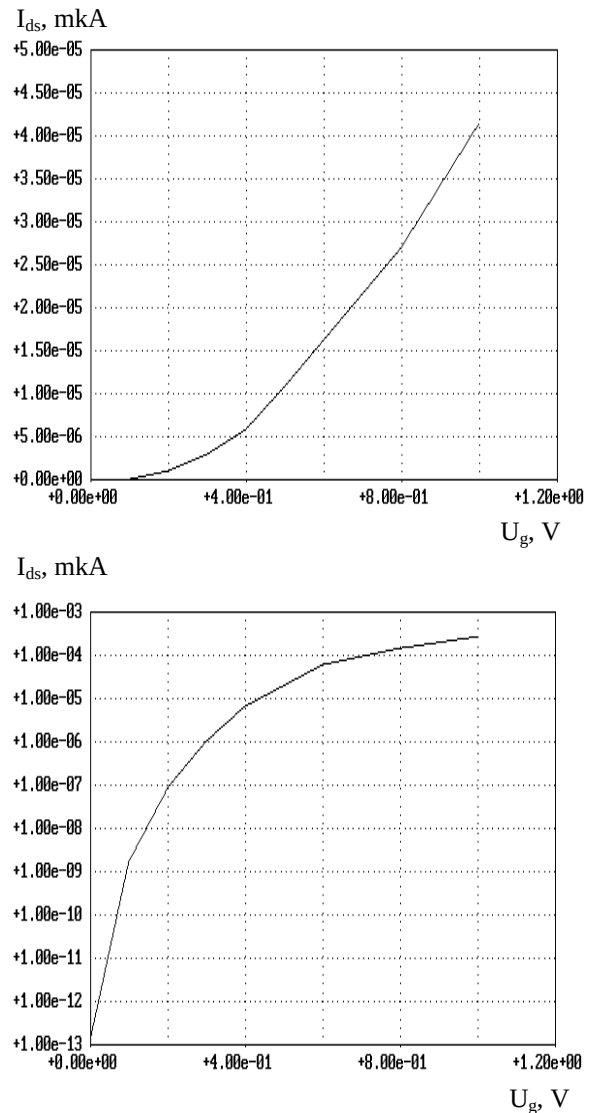


Рис. 2. ВАХ $I_{ds}(U_g)$ для различных U_{ds} , где верхний рисунок $U_{ds} = 50$ мВ, нижний $U_{ds} = 1.0$ В

Для данного случая пороговое напряжение составляет 268 мВ. Данный результат получен из анализа распределения потенциала на обратной поверхности исследуемой структуры. Как известно, пороговым напряжением обычно управляют минимумы потенциала канала. Поэтому необходимо проанализировать вариации трансформирования распределения потенциала вдоль обратной поверхности, и влияние этого эффекта на изменения фронтального поверхностного потенциала вдоль канала. Результаты моделирования и данные,

полученные при помощи программы ATLAS, находятся по большей части в хорошем согласовании.

Данные на рис 2 также показывают, что максимальное значение тока для значения затворного напряжения $U_g = 1$ В составляет 42.3 мкА при $U_{ds}=50$ мВ и 268 мкА при $U_{ds}=1$ В.

Подпороговый наклон можно определить как изменение напряжения затвора U_g требуемое для уменьшения подпорогового тока на одно декаду при $U_{ds} = 50$ мВ, а DIBL-эффект можно оценить из разности порогового напряжения при $U_g=50$ мВ и 1.0 В

Характеристики транзисторной структуры с высоко и низко легируемыми областями моделировались по тому же алгоритму, что и базовое устройство. Для исследований был выбран прототип двух затворного симметричного КНИ транзистора с параметрами $L_g=45$ нм, $t_s=10$ нм, $t_g=1.8$ нм, $N_{A1}=5.5 \times 10^{17} \text{ см}^{-3}$, $N_{A2}=5.5 \times 10^{15} \text{ см}^{-3}$. На рис. 3 приведено характерное распределение поверхностного потенциала в рабочей области транзистора.

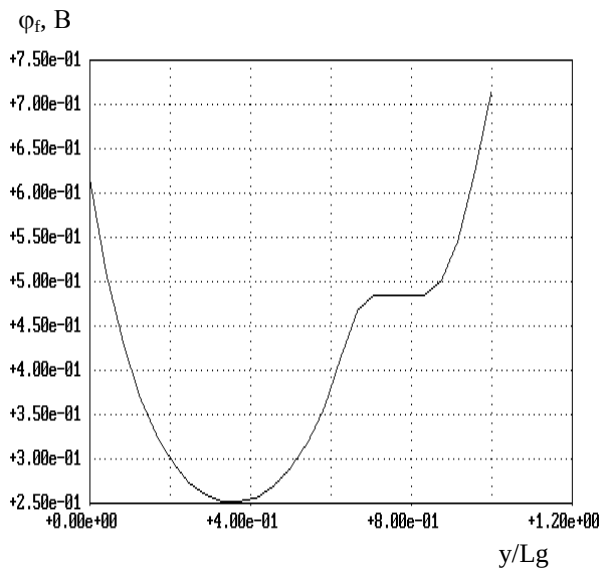


Рис. 3. Распределение поверхностного потенциала высоко низко градиентно-легированной структуры при $U_{ds} = 0.1$ В

В представленном случае минимум поверхностного потенциала расположен в высоко легированной области, примерно по середине между центром рабочей области и центром высоко легированной области. Распределение потенциала определяется отношением концентраций N_{A1} и N_{A2} . При снижении параметра N_{A2} минимум потенциала сдвигается влево и разница между минимальным значением потенциала и потенциалом соответствующим пологой части возрастает. При возрастании N_{A2} пологая часть в распределении потенциала постепенно исчезает и минимум потенциала сдвигается вправо. При выравнивании концентраций распределение потенциала в точности

соответствует распределению потенциала для базового случая.

Для структуры с низко и высоко легированными областями для тех же значений концентрации распределение потенциала представляет собой симметрично отображенное относительно центральной оси (параллельной оси x) распределение, представленное на рис. 2. Следует отметить, что распределение потенциала обладает симметричными свойствами, которые характеризуют распределение потенциала для случая высоко и низко легированной рабочей области.

ВАХ структура с высоко и низко легированной рабочей областью при $U_{ds}=0.1$ В приведена на рис. 4.

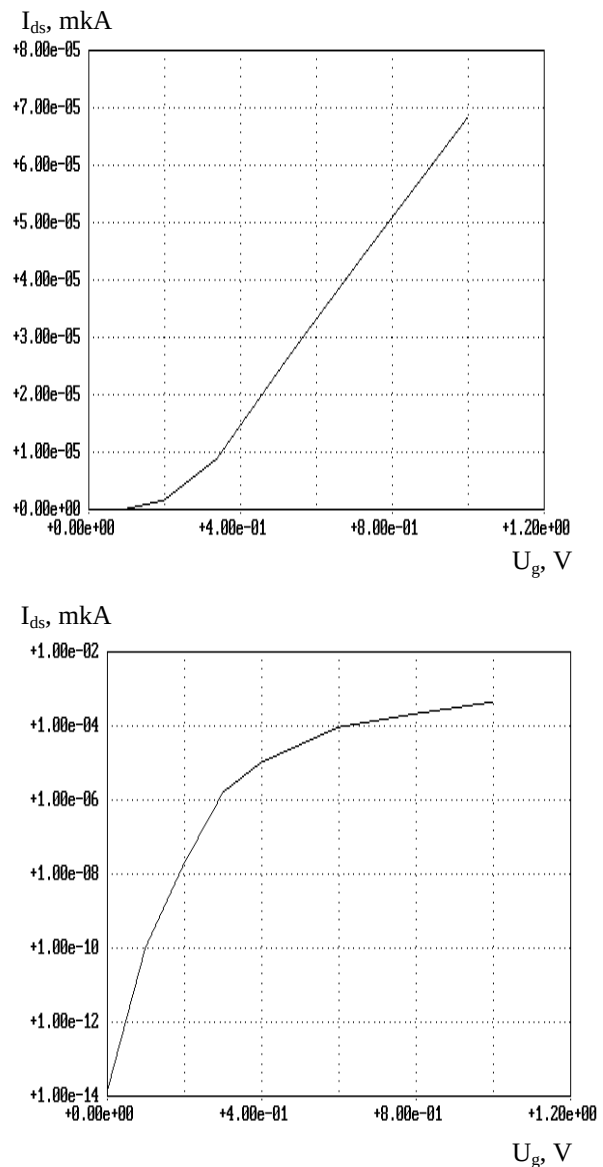


Рис. 4. ВАХ $I_{ds}(U_g)$ для различных U_{ds} , где верхний рисунок $U_{ds} = 50$ мВ, нижний $U_{ds} = 1.0$ В.

Данные на рис. 4 также показывают, что пороговое напряжение составляет 246 мВ, максимальное значение тока для значения $U_g = 1$ В

составляет 68.4 мкА при $U_{ds}=50$ мВ и 435 мкА при $U_{ds}=1$ В. Из данных на рис. 4 следует, что по сравнению с однородно легированной структурой максимальные значения тока I_{ds} существенно выше, более чем в полтора раза, ток I_{OFF} – практически на порядок меньше по сравнению с равномерно легированной структурой. Этот факт может являться предпосылкой для повышения производительности электронных устройств по сравнению с аналогичными, но выполненными на однородно легированных транзисторах.

Этот эффект усиливается при снижении уровня концентрации легирования низко легированной области. Этот вывод основывается на зависимости порогового напряжения от значения параметра N_{A2} . На рис. 5 представлена данная зависимость в удобном формате.

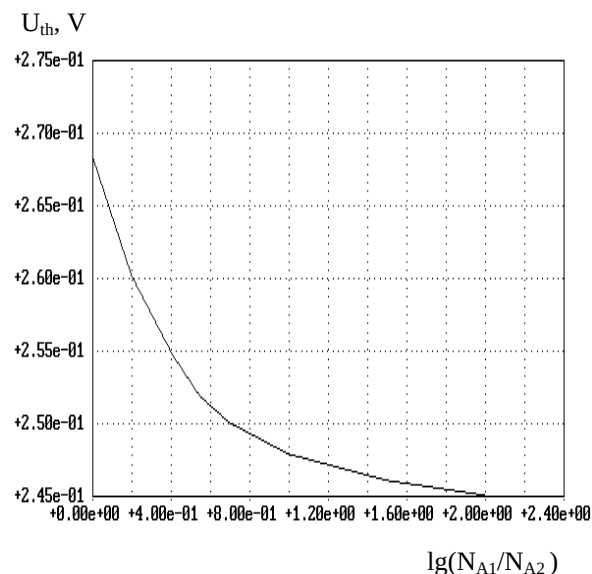


Рис. 5. Зависимость порогового напряжения от логарифма отношения концентраций $\lg(N_{A1}/N_{A2})$

Очень важно, что в области больше 2 – случай низкого легирования – пороговое напряжение практически не меняется, что обусловлено слабым влиянием концентрации легирования на пороговое напряжение. Следующий вывод, который следует из результатов моделирования заключается в том, что чем выше допустимый уровень легирования N_{A1} тем более качественное улучшение таких характеристик как быстродействие и утечка можно получить. Это вызвано тем, что данные свойства напрямую определяются минимумом потенциала канала, инверсионным зарядом которые приходятся на высоколегированную область. Это означает, что ретроградный профиль легирования, имеющий высоко-низкую комбинацию, более выгоден, чем однородно легированный профиль.

Для структуры с низко высоко легированной рабочей областью данные моделирования показывают, что пороговое напряжение составляет 244 мВ, максимальное значение тока I_{ds} при $U_g = 1$ В составляет 51.2 мкА при $U_{ds}=50$ мВ и 282 мкА при $U_{ds}=1$ В. Из приведенных данных следует, что

по сравнению с однородно легированной структурой максимальные значения тока I_{ds} практически не изменилось, ток I_{OFF} – примерно в два раза меньше по сравнению с равномерно легированной структурой.

Еще одна проблема, затронутая в ходе работы – не точность совмещения затворов. Наличие области, где отсутствует покрытие затвором в высоко легированной области приводит к уменьшению тех положительных свойств присущих транзисторной структуре с неравномерно легированной рабочей областью. Следует отметить, что эффект не точного совмещения проявляется в меньшей степени для областей с низким легированием. Поэтому неким компромиссом является структура с низко высоко низко легированными областями. При чем протяженность низко легированной части сопоставима с размером области насыщения. Предварительные численные эксперименты показывают, что для тех же топологических параметров достигается аналогичные ВАХ что и для структуры с высоко низко легированной рабочей областью, которые представлены выше.

Заключение

При помощи численного моделирования проанализированы электро-физические характеристики двух затворных КМОП нанотранзисторов с продольно градиентно-легированной рабочей областью.

На основании полученного квази аналитического решения 2D уравнения Пуассона численно рассчитывается распределения потенциала в рабочей области транзистора, при помощи которого вычислены: электрическое поле, пороговое напряжение, подпороговый наклон и вольт-амперные характеристики.

Проанализированы два варианта неравномерно-легированной рабочей области: высокое низкое легирование и низкое высокое легирование.

В результате показано, что профиль легирования, имеющий высоко-низкую комбинацию, является наиболее оптимальным, чем однородно легированный профиль и профиль с комбинацией низко-высоко легированный.

В такой структуре успешнее всего решается проблема горячих носителей, roll-off порогового напряжения, ограничение уровня подпорогового тока и паразитные биполярные эффекты.

Главная особенность – меньше управляемости низко легированной области по сравнению с высоко легированной областью, т. к. практически все отрицательные эффекты, присутствующие в рабочей области транзистора, возникают у стока.

Таким образом, низко легированная область рядом со стоком может обеспечить эффективное средство, чтобы устранить отрицательные эффекты.

Characteristics of double gate CMO S nanotransistors with a gradient-doping channel

N.V.Masalsky

Abstract: By means of numerical modeling electro-physical characteristics of two gate SOI transistor nanostructures with non-uniform doping channel area are researched. The offered mathematical model implements approach of the numerical solution for a 2D Poisson equation. Based on the received potential distribution a threshold voltage, a subthreshold slope and voltampere characteristics for sub-50 nm structures are calculated.

Key words: symmetric double gate SOI CMOS nanotransistor, 2D Poisson's equation, gradient-doping channel, volt-ampere characteristics

Литература

1. T. Schulz, W. Rosner, E. Landgraf, L. Risch, U. Langmann. Planar and vertical double gate concepts. «Solid-State Electron». v. 46(2002), № 3, 985-994.
2. S. Zhang, X. Lin, R. Huang, R. Han, M.A. Chan. Self-aligned, electrically separable double-gate MOS transistor technology for dynamic threshold voltage application. «IEEE Trans. Electron Devices». v. 50(2003), № 11, 2297-2306.
3. J. Widiez, J. Lolivier, M. Vinet, T. Poiroux, B. Previtali, F. Dauge, M. Moui, S. Deleonibus. Experimental evaluation of gate architecture influence on DG SOI MOSFET's performance. «IEEE Trans. Electron Devices». v. 52(2005), № 5, 1772-1781.
4. M. Reyboz, O. Rozeau, T. Poiroux, P. Martin, J. Jomaah. An explicit analytical charge based model of undoped independent Double-Gate MOSFET. «Solid-State Electronics». v. 50(2006), № 5, 1276-1285.
5. H. Liu, Z. Xiong, K. Sin. Implementation and characterization of double-gate MOSFET using lateral solid-phase epitaxy. «IEEE Trans. Electron Devices». v. 50(2005), № 4, 1552-1557.
6. T. M. Chung, B. Olbrechts, U. Sodervall, S. Bengtsson, D. Flandre, J-P. Raskin. Planar double-gate SOI MOS devices: fabrication by wafer bonding pre-patterned cavities and electrical characterization. «Solid-State Electron». v. 51(2007), № 1, 231-239.
7. C. Tsai. High-performance top and bottom double-gate low-temperature poly-silicon thin film transistors fabricated by excimer laser crystallization. «Solid-State Electron». v. 52(2008), № 2, 365-371.
8. Н.В.Масальский. Характеристики двух затворных КНИ КМОП нанотранзисторов для перспективных технологий с низким уровнем потребляемой мощности. «Микроэлектроника». т. 41(2012). №6, 436-444.
9. Н.В.Масальский. Двух затворные неравномерно легированные полевые нанотранзисторы. 2016. Монография, ISBN: 978-3-659-91279-5, LAP LAMBERT Academic Publishing GmbH & Co, Saarbrücken, Germany.
10. C. Yin, C. H. P. Chan. Investigation of the source/drain asymmetric effects due to gate misalignment in planar double-gate MOSFETs. «IEEE Trans. Electron Devices». v. 52(2005), № 1, 85-90.
11. M. A. Pavanello, J. A. Martino, V. Dessard, D. Flandre. Analog performance and application of graded-channel fully depleted SOI MOSFETs. «Solid-State Electron». v. 44(2000), № 4, 1219-1222.
12. M. A. Pavanello, J. A. Martino, D. Flandre. Graded-channel fully depleted silicon-on-insulator n-MOSFET for reducing the parasitic bipolar effects. «Solid-State Electron». v. 44(2000), № 3, 917-922.
13. M. A. Pavanello, J. A. Martino, J.-P. Raskin, D. Flandre. High performance analog operation of double gate transistors with the graded-channel architecture at low temperatures. «Solid-State Electron». v. 49(2005), № 6, 1569-1575.
14. J. S. Martin, A. Bournel, P. Dollfus. Comparison of multiple-gate MOSFET architectures using Monte Carlo simulation. «Solid-State Electron». v. 50(2006), № 1, 94-101.
15. R. K. Sharma, R. Gupta, M. Gupta, R. S. Gupta. Graded channel architecture: the solution for misaligned DG FD SOI n- MOSFETs. «Semiconductors Science Technology». v. 23(2008), № 11, 75041-75051.
16. Y. Liu. Systematic electrical characteristics of ideal rectangular cross section Si-Fin channel double gate MOSFETs fabricated by a wet process. «IEEE Trans. Nanotechnol.». v. 2(2003), № 1, 198-204.
17. C. Yin, P. C. Chan, V. W. C. Chan. Fabrication of raised S/D gate all around transistor and gate misalignment analysis. «IEEE Electron Devices Lett.». v. 24(2003), № 4, 658-664.
18. X. Lin, C. Feng, S. Zhang, W-H. Ho, M. Chan. Characterization of double gate MOSFETs fabricated by a simple method on a recrystallized silicon film. «Solid-State Electron». v. 48(2004), № 12, 2315-2324.

Моделирование сбоев информационной безопасности в сложных технических системах

М.Ш. Сургуладзе

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: sfm12@mail.ru

Аннотация. Рассмотрены и предложены методы активной диагностики сбоев аппаратно-программных систем: интеллектуальные, электромагнитной совместимости. Представлена концепция комплексного решения задачи повышения надежности космической радиоэлектронной аппаратуры в С-диапазоне частот.

Ключевые слова: моделирование, сбой, техническая система, информационная безопасность.

Развитие элементной базы, усложнение программного обеспечения решаемых задач и ужесточение условий эксплуатации современной аппаратуры требует совершенствования существующих и поиска новых подходов в повышении надежности и безопасности. Особо много проблемных вопросов возникает при создании высоконадежной аппаратуры для сложных технических систем при обеспечении высоких требований безотказности и долговечности в условиях воздействиях сверхнизких и сверхвысоких (до сотен градусов) температур и высоких уровней нагрузок [1, 5 – 9]. Среди перечня отказов аппаратуры самыми проблемными для контроля и диагностики следует признать перемежающиеся отказы, называемые также кратковременными, скрытыми, плавающими, самоустраняющимися или мерцающими отказами. Такие отказы приводят к сбоям в работе всего оборудования и могут привести не только к существенным материальным потерям, но и к человеческим жертвам. Под сбоем понимают самоустраняющееся нарушение нормального функционирования аппаратуры вследствие кратковременных воздействий на некоторый элемент (или совокупность элементов) внешних и внутренних факторов. После сбоя аппаратура длительное время может работать нормально, но при этом может быть искажена информация при операциях передачи, хранения или обработки. Проблема сбоев в последнее время ей уделяется повышенное внимание. Так, например, одной из важнейших характеристик автономной системы навигации в создаваемой сегодня международной космической станции является ее устойчивость к сбоям, причем приоритетность задач обеспечения высокой надежности по параметру сбоев выше, чем приоритетность задач управления объектами, классически стоявших на первом месте.

При разработке аппаратуры, имеющей в своем составе большое (до нескольких тысяч и десятков тысяч) число потенциальных источников сбоя (многоконтактные соединители, контактирующие устройства БИС и СБИС, печатные проводники, линии связи – интерфейсные шины, шины электропитания и заземления и т. д.), ключевой проблемой кардинального повышения надежности является диагностика сбоев, что непосред-

ственно связано с обнаружением и регистрацией источников сбоев в аппаратуре.

Анализ отечественных и зарубежных исследований по данному вопросу показал, что существуют многочисленные методы контроля, направленные не на выявление и устранение источников сбоя, а на результаты их проявления, при этом устраняется не сама причина (в данном случае источник сбоя), а ее следствие (т.е. ошибка, вызванная сбоем), оставляя тем самым потенциальную возможность существования в аппаратуре скрытых дефектов [4, 5]. Принципиальная особенность новой концепции резкого повышения надежности аппаратуры за счет исключения воздействия на нее сбоев состоит в том, что в отличие от всех ранее используемых подходов к решению данной проблемы обнаруживаются и регистрируются не места сбоев, а источники сбоев. В зависимости от принципов формирования и получения информативных признаков, по совокупности которых оценивается сбойное состояние элементов аппаратуры как источников сбоев, предложены различные методы обнаружения и регистрации источников сбоев (рис. 1), имеющие значительный приоритет.

Причинами сбоев в пассивных элементах (ПЭ) современной радиоэлектронной аппаратуры (РЭА) могут быть как различные внешние воздействия (вибрации, силовые электрические поля, температурные и химические воздействия), так и многочисленные скрытые дефекты аппаратуры, ускоренно приводящие к её деградации.

Наиболее сложным для описания является такой важный элемент аппаратуры как соединитель (электрический контакт).

Современная теория представляет электрический контакт в виде колебательного контура с элементами R_n , L_n и C_n – соответственно омическим, индуктивным и емкостным сопротивлением переходной зоны контактов (переходным сопротивлением) [4].

Данные параметры, определяющие резонансные колебания в области высоких частот, являются зависимыми от множества факторов, трудно контролируемых при эксплуатации и, следовательно, делающие актуальной задачу интегральной и оперативной их оценки.

Диагностирование пассивных элементов РЭА как процесс определения их сбойного состояния включает решение трех задач: изучение пассивных элементов как объекта диагностирования сбоев, построение алгоритмов диагностирования сбоев; разработка средств диагностирования сбоев.

Теоретическое обобщение процесса диагностирования сбоев пассивных элементов РЭА при ограниченной информации об их техническом состоянии предполагает использование формального описания ПЭ РЭА, то есть их математической модели диагностиро-

для измерения коэффициента отражения K^{omp} . Типы используемых при этом измерителей – Е7-8 (погрешность $\pm 0,001$ от измеренного значения ± 1 единица счета на табло индикатора) и стробоскопический осциллограф (границная частота $f_{gp} \leq 10$ ГГц).

При экспериментах по устойчивому получению режима сбоя решались следующие задачи: определение закона распределения сопротивлений контактных пар как функции включений-выключений N ; расчет статистических характеристик математического ожи-



Рис.1. Классификация методов обнаружения и регистрации источников сбоев

вания. Последняя должна обеспечивать диагностирование сбоев и быть пригодной для дальнейшего синтеза и реализации алгоритмов диагностирования.

Для построения математической модели диагностирования сбоев в ПЭ РЭА в качестве основы использованы структурные схемы кодо-импульсного управления асинхронными двухфазными двигателями, так как они дают возможность задать не только максимальную глубину диагностирования, позволяющую регистрировать сбой, но и отразить конструктивные решения реального ПЭ, допускающие решить вопросы практической реализации системы диагностики.

Экспериментальные доказательства. На основании предложенной кодо-импульсной модели источников сбоев экспериментально обнаружено сбойное состояние и получен нормальный закон распределения параметров и пуассоновский закон появления сбоев на примере наиболее перспективных соединителей типа РППМ 27. Количество контактных пар «п» было выбрано порядка 100 (точная цифра – 88), как достаточно представительная выборка. Динамика изменения параметров контактов в простейшем случае обеспечивалась их ускоренным износом за счет изменения числа включений-выключений и наблюдением за изменением одного из них – омического сопротивления. Испытания контактных пар проводилась в соответствии с техническими условиями (ТУ) методом миллиметра для измерения переходного сопротивления контактных пар R и методом падающей и отраженной волны

дания, дисперсии, среднеквадратического отклонения; проверка полученных экспериментально законов распределения существующим теоретическим по критерию:

Пирсона (χ^2) и Колмогорова А.Н. Кроме того, решались задачи экспериментального определения числа серий опытов N , приводящих гарантированно к сбойному состоянию контактных пар, а также определение закона появления сбоев и проверка достоверности полученных результатов.

В соответствии с существующими нормами одним из важнейших показателей качества является надежность технической системы (объекта). С другой стороны, работа любой технической системы может характеризоваться ее качеством, под которым понимается совокупность свойств, определяющих способность системы выполнять при ее создании определенные задачи. В свою очередь, одной из характеристик надежности является безотказность – свойство объекта непрерывно сохранять работоспособность в течение некоторого времени. Характеризуется сбой в соответствии с ГОСТом как самоустранимый отказ, поэтому очевидно, что в понятие «безотказность» входит и понятие «сбой». Таким образом, вводя активную диагностику сбоев (посредством фиксации их источников) через регистрацию промежуточного состояния («предсбой») и различные информативные параметры выводим понятие «сбой» в самостоятельную категорию («безсбойность»). Учитывая, что понятие «качество»

оперирует с такими понятиями как дефекты явные и неявные, т.е. не выявляемые по нормативно-

многослойной печатной платы — это частичное нарушение контакта проводника с контактными площадками

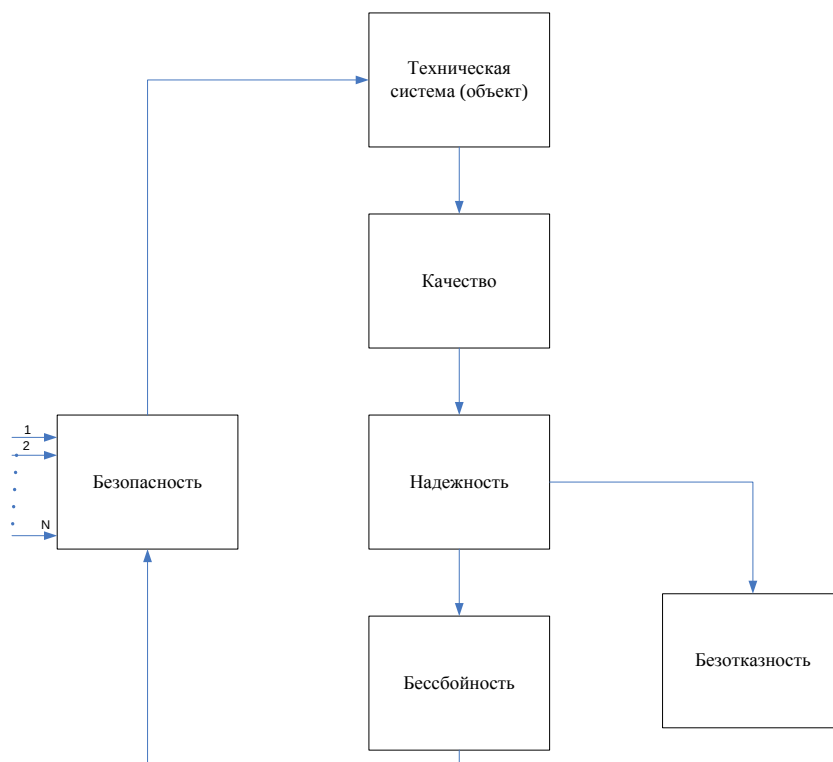


Рис. 2. Новое свойство технических систем – «бессбойность» - и ее связь с существующей структурой и безопасностью

технической документации (НТД), устанавливаем связь между сбоями и скрытыми дефектами. Поэтому существующие основные свойства надежности технических систем целесообразно добавить новым свойством – «бессбойность». Введение такого понятия в качество, в частности через измерения параметров сбоев бесконтактными и контактными методами, позволит управлять им, используя тонкий (до 10-15 мкм, размеров) пограничный слой предсбойного состояния пассивных элементов аппаратуры. В соответствии с изложенным, данную структуру можно представить в следующем виде (рис. 2). На рис. 2 надежность связана с качеством через бессбойность и безопасность обратной связью, причем N – число опасных воздействий.

Роль и место сбоев в методах электромагнитной совместимости. С точки зрения проблем электромагнитной совместимости (ЭМС) аппаратуры важно отметить, что сбой представляет собой кратковременное невыполнение требований по электромагнитной изоляции от внешних источников (помех), которые ведут к сбоям. С другой стороны, внутреннее состояние аппаратуры может само быть источником электромагнитных помех. Другими словами, между электромагнитными помехами и сбоями существует тесная взаимная связь.

Рассмотрим проблему ЭМС при сбоях на примере многослойных (в частном случае, однослойных) печатных плат. Одна из причин сбоев в проводниках

ми внутренних слоев. Известным способом оценки качества электрических соединений является анализ моделей, характеризующих линию связи, в частности, соединяющую БИС между собой, по заданным параметрам омического сопротивления R и полученным параметрам паразитных индуктивности L и емкости C . В состав таких моделей входят: передатчик — генератор импульсов, приемник — блок высокочастотных логических схем, соединенный с генератором через линию связи. Анализ таких моделей проводится временным методом. Однако представляется проблематичным использование данного метода для обнаружения и фиксации сбоев в линиях связи по двум причинам. Первая из них заключается в малых диапазонах изменения параметров режима "сбой", поэтому изменения, вносимые, например, за счет ослабления контакта в линиях связи, соизмеримы с изменениями сопротивлений, вызванных допустимыми изменениями геометрии печатных элементов вследствие нестабильности технологических процессов. Вторая причина заключается во вносимой погрешности средств тестирования при контактировании с диагностируемыми элементами.

Под сбоем или сбойным состоянием электрического соединения печатной платы (линии связи или соединителя) будем понимать промежуточное состояние соединения между двумя другими предельными состояниями. Одно из них — отказное состояние — со-

ответствует обрыву линии связи или состоянию выключенного соединителя с параметром омического сопротивления, равным бесконечности ($R = \infty$), а паразитными индуктивностью (L) и емкостью (C), равными нулю ($L, C = 0$). Другое устойчивое состояние соответствует исправному состоянию линии связи или состоянию включенного соединителя с параметром омического сопротивления меньше номинально допустимого

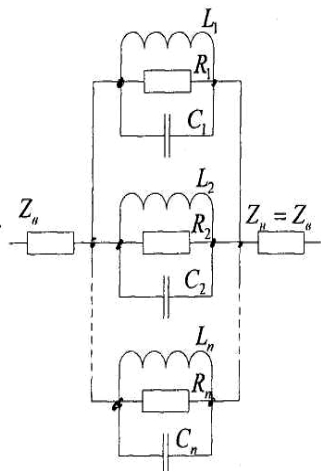


Рис. 3. Модель соединителя как совокупность «n» резонансных контуров

($R < R_{ном}$) и с индуктивным параметром паразитной реактивности. Тогда третье (сбойное) состояние характеризуется промежуточным значением омической составляющей и емкостным характером паразитной реактивности. Это состояние соответствует, например, микрообрыву, микротрещине или микрозазору печатного проводника.

Данное обстоятельство является причиной того, что соприкасающиеся точки имеют разные значения R , L и C . Параллельное соединение элементов R , L и C образует параллельный колебательный контур с потерями. В общем случае таких контуров будет n . На рис. 3 обозначено: Z_B - волновое сопротивление соединительного кабеля; Z_H - сопротивление нагрузки; R_i , L_i , C_i ($i = 1, 2, \dots, n$) — совокупности дискретных значений сопротивления, индуктивности и емкости в процессе деградации соединителя соответственно. Однако, когда диагностика сбоев во временной области невозможна, можно использовать другой информативный параметр - резонансную частоту.

Пример регистрации режима "сбой" соединителя типа РППМ 27, применяемого в отечественной супер-ЭВМ серии "Эльбрус", при тестировании показан на рис. 4. Кривая 1 на указанном рисунке — это тестируемое соединение (в качестве приемника выбран один из каналов осциллографа), кривая 2 — регистрируемые (на втором канале осциллографа) датчиком-соединителем (проводником) в режиме "сбой" частоты. Из рис. 4 видно, что сбой соединителя имеет несколько резонансных максимумов на частотах 4, 15, 55 МГц. Тенденция к образованию "сбойных" резонансных частот наблюдается и при повышении рабочего диапазона частот. Широкий диапазон как генерируемых, так и принимаемых частот в режиме "сбой" позволяет сделать вывод о надежности регистрации дан-

ного режима описанными методами, а также позволяет выполнять аппаратуру для регистрации портативной.

С этой точки зрения более универсальным и лишённым указанных недостатков является код Вьюшкова-Дианова, названный по фамилиям авторов, впервые предложивших его для решения задач прямого цифрового управления объектами. Основная идея — "цифровизировать" объект (в данном случае исполнительный механизм) и управлять им от цифровой машины, минуя цифро-аналоговый преобразователь. Разработанная теория, основой которой явилось совместное рассмотрение преобразования сигналов как в частотной, так и во временной областях, позволила не только универсализировать управление от цифровой ЭВМ с любым числом разрядов, начиная с восьми (при меньшем числе цифровое или кодовое управление вырождается в число-импульсное или широтно-импульсное), но и оптимизировать его с точки зрения электрической нагрузки исполнительного механизма. В частности, предложенный метод управления исполнительными механизмами, названный кодо-импульсным управлением, оказался эффективным как при управлении исполнительными двигателями малой (десятки и сотни ватт) мощности (летательные аппараты), так и большой (тысячи киловатт) мощности (блуждающие, прокатные станы). Имеет место и другой важный параметр универсализации - тактовые частоты работающих ЭВМ (от низких частот до десятков и сотен мегагерц) и частот питающих напряжений нагрузки (от нулевых и низких, 50-герцовых частот до 400 и 1000 Гц). Характерен и тот факт, что данный метод обработки сигналов предложила в конце 80-х годов прошлого столетия одна из ведущих зарубежных фирм в области вычислительной и контрольно-диагностической аппаратуры — Хьюлет-Паккард — и использовала его при построении сигнальных анализаторов. В общем виде код ЭВМ, "привязанный" к нагрузке, представляет совокупность единиц (импульсов) и нулей (отсутствие импульсов) и может иметь форму [2, 3, 10] ($k_1 + k_2 + \dots + k_m = N$, N - разрядность ЭВМ)

$$\begin{array}{ccccccc} 111\dots 1 & 000\dots 0 & 111\dots 1 & \dots & 111\dots 1 & \dots & 000\dots 0 \\ k_1 & k_2 & k_3 & & k_{i+n} & & k_m \end{array}$$

Предлагаемый метод для интеллектуальной диагностики сбоев датчиков и исполнительных механизмов заключается в том, что сбойные состояния определяются по образованию резонансных частот в датчиках и исполнительных механизмах и наличию повышенного (более 2-4 раза) электромагнитного излучения, дополнительному сдвигу фронтов импульсных сигналов, смене фазы гармонических сигналов проходящих через датчики и исполнительные механизмы, наличию субгармонических колебаний, изменению амплитудно-частотных характеристик, изменению соотношения падающей и отраженной волны, появлению эффекта дифференцирования и интегрирования сигналов. Из всех рассмотренных перспективных направлений развития вопросов изучения и исследования сбоев в современных системах автоматики особо хотелось бы отметить направление разработки сбоеустойчивой аппаратуры для космических аппаратов. В частности, одним из важнейших дополнительных требований к бортовой вычислительной технике является необходимость разработки повышенных мер защиты от сбоев.

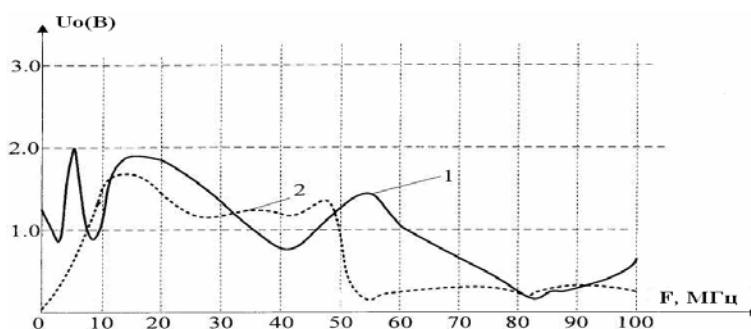


Рис. 4. Соединитель-излучатель (кривая 1) и соединитель-приемник (кривая 2) в режиме "сбой"

При этом необходимо учитывать различные условия возникновения сбоев во время полета (в первую очередь, виды орбит, радиационные условия). В частности, наиболее подвержена сбоям аппаратура космических аппаратов, вращающихся по высокоэллиптическим орбитам. Вместе с тем, следует отметить, что, наряду с исследованием сбоев в активных элементах аппаратуры (например, СБИС динамической памяти), недостаточно уделяется внимания "пассивной" части аппаратуры (линиям связи, контактирующим устройствам, контактными площадкам, проводникам печатных

плат и т.д.). При этом следует иметь в виду, что если для активных элементов аппаратуры первопричиной сбоев зачастую является радиация (например, высокоэнергетичные частицы химических элементов), то для пассивных элементов первопричиной сбоев может выступать вибрация (например, в длительных космических полетах).

В заключении приношу свою благодарность акад. В.К. Левину, д.т.н., проф. Н.А. Северцеву и д.т.н., проф. В.Д. Дедкову, за плодотворное обсуждение результатов работы.

Modeling of information security failures in complex technical systems

M.Sh. Surguladze

Abstract. Examined and proposed active diagnostic methods failure of hardware and software systems: intel-tual, electromagnetic compatibility. The concept of an integrated solution to the problem rose-sheniya reliability of space electronic equipment in the frequency C-band.

Keywords: modeling, failures, technical system, information security

Литература

1. А.И.Андреев, А.В.Баюков. Проблемные вопросы создания высоконадежной радиоэлектронной аппаратуры для ракетной, космической и авиационной техники, вытекающие из требований комплекса государственных стандартов «Мороз – 6». Международная НТК «Системные проблемы качества, математического моделирования, информационных электронных и лазерных технологий». Материалы конференции, ч. 1, 45-47, Москва-Сочи, 2001.
2. В.Н.Дианов. Радиоизмерительные методы в задачах обнаружения сбоев в ЭВМ // Вопросы радиоэлектроники. Серия ЭВТ. – 1992 – Вып. 6, 102-114.
3. Ю.Г.Евтушенко. Методы решения экстремальных задач и их применение в системах оптимизации. – М.: Наука, 1982.
4. А.П.Левин. Контакты электрических соединителей радиоэлектронной аппаратуры. (Расчет и конструирование). М., Советское радио, 1972.
5. А.Г.Мадера. Моделирование теплообмена в технических системах. – М.: НФ им. ак. В.А. Мельникова, 2005.
6. А.Г. Мадера. Концепция математического и компьютерного моделирования тепловых процессов в электронных системах // Программные продукты и системы. – 2015. – № 4. – С.79 – 86.
7. П.И. Кандалов. Мадера Моделирование температурных полей в многослойных структурах // Программные продукты и системы. – 2008,. № 4, с. 11.
8. А.Г. Мадера, П.И. Кандалов Моделирование трехмерных температурных полей в электронных модулях // Программные продукты и системы. – 2010, №2, с. 36.
9. A.G. Madera Simulation of stochastic heat conduction processes // International Journal of Heat and Mass Transfer. – 1994. т. 37, № 16, с. 2571 – 2577.
10. Р.Хольм. Электрические контакты. – М., ИИЛ, 1961.

Анализ причин неработоспособности кристаллов Ethernet контроллера 1990ВГ3Т

В.Ю. Троицкий¹, М.В. Орешков², А.А. Захарченко³, А.П. Трепалин⁴, Т.В. Петлицкая⁵,
В.А. Филипеня⁶

^{1, 2, 3, 4} ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail's: ¹troitskiy@srisa.ru, ²oreshkov@srisa.ru, ³zakharchenko@srisa.ru, ⁴ap@cs.niisi.ras.ru,

^{5, 6} НТЦ «Белмикросистемы», ОАО «Интеграл», Минск, Белоруссия, E-mail's: ^{5, 6}office@bms.by

Аннотация. Разработана и успешно опробована методика поэтапного исследования физико-технологических причин неработоспособности кристаллов интегральной микросхемы (ИМС) Ethernet контроллера 1990ВГ3Т с применением неразрушающих и разрушающих методов:

- локализации дефектов по результатам функционального тестирования и регистрации фотоэмиссии,
- послойного удаления слоёв металлизации,
- нанозондирования электрофизических характеристик транзисторных структур,
- исследования локализованной области на ионно-электронном микроскопе.

Получены важные результаты, раскрывающие технологическую причину неработоспособности ИМС.

Ключевые слова: анализ брака, локализация дефектов

Введение

Анализ причин неработоспособности кристаллов ИМС является важным направлением исследований не только на этапе отладки проекта топологии в конкретном технологическом процессе, но и в случае появления брака в кристалльном производстве. По мере уменьшения размеров элементов ИМС такой анализ существенно усложняется и требует применения разнообразных методов исследования.

Существуют как разрушающие, так и неразрушающие методы, позволяющие локализовать и установить причину неработоспособности кристалла ИМС, такие как:

- обратное проектирование по результатам функционального тестирования ИМС,
- метод регистрации фотоэмиссии,
- послойное удаление диэлектрических и металлических слоёв с визуальным контролем на каждом этапе,
- электронная растровая (SEM) и просвечивающая (TEM) микроскопия и ионно-электронная микроскопия.

Применение каждого метода сопряжено с определёнными ограничениями и требованиями к исследуемой ИМС, что будет рассмотрено более подробно. Кроме того, организация на мини-фабрике участка, содержащего полный комплект оборудования, позволяющего реализовать все перечисленные методы, нецелесообразна, ввиду малых объемов производства, высокой стоимости оборудования и инфраструктурных ограничений.

Тем не менее, возможность сочетания в процессе анализа различных методов позволяет снять многие ограничения, провести комплексный, законченный анализ и дать конкретные рекомендации технологам

или разработчикам. Именно такой комплексный подход, сочетающий использование оборудования различных организаций и тесное взаимодействие со специалистами этих организаций был применён в данной работе, что определяет её новизну и актуальность.

1. Описание образцов

В работе исследовались кристаллы микросхем 1990ВГ3Т, изготовленные по КМОП технологии с минимальным топологическим размером 0.35 мкм на объёмной кремниевой подложке с эпитаксиальным слоем. Функционально ИМС представляет собой Ethernet-контроллер 10/100 Мбит. Топология схемы содержит четыре блока двухпортовой кэш-памяти размерами: 512x35, 512x32, 512x25 и 32x35. По результатам расширенного функционального тестирования кристаллов неработоспособных микросхем, проведённого сектором электрических измерений отделения микротехнологии (ОМТ) совместно с отделением разработки вычислительных систем (ОРВС) НИИСИ РАН, на различных кристаллах зафиксирована неработоспособность одного или нескольких блоков кэш-памяти со следующими проявлениями:

- присутствуют дефектные ячейки, как с постоянной, так и с «плавающей» неработоспособностью;
- работоспособность дефектных ячеек памяти в ряде случаев зависит от напряжения питания, при повышении напряжения происходит «залипание» ячейки в одном из логических состояний.

Также на данном этапе была установлена возможность выхода на конкретную дефектную ячейку по топологии микросхемы для её дальнейшего

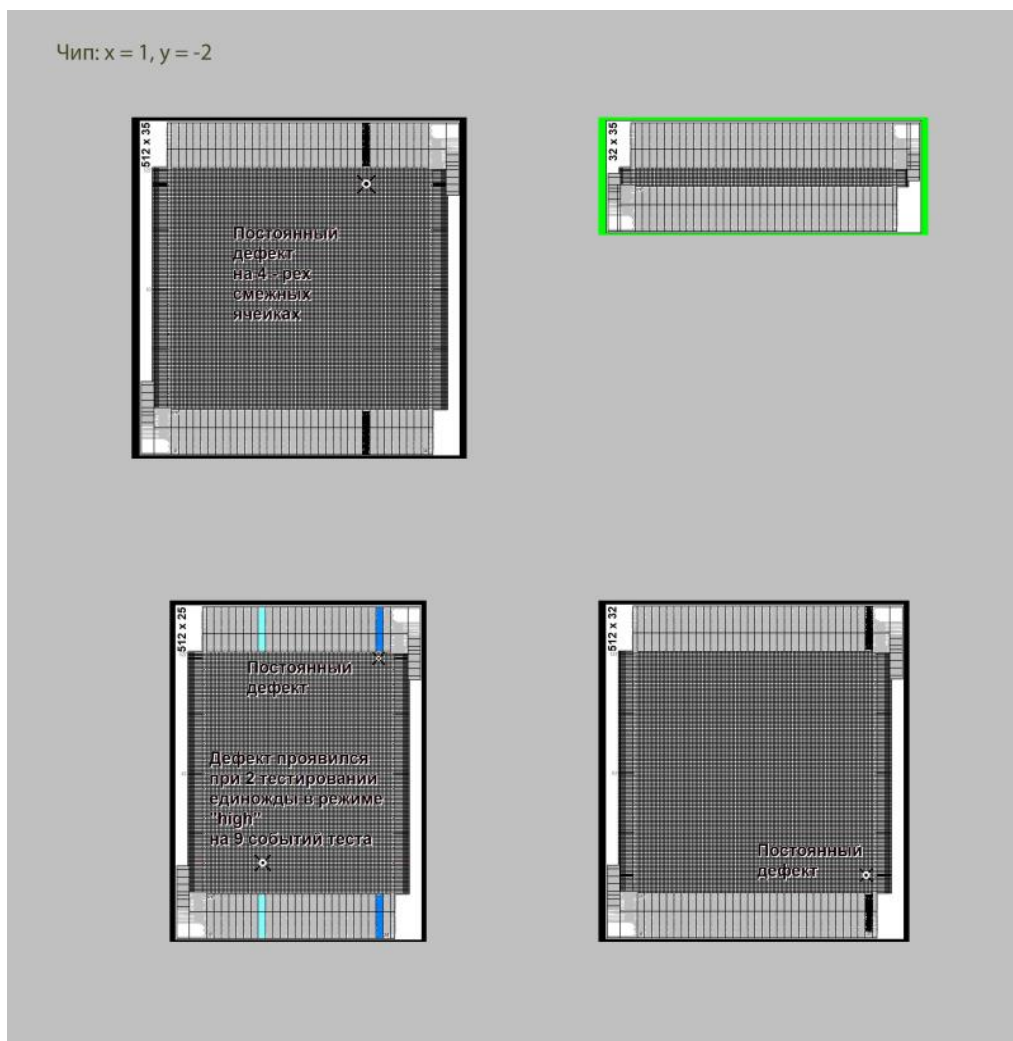


Рис. 1. Локализованное расположение дефектных ячеек памяти в исследованном кристалле

исследования разрушающими методами. Следующим этапом исследования стала локализация дефектных ячеек.

2. Локализация дефектов

Среди методик, позволяющих локализовать дефекты, сопоставимые с размерами транзисторов в исследуемой ИМС, можно выделить следующие: фотонно-эмиссионная микроскопия [1], обратное тополого-электрическое моделирование функционирования ИМС и послойное удаление диэлектрических и металлических слоёв с последовательным визуальным контролем всей площади кристалла.

Первые два метода являются неразрушающими, последний – разрушающий, однако каждый метод имеет свои особенности и ограничения. Первый требует наличия специального оборудования для получения фотоэмиссионных изображений кристалла с высоким разрешением и специальной подготовки образца. Кроме того, данный метод годится зачастую только для анализа блоков статической памяти, таких как кэш-память или SRAM, поскольку в КМОП технологии транзисторы в ячейках таких типов памяти не должны испускать фотоны в достаточном для детектирования количестве.

Обратное тополого-электрическое моделирование не требует специального оборудования, кроме функционального тестера, но требует составления специальных алгоритмов и трудоёмкого анализа полученных результатов, а также может не позволять достаточно точно выходить на неработоспособный транзистор или логическую ячейку, указывая только на конкретный функциональный блок. В случае сбоев в функционировании памяти у ИМС, таким методом, как правило, можно выйти на неработоспособную ячейку памяти, даже в отсутствие непосредственной адресации к ней (как в случае с анализируемой ИМС).

Третий метод локализации может использоваться и самостоятельно, и в сочетании с другими методами. Ограничениями метода являются его разрушающий характер, необходимость проведения операций травления в жидкой химии и высокие требования к прецизионности оборудования механической обработки поверхности кристаллов.

Трудоёмкость метода определяется сложностью и площадью кристалла ИМС, а его информативность – типом дефекта, то есть возможностью обнаружения этого дефекта в оптическом или электронном микроскопе и его сохраняемостью при проведении химических и механических обработок.

Результаты анализа функционирования.

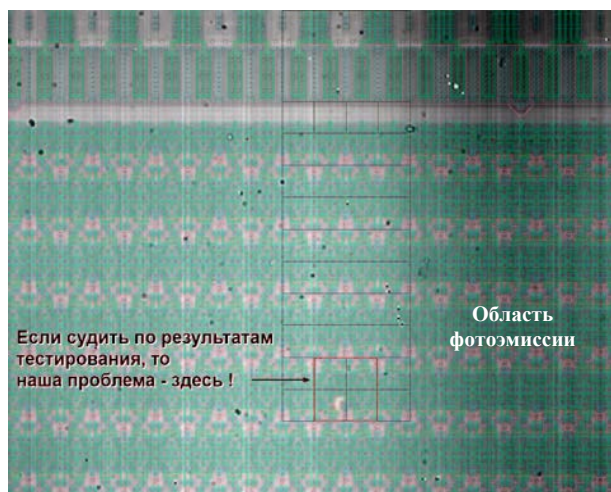


Рис. 2. Совмещённое изображение фотоэмиссии и топологии для кристалла с комментарием разработчика

По результатам функционального тестирования кристаллов ИМС дефектные ячейки встречались в любом из блоков памяти абсолютно случайно. Для физического анализа был выбран неработоспособный кристалл из центральной области пластины. Расположение локализованных разработчиком дефектных ячеек в блоках памяти по результатам тестирования данного кристалла представлено на рис. 1.

В этом кристалле разработчиком локализованы сбойные ячейки в блоках памяти 512x35, 512x32 и 512x25.

Следует обратить внимание, что в блоке 512x35 стабильно не работает группа из 4-х смежных ячеек, имеющих общий карман (ACTIVE) для n -канальных транзисторов (см. рис. 6).

Результаты фотонно-эмиссионной микроскопии.

Для проведения фотоэмиссионных исследований с использованием экспериментального программно-аппаратного комплекса на базе установки Vistec INM 100IR кристалл был разварен на специализированной плате. Фотоэмиссия из кристалла снималась при напряжении питания 3.6 В, поданного на выводы питания, остальные выводы не подключались.

На рис. 2 приведен результат наложения фрагмента чертежа топологии блока памяти, в котором обнаружена фотоэмиссия, на его фотоэмиссионное изображение. По данному изображению при большом увеличении видно, что эмиссия предположительно исходит от n -канального транзистора одного из инверторов триггера ячейки памяти (см. рис. 3).

По итогам анализа результатов функционального тестирования микросхемы разработчиком были локализованы 4 соседние неработоспособные ячейки в блоке памяти. В одной из этих ячеек и была зафиксирована описанная выше фотоэмиссия. Таким образом, результаты экспериментальной локализации дефектных ячеек и локализации по результатам анализа функционального тестирования микросхемы совпали.

Результаты исследований после удаления слоев металлизации.

Для проведения дальнейшего физического анализа кристалл был передан в ГЦ «Белмикрoанализ» ОАО

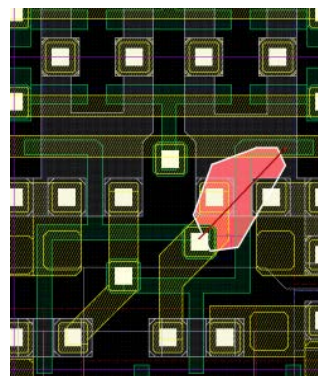


Рис. 3. Область свечения в одной из ячеек памяти

«Интеграл», г. Минск. На кристаллах было проведено прецизионное удаление слоёв пассивации, металлизации и межслойных диэлектриков вплоть до контактных окон к транзисторным структурам. При визуальном контроле открывшейся структуры транзисторных слоев неработоспособных блоков кэш-памяти были обнаружены локальные цветовые вариации топологического рисунка (рис. 4), причем эти цветовые вариации расположены именно в области четырех неработоспособных ячеек памяти, локализованных разработчиком, и совпадают с областью зафиксированной фотоэмиссии.

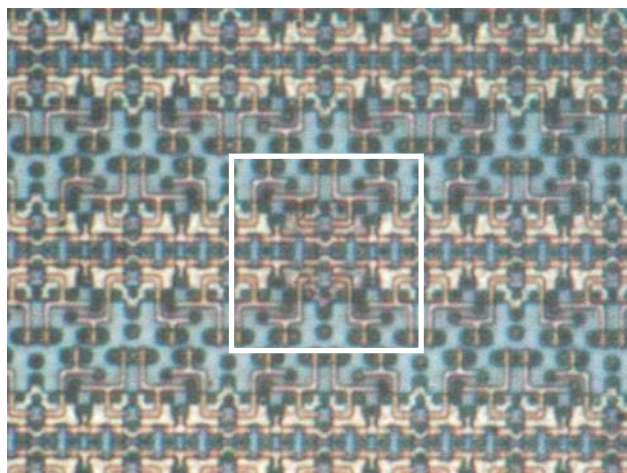


Рис. 4. Оптическое изображение дефектных ячеек в блоке 512x35 кристалла. Увеличение 3000х.

Все обнаруженные дефекты объединяет изменение цвета области ACTIVE, однако характер изменения отличается, что должно быть обусловлено разной природой дефектов. Так, в случае, показанном на рис. 4, вероятно имело место некачественное образование силицида титана по всему островку области ACTIVE.

3. Анализ обнаруженных дефектов

В качестве объекта для последующего разрушающего и электрофизического анализа были выбраны четыре смежные ячейки в блоке кэш-памяти 512x35 кристалла, локализованные всеми использованными методиками и представляющие собой информативный объект для проведения исследования на ионно-электронном микроскопе.

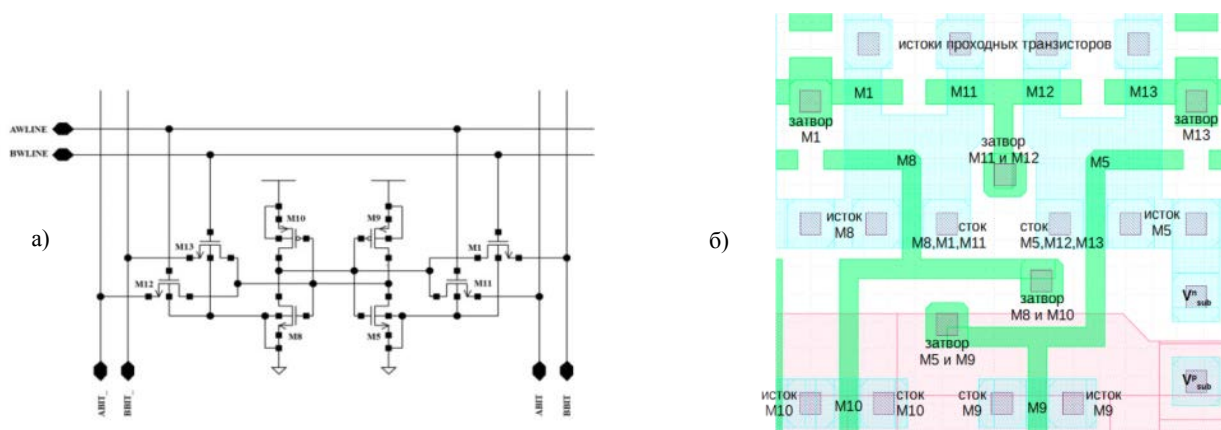


Рис.5 Электрическая (а) и топологическая (б) схемы двухпортовой ячейки памяти (DPRAM).

Нанозондирование электрофизических характеристик.

Для выявления возможного влияния обнаруженных дефектов на работоспособность транзисторов в локализованных неработоспособных ячейках кэш-памяти было проведено электрофизическое тестирование этих транзисторов с помощью нанозондов в электронном микроскопе [2,3]. Контакт зондов с электродами транзисторов на кристалле осуществлялся через вольфрамовые столбики в контактных окнах к активным областям транзисторной структуры после удаления всех слоёв металлизации.

На рис. 5 представлены электрическая схема анализируемой двухпортовой ячейки памяти и ее топология – слои карманов транзисторов, поликремния и контактов к транзисторным структурам.

На топологическом чертеже (рис. 6) четырех смежных неработоспособных ячеек в центре выделена область «Active», являющаяся общей для транзисторов M5, M12, M13, M1, M11, M8, находящихся в этих четырех ячейках. Именно в этой области наблюдались цветовые вариации, описанные при проведении визуального контроля после удаления слоёв металлизации. Также выделен затвор транзистора M5 в

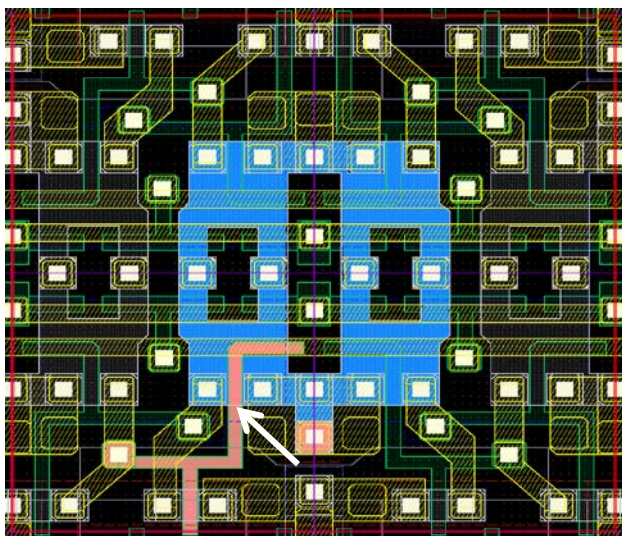


Рис. 6. Топологический чертеж четырех смежных неработоспособных ячеек. Стрелка указывает на затвор транзистора, в области которого наблюдалась фотоэмиссия.

нижней левой ячейке, в области которого наблюдалась фотоэмиссия.

Прецизионное электрофизическое тестирование транзисторов в области неработоспособных ячеек проводилось в ГЦ «Белмикроанализ» ОАО «Интеграл», г. Минск. Кристалл был помещен в растровый электронный микроскоп Hitachi S-4800, оснащенный зондовыми головками с электромеханическими манипуляторами производства фирмы Kleindiek Nanotechnik.

Электронно-микроскопическое изображение области вскрытых контактов к транзисторному слою, где проводилось зондирование, представлено на рис. 7. Местоположение контактов к электродам транзисторов исследуемых ячеек определялось по топологическому чертежу соответствующего слоя контактов.

Первоначально была поставлена задача провести измерения ВАХ всех транзисторов ячейки, в которой наблюдалась фотоэмиссия, однако из-за высокой трудоемкости и большой длительности процесса зондирования программа измерений была сокращена. Были проведены измерения семейства выходных ВАХ транзисторов M5 из соседней работоспособной ячейки и того же транзистора M5 из левой нижней неработоспособной ячейки (рис. 6), на котором наблюдалась фотоэмиссия. Потенциал кармана на соответствующем контакте V_{sub}^n (см. рис. 5) устанавливался равным истоковому потенциалу

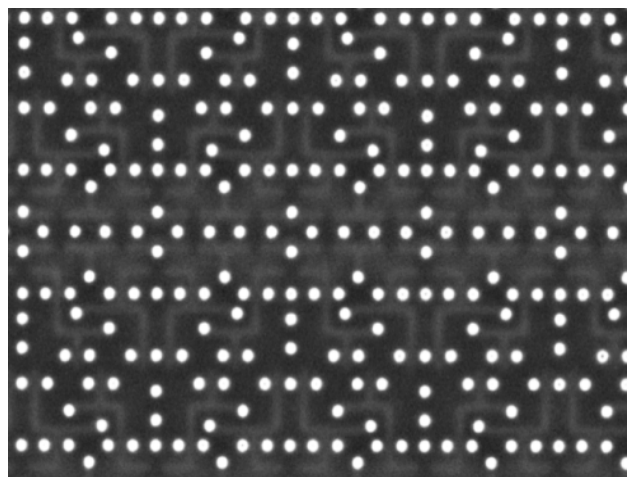


Рис. 7. Электронно-микроскопическое изображение области нанозондирования.

транзистора. Напряжение на затворе изменялось в диапазоне от -0,5 до +3 В.

Измерения в камере электронного микроскопа при разных ускоряющих напряжениях электронного луча на работоспособном n -канальном транзисторе M5 показали, что целесообразно проводить измерения при минимальном ускоряющем напряжении электронного пучка для уменьшения его влияния на измеряемые ВАХ. В дальнейшем измерения проводились при ускоряющем напряжении 1 кВ.

Провести измерение ВАХ неработоспособного транзистора M5 не удалось. В измерительной цепи наблюдался обрыв – отсутствие контакта к одному или нескольким электродам транзистора. Аналогичная ситуация сложилась с p -канальным транзистором M9, имеющим общий с M5 затвор, но не имеющим признаков дефектности. На рис. 8 представлены результаты измерения передаточных ВАХ этих транзисторов из нижней левой ячейки, в которой наблюдалась фотоэмиссия. Определить отсутствие конкретного контакта к стокам, истокам или к затвору не удалось из-за одиночных контактов к этим областям.

Следующим шагом стала проверка транзисторов, имеющих общий с излучающим транзистором M5 из нижней левой ячейки p -карман (выделенная область вокруг центра на рис. 6). В нижней правой ячейке проверялись транзисторы M5, M12, в верхних ячейках – M8. Так как предшествующие измерения выявили проблемы с качеством контактов к электродам

транзисторов, а также в связи с тем, что основные временные затраты при проведении измерений связаны с осуществлением контактирования нанозондов к электродам транзисторов (в общем случае используются 4 контакта), было принято решение провести ускоренное тестирование состояния транзисторов путем проверки качества контактов к стоковому и истоковому p - n -переходам этих транзисторов.

С этой целью один зонд был постоянно установлен на контакте к p -карману V_{sub}^n , расположенном между нижними ячейками, а второй зонд устанавливался на стоковые и истоковые контакты тестируемых транзисторов.

На рис. 9 представлены результаты тестирования контактов к стокам у транзисторов M5 в неработоспособной и годной ячейках (для сравнения). Аналогичные рис. 9(а) результаты были получены для транзистора M12 из нижних ячеек, в том числе и для контактов к истоку.

Из рис. 9(а) видно, что имеется только нелинейный ток утечки порядка 10^{-9} А, а контакт к самому диоду отсутствует. Однако нельзя утверждать, что контакты отсутствуют для всех стоков и истоков, поскольку все измерения проводились с использованием единственного общего контакта к карману этих транзисторов, который мог также отсутствовать.

Следует отметить, что хороший контакт к вольфрамовым столбикам, при котором получается ВАХ на рис. 9(б), с первого раза получить довольно сложно. На это влияет как форма и загрязнённость

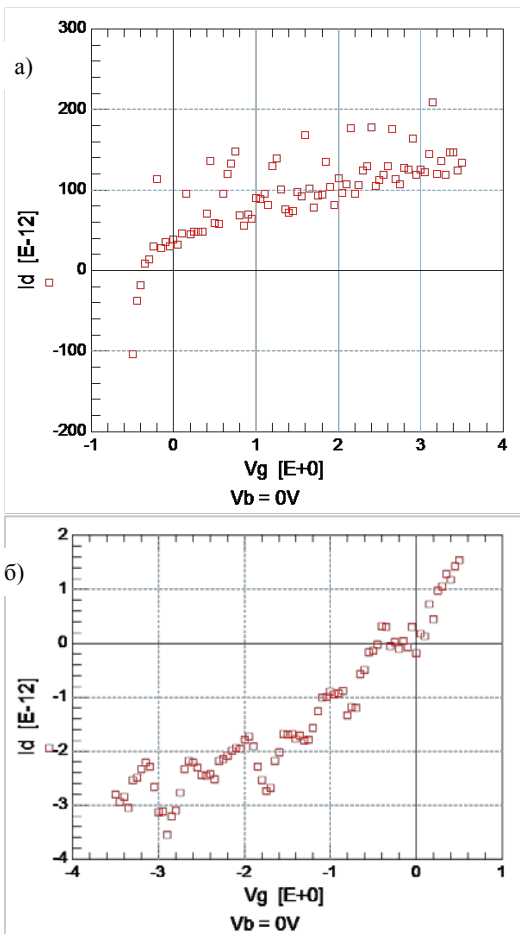


Рис. 8 Результаты измерения передаточных ВАХ транзисторов M5 (а) и M9 (б).

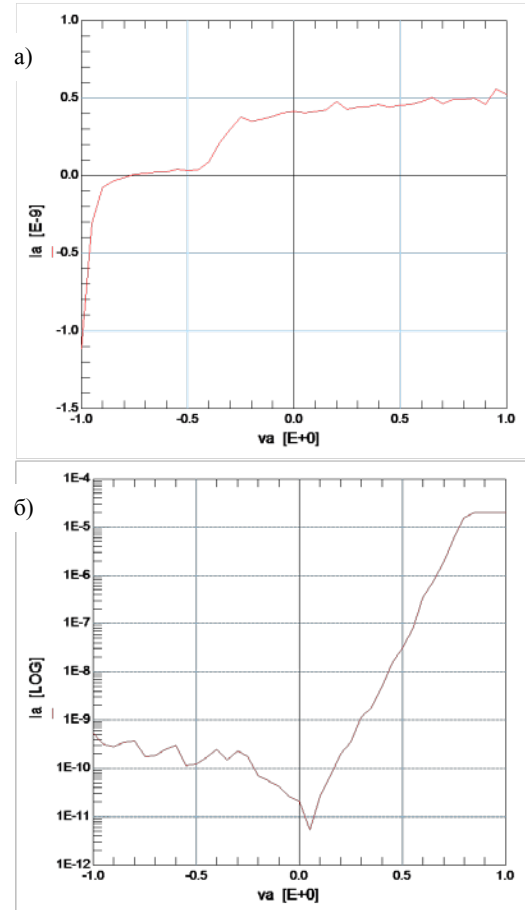


Рис. 9. ВАХ стоковых диодов транзисторов: а) транзистор M5 с фотоэмиссией; б) транзистор M5 из годной ячейки.

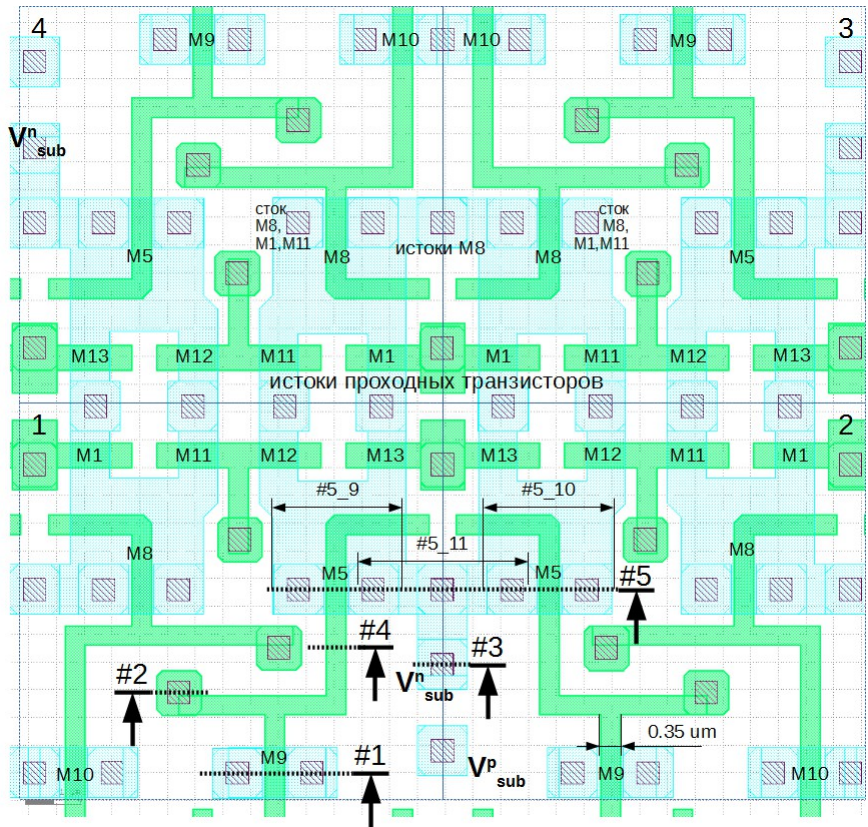


Рис.10. Топология четырёх смежных боковых ячеек кристалла с указанием изготовленных на установке FIB/SEM поперечных сечений.

нанозондов, так и рельеф самих столбиков, получившийся в результате удаления слоёв. При возникновении омического контакта появляется типичная ВАХ диода, ограниченная последовательным сопротивлением контактов.

Исследования методами FIB и электронной микроскопии.

Для локальных электронно-микроскопических исследований с использованием фокусированного ионного пучка (FIB) был выбран характерный дефект на кристалле, который подтвердился всеми задействованными ранее методами исследования – четыре неработоспособные смежные ячейки «ACTIVE», которая даже визуально отличается от аналогичных областей в работоспособных ячейках. Исследование проводилось в центре коллективного пользования «Диагностика и модификация микроструктур и нанообъектов» НИУ «МИЭТ».

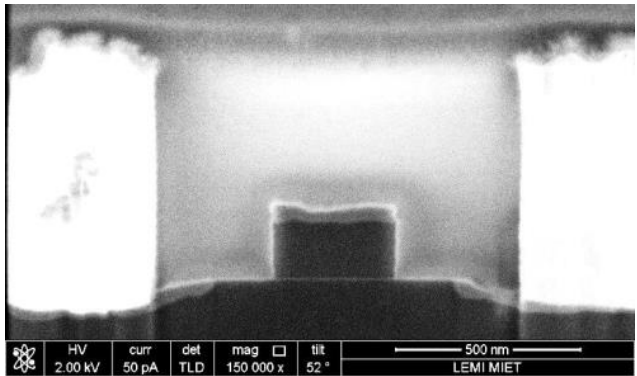


Рис.11. Сечение №1, р-канальный транзистор M9, увеличение 150000х.

По результатам нанозондирования транзисторов в дефектных ячейках были определены области формирования поперечных сечений для последующих электронно-микроскопических исследований.

На рис. 10 обозначены расположение и границы выполненных поперечных сечений структуры с помощью FIB.

Существенных отклонений в структуре p-канального транзистора M9 в ячейке №1 не обнаружено (рис. 11), в том числе и с контактом к общей для M5 и M9 затворной шине (сечение №2, не приводится). Вероятно, отсутствовал контакт нанозонда(ов) с вольфрамовыми столбиком(ами). Силицид на поликремниевой шине был обнаружен на обоих сечениях (№2 и №4).

Из рис. 12 и 13 видно, что силицид титана практически везде отсутствует, за исключением небольших островков, что объясняет «пятнистую» текстуру данной области при визуальном контроле. Под контактом к карману силицида

титана нет даже по периметру столбика, что может весьма негативно (привести к кинк-подобному эффекту) сказаться на характеристиках даже тех транзисторов, контакты к которым сохранились.

Причина отсутствия силицида титана заключается в наличии остатков вспомогательного слоя поликремния, используемого при формировании STI-изоляции. При дальнейших обработках слой поликремния должен быть полностью удален. Электронно-микроскопические исследования показали, что в данном кристалле присутствуют локальные тонкие слои неудаленного поликремния, а под ним остаётся слой screen-окисла порядка 150Å, который и препятствует формированию силицида титана, даже если остатки поликремния были удалены на последующих операциях. Более того, на рис. 14 и 15

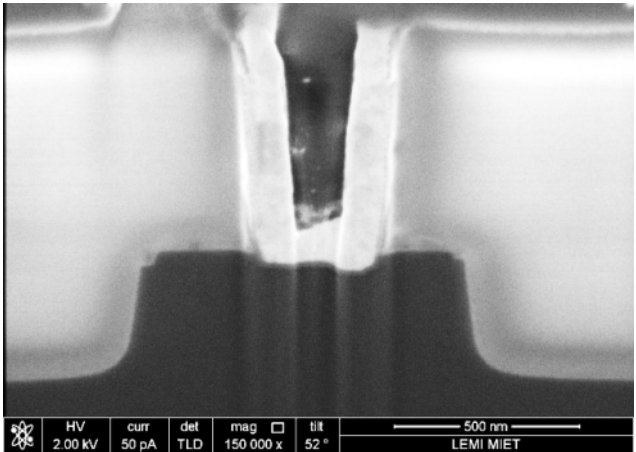


Рис.12. Сечение №3, контакт к карману п-канальных транзисторов M1, M5, M11-13.

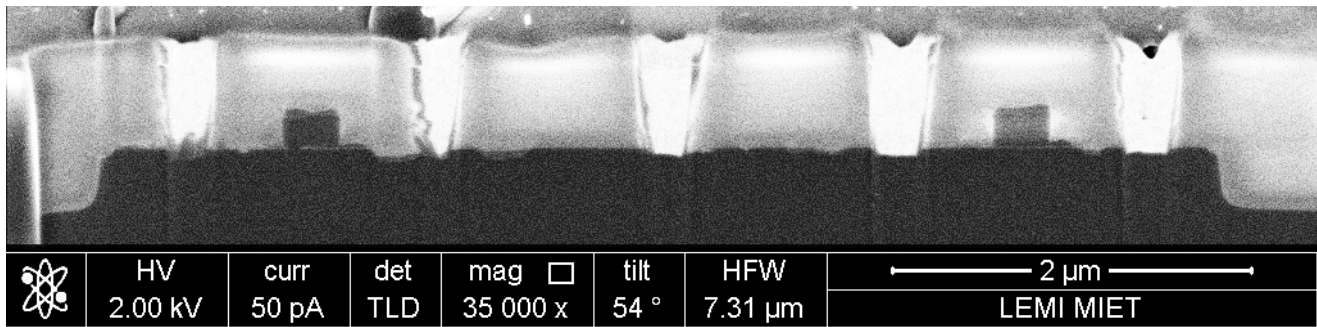


Рис.13. Сечение №5. Общий план контактов к стокам и истокам транзисторов M5 двух смежных ячеек.

показаны обнаруженные остатки слоя поликремния и screen-окисла непосредственно в подзатворной структуре транзисторов, замаскированные поликремниевой шиной и спейсером от удаления.

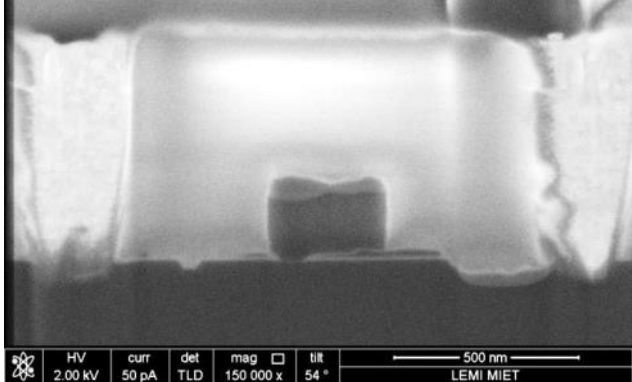


Рис.14. Сечение №5_9. Крупный план транзистора M5 в левой ячейке, у которого была обнаружена фотоэмиссия.

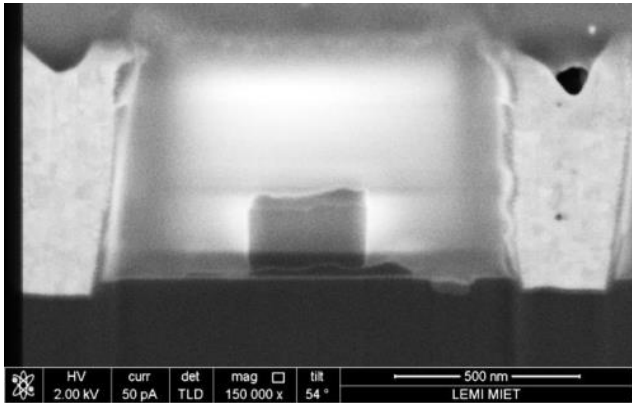


Рис.15. Сечение №5_10. Крупный план транзистора M5 в правой ячейке.

Обнаруженные структурные дефекты могут приводить к совершенно непредсказуемому поведению ячейки кэш-памяти в зависимости от площади, занимаемой неуда­ленными слоями под затвором, от сопротивления и линейности контактов к стокам, истокам и карману. Это может приводить к «залипанию» триггера ячейки, невозможности его переключения проходными транзисторами, зависимости от напряжения питания, нестабильной (невоспроизводимой) работе, а также к полному отказу ячейки памяти.

В таблице 1 представлены все обнаруженные конструктивные несоответствия, зафиксированные по результатам проведенного анализа в области четырех неработоспособных ячеек.

Таблица 1.

Перечень выявленных конструктивных несоответствий в различных поперечных сечениях в области неработоспособных ячеек.

№ п/п	Несоответствие	№ сечения (рис.10)
1	Полное отсутствие слоя TiSi	3
2	Островковое образование TiSi из-за оставшегося screen-окисла под неуда­ленным поликремнием, который был удалён в результате последующих операций	5, 5_9, 5_10, 5_11
3	Наличие неуда­ленного слоя поликремния в транзисторах под затвором и спейсерами, которые сыграли роль маски и защитили поликремний от удаления	5_9, 5_10
4	Ямка анизотропного под­травливания кремния (область ACTIVE) между затвором и контактом к стоку	5_9

Закключение

В процессе проведения работы по выявлению причин неработоспособности кристаллов микросхем Ethernet-контроллера 1990ВГ3Т была разработана и успешно опробована методика комплексного прецизионного локального анализа дефектов в рабочей области кристалла.

Методика включает в себя четыре основных этапа:

1. Локализация дефекта. Для этой цели могут использоваться методы:
- регистрация фотоэмиссии в статическом режиме работы кристалла микросхемы;
 - обратное моделирование результатов функционального тестирования;
 - послойное удаление конструктивных слоев в сочетании с визуальным, в том числе электронно-микроскопическим контролем.
2. Совмещение физического расположения (изображения) локализованного дефекта с топологическим чертежом слоя, в котором он обнаружен.
3. Локальное электрофизическое тестирование элементов в области локализованного дефекта в электронном микроскопе с использованием нанозондов. По результатам тестирования – разработка плана эксперимента по электронно-микроскопическому исследованию локальных

поперечных сечений элементов с использованием FIB, SEM, TEM.

4. Проведение электронно-микроскопических исследований сформированных с использованием FIB поперечных сечений выбранных элементов и, при необходимости, энергодисперсионного элементного анализа. В случае недостаточности полученных результатов для выявления причины неработоспособности выявленных элементов – проведение электронно-микроскопических исследований «на просвет» путем изготовления ламели выбранного поперечного сечения с помощью FIB и ее исследования в просвечивающем электронном микроскопе.

В процессе анализа кристалла ИМС проведена локализация дефектной области кристалла тремя независимыми способами: фотоэмиссионным, обратным моделированием результатов функционального тестирования и разрушающим методом с последующим визуальным контролем. Все методы указали на одно и то же проблемное место в кэш-памяти кристалла микросхемы.

Обнаружено неустойчивое образование или полное отсутствие силицида титана на карманах в областях ACTIVE. В транзисторной структуре на данных карманах присутствуют дополнительные слои поликремния и SiO₂ под затвором, имеющие разную остаточную толщину и площадь под затвором.

Источником обнаруженных дефектов является не полностью удаленный поликремний, осаждаемый на исходную пластину при подготовке к формированию STI (мелкощелевой изоляции).

Данный технологический дефект, в зависимости от конечного проявления после формирования транзисторной структуры, способен приводить к любому из обнаруженных типов сбоев при функциональном тестировании микросхем данного типа.

Для своевременной отбраковки обнаруженных типов технологических дефектов в рамках операционного контроля необходимо проведение дополнительной операции автоматизированного поиска дефектов топологического рисунка ввиду возможности их оптического обнаружения.

Malfunction investigation of Ethernet controller dies 1990VG3T

V.Y. Troitskiy, M.V. Oreshkov, A.A. Zakharchenko, A.P. Trepalin, T.V. Petlitskaya, V.A. Filipenya.

Abstract. Thorough multistage malfunction investigation of integrated circuit (IC) is carried out in this work. The IC is Ethernet controller 1990VG3T. Several consecutive non-destructive and destructive methods are used during investigation:

- defect localization by functional testing and by static photoemission detection;
- consecutive removal of metallization and dielectric layers;
- electro physical measurements by nano-probes;
- focused ion beam SEM research of selected areas.

Important results are obtained which reveal technological cause of IC malfunction.

Keywords: failure analysis, defect localization

Литература

1. А.А. Захарченко, М.В. Орешков, В.Ю. Троицкий. Анализ дефектов интегральных микросхем с помощью фотонно-эмиссионной микроскопии. Труды НИИСИ РАН. Том 3, №1, 2013, с. 67-71
2. А.Н. Петлицкий, Д.В. Жигулин, А.К. Панфиленко и др. Анализ электрофизических параметров элементной базы субмикронных ИМС с использованием четырехзондового наноманипулятора. Труды II российско-белорусской научно-технической конференции «Элементная база отечественной радиоэлектроники: импортозамещение и применение» им. О.В. Лосева, под ред. А. Э. Рассадина. – Н.Новгород: Изд-во ННГУ им. Н.И. Лобачевского, 2015, с. 230-232
3. В.А. Пилипенко, В.А. Солодуха, А.Н. Петлицкий и др. Использование четырехзондового наноманипулятора для измерения вольтамперной характеристики биполярного n–p–n-транзистора. Труды XVII международной научно-практической конференции "Современные электронные и информационные технологии". Украина, Одесса, 23-27 мая 2016 г. – Одесса: Изд-во Политехпериодика, 2016, с.148-149

Разработка банка алгоритмов и основ языка описания проблемы с целью минимизации числа программных ошибок

Ю.А. Семенов¹, А.П. Овсянников², Т.В. Овсянникова³

¹ НИЦ «Курчатовский институт», ФГБУ «Государственный научный центр РФ, Институт Теоретической и Экспериментальной физики» - Московский физико-технический институт, Факультет нанотехнологии и информатики, Москва, Россия,
E-mail: semenov@itep.ru

^{2,3} ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Межведомственный Суперкомпьютерный центр, Москва, Россия,
E-mail's: ² apo@jscc.ru, ³ tat@jscc.ru

Аннотация: Программное обеспечение становится неотъемлемой составляющей всех современных технологий. По этой причине становится крайне важно минимизировать число программных ошибок. Рассмотрены некоторые пути, которые могут помочь решить эту проблему, включая возможность создания языка описания проблемы, а не алгоритма.

Ключевые слова: качество программного обеспечения, банк алгоритмов, язык описания проблемы

На современном уровне технического развития первостепенное значение приобретает вопрос надежности применяемых информационно-коммуникационных технологий. Ключевой проблемой являются ошибки программного обеспечения, причем количество этих ошибок растет с ростом объема программного кода и сложности программ. Одним из подходов решения этой проблемы является верификация программ, другим - разработка средств автоматизации программирования. Существуют различные отладчики, ищущие ошибки синтаксиса языка программирования, разрабатываются статические и динамические код-чекеры, призванные выявлять ошибки алгоритма программы. Код-чекеры разрабатываются в США уже более 30 лет, есть и отечественные разработки. Достигнут заметный прогресс, но число ошибок только растет. Это связано с быстрым увеличением сложности программ.

Ядро ОС Windows NT 3.1 имело 4,5 миллионов строк кода, версия 3.5 – в 1994 г. – 7,5 млн. строк кода. Windows XP (2001r) содержит в себе 45 миллионов строк кода, практически все версии ядра Linux содержат столько же или даже больше строк кода (тезисы доклада Иванникова В.П. на семинаре ВМК МГУ). Число ошибок в коммерческих программах компании Microsoft составляет 0,5 ошибок на 1000 строк кода, отсюда предполагаемое число ошибок в ядре ОС Windows XP превышает 20000. Ошибки программного обеспечения достаточно критичны в основных областях применения компьютеров: проектировании и моделировании, бизнес-процессах, банковской сфере и др.; в системах управления реального времени, управления медицинским или промышленным оборудованием цена программной ошибки неизмеримо выше. Актуальность минимизации числа программных ошибок очевидна.

По данным института NIST программные ошибки обходятся экономике США в 22 млрд. долларов в год [1]. Важным параметром качества программы становится число ошибок на 1000 строк кода.

Возможным шагом на пути сокращения числа программных ошибок может стать минимизация участия программиста в создании программы путем использования **мета языка программирования** (meta language - **M_Lan**), который будет служить не для описания алгоритма, а для описания стоящей проблемы. Почему такой путь может гарантировать уменьшение числа ошибок? Мы сегодня с высокой степенью достоверности уверены в отсутствии ошибок в программах вычисления значений стандартных функций (тригонометрических, логарифмов и пр.). Это связано с тем, что они проверены десятками тысяч пользователей для огромного многообразия аргументов. После создания банка алгоритмов и начала его использования достаточно быстро будут выявлены и устранены ошибки из этих алгоритмов, так как они будут опробованы тысячами пользователей при решении самых разных задач во всем возможном диапазоне параметров.

Алгоритмические модули, лежащие в банке данных, должны иметь максимально возможную универсальность, ограниченные размеры и максимально возможную прозрачность (понятность). Кроме самих текстов программ в базе данных должны содержаться их описания, диапазоны параметров, возможная диагностика и идентификатор алгоритмического языка, на котором программа написана, имена библиотек, используемых программой, идентификатор автора, дата создания, даты и авторы всех коррекций.

При создании базовых модулей разработчикам следует пользоваться правилами Хольцмана для программистов [2], это снизит вероятности ошибок.

Правила Холлмана использовались при написании программ управления американскими шатлами, где была достигнута вероятность ошибки менее 0,003 на 1000 строк кода.

Для каждой области (физика, биология, химия, лингвистика, экономика, медицина, инженерия и т.д.) может быть создан банк известных алгоритмов, уже опробованных для решения различных задач в данной области.

Стандартное описание алгоритма сегодня - это текст программы на одном из известных алгоритмических языков. Но использование этих описаний сопряжено с определенными трудностями. Эти описания содержат внешние вставки, макросы и ссылки на специфические библиотечные программы, вызовы других подпрограмм или функций. Кроме того, подавляющее их число не имеет сколько-нибудь подробного описания. При создании базы данных алгоритмов надо будет решить эти проблемы. Каждому алгоритму должно быть присвоено уникальное имя-идентификатор (ALID).

Банк алгоритмов может стать мишенью сетевых атак. Препятствовать этим атакам будет регистрация всех пользователей и их сертификация. Любые запросы от несертифицированных пользователей будут отклоняться. Вообще сетевые пакеты, адресованные базе данных от IP незарегистрированных пользователей, должны блокироваться и не пропускаться к серверу банка алгоритмов.

Ситуация усложняется тем, что существующие алгоритмы могут быть привязаны к определенным алгоритмическим языкам (Perl, Java, Python, C, C++, Cobol, ProLog, SmallTalk, и т.д.). В некоторых случаях, например, для обработки прерываний используются программы, написанные на ассемблере конкретного процессора. По существу M_Lap будет формировать исполняемый образ программы на основе библиотеки описаний алгоритмов.

В идеале нужна машина с искусственным интеллектом, только такой компьютер сможет создавать новые алгоритмы, которых нет в банке. Здесь уместно заметить, что нужна машина с высоким уровнем интеллекта. По американским прогнозам

такая машина появится в 2023 году, а к 2045 году будет создана персональная машина, существенно превосходящая человека по интеллекту [3]. Будет ли такая машина всегда безошибочна? Если вероятность ее ошибки будет равна вероятности ошибки человека, пользы от нее для минимизации программных ошибок не будет.

Но начинать можно и без этого. Ведь очень часто создаются программы, алгоритмы работы которых известны, и нужно лишь адаптировать их к требованиям конкретной задачи. Решением проблемы может быть суперпозиция известных алгоритмов.

В процессе реализации будут созданы библиотеки с открытым кодом, которые должны быть доступны для использования всем желающим. Это позволит со временем устранить из них имеющиеся ошибки.

Некоторые алгоритмы носят универсальный характер (статистическая обработка, аппроксимация, вычисление стандартных функций или интегралов, компьютерная графика и пр.). Такие алгоритмы могут храниться в одном из банков удаленных ветвей семантического дерева. По этой причине нужно будет создать систему универсальных идентификаторов алгоритмов и общую базу данных, обращение к которой упростит поиск нужного алгоритма.

В перспективе описание проблемы должно проводиться на естественном языке, как это делается в машине Watson (IBM) [4]. Для этого программа должна научиться распознавать контекст проблемы. На начальном этапе контекст может выявляться в результате диалога между машиной и программистом.

Все многообразие проблем будет характеризоваться семантическим деревом. В каждом узле этого дерева будет размещаться банк алгоритмов, которые соответствуют контексту этого узла. Таким образом, банк алгоритмов будет распределенным.

Учитывая многообразие отраслей и алгоритмов, могут потребоваться программные интерфейсы для согласования описаний алгоритмов из разных сфер программирования (см. рис.1).



Рис. 1. Дерево отраслей науки и проблем

После того как контекст проблемы определен, начинается описание конкретной задачи. Эта часть

может выполняться также в диалоговом режиме. После того как задача описана, машине передаются

входные параметры задачи. Машина может запрашивать дополнительные данные, необходимые для решения или уведомлять об отсутствии нужного алгоритма или данных. В языке может быть предусмотрена возможность описания объектов, методов и атрибутов для конкретной проблемы.

Формат описания алгоритма может иметь несколько форматов:

- Текст программы-функции, для реализации которой необходимы только входные параметры.
- Описание последовательности операций, которые надо выполнить для формирования текста программы, реализующей алгоритм.

Если алгоритма в банке алгоритмов нет, программа может быть создана самим программистом в соответствии с определенными правилами и помещена в банк алгоритмов с указанием имени, адреса автора, даты занесения, а также подробного описания. Этому алгоритму присваивается уникальный идентификатор.

Все тексты описаний алгоритмов должны быть *open source*. Создатель описания алгоритма и все программисты, внесшие свои вклады должны быть зарегистрированы и им присваиваются уникальные идентификаторы (**PRID**). В базе данных фиксируется время создания описания, авторы и моменты внесения всех правок, а также идентификаторы программистов, их внесших. В базу данных записываются все случаи использования алгоритма и параметры использования. Должны сохраняться несколько версий описаний с идентификаторами контрибуторов. Данная процедура необходима для предотвращения преднамеренного внесения вредоносных вставок в программы. Создается форум, на котором обсуждаются алгоритмы и случаи их использования, предложения по их совершенствованию. При поступлении предложения нового алгоритма или модификации уже существующего, соответствующая программа попадает в банк кандидатов, откуда ее может извлечь любой желающий для использования. После определенного периода апробации или после оговоренного числа применений описание алгоритма (или библиотечная функция) переносится в основной банк алгоритмов.

Программы-функции должны быть написаны на одном языке или совокупности языков, допускающих кросс-компиляцию или взаимные вызовы с совместимостью по входным и выходным данным.

- Алгоритмы представляются в виде стандартных библиотек. Компьютер анализирует текст описания проблемы на *M_Lan* и формирует исполняемый образ программы на основе имеющихся библиотек. При этом возможен диалог программист-компьютер с целью уточнения задачи, входных параметров или данных.

- После создания обширного банка алгоритмов и разработки методов подбора алгоритмов на основе текста программы на *M_Lan*

начинается следующий этап. Разрабатывается техника формирования *M_Lan*-программы на основе определенного контекста проблемы и алгоритмов, найденных в банке. Использование тех или иных алгоритмов из банка может также согласовываться с программистом.

- При наличии компьютера с искусственным интеллектом, который превосходит уровень среднего программиста и при условии, что он не может делать ошибки (не ясно, достижимо ли это), задача формулируется на естественном языке. Если нужно он задает уточняющие вопросы постановщику задачи. Компьютер анализирует текст на *M_Lan* и подбирает список нужных алгоритмов. При подборе алгоритма возможен диалог программист-компьютер. В результате компьютер формирует исполняемый образ программы.

Идентификатор контекста C_p в дальнейшем является индексом при выборе базы данных описаний алгоритмов или библиотеки функций (из базы баз алгоритмов).

Параметрами программы могут быть имена других программ. Система должна иметь встроенный механизм контроля входных параметров. Число вложений программ не лимитируется.

Список функций-подпрограмм-алгоритмов параметров обращения имеет вид счетного массива: число, за которым следует список из n имен функций-подпрограмм-алгоритмов (f_names), разделенных запятыми. Число характеризует количество элементов в списке. Допускается и одно имя в формате " $1, f_name$ ". У f_name в свою очередь могут быть параметры, значения которых записываются в скобках. Например, $poly(x_1, x_2, \dots, x_n)$, если число параметров больше или меньше оговоренного, выдается уведомление об ошибке. Рассмотрим пример. Имеется функция-подпрограмма $Fun=aX^2+bY+c$. Возможно обращение:

$A=Fun(a=10; X=(x=1); b=5; Y=(y=0.5); c=1);$

Тогда будет вычислено значение полинома второго порядка. Но возможно обращение и:

$A=Fun(a=10; X=\sin(x=1); b=5; Y=\cos(x=0.5); c=1).$

В идеале текст программы будет выглядеть как цепочка вызовов базовых модулей, некоторые из которых содержат в себе вызовы каких-то других базовых модулей.

В предлагаемом языке определяются **объекты**, над которыми должны быть совершены определенные действия:

Файл или список файлов (напр., `file=access_log; file=error_log`; или `(2, access_log; file=error_log;)`)

- Каталог или список каталогов
- Вектор или список векторов
- Матрицы
- Функции

- Переменные

У объектов могут быть атрибуты, которые характеризуют их свойства и свойства составляющих их элементов. Объекты могут иметь также определенные параметры-аргументы.

Операторы - определяют то, что должно быть проделано с объектами.

Операторами могут быть идентификаторы алгоритмов из базы данных. Объекты являются параметрами операторов.

Представление – описывает форму выдаваемого результата, например:

- Таблицы
- Графические образы
- Числа
- Массивы чисел
- Файлы
- Каталоги
-

У представлений могут быть атрибуты, определяющие их свойства и свойства составляющих их элементов.

Именем представления может быть идентификатор из базы данных алгоритмов.

Базовые модули алгоритмов должны иметь ограниченный размер (согласно правилам Хольцмана ~60 строк).

Вероятность ошибки в таком модуле будет пропорциональна длине модуля и будет падать по мере его использования большим числом пользователей (ошибки выявляются и устраняются по мере использования).

Если предположить, что в базовом модуле ошибок нет, то снижение вероятности программной ошибки при работе с модулем будет уменьшена как минимум в 60 раз (ведь ошибка будет возможна только в строке вызова).

Если программа содержит N обращений к разным базовым модулям, то вероятность программной ошибки W будет пропорциональна

$$W = \sum_{i=1}^N P_i + \sum_{i=1}^N m_i$$

где P_i – вероятность программной ошибки в i -ом модуле, а m_i – вероятность ошибки в строке обращения к базовому модулю. Первая сумма больше второй пропорционально отношению числа строк кода. Если модули программ имеют ~50-60 строк кода, то первая сумма в 50-60 раз больше. Здесь предполагается, что число ошибок пропорционально числу строк кода.

Первая сумма по мере использования базового программного модуля будет экспоненциально приближаться к нулю. Вторая сумма будет оставаться почти неизменной, хотя по мере использования программы и она будет уменьшаться.

Само значение P_i зависит от числа строк кода в модуле:

$$P_i = \sum_{j=1}^K e_j,$$

где K – число строк кода в модуле, e_j – вероятность программной ошибки в строке j . Следует учитывать, что точного числа ошибок как в отдельной строке, так и в программном модуле мы никогда не знаем. Приводимые цифры числа ошибок на 1000 строк кода являются эмпирическими оценками.

Вначале, когда базовые алгоритмы-функции использованы ограниченное число раз, вероятность наличия программной ошибки в тексте может быть аналогичным с традиционными методами программирования, но со временем, когда число ошибок в базовых модулях упадет, можно ожидать падения числа программных ошибок в 10 и более раз. Удастся ли улучшить это отношение в случае появления искусственного интеллекта, покажет время.

Development of the algorithm bank and basics of the language for problem description to minimize a number of program errors

Yu.A.Semenov, A.P.Ovsyannikov, T.V.Ovsyannikova

Abstract: As the software becomes an inherent part of all used technologies, it is important to minimize a number of program errors. The article presents some considerations which can help to reduce a probability of program errors, including the possibility of development of a new language, describing problem, but not an algorithm.

Keywords: software quality, algorithm library, problem definition language

Литература

1. URL: http://book.itep.ru/depositary/security/software_security/open_source_quality_report.pdf. "Measuring software quality. A Study of Open Source Software", Ben Chilf, Coverity, Inc
2. URL: http://book.itep.ru/10/holz_rules.htm. Правила Хольцмана
3. "Когда компьютеры станут умнее человека", Chip 09/14, стр. 24-25
4. URL: <http://book.itep.ru/10/supercomp.htm#2>. Компьютер Watson

Внедрение в библиотеку FFTW собственных модулей, применяющих специализированный сопроцессор для ускорения операций преобразования Фурье

А.А. Бурцев

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: burtsev@niisi.msk.ru

Аннотация: В составе созданного в ФГУ ФНЦ НИИСИ РАН микропроцессора BM8 семейства КОМДИВ MIPS-архитектуры предусмотрен арифметический специализированный 128-разрядный сопроцессор, позволяющий ускорять вычисления над векторами комплексных и вещественных чисел одинарной и двойной точности. В статье представлены результаты внедрения в широко известную и общедоступную библиотеку FFTW собственных модулей, которые позволяют применять этот специализированный сопроцессор для повышения производительности операций преобразования Фурье над векторами комплексных чисел.

Ключевые слова: MIPS-архитектура, микропроцессоры семейства КОМДИВ, сопроцессор комплексной арифметики, векторный сопроцессор (CPV), цифровая обработка сигналов (ЦОС), операция быстрого преобразования Фурье (БПФ), библиотека FFTW.

Введение

Для обеспечения информационной безопасности страны необходимо создавать свои собственные высокопроизводительные вычислительные системы [1,2], способные решать сверхсложные научно-технические задачи за приемлемое время и при соблюдении жёстких требований к условиям их эксплуатации.

В ФНЦ НИИСИ РАН, где уже на протяжении многих лет создаётся семейство отечественных высокопроизводительных микропроцессоров КОМДИВ, наряду с универсальными процессорами MIPS-архитектуры разрабатывает также семейство специализированных сопроцессоров, ориентированных на ускоренное исполнение определённого набора математических операций. Такой подход к повышению производительности вычислительных систем в НИИСИ РАН получил название встречной оптимизации [1,с.8].

Так, при создании 64-разрядного микропроцессора 1890BM6 (семейства КОМДИВ) в его составе наряду с обычным сопроцессором плавающей вещественной арифметики (CP1) был предусмотрен 64-разрядный сопроцессор комплексной арифметики (CP2). А при дальнейшем развитии этот CP2 был расширен до 128-разрядов и дополнен рядом полезных команд, позволяющих ускорять исполнение ряда типичных операций обработки векторов и матриц. Получившийся в результате 128-разрядный сопроцессор, условно названный векторным (CPV), был включён в состав нового высокопроизводительного микропроцессора 1890BM8.

Применение такого CPV позволяет значительно ускорить типичные функции обработки векторов и матриц, характерные для задач линейной алгебры и цифровой обработки сигналов. И путём разработки ряда экспериментальных тестовых программ этот вывод был реально подтверждён на практике [3,4].

Но чтобы такие возможности CPV поставить на службу широкому кругу потенциальных пользователей микропроцессоров семейства КОМДИВ, необходимо

также обеспечить и программную поддержку для его применения, т.е. снабдить CPV сопутствующим ПО и при необходимости включить его в систему общего программного оснащения, предусмотренного для эксплуатации на микропроцессорах данного семейства.

В настоящее время создание прикладных программ для решения задач цифровой обработки сигналов (ЦОС) и/или цифровой обработки изображений (ЦОИ) редко обходится без использования какой-либо хорошо зарекомендовавшей себя библиотеки функций, ориентированной на данные области применения. Одной из таких широко известных и свободно распространяемых библиотек является библиотека FFTW [5], предназначенная для выполнения эффективных преобразований Фурье над комплексными и вещественными числами, организованными в массивы любой размерности. Исходный код этой библиотеки (на языке Си), а также подробное руководство по её использованию доступны (для скачивания) в Интернете [6].

Библиотека FFTW может функционировать на компьютерах самой разнообразной архитектуры и под различными ОС. В частности, её можно сконфигурировать для работы под ОС Linux на процессорах MIPS-архитектуры. Это позволяет использовать библиотеку FFTW для решения задач ЦОС и ЦОИ на микропроцессорах семейства КОМДИВ.

Чтобы с помощью CPV значительно повысить производительность решения задач, предполагающих многочисленные операции преобразования Фурье, вызовы которых реализуются функциями библиотеки FFTW, требуется обеспечить применение в этих функциях высокоэффективных команд сопроцессора CPV. Другими словами, необходимо решить проблему адаптации библиотеки FFTW к CPV, т.е. оптимальным способом настроить её на применение CPV в тех случаях, когда это позволяет ускорить исполнение заданной операции преобразования Фурье.

В данной статье предлагается один из возможных путей решения этой проблемы и описываются резуль-

таты работы по внедрению в библиотеку FFTW собственных модулей, выполняющих операции преобразования Фурье над векторами комплексных чисел с применением специализированного сопроцессора CPV.

1. Краткая характеристика CPV

Векторный сопроцессор (CPV) содержит 64 128-битных регистра, в каждом из которых можно хранить:

- 1 комплексное число двойной точности (DC);
- 2 комплексных числа одинарной точности (SC);
- 2 вещественных числа двойной точности (DR);
- 4 вещественных числа одинарной точности (SR).

Для каждого из этих 4-х форматов значений, помещённых в его регистры, CPV обеспечивает соответствующие ему вычислительные команды. Например, каждая из команд умножения с накоплением (см. таблицу 1) выполняет свойственную ей операцию для одной тройки (z,y,x) величин типа DC, либо двух троек типа SC или DR, либо сразу для четырёх троек типа SR.

Таблица 1. Вычислительные команды группы умножения

1 DC (9)*	2 SC (7)*	2 DR (5)*	4 SR (4)*	cmd z,y,x
cmul.d	cmul.s	vmul.d	vmul.s	$z=y \cdot x$
cmadd.d	cmadd.s	vmadd.d	vmadd.s	$z=z+y \cdot x$
cmsub.d	cmsub.s	vmsub.d	vmsub.s	$z=z-y \cdot x$
cmaddsub.d	cmaddsub.s	vmaddsub.d	vmaddsub.s	$z=z+y \cdot x$
(t) - длительность в тактах (если команда уже в кэше)				$y=z-y \cdot x$

Длительность вычислительных команд зависит от типа обрабатываемых величин (см. в скобках: 4 такта для SR, 5 для DR, 7 для SC и 9 для DC). Но поскольку CPV разрешает на каждом такте начинать исполнение новой команды вычислительного потока, то в итоге можно добиться такой максимальной производительности CPV, при которой n его вычислительных команд смогут исполниться всего за n+8 тактов.

Команды CPV для работы с памятью (см. таблицу 2) позволяют загрузить 128-битное значение целиком в регистр или заполнить его старшую и младшую половину 64-битными значениями по отдельности. Можно одной командой (vldq) загрузить два соседних регистра двумя 128-битными смежными словами из памяти. Аналогичные команды (vsd, vsdm, vsdq) предусмотрены и для сохранения значений регистров в памяти.

Таблица 2. Команды работы с памятью

	команды загрузки	команды сохранения
256 / 32 L2(5)*	vldq, vldqx	vsdq, vsdqx
64 / 8 L1(3)*	vld, vldh, vldx, vldhx, vldlx	vsd, vsdh, vsdx, vsdhx
128 / 16 L1(3)* *если данные уже в L1(L2)-кэше	vldm, vlidd, vliddh, vldmx, vliddx, vliddhx, vliddlx	vsdm, vsdd, vsddh, vsdmx, vsddx, vsddhx

Для указания виртуального адреса в этих командах можно использовать базовую адресацию с относительным смещением (vld) или базовую индексную (vldx). Получаемый адрес должен быть выровнен на границу обрабатываемого слова, т.е. кратен 8, 16 или 32 (см./d).

Архитектурой КОМДИВ предусмотрена возможность в каждом такте взять на исполнение две очередные команды, если эти команды разных потоков. Это позволяет совместить во времени вычислительные команды CPV над одной группой данных с командами

CPV для загрузки/сохранения в/из памяти другой группы данных.

Самая высокопроизводительная команда (**cmaddsub**), предусмотренная в CPV над комплексными числами, осуществляет 10 арифметических операций (4 умножения и 6 сложений) над вещественными числами двойной точности или 20 арифметических операций над вещественными числами одинарной точности. Именно при её исполнении CPV достигает пика своей производительности (16 ГФлопс на частоте 800 Мгц).

2. Ускорение БПФ (FFT) на CPV

Команда **cmaddsub** предназначена для аппаратной реализации базовой операции цифровой обработки сигналов — так называемой бабочки Фурье (БФ).

Бабочкой Фурье BF(A,B,V) с коэффициентом V над комплексными величинами A и B называется такая единая операция, в результате которой новые значения A' и B' для этих величин вычисляются на основе их предыдущих значений по формулам (см. рис. 1):

$$A' = A + B \times V, B' = A - B \times V$$

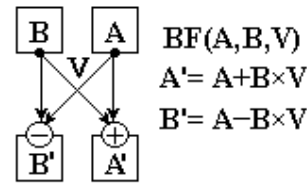


Рис. 1. Операция «бабочка Фурье»

Эта операция БФ является ключевой при реализации целого класса алгоритмов, известных под общим названием БПФ — Быстрое Преобразование Фурье (FFT — Fastest Fourier Transform).

Для заданного вектора комплексных чисел X_n длиной n результатом одномерного дискретного преобразования Фурье (ДПФ или DFT) является комплексный вектор Y_n , элементы которого Y_m ($m=0,1,\dots,n-1$) вычисляются по формуле (см. Ф1):

$$\{F1\} Y_m = \sum \{X_k \cdot w_n^{m \cdot k}\} \quad k=0,1,\dots,n-1$$

$$\text{где } w_n^{m \cdot k} = \exp(-I \cdot m \cdot k \cdot 2\pi/n) \quad (I - \text{мнимая единица}).$$

Вычисление ДПФ непосредственно по этой формуле требует порядка $O(n^2)$ арифметических операций с вещественными числами. А алгоритмы БПФ позволяют вычислить ДПФ всего за $O(n \cdot \log_2 n)$ арифметических операций. И львиную долю вычислений в таких алгоритмах занимают операции БФ, многократно исполняемые над различными парами элементов вектора. Поэтому ускорение (а тем более аппаратная реализация) операции БФ позволяет существенно поднять общую производительность БПФ (FFT).

Для реализации БПФ на CPV был опробован вариант известного алгоритма Кули-Тьюки с прореживанием по времени для длины $n=2^p$ (степени двойки). В результате выполненной реализации (подробное описание которой приведено в [4]) было подтверждено на практике, что с помощью сопроцессора CPV можно существенно ускорить операцию БПФ над векторами комплексных чисел (по сравнению с использованием обычного сопроцессора CP1 плавающей вещественной

арифметики). Для операций БПФ над непрерывными векторами комплексных чисел двойной точности длиной $n=2^p > 32$, размещённых в кэш-памяти, удастся добиться ускорения в 8-9 раз, а для таких же векторов длиной 32 можно достичь даже 20-кратного ускорения.

3. Библиотека FFTW

Библиотека FFTW (FFT in the West) представляет собой определённым образом организованную коллекцию Си-функций, реализующих разнообразные методы вычисления ДПФ (DFT) над массивами комплексных и вещественных чисел любой длины и размерности. Причём FFTW позволяет обрабатывать не только непрерывные массивы, элементы которых располагаются в памяти друг за другом, но и любые произвольные массивы, элементы которых размещаются в памяти с некоторым шагом – страйдом. (Примерами таких массивов могут служить столбцы матрицы, объявленной в Си-программе).

Поскольку библиотека FFTW написана на языке Си, она может быть скомпонована для функционирования на любой компьютерной платформе, для которой имеется подходящий Си-компилятор. В зависимости от выбранной точности представления вещественных чисел (float, double, long double) для одной и той же платформы (компьютерной архитектуры и ОС) можно скомпоновать три различных версии библиотеки: для одинарной (fftwf), двойной (fftw) и учетверённой (fftwl) точности.

При генерации конкретной версии библиотеки можно также указать, будут ли использоваться специальные возможности системы команд (например, расширенные наборы SSE, SSE2, AVX, AltiVec, MIPS PS) или особые возможности ОС для параллельного (multithread) и распределённого программирования (MPI).

Главная отличительная особенность библиотеки FFTW заключается в её способности самоадаптации к нуждам пользователя. Всякий раз из совокупности предусмотренных в ней методов (алгоритмов) вычисления ДПФ она сама подбирает в зависимости от требований пользователя наиболее оптимально отвечающий его нуждам метод, который на используемой компьютерной платформе позволяет выполнить ДПФ над заданным набором данных за минимальное время.

Подбор нужного метода преобразования или FFT-плана (в терминологии библиотеки FFTW) осуществляет специальная программная компонента библиотеки — так называемый планировщик (the planner). В FFTW собрана богатейшая коллекция методов реализации FFT-преобразований, что позволяет планировщику для любого размера данных N создать FFT-план оптимальной сложности $O(N \cdot \log_2 N)$.

Всякая программа, использующая библиотеку FFTW, должна сначала вызвать планировщик для формирования плана требуемого FFT-преобразования. А затем такой однажды сформированный FFT-план можно уже исполнять многократно, причём над многими данными. Эти данные могут быть расположены в различных местах памяти, но они должны быть того же самого размера и иметь такую же структуру, которые были заявлены при формировании плана. Описа-

ние набора параметров, характеризующих, какое именно FFT-преобразование и над какой структурой данных необходимо выполнить, называется FFT-задачей (в терминологии библиотеки FFTW).

Предлагаемые для использования функции библиотеки в зависимости от трудоёмкости их интерфейса распределены на 3 уровня сложности: 1) базовый (basic), 2) продвинутый (advanced) и 3) гурзу (guru). Функции 1-го уровня позволяют выполнить одиночное преобразование Фурье над массивом данных, непрерывно расположенных в памяти. Функции 2-го уровня допускают множественные преобразования Фурье над произвольно расположенными массивами, элементы которых следуют в памяти с некоторым шагом. С помощью функций 3-го уровня можно выполнять преобразования Фурье над массивами данных любой структуры, предусмотренной библиотекой.

Рассмотрим пример простой программы, использующей библиотеку FFTW (версии 3.2.2):

```
#include <fftw3.h>
...
fftw_complex *in, *out;
fftw_plan p; int dir; unsigned int flags;
... /* точка 0 */
in=(fftw_complex*)fftw_malloc
    (sizeof(fftw_complex)*N);
out=(fftw_complex*)fftw_malloc
    (sizeof(fftw_complex)*N);
dir=FFTW_FORWARD; flags=FFTW_ESTIMATE;
p=fftw_plan_dft_1d(N, in, out, dir, flags);
... /* точка 1 */
fftw_execute(p); // исполнение FFT-плана
... /* точка 2 */
fftw_destroy_plan(p);
fftw_free(in); fftw_free(out);
```

В ней выполняется одномерное прямое преобразование Фурье длиной N над комплексными числами двойной точности. По указателям `in` и `out` в результате вызовов функции `fftw_malloc` динамически выделяется память для хранения двух массивов комплексного типа `fftw_complex`: входного `in`, куда должны помещаться исходные данные (в точке 1), и выходного `out`, куда принимаются результаты FFT-преобразования (доступные в точке 2). В результате вызова функции `fftw_plan_dft_1d` в структуре данных типа `fftw_plan` по указателю `p` формируется план FFT-преобразования, который затем исполняется путём вызова (возможно многократного) функции `fftw_execute`.

Память, выделяемая функциями библиотеки для порождаемых ими объектов, гарантировано выравнивается по кратному адресу (требуемая кратность устанавливается при конфигурации библиотеки). При завершении программы все ресурсы, которые были выделены для функционирования FFT-плана и массивов данных, должны быть корректным образом возвращены, для чего требуется вызвать функции: `fftw_destroy_plan` и `fftw_free`.

Используя уже сформированный FFT-план `p`, можно выполнить аналогичное FFT-преобразование, но уже над другими массивами (X, Y), вызвав функцию:

```
fftw_execute_dft(p, X, Y);
```

Параметр `dir=FFTW_FORWARD` при вызове функции построения плана задаёт прямое направление преобразования Фурье. Для формирования плана обратного преобразования Фурье в качестве этого параметра `dir` следует задавать значение `FFTW_BACKWARD`.

В случае выполнения двумерных или трёхмерных преобразований Фурье над матрицами ($N \times M$ или $N \times M \times K$) комплексных чисел для формирования плана следует использовать соответствующие им функции:

```
p=fftw_plan_dft_2d(N,M,in,out,dir,flags);
p=fftw_plan_dft_3d(N,M,K,in,out,dir,flags);
```

А для создания плана многомерного преобразования Фурье (над массивом $L_1 \times L_2 \times \dots \times L_r$ комплексных чисел) предусмотрена универсальная функция:

```
int L[r]={L1,L2,... Lr};
p=fftw_plan_dft(r,L,in,out,dir,flags);
```

В библиотеке FFTW предусмотрены аналогичные функции и для выполнения разнообразных преобразований Фурье над массивами вещественных чисел. Но в данной статье они не рассматриваются.

Последним параметром при вызове всех рассмотренных функций формирования FFT-плана задаются флаги `flags` режима работы планировщика. Они определяют, насколько глубоко и тщательно должен быть проведён сравнительный анализ предусмотренных библиотекой различных методов исполнения преобразования Фурье при выработке оптимального FFT-плана с учётом конкретных заданных параметров подвергаемых обработке данных и особенностей системы команд имеющейся компьютерной архитектуры. Наиболее часто для `flags` используются значения `FFTW_ESTIMATE` и `FFTW_MEASURE`.

Значение `FFTW_ESTIMATE` позволяет быстро получить подходящий план FFT-преобразования. Но он может оказаться не самым оптимальным, т.к. в этом режиме просматривается некоторое разумно ограниченное подмножество методов исполнения FFT, причём каждый из них сравнивается с другими лишь по тем вычисленным заранее показателям производительности, которые характеризуют его теоретически. В качестве такого показателя принято считать количество машинных команд: сложений, умножений, умножений с накоплением и др., которые в результате придётся исполнить для выполнения преобразования Фурье принятым методом.

В другом режиме (`FFTW_MEASURE`) планировщик несколько раз запускает на выполнение каждый формируемый им FFT-план и оценивает его реальную производительность по некоторому усреднённому показателю. Это позволяет выбрать среди планов-претендентов тот, наилучшая производительность которого подтверждается практически. Но сам процесс поиска такого оптимального плана занимает довольно значительное время. К тому же применение этого режима предполагает использование памяти, выделенной под обрабатываемые массивы, и как следствие, накладывает жёсткое требование: исходные данные в обрабатываемый массив можно помещать только после завершения выработки оптимального FFT-плана.

Множество FFT-методов, подвергаемое анализу при поиске оптимального FFT-плана, можно существенно расширить, задав для `flags` значение `FFTW_PATIENT`. Возможно, это позволит найти более оптимальный план, но ценой значительного увеличения времени работы планировщика.

И наконец, задав для `flags` значение `FFTW_EXHAUSTIVE`, можно потребовать от планировщика найти самый оптимальный план для выполнения заказанного преобразования Фурье, проведя для этого исчерпывающий поиск с анализом всевозможных FFT-методов, предусмотренных в библиотеке FFTW, не считаясь с затраченным на это времени.

Программа, использующая библиотеку FFTW, может сохранять сформированные ею FFT-планы в особом хранилище, называемом `wisdom`. Вместе с каждым FFT-планом там сохраняются и параметры выполняемой им FFT-задачи. Содержимое этого хранилища можно записывать в файл и восстанавливать из файла. А при вызове функции построения плана установкой флага `FFTW_WISDOM_ONLY` можно сообщить планировщику, что необходимый оптимальный план вовсе не обязательно формировать заново, ибо возможно его удастся просто воссоздать из хранилища.

Это позволяет использовать библиотеку FFTW в том числе и для эффективного выполнения критичных по времени преобразований Фурье. Для этого надо лишь предварительно (с помощью вспомогательной программы) выработать все необходимые оптимальные FFT-планы и сохранить их в этом хранилище, чтобы затем в основной программе после их быстрого воссоздания сразу же запускать их на исполнение.

4. Механизм функционирования планировщика библиотеки FFTW

Познакомимся подробнее, как устроен и как функционирует планировщик библиотеки (`planner`), т.е. тот её важнейший механизм, который как раз и обеспечивает формирование оптимального плана для исполнения заказанного FFT-преобразования.

В качестве входного параметра планировщик получает дескриптор FFT-задачи (указатель на структуру данных типа `problem`), в котором описано, какое FFT-преобразование и над какими данными предстоит выполнить. А в качестве результата планировщик должен выдать дескриптор сформированного им оптимального FFT-плана (указатель на структуру данных типа `plan`), в котором содержатся характеризующие его величины, а также приготовлены необходимые таблицы и функции для его реального исполнения.

У планировщика имеется таблица так называемых решателей FFT-задачи. Решатель (`solver`) – это отдельный модуль, реализующий определённый метод выполнения FFT-преобразования. На этапе инициализации планировщика каждый решатель добавляет в эту таблицу запись с указателем на дескриптор решателя (структуру данных типа `solver`), таким образом регистрируя себя в качестве потенциального кандидата, способного выполнить FFT-задачу.

Будучи вызванным в работу и получив задание (дескриптор FFT-задачи), планировщик последовательно перебирает сформированную у себя таблицу решателей и вызывает у каждого из них специально для этого предусмотренную (в дескрипторе решателя) функцию построения плана `mkplan`, передавая ей дескриптор FFT-задачи в качестве параметра. Если решатель способен исполнить FFT-преобразование, описываемое этим дескриптором, то его функция `mkplan` должна сформировать дескриптор FFT-плана и вернуть указатель на него в качестве результата. Если же она вернёт пустой указатель, это будет означать, что данный решатель не способен выполнять такое FFT-преобразование.

Планировщик оценивает выдаваемые вызванными решателями FFT-планы и сравнивает их между собой, а затем выдаёт наилучший из них в качестве результата своей работы (или пустой указатель, если ни одного плана сформировать не удалось). Совокупность вызываемых решателей и критерии, которыми руководствуется планировщик для оценки и сравнения получаемых планов, зависят от флагов его режима работы.

Каждый из рассматриваемых далее дескрипторов по сути определяется как новый тип объектов – класс в терминах ООП, но в программе на языке Си объявляется как структура с полями, содержащими не только значения характеризующих его величин, но и указатели на функции, вызовы которых обеспечивают исполнение предусмотренных над ним операций.

Сразу отметим, что в каждом таком дескрипторе, как правило, предусмотрена функция `destroy` для его ликвидации, а у дескриптора задачи или плана — ещё и функция `print` для печати его характеристик.

4.1. Дескриптор FFT-задачи

В дескрипторе FFT-задачи (типа `problem`) в общем случае задаются следующие характеристики:

- 1) I/O-тензор $S=\{r,(n_1,i_1,o_1),\dots,(n_r,i_r,o_r)\}$, описывающий размерность r FFT-преобразования, а также размеры $n_1 \times n_2 \times \dots \times n_r$ (и границы) по каждому измерению, а также страйды $(i_1, \dots, i_r)$ и $(o_1, \dots, o_r)$ для входного и выходного массивов, в нём участвующих;
- 2) I/O-тензор V , описывающий группу массивов, над каждым из которых требуется выполнить FFT-преобразование, определяемое тензором S ;
- 3) указатели расположения в памяти начала группы входных I и выходных O массивов.

В простейшем случае, когда требуется исполнить одномерное FFT над одиночным непрерывным вектором X размером n , оставляя результат на том же месте (FFT в режиме `in-place`), такую FFT-задачу можно охарактеризовать так: $S=\{1,(n,1,1)\}$, $V=\{0\}$, $I=X$, $O=X$. Если при этом требуется поместить результат в другой массив Y (FFT в режиме `out-of-place`), то $O=Y$.

Когда требуется выполнить несколько одномерных FFT (`in-place`) над непрерывными векторами, например, над строками матрицы $A[m][n]$ (объявленной так в Си-программе), то: $S=\{1,(n,1,1)\}$, $V=\{1,(m,n,n)\}$, $I=A$, $O=A$.

Если же результат такого FFT (`out-of-place`) надо записать в другую матрицу $B[m][n]$, то $O=B$.

Однако, если аналогичную серию FFT: $B=\text{FFT}(A)$ потребуется исполнить над n столбцами матриц, т.е. над векторами длиной m , элементы которого следуют в памяти с шагом (страйдом) m , тогда: $S=\{1,(m,n,n)\}$, $V=\{1,(n,1,1)\}$, $I=A$, $O=B$. Результат одномерного FFT-преобразования столбцов матрицы A можно записать в качестве строк в другую матрицу $C[n][m]$, которая будет транспонированной по отношению к матрице B : $S=\{1,(m,n,1)\}$, $V=\{1,(n,1,m)\}$, $I=A$, $O=C$.

Для характеристики двумерного преобразования Фурье размером $m \times n$ над одиночной матрицей $A[m][n]$ надо задать: $S=\{2,(m,n,n),(n,1,1)\}$, $V=\{0\}$, $I=A$, $O=A$. А для такого же группового FFT-преобразования над k матрицами, которые являются плоскостями трёхмерного массива $Z[k][m][n]$, потребуется сформировать такой дескриптор FFT-задачи: $S=\{2,(m,n,n),(n,1,1)\}$, $V=\{1,(k,m \cdot n, m \cdot n)\}$, $I=Z$, $O=Z$.

Анализируя дескриптор задачи, каждый решатель должен дать ответ, способен ли он выполнять такую задачу или нет. Для этого ему возможно потребуется проверить, удовлетворяют ли характеристики S , V , I , O FFT-задачи определённым условиям.

Если метод, который применяет модуль решателя для исполнения FFT, работает только для случая совмещения входного и выходного массива (`in-place`), то должны совпадать значения указателей I и O ($O=I$), а также попарно все значения входных и выходных страйдов $i_k=o_k$ тензора S по всем измерениям $k=1, \dots, r$.

Если для исполнения БПФ требуется, чтобы входной и выходной массивы были расположены выровненными в памяти по адресу, кратному значению K , то необходимо проверить, что $I \bmod K = 0$ и $O \bmod K = 0$.

Возможно также, что тот или иной метод преобразования Фурье может быть применим лишь к массивам данным определённого размера n . Например, если n не простое число. Или если $n=2^t$, т.е. степень двойки.

Если решатель умеет исполнять только одномерное преобразование Фурье, то он может применяться лишь при значении r у тензора S равным 1. А если он не рассчитан на групповое исполнение нескольких FFT, то следует убедиться, что значение r у тензора V равно 0.

Все эти проверки в модулях-решателях библиотеки FFTW, как правило, сосредотачиваются в отдельной функции `applicable`, которая вызывается из функции `mkplan`, чтобы получить ответ на вопрос: можно ли применить данный решатель для выполнения сформулированной в дескрипторе FFT-задачи.

В дескрипторе FFT-задачи содержится также индикатор вида `problem_kind`, характеризующий, какого вида и над величинами какого типа должно выполняться данное FFT-преобразование (над комплексными или над вещественными). Такой же индикатор записан и в дескрипторе (типа `solver`) каждого решателя. Планировщик сам проверяет, что значение этого индикатора у решателя совпадает со значением индикатора в дескрипторе FFT-задачи. И только если эта проверка прошла успешно, он вызывает функцию `mkplan`, предусмотренную в дескрипторе решателя.

4.2. Дескриптор FFT-плана

В дескрипторе FFT-плана задаются две основные функции `awake` и `apply`, которые применяются для исполнения этого плана FFT-преобразования.

Функция `awake` вызывается сначала перед первым вызовом `apply`, а затем ещё раз после последнего вызова `apply`, что позволяет заранее подготовить некоторые вспомогательные величины (например, вычислить таблицы поворотных коэффициентов), необходимые для осуществления основной операции — преобразования Фурье.

Чтобы распознать, с какой целью произведён вызов `awake`, этот вызов сопровождается параметром `wakefulness`. При вызове `awake`, следующего после завершения серии вызовов `apply`, `wakefulness` равен значению `SLEEPY` (=0). Перед первым вызовом `apply` с реальными данными, `awake` вызывается с параметром `wakefulness`, равным значению `AWAKE_SQRTN_TABLE` или `AWAKE_SINCOS`. А перед серией тренировочных вызовов `apply` с пустыми (обнулёнными) данными, которые осуществляются в целях определения реальной производительности оцениваемого плана, `awake` вызывается с параметром `wakefulness`, равным значению `AWAKE_ZERO`.

При вызове функции `apply` ей в качестве параметров передаются указатель на дескриптор FFT-плана и указатели на входной и выходной массивы, для которых требуется исполнить FFT-преобразование в данном вызове. Заметим, что при этом вызове не передаётся указатель на дескриптор FFT-задачи. Поэтому все необходимые величины, характеризующие FFT-задачу (размерность и размеры массивов, их страйды и др.), которые могут потребоваться для выполнения функции `apply`, следует заранее переписать в дескриптор плана (при его построении) из дескриптора задачи.

4.3. О направлении FFT- преобразования

Вызов функции `apply` выполняется всякий раз, когда программа пользователя вызывает функции:

```
fftw_execute(p);
fftw_execute_dft(p, X, Y);
```

В них в качестве параметра `p` (типа `fftw_plan`) используется указатель на структуру данных вида:

```
typedef struct {
    plan *pln; //указатель на дескриптор плана
    problem *prb; //указатель на дескриптор задачи
    int sign; //знак направления FFT-преобразования
} *fftw_plan
```

в которой собраны вместе дескриптор задачи, дескриптор плана и знак направления FFT-преобразования: (`FFTW_FORWARD=-1` или `FFTW_BACKWARD=+1`).

Отметим здесь одну особенность, характерную для реализации библиотеки FFTW. В ней для выполнения преобразования Фурье над комплексными числами знак, характеризующий, какое требуется направление преобразования, прямое или обратное, явно не указывается ни при вызове функции создания плана

`mkplan`, ни при вызове функции `apply`. Не записан он и в дескрипторе FFT-задачи. Возникает вопрос, почему в этих случаях такой знак не задаётся?

Для понимания ответа на этот вопрос сначала объясним следующий фокус. Пусть операция $Y=FFT(X)$ выполняет прямое преобразование Фурье, а операция $Exch(X)$ в каждом комплексном элементе X_k меняет местами реальную (Re) и мнимую (Im) части. Тогда ненормализованное обратное преобразование Фурье $Y=iFFT(X)$ можно выполнить такой последовательностью действий: $Exch(X)$; $Y=FFT(X)$; $Exch(Y)$.

Другими словами, для обратного преобразования Фурье нужно выполнить аналогичное прямое преобразование, но при этом подразумевать, будто реальная и мнимая части обрабатываемых элементов как будто поменялись местами. Почему это так?

Известно, что обратное преобразование Фурье (без нормализации / n) отличается от прямого только тем, что в приведённой ранее формуле (Ф1) нужно использовать (вместо $w_n^{m \cdot k}$) другие коэффициенты $W_n^{m \cdot k}$:

$W_n^{m \cdot k} = \exp(+i \cdot m \cdot k \cdot 2\pi/n)$ (I – мнимая единица). Заметим, что эти коэффициенты $w_n^{m \cdot k}$ и $W_n^{m \cdot k}$ различаются лишь знаком у мнимой степени экспоненты:

$w_n^{m \cdot k} = \exp(-i \cdot f)$, $W_n^{m \cdot k} = \exp(+i \cdot f)$, где $f = m \cdot k \cdot 2\pi/n$

Убедимся, что значение p реальной части и значение q мнимой части комплексного значения $R=p+i \cdot q$, являющегося результатом умножения комплексного числа $z=y+i \cdot x$ на комплексное значение $\exp(+i \cdot f)=\cos(f)+i \cdot \sin(f)$:

$$R = z \cdot \exp(+i \cdot f) = (y+i \cdot x) \cdot (\cos(f)+i \cdot \sin(f)) = (y \cdot \cos(f) - x \cdot \sin(f)) + i \cdot (x \cdot \cos(f) + y \cdot \sin(f)) = p+i \cdot q,$$

где: $p = y \cdot \cos(f) - x \cdot \sin(f)$ и $q = x \cdot \cos(f) + y \cdot \sin(f)$

можно вычислить другим способом, перемножив комплексное число $Z=x+i \cdot y$ на $\exp(-i \cdot f)=\cos(f)-i \cdot \sin(f)$:

$$S = Z \cdot \exp(-i \cdot f) = (x+i \cdot y) \cdot (\cos(f)-i \cdot \sin(f)) = (x \cdot \cos(f) + y \cdot \sin(f)) + i \cdot (y \cdot \cos(f) - x \cdot \sin(f)) = q+i \cdot p$$

и получив в результате число $S = q+i \cdot p$, в мнимой части которого окажется требуемое значение p , а в реальной части — значение q .

Это означает, что операцию обратного преобразования Фурье можно вычислять по той же самой формуле Ф1, указав лишь, что значения реальных частей элементов обрабатываемого массива надо брать с позиций, где расположены их мнимые части, и наоборот, указатели на значения мнимых частей необходимо настроить на позиции реальных частей обрабатываемых элементов.

Вот именно такой трюк и используется в библиотеке FFTW. В качестве указателя P массива комплексных чисел задаётся два значения указателей на вещественные числа $P=(P_{re}, P_{im})$: первое указывает на начало расположения в памяти реальных компонент элементов массива, а второе — на начало мнимых компонент того же массива. Для обратного же преобразования Фурье эти значения меняют местами: $P=(P_{im}, P_{re})$. И вот такие составные указатели задаются для входных и выходных массивов как при вызове функции создания дескриптора FFT-задачи, так и при вызове функции `apply` уже для исполнения FFT-преобразования.

Как же в таком случае определить, какое будет выполняться преобразование Фурье, прямое или обратное? Допустим, пара вещественных указателей P_1 и P_2

задаются в качестве составного указателя $P=(P_1, P_2)$ на обычный комплексный массив, в котором для каждого элемента мнимая часть располагается вслед за его реальной частью. В таком случае знак направления преобразования можно узнать, сравнивая значения этих указателей P_1 и P_2 . Если $P_2=P_1+1$, это значит, что P_1 указывает на реальную часть, а P_2 — на мнимую, и тогда должно выполняться прямое преобразование. Если же $P_1=P_2+1$, это значит, что P_1 указывает на мнимую часть, а P_2 — на реальную, и тогда следует выполнять обратное преобразование.

Задание двойных значений для указателей на комплексные массивы позволяет обрабатывать не только обычные массивы из комплексных чисел, у которых реальные и мнимые части расположены вместе. Открывается возможность, вообще говоря, обрабатывать и такие расщеплённые массивы, реальные и мнимые компоненты которых могут быть заданы двумя отдельными массивами вещественных чисел, размещёнными в различных местах памяти.

4.4. Рекурсивный вызов планировщика

Если число n можно разложить на множители: $n=s \cdot r$, то алгоритм вычисления ДПФ длины n можно свести к последовательным выполнениям сначала r операций ДПФ размера s и затем s операций ДПФ размера r или наоборот.

Для этого представим, будто исходный вектор X_n уложен в матрицу размером $s \times r$ по строкам (в ней s строк и r столбцов). А вектор-результат Y_n будем представлять в виде матрицы размером $r \times s$, состоящей из r строк и s столбцов. Представим $k=i \cdot r + j$ и $m=s \cdot t + t$ и перепишем прежнюю формулу Ф1 вычисления ДПФ в виде эквивалентной ей формулы Ф2 (см. [5]):

$$\{\Phi 2\} Y_{s \cdot t + t} = \sum \left\{ \left[\left(\sum \{ X_{i \cdot r + j} \cdot w_s^{i \cdot t} \}_{i=0,1,\dots,s-1} \right) \cdot w_p^{j \cdot t} \right] \cdot w_p^{j \cdot r} \right\}_{j=0,1,\dots,p-1} \\ t=0,\dots,p-1, \quad t=0,\dots,s-1.$$

По внутренней сумме (в круглых скобках) формулы Ф2 вычисляется ДПФ размера s для j -го столбца воображаемой матрицы X :

$$\{\Phi 3\} Y[j, t] = \sum \{ X_{i \cdot r + j} \cdot w_s^{i \cdot t} \}_{i=0,1,\dots,s-1} \quad \text{для } t=0,\dots,s-1.$$

Допустим, вектора, полученные в результате выполнения r операций ДПФ над всеми j -ми столбцами матрицы X ($j=0,\dots,p-1$), укладываются как строки в воображаемую матрицу $Y[p][s]$.

Далее элементы этой матрицы домножаются на коэффициент $w_p^{j \cdot r}$ ($j=0,1,\dots,p-1$; $t=0,\dots,s-1$).

А теперь по внешней сумме формулы Ф2 выполняется s операций ДПФ размера r над каждым столбцом воображаемой матрицы Y :

$$\{\Phi 4\} Y_{s \cdot t + t} = \sum \{ Y[j, t] \cdot w_p^{j \cdot r} \}_{j=0,1,\dots,p-1} \\ t=0,\dots,p-1, \quad t=0,\dots,s-1.$$

В результате операция ДПФ размера n осуществляется путём выполнения r операций ДПФ размера s и последующих s операций ДПФ размера r .

Многие модули-решатели библиотеки FFTW используют такой приём, чтобы выразить алгоритм ДПФ заданного размера через серию ДПФ меньшего размера. Например, к ним относятся модули DIT (decimation in time) и DIF (decimation in frequency).

Такие решатели разлагают заданную FFT-задачу на серию FFT-задач меньшего размера, а для поиска оптимального исполнителя для них обращаются к планировщику. В результате получается, что планировщик, вызвавший функцию `mkplan` такого решателя, сам вызывается при исполнении этой функции. Так закручивается сложный рекурсивный механизм функционирования планировщика.

Этот рекурсивный механизм прекращает свою раскрутку, когда для FFT-задачи малого размера вызываются модули-решатели, способные окончательно выполнить FFT-задачу уже без услуг планировщика.

Некоторые из таких решателей полностью справляются с выполнением требуемой FFT-задачи сами. А другие прибегают к обширной армии вспомогательных модулей, называемых коделетами (codelets).

4.5. Коделеты

Коделет — это специально оформленный модуль, способный напрямую исполнять ДПФ конкретного малого размера. Каждый коделет реализует ДПФ своего размера над массивами определённого типа, характеристики которых (указатели и страйды) ему передаются в качестве параметров при его вызове. Вызов коделета осуществляет функция `apply` того решателя, который использует данный коделет как помощника.

В библиотеке FFTW приготовлена обширная коллекция коделетов (около 150), код которых оптимизирован для эффективного исполнения на процессорах многих современных компьютерных архитектур.

Тела коделетов (на языке Си) генерируются автоматически с помощью специальной программной компоненты библиотеки FFTW, названной генератором коделетов `genfft`. Такой генератор воспринимает высокоуровневое математическое описание алгоритма вычисления ДПФ, составленное на некотором специализированном языке, и формирует оптимизированный код коделета, обеспечивающий эффективную реализацию ДПФ фиксированного размера.

5. Модификация FFTW для CPV

С помощью CPV можно достичь максимально возможного ускорения БПФ над комплексными числами, лишь используя его высокопроизводительную команду `cmaddsub`, предназначенную для аппаратного исполнения операции БФ («бабочки Фурье»). Следовательно, на CPV надо применять такой алгоритм вычисления БПФ, который рассчитан на многократное употребление операций БФ.

Один из таких алгоритмов — вариант алгоритма Кули-Тьюки с прореживанием по времени для длины $n=2^p$ (степени двойки) — был ранее успешно опробован на CPV и подробно описан в [4]. В ФНЦ НИИСИ РАН была предпринята попытка внедрить модуль такого алгоритма вычисления БПФ в библиотеку FFTW версии 3.2.2, предусматривающей возможность компоновки для компьютеров MIPS-архитектуры.

На первом этапе в FFTW был встроен модуль-решатель `vffft2`, который обеспечил ускоренное выполнение БПФ для непрерывных векторов комплекс-

ных чисел двойной точности длиной $n=2^p$ (степени двойки) с использованием команд CPV. Этот модуль регистрирует своего решателя (solver) в таблице планировщика. Его функция `mkplan` успешно откликается на вызов планировщика и формирует FFT-план при соблюдении следующих условий:

- 1) ранг FFT-преобразования $S.r = 1$;
- 2) длина массива $n=S.n_1=2^p$, $8 \leq n \leq 32768$;
- 3) страйды $i_1=1$ и $o_1=1$, т.е. элементы массива следуют в памяти непрерывно друг за другом;
- 4) входные и выходные массивы выровнены по адресу, кратному 32 (байта);
- 5) ранг $V.r \leq 1$ (лишь одноуровневая группа);
- 6) адреса реальной и мнимой частей у составных указателей комплексного массива $I=(I_1, I_2)$, $O=(O_1, O_2)$ должны быть смежными: $|I_1-I_2|=1$, $|O_1-O_2|=1$;

Причём по знаку разницы (I_1-I_2) определяется знак направления FFT-преобразования (-1 для прямого и $+1$ для обратного).

На втором этапе в FFTW был встроен модуль-решатель `vfftp2`, который обеспечивал выполнение такого же БПФ, но уже для произвольных векторов комплексных чисел двойной точности длиной $n=2^p$, элементы которых могли располагаться не обязательно непрерывным блоком в памяти, а следовать в ней с некоторым страйдом. И этот модуль-решатель уже не требовал соблюдения условия 3 ($i_1=1, o_1=1$) для построения FFT-плана.

Конечно, по быстродействию выполнения БПФ модуль `vfftp2` проигрывает модулю `vfftp2` (это подтверждается далее в п.6). Но зато область его применения гораздо шире. Во-первых, он позволяет подвергать FFT-обработке массивы длиной $n=2^p$ с любыми страйдами, например, не только строки, но и столбцы комплексной матрицы.

А во-вторых, он разрешает решателям, строящим иерархический план FFT-преобразования, использовать себя в качестве исполнителя базового FFT-плана. В результате в сочетании с другими решателями модуль `vfftp2` позволяет ускорить обработку массивов не только длины $n=16, 32, 64, \dots$, но и длин, кратных этим значениям. Вот, например, какой FFT-план сформирует планировщик для обработки массива длиной 48:

```
(dft-ct-dit/3
(dftw-direct-3/4 "t1_3")
(dft-vfftp2![16,6,2]F) ((3 2 32)))
```

6. Результаты применения CPV для ускорения БПФ в FFTW

Исходная версия 3.2.2 библиотеки FFTW и её модифицированные версии с добавленными модулями `vfftp2` и `vfftp2` подвергались сравнительному тестовому исследованию с целью определения коэффициентов ускорения операций БПФ, которые обеспечивают полученные модифицированные версии библиотеки по отношению к её исходной версии.

Тестирование проводилось на ПЛИС-прототипе микропроцессора VM8 с частотой ядра 62,5 МГц. Для получения показателей производительности в среде ОС Linux запускалась стандартная утилита `bench`, ко-

торая поставляется вместе с самой библиотекой и обычно компонуется вместе с ней. Для каждой испытываемой версии библиотеки утилита запускалась многократно для $n=2^p$ ($p=3, 4, 5, 6, \dots, 14$) командой:

```
# bench -v2 -s n
```

В качестве результирующего показателя производительности принималось выдаваемое этой утилитой среднее значение (avg) времени (в мксек.), затраченное ею на исполнение операции БПФ заданной длины.

Результаты проведённых испытаний приведены в таблицах 3 и 4. Таблица 3 демонстрирует, во сколько раз операция БПФ над комплексными числами выполняется быстрее в модифицированной версии библиотеки с добавленным модулем `vfftp2`, применяющим CPV. Полученные коэффициенты ускорения записаны в её в правом столбце (К).

Таблица 3. Ускорение БПФ в FFTW с модулем vfftp2

N=2^p	FFTW	FFTW + vfftp2	К
8	2.64	2.15	1.23
16	6.17	2.60	2.37
32	17.34	3.54	4.9
64	33.99	22.47	1.51
128	75.84	47.31	1.6
256	205.42	99.58	2.06
512	448.52	228.69	1.96
1024	1020	508.41	2.01
2048	2530	1100	2.3
4096	5360	2440	2.2
8192	1182	5960	1.98
16384	3492	3283	1.06

Таблица 4 характеризует коэффициенты ускорения модифицированной версии библиотеки со встроенным в неё модулем `vfftp2` (но без модуля `vfftp2`).

Таблица 4. Ускорение БПФ в FFTW с модулем vfftp2

N=2^p	FFTW	FFTW + vfftp2	К
16	6.17	3.05	2.02
32	17.34	4.39	3.95
64	33.99	22.76	1.49
128	75.84	49.59	1.53
256	205.42	103.89	1.98
512	448.52	318.16	1.41
1024	1020	871.96	1.17
2048	2530	1950	1.23
4096	5360	4410	1.22
8192	1182	1056	1.12
16384	3492	3399	1.03

Как и ожидалось, версия с модулем `vfftp2` обеспечивает ускорение операции БПФ (для любых $n=2^p$) в большей степени, чем версия с модулем `vfftp2`. Наилучший показатель ускорения достигается на длине $n=32$ (почти в 5 раз!). Но для больших значений n выигрыш от применения CPV становится менее заметным. Это объясняется тем, что при обработке данных больших размеров переполняется кэш-память, из-за чего существенно замедляется работа команд доступа к памяти, а в этих случаях высокая производительность команд CPV уже не играет заметной роли.

Версия библиотеки с модулем vffts2 позволяет также ускорить операции БПФ над массивами длиной, кратной 16 и 32, что и демонстрирует таблица 5.

Таблица 5. Ускорение БПФ в FFTW с модулем vffts2

N	FFTW	FFTW + vffts2	K
48= 3×16	26.80	21.14	1.27
80= 5×16	47.67	38.62	1.23
112= 7×16	79.29	61.09	1.3
96= 3×32	51.18	36.67	1.48
160= 5×32	105.08	70.83	1.48
192= 6×32	51.18	83.11	1.57
320= 10×32	279.67	155.84	1.79

Заключение

Предложен способ модификации библиотеки FFTW, заключающийся во внедрении в её механизм построения плана своих модулей, применяющих для ускорения операций преобразования Фурье высокопроизводительные команды векторного сопроцессора микропроцессора BM8 семейства КОМДИВ.

В результате проведённой модификации для микропроцессора BM8 создана новая версия библиотеки FFTW, которая позволяет существенно повысить производительность программ, выполняющих преобразования Фурье над массивами комплексных чисел двойной точности.

Автор выражает благодарность П.А. Мамонову за квалифицированную помощь при подготовке make-файлов для сборки новой версии библиотеки.

Implementation in FFTW library of own modules using the specialized coprocessor to an acceleration of operations of Fourier transform

A.A.Burtsev

Abstract. In Scientific Research Institute for System Analysis of the Russian Academy of Sciences (SRISA RAS) as expansion of universal microprocessors of the KOMDIV family the specialized 128-digit coprocessor allowing to accelerate calculations over vectors of complex and real numbers of unary and double accuracy has been created. The article presents results of implementation to widely known and generally available library FFTW of own modules which allow to use this specialized coprocessor to increase of productivity of operations of Fourier transform over vectors of complex numbers.

Keywords: MIPS architecture, microprocessors of the KOMDIV family, coprocessor of complex arithmetics, vector coprocessor (CPV), digital signal processing (DSP), operation of the fast Fourier transform (FFT), library FFTW.

Литература

1. В.Б. Бетелин. Отечественные суперкомпьютерные технологии экзафлопсного класса – необходимое условие обеспечения технологической конкурентоспособности России в XXI веке. // Программные продукты и системы. 2013. №4 (104). С.4-9.
2. С.Г. Бобков. Высокопроизводительные вычислительные системы / под ред. академика РАН В.Б. Бетелина. – М.: НИИСИ РАН, 2014. – 299 с.
3. А.А. Бурцев. О возможности оптимизации некоторых функций библиотеки линейной алгебры с помощью векторного сопроцессора // Труды НИИСИ РАН, т.4 №2. М.: Изд-во НИИСИ РАН, 2014. с.5-15.
4. А.А. Бурцев. О возможности применения векторного сопроцессора для ускорения операции быстрого преобразования Фурье // Труды НИИСИ РАН, т.5 №2. М.: Изд-во НИИСИ РАН, 2015. с.138-147.
5. M. Frigo, S.G. Johnson. The design and implementation of FFTW3 // Proceedings of IEEE, vol. 93, №2, 2005, pp. 216-231.
6. M. Frigo, S.G. Johnson. The FFTW web page. URL: <http://www.fftw.org/>, 2004.

Способ интеграции табличных данных с целью оптимизации поиска

Г.Л. Левченкова

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: galka_lev@mail.ru

Аннотация: Рассматривается метод слияния табличных данных из нескольких документов с целью оптимизация поиска информации. В качестве исходных документов могут использоваться файлы, подготовленные вручную либо полученные путем экспорта из баз данных.

Ключевые слова: информация, база данных, табличные данные, импорт

1. Введение

Предположение, что каждый из нас хотя бы раз сталкивался с задачей поиска нужного документа, однозначно является верным. Допущение, что где-то имеются сведения об интересующем нас документе, кажется вполне разумным. И вероятнее всего это место – оглавление/список, оформленный в виде, например, электронной таблицы Microsoft Excel: аккуратные предусмотрительные работники обычно ведут таковой, заполняя вручную, либо предприятие оснащено соответствующим программным обеспечением, позволяющим сформировать перечень нужного содержания путем экспорта из какой-либо базы данных.

Как поступает человек, которому нужно найти документ? Он открывает файлы-списки, которые могут содержать нужные сведения, и, отбирая (фильтруя или используя поиск по файлу) ищет необходимое. Но файл, в котором производится отбор-поиск, не один. Зачастую их десятки. Например, нужны сведения об изменении состава постоянно действующей комиссии (с датами и фамилиями). В этом случае придется вести поиск по оглавлениям нескольких годов. А в случае поиска документов, отражающих историю перемещения какой-либо материальной ценности (заявка на приобретение, закупка, передача между материально ответственными лицами или отделами, изменение цены), придется прочесывать несколько файлов, сформированных из различных баз данных.

Поиск, таким образом, представляет собой Работу (с большой буквы), которую нередко поручают специально для этой цели выбранному сотруднику. Процесс визуального отбора нужных сведений можно упростить путем объединения (копированием) строк таблиц в одну сводную таблицу и фильтрации информации в полученном файле: уже в одном, а не в десятках. Но при следующем "задании на поиск" придется делать сводный файл заново, потому что сведения в базах данных и таблицах постоянно пополняются. Вот если бы можно было сразу искать информацию в одном месте, да еще со "свежими" данными! Оказывается, такая задача вполне решаема.

Подавляющее большинство всевозможных учетных систем позволяет экспортировать данные в формат Microsoft Excel (*.xls, *.xlsx). Рабочие места организаций практически всегда оснащены компьютерами с установленным пакетом Microsoft Office, и, естественно, сотрудники используют Microsoft Excel ([1], [2]) для целей ведения учета. Таким образом, остается лишь "изобрести" способ объединять эти данные с минимальными трудозатратами.

Аспекты учета и мобильности сведений уже были рассмотрены ранее (см. [3], [4] и [5]). В настоящей статье рассматривается схема оформления объединенной информации, обеспечивающая быстрый и удобный последующий поиск данных в одном файле, отражающем актуальные на настоящий момент сведения.

2. Процесс интеграции

Рассмотрим способ объединения информации в файл формата Microsoft Excel (*.xls, *.xlsx) с использованием стандартного механизма - импорта внешних данных. Условие импорта "обновление при открытии файла" позволит иметь актуальную сводку на текущий момент.

Допустим, что нам известны места хранения и имена файлов формата Microsoft Excel, которые организованы в одинаковые таблицы по виду информации, содержащейся в колонках.

Эти файлы могут обновляться с разной частотой по мере пополнения сведениями. Некоторые из них выгружаются (экспортируются) из баз данных с определенной частотой. Другие заполняются вручную: новые записи появляются по мере поступления документов.

Пусть информация, которая подлежит дальнейшему анализу, выглядит так, как приведено на Рис. 1.

Для слияния будем брать в качестве исходных данных так называемые "файлы-оглавления" различных подразделений предприятия за разные года:

Местонахождение (пор. № документа в папке подразделения)	Местонахождение (Год регистрации)	Документ (номер или название)	Дата документа	Номер документа по системе ДЕЛО	Исполнитель (организация или ФИО)	Предмет (тем а)	Связанные документы	Признак "исполнено"	Примечание
34	2015	Приказ № 45	12.11.15	02-14/33	Директор	Назначение	вх. 33/2014	ОК	Новиков
33	2015	154-22	10.11.15	02-09/118	НПО КТТ	Командировка...	исх. 25/2015		Потапов

Рис.1. Вид исходных данных

D:\Документооборот\2016\Вход\oglavlenie_PR_2016.xls
D:\Документооборот\2016\Вход\oglavlenie_PR_2015.xls

...
\\OK\D\DOCU\ДО_2016\Вход_OK.xls
\\OK\D\DOCU\ДО_2015\Вход_OK.xls

...
\\Канц\D\2016\ДЕЛО.xls
\\Канц\D\2015\ДЕЛО.xls

...
\\1C\D\2016\export.xls
\\1C\D\2015\export.xls

1. Создаем СВОДНЫЙ (резльтирующий) файл

Файл создается, например, как "Лист Microsoft Excel".

Устанавливаем во всех ячейках файла формат "текстовый", на вкладке "Выравнивание" выставляем значения, приведенные на Рис. 2. :

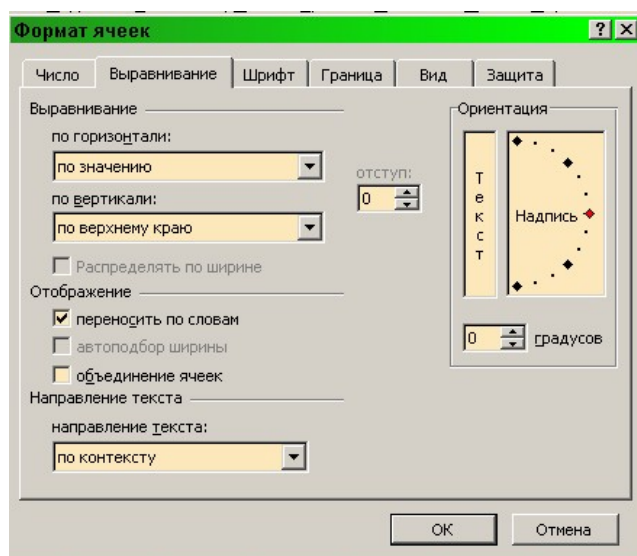


Рис.2. Параметры выравнивания ячеек

Прорисовываем границы ячеек (в меню "Нарисовать границы" выбираем пункт "Все границы").

Далее рекомендуется расширить колонки до нужного размера, чтобы размещаемый в них текст не занимал много места в высоту. Для этого можно перенести копированием информационную строку из любого из объединяемых исходных файлов и, растянув/сузив в соответствии с размещенным текстом соответствующие ячейки, вписать в эту верхнюю строку названия будущих колонок.

В результате подготовительных действий получаем файл следующего вида (см. Рис. 3).

Рис.3. Пустой сводный файл

Рекомендуется выйти из сводного файла перед началом последующих действий. Это позволит "откатиться" до пустого файла, если будет сделана неточность при импорте.

2. Импортируем данные

Для эффективности последующей работы с объединенной информацией следует определить (каждому пользователю самостоятельно в соответствии со своими потребностями) порядок файлов-источников данных в плане расположения сведений в результирующем файле. То есть, удобнее ли Вам сначала (сверху) видеть "свои" оглавления, а потом оглавления других подразделений, либо удобнее чередовать оглавления, сортируя их по годам, и т.д. После того, как очередность первоначальной загрузки определена, приступаем к импорту.

Первое наполнение сводного файла будем начинать сверху вниз. По мере появления новых исходных файлов данных можно будет их добавлять в любое место сводки (как это делать, рассмотрим позже), но новый файл легче заполнить последовательно.

ВАЖНО: Устанавливаем курсор в самую левую ячейку (колонок А). Импортированные данные представляют собой прямоугольник ячеек с верхним левым углом, являющимся исходной точкой ввода данных. При обновлении (добавлении строк в исходный файл) вниз "поползет" этот блок целиком. Если курсор в начале импортирования сведений будет располагаться, например, в колонке В, то текст колонки А не будет являться частью загружаемого диапазона, и при добавлении строк в исходный файл записи, расположенные в строках результирующего файла, сместятся и окажутся некорректными.

Итак, курсор находится на пустой ячейке колонки А ниже строки с названием столбцов. Активируем пункт меню: "Данные" – "Импорт внешних данных" – "Импортировать данные". Выбираем файл Microsoft Excel, содержащий нужное оглавление, и перед нами следующий выбор таблицы (см. Рис. 4)

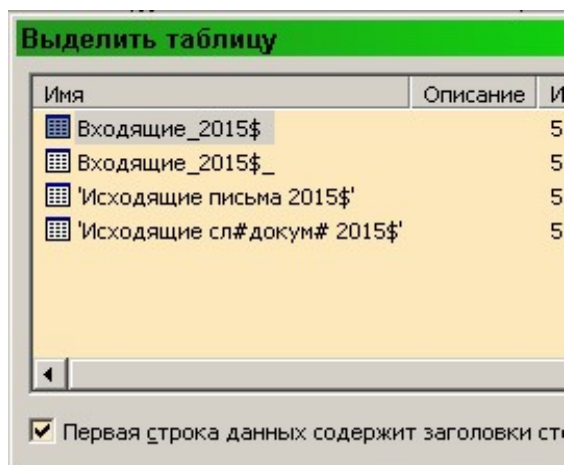


Рис.4. Выбор таблицы загрузки

Здесь мы видим наименования Листов, которые входят в выбранном нами файле, и "временные" Листы (которые заканчиваются символом _ (подчеркивание)). Оставляем "галочку" в окошке "Первая строка содержит заголовки..." и указываем нужный лист для загрузки. В приведенном примере это будет лист с именем "Входящие_2015\$" (см. Рис. 4).

После выбора источника данных на экране появляется следующее меню "Импорт данных", в котором нужно немного откорректировать "Свойства". Например, чтобы результат был в том виде (в частности, ширина колонок) в котором вы хотите (как в верхней строке с названиями колонок), то нужно убрать галочку около "задать ширину столбца". Свойства рекомендуется выставить согласно рисунку 5. После выставления "Свойств" нажимаем "ОК" и еще раз "ОК" в меню "импорта данных". Результат: в сводный файл добавились строки, которые будут загружаться из внешнего источника данных в момент "обновления" (ранее мы установили "обновление" при открытии сводного файла – см. Рис. 5).

ВАЖНО: Если Вы где-то промахнулись, и диапазон находится не на том месте, где ожидалось, либо выбран не тот лист, который надо было загрузить, то следует выйти из сводного файла без сохранения, открыть его заново и произвести импорт еще раз. Откатиться после импорта данных невозможно. Можно, однако, удалить весь диапазон "убиением" строк, но при этом есть вероятность захватить не весь диапазон или изменить формат ячеек.

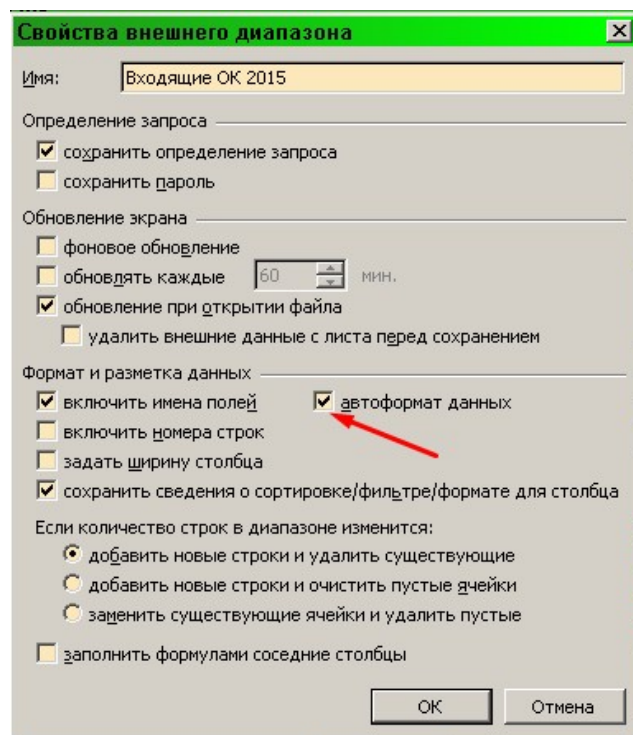


Рис.5. Свойства диапазона импортируемых данных

Совет: Для удобства дальнейшей ориентации по импортированным данным рекомендуется указывать в свойствах добавляемого диапазона понятное имя (см. верхнее окошко на Рис. 5). Место, откуда импортированы сведения, скрыто внутри свойств диапазона и нет никакой возможности добавить словесные комментарии о нем в текст сводного файла (данные все время "ползут" при обновлении, и комментарии оказываются не на тех строках, которые помечались). Кроме того, можно окрасить диапазоны "заливкой" определенным цветом, пометив таким образом разные года, подразделения др. Это позволит визуально ориентироваться в данных, не прибегая к открытию свойств диапазона. Памятку типа "Имя диапазона – источник данных" или "Тип диапазона - цвет" потом можно будет добавить в качестве условных обозначений.

ВАЖНО: Чтобы впоследствии можно было без ущерба для импортированных данных переставить колонки местами, или удалить лишнюю информацию, следует убедиться, что количество значащих колонок в исходном файле превышает количество озаглавленных колонок в сводной таблице (в примере на Рис. 3 их шесть штук). Если в импортируемом

файле колонок меньше, чем нам надо иметь в сводке, то следует сократить таблицу-сводку, убрав колонки, которые содержат не очень важную информацию, либо нужно позаботиться, чтобы в исходном файле было достаточно столбцов (вставить пустые при необходимости).

Переходим к добавлению следующего диапазона. Для этого следует сначала найти конец только что импортированных данных. Прокручиваем файл вниз, находим пустые строки. "Встаем курсором" в колонку А. Если при нажатии правой кнопки мыши на ячейке не видна строка меню "Свойства диапазона данных" (см. Рис. 7), значит загруженный ранее диапазон находится выше, и в эту строку можно добавлять новые данные. Последующую работу с диапазонами облегчит наличие пары строк между ними в сводном файле. Рекомендуется располагать диапазоны не очень далеко друг от друга. Заметим, что пустые строки между загружаемыми диапазонами не будут мешать в дальнейшем. Заводим следующий диапазон аналогично описанной выше процедуре.

Совет: После каждого удачного добавления диапазона рекомендуется выйти из сводного файла с сохранением данных и открыть его заново.

3. Добавление нового диапазона между уже импортированными

Отдельно опишем процедуру вставки диапазона между уже заведенными. Первоначально заполняя сводный файл, мы могли не обращать внимания на строки, расположенные ниже курсора, поскольку они были пустыми. Импортирование диапазона между уже заведенными данными может нарушить логическую структуру. Мы запросто можем записать данные поверх нижерасположенных строк. Следует сначала добавить между диапазонами пустые строки в количестве, чуть превосходящем количество значащих строк нового исходного файла, и лишь затем проводить импортирование. Обратите внимание: количество строк в новом_исходном файле обманчиво может показаться меньше, чем их там на самом деле. Их не видно, даже если перейти в режим "Вид - Разметка страницы". При переходе в режим "Предварительный просмотр" перед печатью документа также не заметно величины диапазона, который будет импортироваться. "Непустыми строками импортируемого диапазона", которых не видно, являются, например, строки, содержащие ячейки, в которых установлен некий формат, отличный от "текстового", но не содержится значения.

Для вставки необходимого для импорта количества пустых строк между уже заведенными диапазонами результирующего файла сначала подсчитаем их предполагаемое количество. На любом пустом временном "Листе Microsoft Excel" покрасим (заливкой) два первых столбца (А и В) в какой-либо цвет (см. Рис. 6). Затем поместим курсор в ячейку В1 и импортируем диапазон, в котором нужно пересчитать

строки, отключив в "Свойствах" диапазона "автоформат данных" - см. Рис.5.: надо убрать галочку.

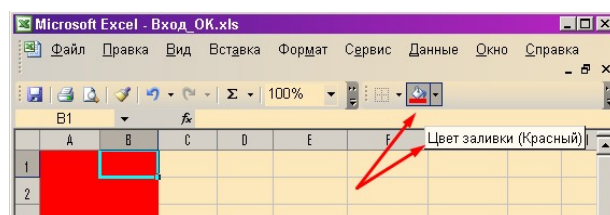


Рис.6. Заливка колонок цветом для подсчета строк диапазона

После такого импорта на пустой лист, пролистаем вниз изображение и визуально сможем определить конец вставленного диапазона (см. Рис 7).

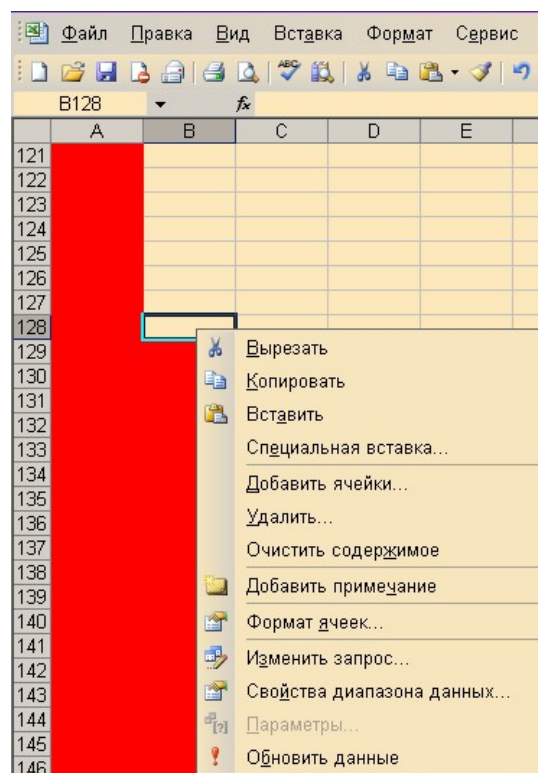


Рис.7. Конец импортируемого диапазона

Ячейка В128 является частью импортированных данных, а ячейка ниже уже их не содержит. Таким образом можно утверждать, что в нужном нам диапазоне содержится 128 строк. Поэтому вставлять между уже заведенными диапазонами в результирующий файл следует минимум 129 строк (одну запасную строку – разделитель диапазонов), и можно быть уверенными, что данные, которые расположены в сводном файле ниже, не затрут загружаемой информацией.

4. Оформляем данные

При объединении "своих" файлов с файлами, экспортируемыми из баз данных, может возникнуть ситуация, когда колонки из стандартным образом

экспортируемых файлов содержат информацию другого вида (формата), или расположенную не в том порядке, в котором располагаются столбцы в результирующем файле. В этом случае придется прибегнуть к помощи специалистов, сопровождающих эту базу данных, для написания несложной дополнительной программы экспорта, выдающей только нужные сведения в должном порядке и формате, совпадающем с "нашими" данными. Одним из требований к экспортирующей программе должно являться отсутствие объединения ячеек в получаемом файле. Если в базе данных нет той категории сведений, которую мы предполагаем видеть в сводном файле, то эти колонки при экспорте (программой экспорта) следует оставить пустыми. Другими словами, объединять импортированием мы будем файл, подготовленный программой экспорта из базы данных в нужном нам виде: порядок данных и форматы данных в ячейках должны удовлетворять заданным названиям колонок в сводном файле.

Столбцы внутри каждого диапазона можно переставить местами, чтобы импортируемые данные разместились в результирующем файле в нужной колонке согласно Заголовкам столбцов. Выполняя операции "Вырезать – Вставить" (**ВАЖНО:** строго в области каждого из диапазонов!) меняем местами столбцы в результирующем файле.

ВАЖНО: Чтобы прямоугольник диапазона был целостным, следует при перестановке колонок применять команду "Вырезать" (вместо "Копировать") и ни в коем случае не удалять столбцы диапазона. То есть в области таблицы (озаглавленной в результирующем файле – см. Рис. 3) обязательно все столбцы должны являться импортируемыми данными диапазона. Если вам нужно поменять колонки 4 и 5 местами, то вырезайте колонку 4 диапазона, вставляйте её на место совсем неиспользуемой (столбец Z, например) колонки справа, затем вырезайте-вставляйте на нужное место колонку 5, а потом возвращайте из "запасного столбца" вырезанием 4-ю запасенную колонку на место пятой.

Столбцы диапазона, которые не несут нужной для отбора-поиска информации, можно вырезать (**ВАЖНО:** строго в области каждого из диапазонов!). Удаление следует проводить с помощью операции "Удалить ячейки, со сдвигом влево", чтобы информация, размещенная в последующих столбцах диапазона, придвинулась вплотную к предыдущей, и в диапазоне не образовался бы разрыв.

Поскольку импорт данных в один файл задуман для оптимизации организации поиска, то нам не нужна здесь полная информация об искомом "предмете", а лишь указатель на его местонахождение. Поэтому в сводном файле не требуется наличие всех полей с описаниями, пустующие ячейки вполне допустимы, а лишнюю информацию даже рекомендуется удалить, лишь бы окончательная таблица могла содержать однотипные данные из файлов как можно большего количества подразделений организации. Таким

образом, для поиска "Что?, Где?, Когда?" нам будет вполне достаточно в результирующем файле видеть заголовки столбцов "Предмет", "дата" и, может быть, еще не лишним будет номер документа. Если названия окончательно выбранных вами для сводки столбцов позволяют получить данные из разных источников в нужном формате, то, конечно, следует оставить их все в действии. Но нам надлежит удалить правые колонки, которые расположены "за пределами важных сведений".

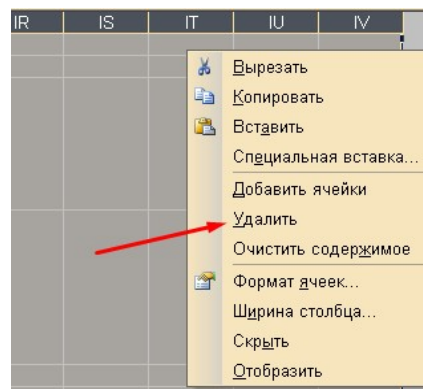


Рис.8. Удаление всех столбцов правее таблицы

"Отрезание хвоста" является необходимым условием корректной работы объединения данных путем импорта. Дело в том, что при импортировании диапазонов мы соблюдали аккуратность при вставке строк, обеспечив таким образом корректное изменение количества записей в результирующем файле при изменении количества записей в исходных файлах: строки диапазонов сейчас не накладывают друг на друга. Но нужно ещё учесть, что диапазон является прямоугольником данных.

При добавлении/уменьшении количества строк в исходном файле будет происходить "вклинивание сведений внутрь" следующего/предыдущего импортированного диапазона сводного файла со сдвигом вниз/вверх соответствующей клину части. Таким образом, нарушится целостность информации в отдельно взятой строке: данные перемешаются. "Испортятся" также и "Свойства диапазона данных": станет невозможно определить, откуда взяты данные, отображаемые в строке.

ВАЖНО: Чтобы изменение прямоугольника импортируемых данных не влияло на результат отображения сведений, в результирующем файле следует выделить ВСЕ КОЛОНКИ, РАСПОЛОЖЕННЫЕ ПРАВЕЕ НУЖНЫХ (оставляемых в сводке согласно Рис. 3) сведений, и выполнить операцию Удалить (см. Рис. 8). После такого удаления нижний "бегунок" будет показывать, что ваш файл почти весь виден на экране (см. Рис. 9). Отсутствие данных, расположенных правее таблицы, можно также проверить установкой в файле "Автофильтра": стрелочками будут помечены только значащие столбцы.

Если в дальнейшем потребуется добавить в сводный файл новый диапазон данных, то процедуру "отрезания хвоста" нужно будет повторить.



Рис.9. Размер нижнего бегунка при отсутствии заполнения строк правее таблицы с данными

Теперь можно озаботиться (если эта процедура не была сделана ранее) выделением каждого диапазона цветом, чтобы визуальное было легче ориентироваться по файлу-сводке. При обновлении файла в результате изменения исходных диапазонов (увеличении/уменьшении количества строк) цвет будет "путешествовать" вместе с данными конкретного диапазона.

После того, как сводный файл заполнен всеми ссылками на источники данных, которые нам нужны для поиска информации, можно сделать Указатель (перечислить условные обозначения, связанные со сведениями определенной категории: по годам, по подразделениям, по цвету диапазона).

Не помешает установка "Автофильтра" (меню Данные - Фильтр) в файле-результате. Измененные строки диапазонов и добавленные в файл новые диапазоны будут автоматически учитываться фильтром.

3. Особенности интеграции методом импортирования данных

Отметим, что наряду с несомненными "плюсами" в сборе информации полуавтоматическим способом (посредством создания файла, импортирующего данные) в таком способе объединения есть некоторые недостатки. Можно назвать их "незначительными неудобствами", но нужно иметь о них представление, чтобы вовремя предотвратить искажение данных.

Например, установив "скрытие столбцов" в сводном файле перед импортированием сведений, мы рискуем поместить в результирующий файл данные со смещением колонок. Скрытые ячейки являются рабочими, а поскольку на экране при скрытии столбцов они не отображаются, то легко можно этот момент упустить, и после перестановки столбцов и выходе из режима "скрыть" увидеть совершенно непонятно откуда взявшиеся пробелы в данных либо наоборот – что-то лишнее.

Опишем последствия изменения формата в какой-либо ячейке_исходного_файла (в исходном файле), которая импортируется в сводную таблицу. Колонка, в которой размещается измененная ячейка, удаляется из того места сводного файла, куда была помещена при

импорте; импортированные колонки этого диапазона, размещенные правее только что исчезнувшей, сдвигаются влево; а новая колонка добавляется в конец данного диапазона. При этом этот "новый для рассматриваемого диапазона" столбец вклинивается "пустым местом" внутрь всех остальных размещенных на листе данных, отодвигая ячейки вправо. Структура данных колонок результирующего файла нарушается. Придется вручную сместить влево отодвинутые вправо колонки пострадавших диапазонов (которых большинство), что очень неудобно. Либо удалить весь "новый" столбец, удалить "плохой диапазон", загрузить его заново и, установив в нем колонки в нужном порядке, "отрезать хвост".

Добавление нового столбца в какой-либо из исходных файлов не очень осложнит работу со сводкой. После открытия результирующего файла мы увидим, что для добавленной информации создан столбец в конце диапазона (справа). Следует либо переместить новый прямоугольник ячеек в нужное место затронутого диапазона, либо произвести внеплановую операцию "отрезания хвоста" у сводки.

Еще раз отметим, что операции "Копировать-Вставить" внутри диапазона, как, собственно, и во всем результирующем файле, приводят к искажению импортирующихся данных. Следует строго придерживаться правила: "Вырезать-Вставлять" только полные_столбцы_диапазона.

Следующий нюанс, о котором следует знать, связан с правами доступа к исходным файлам. После того, как вы, включив "обновление", откроете сводный файл, заполняемый сведениями импортированием данных, открытие всех исходных файлов будет разрешено только на "чтение". И если вы предполагаете длительное время проводить, занимаясь анализом или поиском в сводке, то нужно иметь в виду, что остальным пользователям, желающим получить доступ на редактирование "их файла-оглавления", который вы "захватили", это может не понравиться. Перезапись файла, входящего в импортируемые данные и получаемого путем экспорта из какой-либо базы данных, в этом случае тоже окажется невозможной. Выход из этой ситуации можно предложить следующий: после обновления сведений в результирующем файле (после его открытия в режиме "Включить обновления") надо сразу выйти из этого файла с сохранением, а затем открыть его еще раз, но на этот раз "Выключить обновление". Таким образом, вы получите доступ к данным, актуальным на настоящий момент, но не будете препятствовать работе с исходными файлами других пользователей. Все добавления и изменения, касающиеся диапазонов данных в этом файле с "Выключенным обновлением", будут отражены при следующем открытии файла в режиме "Включенных обновлений". Не забудьте только выйти из файла-сводки с сохранением.

Рассмотрим некоторые характеристики диапазона, которые мы выставляем в процессе добавления нового куска информации в сводку (см. Рис. 4 и Рис. 5). Особое внимание обратим на "галочку" в окошке "Первая строка содержит заголовки..." (Рис. 4). Если её убрать, то в верхней строке загруженного диапазона будут видны символы F1, F2, F3, ... в качестве заголовков импортированных столбцов (см. таблицу внизу на Рис. 10). Но при таком раскладе ползут форматы ячеек и искажается содержание данных. А именно: как можно видеть в исходном файле (на Рис.10 таблица сверху) у колонок 2 и 3 заданы форматы "дата" и "число" соответственно. При импорте с указанием "Первая строка содержит заголовки..." (см. среднюю таблицу на Рис. 10) внешне выглядит все совсем неплохо, хотя названия 2 и 3 колонок имеют дополнительные символы. А вот в нижней таблице заголовки у второй и третьей колонок пропали совсем! Это произошло, потому что формат у каждой ячейки всех исходных строк 86 и 87 уехал вверх, применившись к соответствующим вышерасположенным ячейкам в файле-сводке. В случае средней таблицы это не очень сильно отразилось на данных, а вот нижняя таблица диапазона, имеющая вверху названия колонок, вставленных автоматически, а не взятых из файла, пострадала больше всего. В ячейках результирующего файла (нижняя таблица) содержатся теперь следующие форматы:

- F1, F2, F3... (103 строка) - взяты из строки 86 (в исходном файле у заголовков столбцов был текстовый формат);

- 104 строка должна была принять соответствующие форматы ячеек снизу, то есть из 87 строки должны были унаследоваться "текстовый, дата, числовой, текстовый...".

Однако, в результирующем файле в самом начале мы задали у всех ячеек формат "Текстовый", и числа размещенные теперь в сводке, имеют формат "текст". Самое непредвиденное превращение испытывают ячейки, где мелькает формат "дата". Если бы можно было организовать работу так, чтобы сведения о датах заносились в текстовом формате, то проблем было бы намного меньше. Но поскольку гарантировать такой ввод данных мы не можем, то при выборе импортируемой таблицы лучше поставить "галочку" в поле "Первая строка содержит заголовки...", чтобы потерь информации было меньше.

Если вы видите в ячейках числа вида 42106, это значит, что значение "дата" преобразовано в "текст". Чтобы вернуть ячейке первоначальный вид, установите в сводном файле у этих ячеек формат "дата", только потом при отборе не забудьте, что поиск по этим ячейкам нужно будет задавать не текстовый, или вообще не пользуйтесь этими ячейками для фильтрации данных, а лишь для информации.

Еще один неприятный момент связан с "включением"/"не включением" номеров строк в импортируемый диапазон (см. Рис. 5). С одной стороны, было бы очень удобно ориентироваться по

исходному файлу, если в сводке присутствовала бы ссылка на номер строки, откуда взята информация. Но, к сожалению, в этом случае также "ползут" форматы ячеек: происходит сдвиг форматов влево, распространяясь на новоявленные номера строк, и при этом портится содержание всех остальных ячеек справа. Таким образом, настоятельно рекомендуется снять "галочку" около поля "включить номера строк" в Свойствах диапазона (Рис. 5).

	Тема (формат - текст)	дата (формат - дата)	порядк. номер (формат - число)	Фамилия (формат - текст)	...	...
86						
87	текст	12.04.2015	1	текст		
88	текст	11.11.2011	2	текст		
89	текст	12.12.2012	3	текст		
90	текст	01.01.2001	4	текст		
91	текст	06.06.2006	5	текст		
92						
93	содержит					
	Тема (формат - текст)	дата (формат - дата)	порядк. номер (формат - число)	Фамилия (формат - текст)	...	...
94						
95	текст	12.04.2015	1	текст		
96	текст	11.11.2011	2	текст		
97	текст	12.12.2012	3	текст		
98	текст	01.01.2001	4	текст		
99	текст	06.06.2006	5	текст		
100						
101						
102	не содержит					
103	F1	F2	F3	F4	F5	F6
	Тема (формат - текст)			Фамилия (формат - текст)	...	...
104						
105	текст	12.04.2015	1	текст		
106	текст	11.11.2011	2	текст		
107	текст	12.12.2012	3	текст		
108	текст	01.01.2001	4	текст		
109	текст	06.06.2006	5	текст		
110						
111						

Рис.10 . Результат выбора "Первая строка содержит / не содержит заголовки столбцов"

К сожалению, импорт данных в Microsoft Excel не может быть произведен из файла Word, несмотря на то, что таблицы в Word'e широко используются для создания и ведения всевозможных списков, которые хотелось бы объединять с таблицами Excel. Однако Word умеет экспортировать данные в текстовый формат, а в Microsoft Excel доступна операция импорта их текстового файла. Для получения текстового файла, который подлежит импорту в нашу сводку, можно написать макрокоманду в Word'e, которая будет готовить нужный для импорта файл. Исходим из предположения, что обрабатываемый исходный Word-файл, содержащий необходимые данные для слияния, состоит из одной таблицы.

1. Переводим таблицу в текст:
 - открываем Word-файл с таблицей;
 - выделяем таблицу, выбираем "преобразовать таблицу в текст", в качестве разделителя указываем символ @ (как наименее распространенный. Можно указать любой другой, который практически никогда не используется в файле).
2. Сохраняем файл как "обычный текст .txt" (кодировка Windows). Название файла задаем как xxxxx.txt
3. Теперь файл xxxxx.txt можно импортировать в сводку:

- после выбора файла выставляем параметры, как указано на Рис. 11;
- формат данных столбца у всех (перебором вручную) колонок лучше установить как "текстовый";
- в "Свойствах внешнего диапазона" делаем выбор, как указано на Рис. 5.

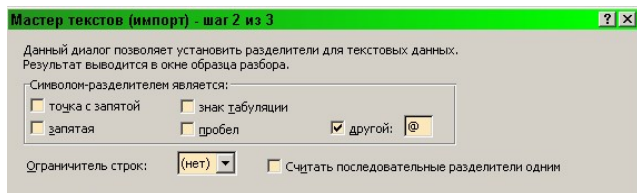


Рис.11. Оформление импорта текстового файла

При перезаписи файла xxxxx.txt, соответствующая информация будет обновляться в сводке стандартным образом – при открытии результирующего файла с параметром "Включить обновление".

4. Отбор данных и поиск

После того, как сводный файл был заполнен импортированными диапазонами, "причесан" и сохранен, можно приступить к поиску данных в любой момент. Открываем файл с "Включенным обновлением" и выходим из него с сохранением, чтобы видеть в сводке последние изменения исходных данных и не препятствовать работе других сотрудников, которые должны иметь право редактировать эти исходные данные. Открываем результирующий файл, выключив обновление, и начинаем работать с ним почти как с обычным документом Microsoft Excel. Почти, потому что если вам не удалось получить (найти) нужную информацию сходу, поставив автофильтр или применив обычную операцию поиска, то к сортировке данных и каким-то изменениям в колонках текста или копирования-перемещениям лучше не прибегать. На том листе, на котором вы находитесь – куда импортировались данные, отображаются строки диапазонов внешних данных и, чтобы не нарушить структуру, следует сначала поместить сведения в другое место, а потом прибегать к перестановкам и подготовке отчета.

Самое простое решение – скопировать весь лист "как есть", однако, не годится. При копировании вы захватите ссылки на диапазоны тоже, а нам в отчете совсем не нужны лишние проблемы с отслеживанием, "поехали" строки данных, или нет. Поэтому, выделяем данные, "Копируем", переходим на другой лист (можно в этом же файле сделать лист для копий, предварительно установив в нем формат всех ячеек текстовым), но вставляем сначала лишь "Форматы", выбрав этот пункт в меню "Специальная вставка". (см. пометку 1 на Рис. 12).

При таком копировании мы получим раскраску файла и ширину колонок, соответствующую основному листу. И сразу, тут же, еще раз из того же самого места, где находится курсор, вставляем

специальной вставкой "значения" (см. Рис. 12: пометка 2). Теперь на листе копий у нас получилась точная копия того, что и на основном листе, только без ссылок на диапазоны внешних данных.

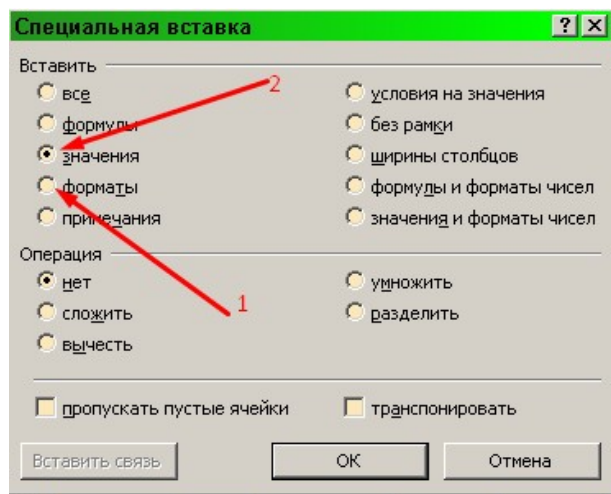


Рис.12. Меню ВСТАВКА - Специальная вставка

В этой таблице вы можете делать всё, что заблагорассудится, лишь бы найти нужные сведения: искать по файлу, ставить автофильтр, менять местами строки и столбцы, запускать сортировку данных, скрывать колонки и т.д. Можно даже написать макрокоманду, которая будет выставлять в колонках нужный формат, если нужно оперировать датами, например, а не их текстовыми представлениями.

Теперь самый важный момент: как найти "оригинал" документа, если данные о нем обнаружены в сводном файле. Если сведения были найдены на основном листе импорта данных (не в копии), то легко можно определить диапазон, откуда взята эта информация. Если "раскраска" строки не информирует об этом с определенностью, то следует установить курсор в содержащую нужные сведения ячейку и нажать правую кнопку мыши. В появившемся меню выбрать "Изменить запрос" и информация о файле исходных данных отобразится на экране (см. Рис. 13).

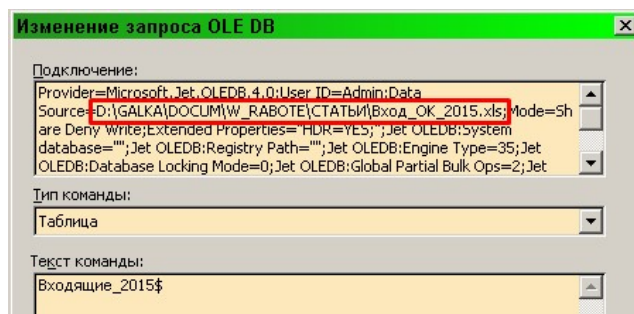


Рис.13. Файл-источник диапазона данных

В случае, если вы обнаружили нужную информацию на листе "копия сводки", то поиском найдите нужную ячейку на основном листе и посмотрите на источник данных.

Дальше всё проще простого: обращаетесь в подразделение, откуда взят источник, и просите копию документа с номером таким-то, или от такого-то числа. Вы можете открыть файл-источник сами и найти

нужную строку в описи, тогда будете знать регистрационный номер документа в подразделении, и поиск "оригинала" займет всего несколько минут.

5. Заключение

Объединение в один файл множества информации импортированием имеет большой плюс. Даже если данные в обрабатываемых файлах разношерстные, то обычный поиск нужно будет проводить всего по одному файлу, причем содержащему самые свежие сведения, а не по нескольким местам и не по "бумажным копиям".

Трудоемкость заполнения (как первоначального, так и в процессе появления новых источников данных) сводного файла невелика. Один человек способен сделать первоначальный вариант, содержащий сведения из около 30-ти оглавлений, всего за 2 часа. А добавление одного нового диапазона в уже "работающую" сводку занимает около 5-ти минут. Согласитесь, лучше затратить в начале каждого года немного времени, чтобы добавить в результирующий файл новые ссылки, чтобы потом в течение 10-ти минут можно было бы найти любую интересующую вас информацию, чем каждый раз, когда нужно что-то искать, обращаться за помощью к сотрудникам других (еще даже неизвестно каких) подразделений или поручать изыски отдельно взятому работнику, который

будет заниматься этим делом отнюдь не такое малое время.

Некоторые дополнительные усилия придется приложить, если экспортируемые данные, которые вы хотите использовать для поиска, автоматически не выгружаются из баз данных в нужном формате, или если используемые оглавления являются файлами Word. Несомненным "плюсом" является возможность импорта перезаписываемого в процессе экспорта или сохраненного "заменой" файла, сведения из которого учитываются в сводке. Никаких дополнительных настроек в результирующем файле при "подмене исходного" не требуется, разве что файл должен иметь такое же название, как в прошлый раз, и имя листа (если это был файл Microsoft Excel) не должно измениться.

В итоге можно сформулировать основные принципы, которым желательно следовать при объединении импортированием. Их всего два:

1) Соблюдение правил добавления новой и удаления ненужной информации четко в рамках полных столбцов строго внутри импортированного диапазона сохранит структуру результирующего файла корректной.

2) Искажения данных будут минимальны, если вы установите в сводном файле текстовый формат у всех импортируемых данных.

A Technique for Integration of Tabular Data to Optimize Search

G.L. Levchenkova

Abstract. A method for merging tabular data from multiple documents to optimize data search is presented. The source documents may be files either prepared manually or produced as a result of export from databases.

Keywords: information, data bases, data tables, import of data

Литература

1. С.Э.Зелинский Excel 2003: Самый популярный табличный процессор - СПб: Лидер, 2005.
2. О.Б. Калугина, В.С. Люцарев Работа с электронными таблицами. Microsoft Office Excel 2003. – М.: ИНТУИТ, 2006.
3. Г.Л. Левченкова, И.В. Холмов Система ДОКУМЕНТООБОРОТ. Организация документооборота и учета на рабочих местах и в подразделениях организаций // под редакцией академика РАН В.Б.Бетелина, Москва, НИИСИ РАН, 2006, С. 4-75
4. Г.Л. Левченкова, А.Г. Прилипко Перенос информации между базами данных, существующими в средах с разными операционными системами // Труды НИИСИ РАН. - Том 1 №1. Математическое и компьютерное моделирование систем: теоретические и прикладные аспекты. ISSN 2225-7349, Москва, 2011, С. 77-85
5. Г.Л. Левченкова Проблемы мобильности данных числового формата в отчетных документах // Труды НИИСИ РАН. - Том 5 №2. Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты. ISSN 2225-7349, Москва, 2015, С. 86-92

Повышение надёжности и масштабирование многопроцессорных систем в ОС РВ Багет

А.Н.Годунов¹, В.А.Солдатов²

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail's: ¹ nkag@niisi.ras.ru, ² nkvalera@niisi.ras.ru

Аннотация: При построении многопроцессорных вычислительных систем на основе коммуникационной среды RapidIO для пользователей возникает ряд специфических задач, связанных с конфигурированием и масштабированием системы, прокладкой маршрутов, повышения надёжности функционирования за счёт резервирования части оборудования системы. В работе рассматриваются основные средства операционной системы реального времени (ОС РВ) Багет, предназначенные для решения этих задач.

Ключевые слова: RapidIO, резервирование, подсистема, маршрутизация, ОС РВ Багет.

Введение

В НИИСИ РАН разработана линейка аппаратных средств, предназначенных для построения высокопроизводительных многопроцессорных систем [1]. Для взаимодействия между собой различных устройств используется коммуникационная среда RapidIO [2-3], которая по таким характеристикам, как пропускная способность, задержки при передаче пакетов в сети, надёжность и масштабируемость, превосходит многие другие коммуникационные средства [4]. Операционная система реального времени (ОС РВ) Багет [5,6] является основным программным средством, предназначенным для создания систем на указанном оборудовании.

Накопленный опыт проектирования многопроцессорных систем, которые могут включать в свой состав до нескольких сотен процессоров, показал, что особое значение приобретают такие задачи как:

- определение топологии сети, которая связывает узлы системы между собой;
- согласованное конфигурирование программного обеспечения всех узлов распределённой вычислительной системы;
- начальная инициализация коммуникационной среды, согласованная загрузка и запуск всех процессоров системы;
- резервирование некоторой части системы для повышения надёжности функционирования;
- сохранение ограниченной работоспособности системы после выключения некоторых подсистем;
- восстановление работоспособности после включения ранее выключенных подсистем.

Для решения первых двух задач, был разработан язык высокоуровневого описания системы (ВОС) [7], который позволяет описывать состав многопроцессорной системы, и указывать взаимосвязь между её отдельными узлами.

При компиляции исходного описания строятся маршруты, по которым будут передаваться сообщения между узлами системы. Результатом компиляции описания системы на языке ВОС является целевое описание системы (ЦОС) — файл формата ELF, который включается в целевой образ. В данной работе рассматриваются способы решения остальных задач приведённого списка средствами ОС РВ Багет на основании информации, содержащейся в ЦОС.

Оборудование

Коммуникационная среда RapidIO — это сеть с коммутацией пакетов, состоящая из узлов, соединённых *физическими каналами* связи. У каждого узла есть один или несколько *портов* — конечных точек физических каналов.

Узел может быть одного из следующих видов.

Оконечное устройство (ОУ). Характеризуется уникальным в пределах системы идентификатором RapidIO (далее «RIO-идентификатор»). В большинстве случаев ОУ имеет только один порт (например, BM7 в составе Комдив128-РИО).

Коммутатор (например, КП3). Не имеет RIO-идентификатора. Всегда имеет более одного порта и выполняет маршрутизацию — направляет пакет, пришедший через один порт, в другой порт в соответствии с таблицей маршрутизации.

Гибридное устройство — является одновременно и коммутатором, и ОУ (например, BM6 в составе Комдив64-РИО и ВГ18).

Коммуникационная среда RapidIO является составной частью распределённой вычислительной системы, которая организована в виде иерархии следующих видов компонент, помимо перечисленных выше видов узлов.

Процессорный элемент (ПЭ) (например, Комдив128-РИО, Комдив64-РИО) — микропроцессор с подключенным к нему ОУ, а также ОЗУ, ПЗУ, РПЗУ, наплатными контроллерами на шине РС1, внешними интерфейсами, устройствами хранения данных и прочими устройствами, доступ к которым возможен только от данного микропроцессора.

Мезонинная плата («мезонин») (например, М128): может включать ПЭ.

Модуль (например, ЦП-РИО128, ЦП-РИО64): может включать коммутаторы, ПЭ, ОУ и мезонинные платы.

Машина, крейт, прибор (например, ЦОИ-1, ЦОИ-2): состоит из модулей.

На рисунке 1 представлен пример, показывающий иерархию компонентов системы. Серым цветом выделены компоненты, относящиеся к коммуникационной среде RapidIO. Относительное положение ОУ в иерархии компонент определяется географическим адресом (ГА) – числом, полученным на основе номера прибора, номера модуля и номера ОУ на модуле. ПЭ элемент может самостоятельно определить ГА своего ОУ, прочитав содержимое определённого регистра.

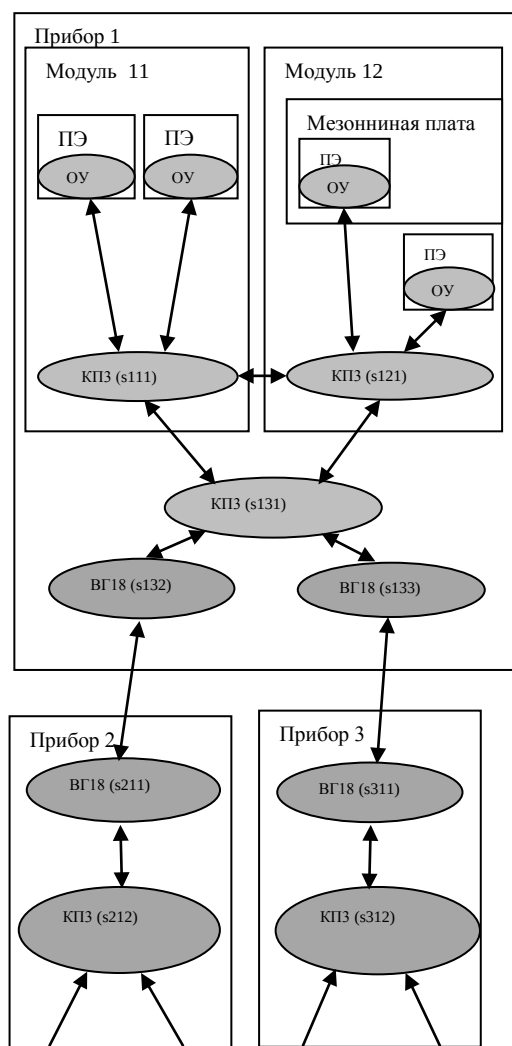


Рис 1. Пример соединения нескольких приборов.

Программное обеспечение

ОС РВ Багет [2,3] предоставляет следующие средства для передачи данных между отдельными ПЭ системы.

Каналы с очередью сообщений и без неё.

Специальные средства передачи сообщений по шине RapidIO (библиотека *mr*).

Эмуляция сети Ethernet на шине RapidIO.

Каналы связи реализованы в соответствии со стандартом ARINC-653 и является универсальным средством, которым можно пользоваться как ARINC-процессах, так и в POSIX-процессах. Каналы должны быть предварительно описаны при конфигурировании системы. Если один из участников обмена данными по каналам будет перезапущен, то это может вызвать лишь потерю некоторого количества сообщений между ним и другими участниками обмена.

Передача сообщений с помощью библиотеки *mr* выполняется по интерфейсу, разработанному специально для этой библиотеки, и отличается максимально возможной скоростью передачи данных, за счёт отсутствия копирования пересылаемых данных в памяти. Функции библиотеки *mr* можно использовать только после их инициализации. При инициализации указываются RIO-идентификаторы всех ПЭ, которые будут обмениваться сообщениями. Пользоваться средствами библиотеки *mr* можно только в POSIX-процессах.

Эмуляция сети Ethernet на шине RapidIO позволяет использовать интерфейс сокетов, который базируется на использовании семейства протоколов TCP/IP, применяемого для передачи данных по сети.

Постановка задачи

Резервирование — это обеспечение возможности отключения от системы одной из подсистем (основной, или резервируемой) и включения вместо неё аналогичной (резервной). При этом предполагается, что существует некоторая общая подсистема, к которой физически подключены обе вышеназванные подсистемы. Общая подсистема может быть не одна. Переключение на резервную подсистему обычно выполняется при обнаружении неполадок в работе основной. Здесь будет рассматриваться только холодное резервирование: если резервная подсистема не используется, питание на нее не подается.

Переключение с точки зрения пользователя заключается в том, что у основной системы отключается питание, а затем подается питание на резервную подсистему. В течение небольшого времени, необходимого для загрузки и инициализации резервной подсистемы, допускаются перебои в работе системы, но после завершения переключения система должна продолжать работу без потери функциональности или производительности.

Единицей резервирования является машина (прибор) или связанная группа машин, так как более мелкие элементы (модули) не имеют независимого питания. Ниже единица резервирования будет называться «подсистемой», независимо от её состава.

Разрыв/восстановление соединения — это отключение от системы одной из подсистем и включение этой же подсистемы через некоторое время. С прикладной точки зрения после выключения подсистемы будет решаться меньший круг задач на меньшем составе оборудования. После включения все ранее разорванные соединения должны быть восстановлены, и система должна продолжить работу в прежнем режиме.

Инициализация среды RapidIO

Инициализация коммуникационной среды RapidIO в пределах машины (прибора) выполняется программой начальной инициализации. Вся информация, необходимая для этой инициализации, содержится в целевом описании системы (ЦОС). После старта программа начальной инициализации определяет свой ГА, по значению которого отыскивается описание соответствующего ПЭ в ЦОС. Один ПЭ каждого прибора является ведущим (master), все остальные — ведомые (slave). На ведущих и ведомых ПЭ инициализация производится по-разному.

Если ПЭ является ведущим, то осуществляется обход компонент прибора по его описанию в ЦОС.

1. При обходе компонентов прибора на каждом компоненте ставится отметка о его посещении. Для этой цели используется регистр Component Tag CSR (регистр Tag), в который записывается ГА ведущего ПЭ. Таким образом, если прочитанное значение регистра Tag совпадает со значением ГА ведущего ПЭ, тогда считается, что этот компонент уже проинициализирован, иначе в регистр Tag записывается ГА ведущего ПЭ.

2. Если текущий компонент оказался коммутатором, то прежде чем настроить его, рекурсивно осуществляется обход компонент, соединённых с портами этого коммутатора.

3. При настройке коммутатора для каждого порта записывается таблица маршрутизации, которая определяет в какой порт будет направлен пришедший пакет в зависимости от содержащегося в нём RIO-идентификатора адресата.

4. При настройке ОУ в регистр Base Device ID CSR записывается RIO-идентификатор этого ОУ, значение которого берётся из ЦОС в зависимости от значения его ГА.

5. После инициализации всех компонент прибора в регистры Tag ОУ каждого ведомого ПЭ записывается предопределённое значение — DONE_TAG, означающее успешное окончание инициализации. Такой алгоритм гарантирует, что если для какого-либо ведомого ПЭ выполнена запись значения DONE_TAG в регистр Tag, то к этому моменту настроены все коммутаторы в пределах этого прибора и, следовательно, существуют

маршруты между ведущим и всеми ведомыми ПЭ.

Если ПЭ является ведомым, тогда программа начальной инициализации ожидает окончания инициализации прибора ведущим ПЭ, для этого выполняются следующие действия.

1. Считывает значение регистра Tag своего ОУ.

2. Если это значение равно DONE_TAG, то инициализация коммуникационной среды успешно закончена, иначе выполняется небольшая задержка и осуществляется переход к предыдущему шагу. Если число считываний регистра Tag превысило максимально допустимое значение, тогда считается, что инициализация коммуникационной среды не произошла из-за ошибок.

Если система состоит более чем из одного прибора, тогда в ней существуют граничные коммутаторы, которые соединяют различные приборы, что схематично представлено на рис. 1.

Граничными коммутаторами на рис.1 являются коммутаторы ВГ18, помеченные как s132, s133, s211 и s311. Каждый граничный коммутатор может быть инициализирован ведущим ПЭ двух приборов. Информация о том, что коммутатор является граничным, содержится в ЦОС. Инициализация граничных коммутаторов несколько отличается от инициализации остальных коммутаторов.

Для синхронизации доступа к граничному коммутатору с различных ведущих ПЭ предназначен регистр захвата компонента ID Lock CSR (регистр Lock). Если значение регистра Lock равно его начальному значению (0xffff), тогда запись нового значения в этот регистр заканчивается успешно. В остальных случаях, если в регистр Lock записывается значение, совпадающее с его содержимым, тогда операция также заканчивается успешно и значение регистра Lock становится равным 0xffff, а запись значения, отличного от содержимого этого регистра, заканчивается ошибкой и его содержимое не изменяется. При захвате коммутатора делается попытка записи в регистр Lock значения, совпадающего с ГА ведущего ПЭ. Если запись прошла без ошибок, то коммутатор считается захваченным, если же была выдана ошибка, то прочитав значение регистра Lock, можно узнать ГА ПЭ, который захватил этот коммутатор. Для освобождения захваченного коммутатора ведущий ПЭ записывает в регистр Lock значение, совпадающее с его ГА.

Если при попытке захвата граничного коммутатора выясняется, что он уже захвачен другим ведущим ПЭ, тогда считается, что коммутатор будет проинициализирован тем ведущим ПЭ, который его уже захватил, и программа инициализации продолжает обход остальных устройств прибора. Если же захват прошёл успешно, то программа начальной инициализации, которая добралась до него через один из двух портов, проверяет, подключено ли какое-либо устройство к его второму порту. Если устройство подключено, тогда считывается регистр Tag этого устройств. Если прочитать регистр Tag не удаётся, то это означает, что была разорвана связь между приборами в момент передачи пакетов, так как согласно протоколу

физического уровня RapidIO, для подтверждения приёма пакетов между портами коммутаторов используются идентификаторы пакетов `askID` – один на передачу, а другой на приём. Значения этих идентификаторов содержатся в соответствующих регистрах и после начального включения равны 0. Так как один прибор был заново включён, то идентификаторы передаваемых пакетов не будут совпадать с идентификаторами, которые ожидает порт коммутатора прибора, который не выключался. В этом случае запускается функция, которая соответствующим регистрам порта коммутатора присвоит начальные значения и передача данных между коммутаторами восстановится. Далее повторно читает регистр Tag.

Рассмотрим работу программы начальной инициализации в случае резервирования. Пусть на рис.1 прибор 1 – это **общий** прибор, прибор 2 – это **основной** прибор, а прибор 3 – это **резервный** прибор. У каждого прибора имеется один ведущий ПЭ, а все остальные ПЭ являются ведомыми. Обычно ведущий ПЭ настраивает коммутаторы только своего прибора, и производит в случае необходимости восстановление счётчиков неподтверждённых пакетов на граничных коммутаторах соседних приборов. В случае наличия резервирования, коммутатор **общего** прибора, через который подключаются **основной** и **резервный** приборы, может настраиваться с ведущих ПЭ каждого из трёх приборов. Для синхронизации доступа к этому коммутатору также используется ранее описанная процедура захвата коммутатора.

Программа начальной инициализации, выполняющаяся на ведущем ПЭ **общего** прибора (прибор 1 на рис. 1), пытается захватить коммутатор КПЗ (на рис. 1 помечен, как `s131`). Если коммутатор уже захвачен ведущим ПЭ другого прибора, тогда после небольшого ожидания делается повторная попытка захвата; и так до тех пор, пока коммутатор не будет захвачен, или не будет превышено максимально возможное число попыток захвата. После захвата коммутатора считывается регистр Tag. Если содержимое регистра совпадает с ГА ведущего ПЭ **основного** или **резервного** прибора, то считается, что коммутатор уже проинициализирован и его настройки (таблицы маршрутизации) не будут изменены после рекурсивного обхода всех соседних устройств, иначе в регистр Tag записывается ГА ПЭ **общего** прибора.

Если в процессе обхода соседних устройств коммутатора `s131`, программа попадает на граничный коммутатор, соединяющий **основной** или **резервный** прибор с **общим**, читается регистр Tag этого коммутатора. Если значение регистра Tag совпадает с ГА ведущего ПЭ **резервного** прибора, тогда считается, что **резервный** прибор загрузился раньше **общего** прибора, либо произошёл разрыв соединения со стороны **общего** прибора при его работе совместно с **резервным**. В этом случае в коммутатор `s131` **общего** прибора должны быть записаны таблицы маршрутизации, которые соединяет **общий** прибор с **резервным**, а иначе будут записаны таблицы маршрутизации, которые соединяют **общий**

прибор с **основным**. После обхода всех соседних устройств, если содержимое регистра Tag коммутатора `s131` совпадает с ГА ведущего ПЭ **общего** прибора, коммутатор инициализируется (записываются соответствующие таблицы маршрутизации.). Коммутатор `s131` освобождается.

Программа начальной инициализации, в случае своего выполнения на ведущем ПЭ **основного** или **резервного** приборов (прибор 2 или прибор 3) также пытается захватить коммутатор КПЗ общего прибора, после захвата коммутатора производится его инициализация (соответственно без обхода его соседей и без проверки содержимого регистра Tag) и освобождение коммутатора.

Разрыв/восстановление соединения

Процесс разрыва/восстановления соединения состоит из следующих шагов.

1. Отключается питание какой-либо подсистемы или прибора, имеющие соединения с другой подсистемой или прибором. В результате прекращается пересылка данных средствами библиотеки *tr*, пересылка данных по всем каналам связи, начинающимся или оканчивающимся в ПЭ отключённой подсистемы, а также разрываются соединения по сети Ethernet. ПЭ подсистемы, взаимодействовавшие с отключённой подсистемой, диагностируют ошибки передачи, и соответствующие каналы, средства библиотеки *tr* и соединения по сети Ethernet становятся неработоспособными и переходят в режим ожидания. Все каналы и средства библиотеки *tr* и сетевые соединения, не связанные с отключённой подсистемой, продолжают свою работу в обычном режиме.

2. Если через некоторое время подаётся питание на ранее отключённую подсистему или прибор, то на включённом оборудовании запускается программа начальной инициализации коммуникационной среды RapidIO, которая восстанавливает ранее разорванные маршруты. Под управлением ОСРВ Багет выполняется инициализация каналов, библиотеки *tr* и передачи данных по сети. Все ранее разорванные соединения между подсистемами восстанавливаются. Для прикладных программ разрыв/восстановление соединения происходит прозрачно, не считая не переданных (не полученных) сообщений за время разрыва соединений.

Переключение на резервную подсистему

Процесс переключения с основной подсистемы на резервную состоит из следующих шагов.

1. Отключается питание основной подсистемы. В результате прекращается передача по всем каналам, средствам передачи сообщений библиотеки *tr* и

сетевым соединениям, начинающихся или оканчивающихся в ПЭ отключённой подсистемы. ПЭ общей подсистемы, взаимодействовавшие с отключённой подсистемой, диагностируют ошибки передачи, и соответствующие каналы, средства библиотеки *tr* и передачи данных по сети переходят в режим ожидания.

Подаётся питание на резервную подсистему. Подсистема выполняет процедуру старта, в ходе которой запускается программа начальной инициализации коммуникационной среды RapidIO для всех ПЭ резервной подсистемы, включая коммутатор, установленный на общей подсистеме, через порты которого производится соединение с основной или резервной подсистемой. После восстановления маршрутов, ведущих на резервную подсистему, выполняется инициализация каналов, библиотеки *tr* и передачи данных по сети. Все ранее разорванные соединения восстанавливаются, но не с основной подсистемой, а с полностью идентичной ей – резервной. Для прикладных программ, выполнявших взаимодействие между ПЭ общей и основной или резервной подсистемой, переключение должно происходить совершенно прозрачно, не считая приостановки передачи на время переключения. RapidIO-идентификаторы ПЭ в основной и резервной подсистемах одинаковые.

Требования к основной и резервной подсистемам

1. Основная и резервная подсистемы должны быть физически идентичны, включая физические каналы, связывающие их с остальной системой, так как в противном случае невозможно гарантировать неизменность функциональности и производительности после переключения.

2. Основная и резервная подсистема должны быть соединены с остальной системой путем подключения к портам одних и тех же коммутаторов. То есть, если основная подсистема подключена к некоторому коммутатору, то соответствующая резервная подсистема должна быть тоже подключена именно к этому коммутатору. На рис. 1 представлен пример, соединяющий основную и резервную подсистемы с общей через коммутатор s131. Количество таких соединений произвольно, но устроены все они идентично.

3. Образы ОСРВ Багет, выполняющиеся на ПЭ основной и резервной подсистем должны быть полностью идентичны.

4. ARINC-конфигурации у соответствующих ПЭ основной и резервной подсистем должны совпадать.

Конфигурирование

Конфигурирование выполняется в ВОС в рамках описания типа системы или группы машин. Ниже приведено описание системы, состоящей из двух приборов 1-ЦОИ (20 ПЭ) и соединённых между собой по коммуникационной среде RapidIO.

```
<SystemDescription HldName = "s_1-coi_1-coi"
                    FileVersion="1.1">
<!-- Номер версии этого файла -->
<Include> 1COIType.xml </Include>
<Type Class="System" Name="s13">
  <Description>Описание элементов </Description>
  <!-- Основные A12 & A13 -->
  <Component Type="1COI" Name="16E08p00"
PhysAddr="13" />
  <Component Type="1COI" Name="16E08p01"
PhysAddr="0" />
  <InternalLinks> <!-- Связи между приборами
ЦОИ по SRio -->
    <!--+++++ 16E08-00 с 16E08-01 ++++++ -->
    <Link Port1="16E08p00:R8"
Port2="16E08p01:R9" />
  </InternalLinks>
</Type>
<Filter> <Match Pattern="16E" />
  <Var Name="MAIN_EL"
Value="B64p03" />
</Filter>
<!-- Установка ведущих -->
<Master>$(MachineFullName)/$(MAIN_EL)/K64p1
</Master>
  <RioId Name="/16E08p00/B64p11/K64p1"
Id="20" />
  <RioId Name="/16E08p01/B64p11/K64p1"
Id="40" />
  <AConfigPath>tar/arinc_ag$(RioId).config
</AConfigPath>
  <EthernetInterface>
    <Name>sm0</Name>
    <IpAddr>192.168.0.$(RioId)</IpAddr>
    <NetMask>255.255.255.0</NetMask>
    <Type>rio</Type>
  </EthernetInterface>
</SystemDescription>

Для проверки задачи резервирования
использовалась система, состоящая из одного
прибора 2-ЦОИ (8 ПЭ) и двух приборов 1-ЦОИ
(основного и резервного). Ниже приведено описание
этой системы.

<SystemDescription HldName="s_1_2coi"
                    FileVersion="1.1">
<!-- Номер версии этого файла -->
<Include>1COIType.xml</Include>
<Include>2COIType.xml</Include>
<Type Class="System" Name="s12">
  <Description>Описание элементов 0 ПК
  </Description>
  <!-- A12 -->
  <Component Type="1COI" Name="16E00p00"
PhysAddr="13" />
  <Component Type="2COI" Name="16D00p00"
PhysAddr="31" />
  <!-- A13 -->
  <Component Type="1COI" Name="16E00p00r"
PhysAddr="0" ReserveFor="16E00p00" />
  <InternalLinks> <!-- Связи между приборами
ЦОИ по SRio -->
```

```

    <Link                Port1="16E00p00:R2"
Port2="16D00p00:R1" />
    <Link                Port1="16E00p00r:R2"
Port2="16D00p00:R2" />
  </InternalLinks>
</Type>
  <Filter> <Match Pattern="16E" />
    <Var                Name="MAIN_EL"
Value="B64p03" />
  </Filter>
  <Filter> <Match Pattern="16D" />
    <Var                Name="MAIN_EL"
Value="B64p03" />
  </Filter>
  <!--      Установка      ведущих      -->
<Master>$(MachineFullName)/$(MAIN_EL)/K64p1</
Master>
  <RioId      Name="/16E00p00/B64p11/K64p1"
Id="20" />
<AConfigPath>/tar/arinc_ag$(RioId).config</AConfigP
ath>
  <EthernetInterface>
  <Name>sm0</Name>
  <IpAddr>192.168.0.$(RioId)</IpAddr>
  <NetMask>255.255.255.0</NetMask>
  <Type>rio</Type>
</EthernetInterface>
</SystemDescription>

```

Описание тестового примера

Для проверки восстановления работоспособности после разрыва и последующего восстановления соединения, а также для перехода к резервируемой подсистеме, была разработан тестовый пример, который, в зависимости от тестируемой задачи, отличается только составом оборудования, на котором он выполняется. Состав оборудования определяется в разделе «Конфигурирование».

Для проверки взаимодействия по каналам на каждом ПЭ системы запускаются ARINC-процессы, каждый из которых устанавливает по два канала связи с каждым другим ПЭ системы. Один канал на передачу данных, другой на приём данных. ПЭ с RIO-идентификатором, равным *RioMainCh*, получает от каждого ПЭ системы информацию об его обменах с другими ПЭ (сколько сообщений было отправлено и сколько принято). На этом ПЭ элементе выполняется дополнительный служебный поток, предназначенный для обработки статистической информации, полученной от других ПЭ. Обработанная информация выводится на консоль в виде сводной таблицы, в которой соединения между различными ПЭ, в зависимости от того обмениваются они сообщениями или нет, отмечаются символом '+' или '-'. Для задачи разрыв/восстановление соединения по каналам

связаны между собой 40 ПЭ, а для задачи резервирования 28 ПЭ.

Для проверки взаимодействия по библиотеке *tr* на 10 ПЭ системы запускаются POSIX-процессы, каждый из которых устанавливает связь на приём и передачу данных средствами библиотеки *tr* к каждому другим ПЭ из заданных 10 ПЭ. ПЭ с RIO-идентификатором, равным *RioMainMp*, получает от других ПЭ, участвующих в обменах данными, информацию об этих обменах и, с заданной периодичностью выводит на консоль сводную таблицу, аналогичную тесту обмена по каналам.

Для проверки работоспособности эмуляции сети Ethernet на шине RapidIO на любом ПЭ системы в произвольный момент времени может быть запущен контрольный пример, который проверяет связь по сети со всеми остальными ПЭ системы.

Тестирование осуществляется по следующей процедуре.

Загружается система, на всех ПЭ стартует тестовый пример, через 5-10 минут на ПЭ с RIO-идентификатором *RioMainCh* и *RioMainMp* по таблицами проверяется работоспособность системы. Запускается тест проверяющий работоспособность системы по эмуляции сети Ethernet на шине RapidIO. Если по всем соединениям идут обмены, тогда перезагружается один из приборов, участвующих в тестировании. После перезагрузки все связи между приборами должны восстановиться, что проверяется тем же способом, что и после первичной загрузки системы. Через некоторый промежуток времени производится последующая перезагрузка одного из приборов, участвующих в тестировании. Если в течении 10 часов выполнения тестовой задачи все обмены функционируют в полном объёме, тогда считается, что оборудование и программное обеспечение обеспечивают надёжную работу по связи между ПЭ системы по коммуникационной среде RapidIO.

Заключение

Рассмотрены средства конфигурирования для многопроцессорных вычислительных систем, функционирующих в коммуникационной среде RapidIO под управлением ОСРВ Багет. Конфигурирование выполняется системным интегратором, который составляет на языке ВОС описание системы в целом, используя библиотеку описания оборудования, поставляемую совместно с компилятором ВОС. Применение описанных средств позволяет в рамках предложенной технологии существенно упростить сложный процесс решения задачи взаимодействия между отдельными процессорами системы, а также сводит к описательному уровню решения таких задач, как холодное резервирование части оборудования, загрузки многих процессоров однотипными или различными образами ОС, реконфигурирования системы, в связи с заменой сбойного оборудования.

Increasing reliability and scaling of multiprocessor systems in RTOS Baget.

A.N.Godunov, V.A.Soldatov

Abstract: While constructing multiprocessor computing systems based on RapidIO communication environment, developers face a number of specific problems associated with configuring and scaling of the system, routing, increasing reliability of processing by means of hardware reservation. The RTOS Baget's main facilities for solving these problems are considered in this article.

Keywords: RapidIO, Reservation, subsystem, routing, RTOS Baget.

Литература

1. С.Г.Бобков. Импортзамещение элементной базы вычислительных систем – “Вестник Российской Академии Наук”, 2014, том 84, № 11, с. 1010–1016.
2. RapidIO Interconnect Specification. Part1: Input/Output Logical Specification. Revision 3.1 – RapidIO Trade Association, 2014.
3. RapidIO Interconnect Specification. Part 2: Message Passing Logical Specification. Revision 3.1 – RapidIO Trade Association, 2014.
4. С.Г.Бобков, С.О.Задябин, Перспективные высокопроизводительные вычислительные системы промышленного применения на базе стандарта RapidIO – «Электроника, микро и наноэлектроника: сб. научн. тр.», М., МИФИ, 2009, с.114-121.
5. А.Н.Годунов, Операционная система реального времени Багет 3.0 – «Программные продукты и системы», Тверь, 2010, №4, с.15-19.
6. А.Н.Годунов, В.А.Солдатов, Операционные системы семейства Багет (сходства, отличия и перспективы) – «Программирование», Москва, 2014, №5, с. 68-76.
7. А.Н.Годунов, В.А.Солдатов, Конфигурирование многопроцессорных систем в операционной системе реального времени Багет – «Программная инженерия», Москва, 2016, №6, с. 243-251.
8. Г.О.Райко, Ю.А.Павловский, В.С.Мельканович. Технология программирования многопроцессорной обработки гидроакустических сигналов на вычислительных устройствах семейства «КОМДИВ» – Научно-технический сборник «Гидроакустика», СПб.: Концерн «Океанприбор», 2014, вып. 20(2), с. 85-92.

Особые множества и их связь с особенностями решений уравнения эйконала

И.Г. Царьков

ФГБОУ ВО «Московский государственный университет имени М.В.Ломоносова»,
Механико-математический факультет, Москва, Россия,
E-mail : tsar@mech.math.msu.su

Аннотация: Изучаются геометрические свойства особого множества приближаемых множеств, полученные результаты применяются при изучении гладких решений уравнения эйконала.

Ключевые слова: особые и регулярные множества, точки солнечности, уравнение эйконала.

I. История вопроса и основные понятия

Одним из новых и перспективных направлений геометрической теории приближения является изучение ее методами решений уравнений Гамильтона-Якоби. Одним из представителей которого является уравнение эйконала: $|\nabla u| = 1$, где функция $u = u(x)$ обычно задана на открытом множестве $\Omega \subset \mathbf{R}^n$. Из простейших соображений нетрудно получить, что такие решения локально представляют собой функции вида $c \pm \rho(x, M)$ для некоторых констант $c \in \mathbf{R}$ и подмножеств $M \subset \mathbf{R}^n$, где $\rho(x, M)$ – расстояние до некоторого множества M от точки x . Само уравнение эйконала является основным уравнением геометрической оптики, с основами которой можно ознакомиться в монографиях М. Борна, Э. Вольфа, Дж. Бруса, П. Джиблина и Р. Фейнмана, Ю.А. Кравцова, Ю.И. Орлова, В. И. Арнольда, С. Н. Кужжова. Здесь мы обратим внимание на следующий факт: свойство гладкости решения уравнения эйконала взаимосвязано с геометрическо-аппроксимативным понятием локальной солнечности поверхностей уровня этого решения. По этой причине в данной работе мы будем исследовать свойства локальной солнечности (регулярности) и множества особых точек.

Подробнее о связи C^1 -решений уравнения эйконала с геометрической теорией приближения и свойствами функций расстояния до множества можно посмотреть в работе [Б-2], а также ознакомиться со свойствами функций расстояния и свойств солнечности в работах [Б-3]. Далее в качестве X будем рассматривать некоторое

конечномерное линейное нормированное пространство. Для множества $M \subset X$ и точки $x \in X$ через $\rho(x, M)$ обозначим расстояние от x до M , т.е. величину $\inf_{y \in M} \|x - y\|$. Те точки $y \in M$, для которых $\|x - y\| = \rho(x, M)$, будем называть ближайшими к x в M . Через $P_M x$ обозначим множество всех ближайших точек для x . Через $B(x, r)$ обозначим замкнутый шар с центром x радиуса r .

Определение. Точка $x \in X \setminus M$ называется регулярной для замкнутого множества $M \subset X$, если все точки некоторой окрестности $O(x)$ являются точками единственности (т.е. для каждой из них существует единственная ближайшая в M). Точки, не являющиеся регулярными или принадлежащие замыканию $\text{int } M$, будем называть особыми.

Отметим, что для конечномерного пространства X в регулярной точке $x_0 \in X \setminus M$ выполняется свойство локальной строгой солнечности (см обзоры [Б-2]–[Б-4]), т.е. для любой точки $x \in O(x_0)$ существует (единственная ближайшая) y из M такая, что эта точка y будет являться ближайшей в M для всех точек из пересечения $O(x_0)$ и луча ℓ_y , выходящего из y и проходящего через x .

Отметим также, что множество всех регулярных точек (регулярное множество) является открытым, а множество всех особых точек (особое множество) точек является замкнутым. Непустое особое множество является замыканием точек

неединственности (т.е. тех x, y , у которых не менее двух ближайших точек).

II. Свойства особых множеств границы области, основные результаты.

Теорема. Пусть Ω – область в конечномерном гладком строго выпуклом пространстве X и E – множество всех особых точек для границы $\partial\Omega$, лежащих в Ω , и пусть G – объединение всех полуинтервалов $[x, y)$, где x – произвольная точка из E , а y – произвольная ближайшая точка из $\partial\Omega$ для x . Тогда вышеуказанные полуинтервалы не пересекаются со внутренностями других, а множество $\Omega \setminus G$ и $\partial G \cap \Omega$ составлено из открытых непересекающихся лучей, начало которых лежит на $\partial\Omega$ и является единственной ближайшей для всех точек этого луча. При этом множество всех начал лучей – замкнутое подмножество в $\partial\Omega$.

Доказательство. Из строгой выпуклости пространства вытекает, что для любой точки $x \in \Omega$ и ее произвольной ближайшей $y \in \partial\Omega$ все точки $[y, x)$ имеют единственной ближайшей в $\partial\Omega$ точку y . Отсюда следует, что все полуинтервалы вида $[y, x)$, где $x \in E$, а y – ближайшая для x , не могут пересекаться своими относительными внутренностями. Для всех точек $z \in \Omega \setminus G$ в силу их регулярности и рассуждений из доказательства предыдущей теоремы существует открытый луч ℓ , исходящий из некоторой точки $\omega \in \partial\Omega$ и проходящий через точку z , состоит из регулярных точек, у которых точка ω является ближайшей. При этом луч содержится в $\Omega \setminus G$. Из соображений компактности вытекает, что $\partial G \cap \Omega$ также состоит из лучей, для всех точек которых их начало на $\partial\Omega$ является ближайшей. Опять же из строгой выпуклости пространства вытекает, что начало луча является единственной ближайшей для каждой точки этого луча. Кроме того, множество всех начал лучей, рассмотренных выше, замкнуто в $\partial\Omega$.

Замечание. В случае, когда дополнительно, пространство X имеет модуль выпуклости 2-го порядка, а $\partial\Omega$ – след C^2 -поверхности, то указанные в предыдущей теореме полуинтервалы не пересекаются со внутренностями других полуинтервалов, а своими внутренностями – со

множеством E . При этом множество G открыто, а его граница состоит из регулярных точек.

Доказательство. Если y – ближайшая точка для некоторой точки $x \in \Omega$ во множестве $\partial\Omega$, то радиус кривизны $r(y)$ поверхности $\partial\Omega$ относительно нормы пространства не превосходит $\|x - y\|$. Из строгой выпуклости пространства вытекает, что y – единственная ближайшая для любой точки $z \in (x, y)$. В достаточно малой окрестности $O(z)$ все ее точки имеют ближайшие точки в достаточно малой окрестности точки y , поэтому можно считать, что для любой точки $\omega \in O(z)$ радиусы кривизны (относительно нормы пространства X) в ближайших точках к ω строго больше $\rho(\omega, \partial\Omega)$. Отсюда вытекает одноточечность множества ближайших точек к ω , а, следовательно, регулярность точек окрестности $O(z)$. Отсюда $E \cap (x, y) = \emptyset$.

Следствие. Пусть Ω – область в конечномерном гладком строго выпуклом пространстве X , в которой нет конечных особых точек. Тогда множество $X \setminus \Omega$ выпукло.

Теорема. Пусть X – конечномерное гладкое строго выпуклое пространство, на области $\Omega \subset X$ задано C^1 -решение $u: \Omega \rightarrow \mathbb{R}$ уравнения эйконала $\|\nabla u\| = 1$. И пусть для некоторой поверхности уровня $M_c = \{x \in \Omega \mid u(x) = c\}$ найдется такое всюду плотное в ней множество, в точках которого все нормальные прямые к M_c разделяются (этими точками) на два луча, один из которых не пересекается с $\partial\Omega$. Тогда M_c – граница некоторого выпуклого замкнутого тела $G \subset X$, пересеченного с Ω .

Доказательство. Условие $u \in C^1(\Omega)$ означает, что в Ω нет особых точек поверхности уровня M_c . Пусть $A \subset M_c$ – всюду плотное множество в M_c , в точках которого любая ортогональная к M_c прямая делится на два луча, один из которых лежит в Ω . Из гладкости пространства следует, что для каждой точки множества A есть опорное в этой точке к M_c

открытое полупространство (содержащееся в Ω). Из соображений компактности вытекает наличие такого опорного пространства в каждой точке поверхности уровня M_c . Отсюда следует утверждение теоремы.

Пример 1. Пусть Ω открытое множество в двумерном пространстве $X = \Omega \cup \partial\Omega$, и любой кусок границы $\partial\Omega$ не является следом простой липшицевой кривой. В этом случае никакой кусок границы не может быть особым множеством выпуклой поверхности. Отсюда из приведенных выше теорем в частности вытекает, что всякая линия уровня C^2 -гладкого решения уравнения эйконала в $\Omega \subset X = \mathbf{R}^2$ может иметь только точку или пустое множество в качестве особого множества. Поэтому класс всех C^2 -гладких решений уравнения эйконала в каждой связной компоненте Ω состоит из функций вида $c \pm (a, x)$ ($|a| = 1$) и $c \pm |x - x_0|$ ($x_0 \in \partial\Omega$).

Пример 2. Пусть $\Omega \subset \mathbf{R}^n$ такое открытое множество, что $\partial\Omega = \mathbf{R}^n \setminus \Omega$, и граница $\partial\Omega$ – либо связное ограниченное множество, либо множество хаусдорфовой размерности $< \frac{n-1}{2}$. Тогда след всякой поверхности

уровня C^1 -решения уравнения эйконала в Ω представляет собой границу выпуклого тела, содержащего особое множество E этой поверхности уровня. При этом, конечно, $E \subset \partial\Omega$.

Пример 3. Пусть особое множество E некоторой C^2 -поверхности в $\mathbf{R}^n$ представляет собой связное множество. Тогда след этой поверхности представляет собой границу выпуклого тела, содержащего особое множество E .

Работа выполнена при поддержке РФФИ, проект № 16-01-00295-а.

Singular set and its relation with peculiarity of solution to the eikonal equation

I.G. Tsarkov

Abstract: We investigate a geometrical properties of a singular sets of a approximated sets, and received results are used for research a smooth solution to the eikonal equation.

Keywords: singular sets, regular sets, solarity points, eikonal equation.

ЛИТЕРАТУРА

А. КНИГИ:

- [А-1] М. Борн, Э. Вольф. «Основы оптики». Москва, «Наука», 1970.
- [А-2] Дж. Брус, П. Джиблин. «Кривые и особенности. Геометрическое введение в теорию особенностей». Москва, «Мир», 1988.
- [А-3] Р. Фейнман, Р. Лейтон, М. Сэндс. «Фейнмановские лекции по физике. Том 3. Излучение, волны, кванты». Москва, «Мир», 1976.
- [А-4] Ю. А. Кравцов, Ю. И. Орлов. «Геометрическая оптика неоднородных сред». Москва, «Наука», 1980.
- [А-5] В. И. Арнольд. «Особенности каустик и волновых фронтов». Москва, «Фазис», 1996.

Б. СТАТЬИ:

- [Б-1] С. Н. Кружков. «Обобщенные решения уравнений Гамильтона-Якоби типа эйконала. I. Постановка задач, теоремы существования, единственности и устойчивости, некоторые свойства решений». Математический сборник. т. 98 (140), 1975, № 3(11). 450-493.
- [Б-2] А. Р. Алимов, И. Г. Царьков. «Связность и солнечность в задачах наилучшего и почти наилучшего приближения». Успехи математических наук, т. 71 (2016), № 1 (427), 3-84.
- [Б-3] В. С. Балаганский, Л. П. Власов. «Проблема выпуклости чебышёвских множеств». Успехи математических наук, т. 51 (1996), № 6(312), 125-188.
- [Б-4] А. Р. Алимов, И. Г. Царьков. «Связность и другие геометрические свойства солнц и чебышёвских множеств». Фундаментальная и прикладная математика.. т. 63 (2014), № 4, 21-91.

Об интерполяции функций обобщённо-полиномиальными сплайнами, построенными на базе гладких чебышёвских систем

В.Б. Демидович

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН» -
ФГБОУ ВО «Московский государственный университет имени М.В.Ломоносова»,
Механико-математический факультет, Москва, Россия,
E-mail : vasdem@mech.math.msu.su

Аннотация: Рассматривается задача интерполяции функций обобщённо-полиномиальными сплайнами, построенными на базе гладких чебышёвских систем. Предложена структурная формула для соответствующих интерполяционных сплайнов и сформулированы условия их однозначного существования в поставленной задаче.

Ключевые слова: гладкие чебышёвские системы, обобщённые полиномы, обобщённо-полиномиальные сплайны, интерполяция.

I. Исходные сведения

В обзорной статье [Б-5] были подробно изложены основные сведения про фундаментальные (в пространстве $C([a, b])$) обобщённые полные чебышёвские системы функций в связи с задачей интерполирования функции, заданной (в узлах из отрезка $[a, b]$) своими значениями, соответствующими обобщёнными полиномами. Здесь же мы будем использовать эти сведения для исследования более общей задачи, когда заданная (в узлах из отрезка $[a, b]$) функция интерполируется обобщённо-полиномиальными сплайнами, представляющими собой «склею» обобщённых полиномов», вводимых на подотрезках отрезка $[a, b]$, при разбиении $[a, b]$ на эти подотрезки. Кратко напомним нужные далее сведения из указанной статьи.

Совокупность функций

$$\{e_k(\cdot)\}_{k=0}^{\infty} \in C^{(\infty)}([a, b]) \quad (1.1)$$

образует, на отрезке $[a, b] \subset R$ ($a < b$), фундаментальную гладкую чебышёвскую систему в пространстве $C([a, b])$ если:

А) для каждого её (конечного) набора функций

$$\{e_k(\cdot)\}_{k=0}^j \quad (j = 0, 1, 2, \dots) \quad (1.2)$$

построенный по нему нетривиальный обобщённый полином порядка j

$$\phi_j(t) = \sum_{i=0}^j c_i e_i(t) \quad \left(\sum_{i=0}^j c_i^2 \neq 0 \right) \quad (1.3)$$

имеет на $[a, b]$ не более j корней с учётом их алгебраических кратностей (то есть, каждый набор функций (1.2) служит *ECT – системой j -го порядка на отрезке $[a, b]$* , иначе говоря, он является *Extended Complete Tchebycheff systeme* на рассматриваемом отрезке);

Б) соответствующее линейное пространство гладких чебышёвских обобщённых полиномов

$$\Phi([a, b]) = \bigcup_{j=0}^{\infty} \Phi_j([a, b]), \quad (1.4)$$

где

$$\Phi_j([a, b]) = \text{span}\{e_i(t)\}_0^j \quad (1.5)$$

- линейная оболочка гладких чебышёвских обобщённых полиномов порядка $\leq j$ (порождающая в пространстве $\Phi([a, b])$ подпространство размерности $j+1$), плотно в пространстве $C([a, b])$ относительно его равномерной нормировки, и, тем самым, для замыкания пространства $\Phi([a, b])$ справедливо равенство

$$\text{cl } \Phi([a, b]) = C([a, b]). \quad (1.6)$$

Примеры гладких чебышёвских систем функций, фундаментальных в пространстве $C([a, b])$, приведены в [Б-5]. Там подробно изложен и аппарат исследований гладких чебышёвских систем функций, из которого мы выделим лишь следующее.

1. Вводя детерминанты (1.7) и вронскианы (1.8)

$$W\left(\begin{matrix} e_0, & \dots, & e_{j-1}, & e_j \\ s, & \dots, & s, & t \end{matrix}\right) = \left\| \begin{matrix} e_0(s) & \dots & e_{j-1}(s) & e_j(s) \\ e_0^{(1)}(s) & \dots & e_{j-1}^{(1)}(s) & e_j^{(1)}(s) \\ \dots & \dots & \dots & \dots \\ e_0^{(j-1)}(s) & \dots & e_{j-1}^{(j-1)}(s) & e_j^{(j-1)}(s) \\ e_0(t) & \dots & e_{j-1}(t) & e_j(t) \end{matrix} \right\| \quad (j=0,1,2,\dots), \quad (1.7)$$

$$W(e_0(s), e_1(s), \dots, e_j(s)) = \left\| \begin{matrix} e_0(s) & \dots & e_{j-1}(s) & e_j(s) \\ e_0^{(1)}(s) & \dots & e_{j-1}^{(1)}(s) & e_j^{(1)}(s) \\ \dots & \dots & \dots & \dots \\ e_0^{(j-1)}(s) & \dots & e_{j-1}^{(j-1)}(s) & e_j^{(j-1)}(s) \\ e_0^{(j)}(s) & \dots & e_{j-1}^{(j)}(s) & e_j^{(j)}(s) \end{matrix} \right\| \quad (j=0,1,2,\dots), \quad (1.8)$$

там были определены:

А) обобщённо-полиномиальные ядра

$$K_j(t, s) = \frac{W\left(\begin{matrix} e_0, & \dots, & e_{j-1}, & e_j \\ s, & \dots, & s, & t \end{matrix}\right)}{W(e_0(s), \dots, e_{j-1}(s))}, \quad (1.9)$$

$$K_0(t, s) =: e_0(t) \quad (j=1, 2, \dots);$$

Б) обобщённые производные от заданной функции $x(t) \in C^{(j)}([a, b])$ в силу рассматриваемой чебышёвской системы

$$D_j[x(t)] =: \frac{W(e_0(t), \dots, e_{j-1}(t), x(t))}{W(e_0(t), \dots, e_{j-1}(t))},$$

$$D_0[x(t)] =: x(t) \quad (j=1, 2, \dots); \quad (1.10)$$

В) усреднённые обобщённо-полиномиальные ядра

$$K_j^*(t, s) = \frac{K_j(t, s)}{D_j[e_j(s)]} \quad (j=0, 1, 2, \dots). \quad (1.11)$$

2. В связи с задачей чебышёвского обобщённого тэйлоровского разложения гладкой функции

$$x(t) \in C^{(n)}([a, b]) \quad (1.12)$$

относительно заданной начальной точки

$$t_0 \in [a, b] \quad (1.13)$$

отмечалось, что соответствующий гладкий чебышёвский обобщённый полином

$$\phi_n(t) = \sum_{i=0}^n c_i e_i(t) \in \Phi_n([a, b]), \quad (1.14)$$

удовлетворяющий равенствам

$$\phi_n^{(j)}(t_0) = x^{(j)}(t_0) \quad (j=0, 1, \dots, n), \quad (1.15)$$

имеет два основных представления:

А) лагранжа-тэйлоровского типа

$$\phi_n(t) = \sum_{j=0}^n U_j(t; t_0) x^{(j)}(t_0), \quad (1.16)$$

где

$$U_j(t; t_0) =$$

$$\left\| \begin{matrix} e_0(t_0) & \dots & e_n(t_0) \\ \dots & \dots & \dots \\ e_0^{(j-1)}(t_0) & \dots & e_n^{(j-1)}(t_0) \\ e_0(t) & \dots & e_n(t) \\ e_0^{(j+1)}(t_0) & \dots & e_n^{(j+1)}(t_0) \\ \dots & \dots & \dots \\ e_0^{(n)}(t_0) & \dots & e_n^{(n)}(t_0) \end{matrix} \right\| \quad (1.17)$$

$$\frac{W(e_0(t_0), \dots, e_n(t_0))}{W(e_0(t_0), \dots, e_n(t_0))} \quad (0 \leq j \leq n)$$

- соответствующие гладкие чебышёвские обобщённые полиномы;

Б) коши-тэйлоровского типа

$$\phi_n(t) = \sum_{j=0}^n V_j(t; t_0) D_j[x(t_0)], \quad (1.18)$$

где

$$V_j(t; t_0) = K_j^*(t, t_0) \quad (0 \leq j \leq n) \quad (1.19)$$

- также соответствующие гладкие чебышёвские обобщённые полиномы.

3. Наконец, там напоминалось понятие *срезки* функции $f(t, s)$, определяя её традиционным образом:

$$f^+(t, s) =: \begin{cases} f(t, s) & \text{при } t > s \\ 0 & \text{при } t \leq s \end{cases}. \quad (1.20)$$

II. Об интерполировании функций гладкими Чебышёвскими обобщённо-полиномиальными сплайнами

1. Пусть на отрезке $[a, b]$ задано фиксированное разбиение

$$\Omega_N[a, b]: a = t_0 < t_1 < \dots < t_N < t_{N+1} = b, \quad (2.1)$$

образованное набором его внутренних узлов $\{t_j\}_{j=1}^N$ ($N \geq 1$ – целое) и концами исходного отрезка.

Функцию $S(t)$ мы будем называть *гладким чебышёвским обобщённо-полиномиальным сплайном порядка $n \geq 0$* ($n \geq 0$ – целое), характеризуемым *гладкостями*

$$\{v_j\}_1^N \quad (-1 \leq v_j < n - \text{целые})$$

в узлах $\{t_j\}_1^N$ разбиения $\Omega_N[a, b]$, если:

а) на каждом подотрезке

$$[t_j, t_{j+1}] \quad (j = 0, 1, \dots, N) \quad (2.2)$$

$S(t)$ является гладким чебышёвским обобщённым полиномом из $\Phi_n([a, b])$, причём хотя бы на одном подотрезке соответствующий обобщённый полином имеет в точности n -ый порядок;

б) во внутренних узлах разбиения

$$\Omega_N[a, b]$$

$$t_j \quad (j = 1, \dots, N) \quad (2.3)$$

функции $S(t), S^{(1)}(t), \dots, S^{(v_j)}(t)$ непрерывны,

а $S^{(v_j+1)}(t)$ терпит конечный разрыв.

Числом

$$v =: \min_{1 \leq j \leq N} v_j \quad (-1 \leq v < n) \quad (2.4)$$

тогда определяется *гладкость* данного сплайна на всём отрезке $[a, b]$.

Замечание 1. Из требования б) следует, что в точках (2.3) функции $D_i[S(t)]$ ($0 \leq i \leq v_j$) также должны быть непрерывными, а функция $D_{v_j+1}[S(t)]$ могут претерпевать конечный разрыв.

Замечание 2. Для удобства дальнейшего мы включаем в рассмотрение и случай

$v_j = -1$, когда в точке t_j сама функция $S(t)$ терпит конечный разрыв.

С другой стороны, требования выполнения строгих неравенств $v_j < n$ ($j = 1, 2, \dots, N$) логически

оправдано, поскольку при $v_j = n$ мы приходим к ситуации, когда на соответствующем «сдвоенном подотрезке» $[t_{j-1}, t_{j+1}]$ функция $S(t)$ оказалась бы (благодаря «чебышёвости» пространства $\Phi([a, b])$) единым гладким чебышёвским обобщённым полиномом из подпространства $\Phi_n([a, b])$, и говорить о разрыве её производной $S^{(v_j+1)}(t)$ в точке t_j было бы бессмысленно.

Замечание 3. Значения сплайна, или его производных, в точках разрыва, как правило, для нас не будут играть существенной роли. Но для определённости, если только не будет ничего специально оговоренного, мы будем считать их равными «полусуммами односторонних пределов слева и справа».

Замечание 4. При описании сплайнов вместо понятия *гладкости* используется, также, понятие *дефектности*.

Именно, набором чисел

$$\eta_j = n - v_j \quad (0 < \eta_j \leq n + 1) \quad (2.5)$$

определяются *дефекты сплайна $S(t)$ по узлам t_j* ($j = 1, \dots, N$), а величиной

$$\eta =: \max_{1 \leq j \leq N} \eta_j \quad (0 < \eta \leq n + 1) \quad (2.6)$$

- *дефект этого сплайна на всём отрезке $[a, b]$.*

Ясно, что в теории сплайнов понятия «гладкость» и «дефектность» взаимодополняемы.

2. Пусть теперь $\{S(t)\}$ - совокупность гладких чебышёвских обобщённо-полиномиальных сплайнов порядка $n \geq 0$ с гладкостями $\{v_j\}_1^N$ в узлах $\{t_j\}_1^N$ разбиения (2.1). Эта совокупность, вообще говоря, не обладает свойством линейности, поскольку, скажем, при суммировании двух её элементов у соответствующей суммы *гладкость по узлам может измениться*: «понизиться» она, разумеется, не может, но вполне возможно её

«повышение» (например, суммируя сплайн со своим «противоположным сплайном», мы придём «к тождественному нулю», то есть к элементу *со всюду бесконечной гладкостью*).

Однако данную совокупность сплайнов можно *пополнить* аналогичными «склейки из кусков» гладких чебышёвских обобщённых полиномов, где гладкости по узлам $\{t_j\}_1^N$ допустимы и выше V_j , в том числе не исключая возможности наличия склейки в этих узлах *по всем производным вплоть до n -го порядка* (в последнем случае, согласно Замечанию 2, получающийся сплайн $S(t)$ просто обратится в *единый гладкий чебышёвский обобщённый полином из $\Phi_n([a,b])$*). Тогда мы приходим к *совокупности сплайнов*, в узлах $\{t_j\}_1^N$ характеризующихся гладкостями $\{V_j\}_1^N$, для которой операции сложения элементов и их умножения на действительные числа уже не выводят за её пределы. Такое линейное многообразие сплайнов мы будем обозначать через $S_n^V(\Omega_N[a,b])$ и называть (линейным) *пространством гладких чебышёвских обобщённо-полиномиальных сплайнов порядка $n \geq 0$ гладкости V в узлах разбиения $\Omega_N[a,b]$* , а его элементы - по-прежнему, просто *гладкими чебышёвскими обобщённо-полиномиальными сплайнами*. Ясно, что при этом

$$S_n^V(\Omega_N[a,b]) \begin{cases} \supset \Phi_n([a,b]) \\ \text{при } -1 \leq V < n; \\ = \Phi_n([a,b]) \\ \text{при } V \geq n. \end{cases} \quad (2.7)$$

и, тем самым, справедлива цепочка вложений

$$\Phi_n([a,b]) \subseteq S_n^V(\Omega_N[a,b]) \subset C^{(V)}([a,b]), \quad (2.8)$$

верная при любых (допустимых) значениях параметров n, V, N , где под $C^{(0)}([a,b])$ понимается пространство $C([a,b])$, а под $C^{(-1)}([a,b])$ - его *естественное расширение* за счёт пополнения его *кусочно-непрерывными функциями с конечными точками разрыва*.

3. Пусть величиной $\dim S_n^V(\Omega_N[a,b])$ характеризуется *размерность* пространства $S_n^V(\Omega_N[a,b])$. Из (2.7) сразу заключаем, что $\dim S_n^V(\Omega_N[a,b]) \geq \dim \Phi_n([a,b]) = n+1$. (2.9)

С другой стороны, поскольку число промежутков $[t_j, t_{j+1}]$ разбиения $\Omega_N[a,b]$ равно $N+1$, и на каждом промежутке

соответствующий гладкий чебышёвский обобщённый полином из $\Phi_n([a,b])$ задаётся не более чем $n+1$ коэффициентами, то справедливо неравенство

$$\dim S_n^V(\Omega_N[a,b]) \leq (N+1)(n+1). \quad (2.10)$$

Объединяя неравенства (2.9)-(2.10), приходим к *априорной оценке* для *размерности* пространства $S_n^V(\Omega_N[a,b])$:

$$n+1 \leq \dim S_n^V(\Omega_N[a,b]) \leq (N+1)(n+1). \quad (2.11)$$

Подробные доказательства дальнейших трёх основных теорем в отношении элементов пространства $S_n^V(\Omega_N[a,b])$, увы, весьма «канительны» - у меня они заняли, в сумме, около сорока рукописных страниц. Поэтому при их изложении мы ограничимся лишь *эскизами доказательств*.

4. Рассмотрим сначала вопрос о *структурном представлении и характеристиках* элементов пространства $S_n^V(\Omega_N[a,b])$. Здесь имеет место следующее утверждение:

Теорема 1. Всякий гладкий чебышёвский обобщённо-полиномиальный сплайн $S(t)$ порядка $n \geq 0$ ($n \geq 0$ - целое), характеризуемый *гладкостями*

$$\{V_j\}_1^N \quad (-1 \leq V_j < n - \text{целые})$$

по узлам

$$\{t_j\}_1^N \quad (N \geq 1 - \text{целое})$$

разбиения $\Omega_N[a,b]$, представим на $[a,b]$ в виде *линейной комбинации*

$$S(t) = \sum_{i=0}^n c_{i0} K_i(t; a) + \sum_{j=1}^N \sum_{i=V_j+1}^n c_{ij} K_i^+(t; t_j), \quad (2.12)$$

в которой для соответствующих коэффициентов справедливы соотношения

$$c_{i0} = \frac{D_i[S(a)]}{D_i[e_i(a)]} \quad (i=0,1,\dots,n),$$

$$c_{ij} = \frac{D_i[S(t_j+0)] - D_i[S(t_j-0)]}{D_i[e_i(t_j)]} \quad (i=V_j+1, V_j+2, \dots, n; j=1,\dots,N), \quad (2.13)$$

где

$$K_i(t, a) = \frac{W(e_0, \dots, e_{i-1}, e_i)}{W(e_0(a), \dots, e_{i-1}(a))},$$

$$K_0(t, a) =: e_0(t) \quad (i=1, 2, \dots, n);$$

$$K_i(t, t_j) = \frac{W(e_0, \dots, e_{i-1}, e_i)}{W(e_0(t_j), \dots, e_{i-1}(t_j))}, \quad (2.14)$$

$$K_0(t, t_j) =: e_0(t)$$

$$(i=1, 2, \dots, n; j=1, 2, \dots, N)$$

- значения *обобщённо-полиномиальных ядер* $\{K_i(t, s)\}_{i=0}^n$. при $s = a, t_1, t_2, \dots, t_N$ соответственно, а

$$K_i^+(t; t_j) =: \begin{cases} K_i(t; t_j) & \text{при } t > t_j \\ 0 & \text{при } t \leq t_j \end{cases} \quad (2.15)$$

$$(i=1, 2, \dots, n; j=1, 2, \dots, N)$$

- *срезки* этих ядер.

Обратно, любая линейная комбинация вида (2.12), в которой

$$c_{nj} \neq 0 \quad (j=1, 2, \dots, N), \quad (2.16)$$

где $n \geq 0$ – целое, $N \geq 1$ – целое, $-1 \leq \nu_j < n$ – целые, $\{t_j\}_1^N$ набор внутренних точек отрезка $[a, b]$, определяющих соответственное разбиение $\Omega_N[a, b]$, представляет собой некий гладкий чебышёвский обобщённо-полиномиальный сплайн *порядка* n , характеризуемый *гладкостями* $\{\nu_j\}_1^N$ по узлам $\{t_j\}_1^N$ разбиения $\Omega_N[a, b]$.

Эскиз доказательства.

Обоснование этого утверждения можно провести *поэтапно*.

На *1-ом этапе* устанавливается (от противного) *линейная независимость* системы функций

$$\{K_0(t; a), \dots, K_n(t; a); K_{\nu_1+1}^+(t; t_1), \dots, K_n^+(t; t_1); \dots; K_{\nu_N+1}^+(t; t_N), \dots, K_n^+(t; t_N)\}. \quad (2.17)$$

А именно, полагая существование *нетривиального* набора коэффициентов $\{\tilde{c}_{0,0}, \dots, \tilde{c}_{n,0}, \tilde{c}_{\nu_1+1,1}, \dots, \tilde{c}_{n,1}, \dots, \tilde{c}_{\nu_N+1,N}, \dots, \tilde{c}_{n,N}\}$, (2.18) обращающего в *тождественный* нуль соответствующую линейную комбинацию функций (2.17), и «двигаясь слева-направо от подотрезка к подотрезку» (с учётом известной линейной независимости обобщённо-полиномиальных ядер на всяком промежутке в $[a, b]$), мы обнаружим, что все коэффициенты

нетривиального набора (2.18) равны нулю. Полученным противоречием и объясняется линейная независимость системы функций (2.17).

На *2-ом этапе* устанавливаются требуемые соотношения (2.13) для коэффициентов рассматриваемого сплайна $S(t)$, возникающие при «последовательных склейках» как *самих обобщённых полиномов* $\varphi_{j-1}(t), \varphi_j(t) \in \Phi_n([a, b])$ из «соседних подпромежутков»

$$[t_{j-1}, t_j], [t_j, t_{j+1}] \quad (1 \leq j \leq N)$$

(записанных в *коши-тэйлоровской* форме, где у «левого» полинома начальной точкой служит правый конец подотрезка $[t_{j-1}, t_j]$, а у «правого» полинома такой точкой служит левый конец подотрезка $[t_j, t_{j+1}]$), так и соответствующих *производных этих полиномов*.

На *3-ем этапе* непосредственной проверкой можно убедиться в «однозначной представимости» сплайна $S(t)$ в форме линейной комбинации (2.12) с условиями (2.13)-(2.15). И этим завершается обоснование *первой части* Утверждения.

На *4-ом, заключительном этапе*, опять же, непосредственной проверкой, можно убедиться, что всякая линейная комбинация вида (2.12) на каждом подпромежутке $[t_j, t_{j+1}]$ ($j = 0, 1, \dots, N$) сводится к соответствующему обобщённому полиному из $\Phi_n([a, b])$, причём, благодаря «нужной гладкости» *срезок* $K_i^+(t; t_j)$, будет обеспечена «нужная состыкованность» этих полиномов (и их производных) на всём отрезке $[a, b]$. При этом требования (2.16) гарантируют, что хотя бы на одном подотрезке соответствующий обобщённый полином имеет в точности *n-ый* порядок.

На этом мы ограничимся в пояснениях справедливости высказанного утверждения.

Следствие. Система функций (2.17) служит *базисом* в пространстве $S_n^\nu(\Omega_N[a, b])$. Тем самым *точное значение размерности* пространства $S_n^\nu(\Omega_N[a, b])$ определяется выражением

$$\dim S_n^\nu(\Omega_N[a, b]) = (n+1) + N(n-\nu)^+. \quad (2.19)$$

Замечание 5. В классическом степенном случае формулы вида (2.12)-(2.15) широко известны. В русской литературе данная тематика подробно анализируется в работах Николая Павловича Корнейчука /1920-2003/, Сергея Борисовича Стечкина /1920-1995/, Юрия Николаевича Субботина /р. 1936/, Юрия Семёновича Завьялова /р. 1931/, Бориса Ильича

Квасова /р. 1947/, Валерия Леонидовича Мирошниченко /р. 1947/. А немецкий математик Günter Nürnberger /1948-2013/ опубликовал монографию с обзором этой тематики.

5. Для любого гладкого чебышёвского обобщённо-полиномиального **сплайна**

$$S(t) \text{ порядка } n \geq 0 \quad (n \geq 0 - \text{целое}),$$

характеризуемого *гладкостями*

$$\{v_j\}_1^N \quad (-1 \leq v_j < n - \text{целые})$$

по узлам $\{t_j\}_1^N \quad (N \geq 1 - \text{целое})$ разбиения

$\Omega_N[a, b]$, определим понятие его *суммарного порядка* $m(S)$, полагая

$$m(S) =: n + \sum_{j=1}^N (n - v_j)^+. \quad (2.20)$$

В свою очередь, под *суммарным порядком пространства сплайнов*

$S_n^v(\Omega_N[a, b])$ будем понимать величину

$$m = n + N(n - v)^+. \quad (2.21)$$

Ясно, что тогда

$$m(S) \leq m, \quad (2.22)$$

причём

$$\dim S_n^v(\Omega_N[a, b]) = m + 1. \quad (2.23)$$

6. Постановка задачи интерполирования сплайнами функции

$$x(t) \in C^{(n)}([a, b]) \quad (2.24)$$

заключается в следующем: для заданного набора узлов

$$\{\xi_0 < \dots < \xi_p\} \in [a, b] \quad (2.25)$$

кратностей

$$\{n \geq q_k \geq 0\}_{k=0}^p \quad (2.26)$$

нужно построить гладкий чебышёвский обобщённо-полиномиальный сплайн $S_m(t)$

в $(m + 1)$ -мерном пространстве

$$S_n^v(\Omega_N[a, b]) \quad (m = \sum_{k=0}^p (q_k + 1) - 1), \quad (2.27)$$

с реализацией равенств

$$S_m^{(l)}(\xi_k) = x^{(l)}(\xi_k) \quad (k = 0, 1, 2, \dots, p; \quad l = 0, 1, \dots, q_k). \quad (2.28)$$

Функцию $S_m(t) =: S_m(t; x)$ тогда мы будем называть *интерполяционным сплайном* для $x(t)$ в пространстве $S_n^v(\Omega_N[a, b])$, а разность

$$x(t) - S_m(t; x) =: R(t; x) \quad (2.29)$$

- *погрешностью* такого интерполирования.

Замечание 6. При постановке такой задачи некорректно однозначно задавать

информацию о функциях в точках, где допустим их разрыв. Поэтому если мы заранее знаем, что узел *интерполяции* ξ_k должен совпасть с узлом *сетки* t_j , где гладкость сплайна должна быть строго меньше n , то тогда надо ещё позаботиться о том, чтобы выполнялось дополнительное ограничение $q_k \leq v_j$.

Замечание 7. Однозначное

существование (да и вообще *существование*) такого интерполяционного (вообще говоря, «кратного») гладкого чебышёвского обобщённо-полиномиального сплайна уже **зависит от взаимного расположения узлов разбиения и узлов интерполяции.**

7. Для исследования поставленной задачи интерполирования мы введём ещё ряд *понятий*.

А. Прежде всего, нам потребуется понятие *счётчика узлов интерполяции на отрезке* $[t_i, t_j]$

$$\Gamma[t_i, t_j] \quad (0 \leq i \leq j \leq N + 1), \quad (2.30)$$

характеризующее *количество узлов интерполяции*, (с учётом их кратностей), *попадающих в промежуток* $[t_i, t_j]$.

Замечание 8. В частности, здесь имеет смысл и величина вида $\Gamma[t_j, t_j]$, причём, в силу постановки исходной интерполяционной задачи, такая величина не должна превосходить $(n + 1)$.

Б. Поскольку нам нужно будет учитывать возможные «зануления» сплайнов как в отдельных точках, так и на целых промежутках отрезка $[a, b]$, то введём ещё следующие *понятия*.

Точку $\mu \in (a, b)$ мы будем называть *внутренней изолированной нуль-точкой* для сплайна $S(t) \in S_n^v(\Omega_N[a, b])$, если $S(\mu) = 0$, но при любом (достаточно малом) $\delta > 0$, как в интервале $(\mu - \delta; \mu)$, так и в интервале $(\mu + \delta; \mu)$, сплайн $S(t)$ отличен от *тождественного нуля*. Определение *кратности* $v = v(\mu) \geq 1$ такой нуль-точки, в условиях возможных здесь «разрывов», требует определённой аккуратности. Именно:

- *если μ является точкой непрерывности* для $S(t), S^{(1)}(t), \dots, S^{(v)}(t), S^{(v+1)}(t)$, и при этом $S(\mu) = S^{(1)}(\mu) = \dots = S^{(v)}(\mu) = 0$, а $S^{(v+1)}(\mu) \neq 0$, то под кратностью данной нуль-точки понимается число $q = v + 1$;

- если же μ является точкой непрерывности лишь для $S(t), S^{(1)}(t), \dots, S^{(\nu)}(t)$, а $S^{(\nu+1)}(t)$ уже терпит в ней разрыв, и при этом

$$S(\mu) = S^{(1)}(\mu) = \dots = S^{(\nu)}(\mu) = 0,$$

то кратность данной нуль-точки полагается равной $q = \nu + 1$ лишь в нечётном случае, когда $S^{(\nu+1)}(\mu - 0) \times S^{(\nu+1)}(\mu + 0) > 0$ (то есть, когда производная $S^{(\nu+1)}(t)$ в точке μ имеет «скачок, учитывающий сохранение тенденции предыдущей производной $S^{(\nu)}(t)$ в данной точке либо к возрастанию, либо к убыванию»), а в противоположном, чётном случае (когда производная $S^{(\nu+1)}(t)$ в точке μ имеет «скачок, учитывающий смену тенденции предыдущей производной $S^{(\nu)}(t)$ в данной точке либо от убывания к возрастанию, либо от возрастания к убыванию»), кратность такой нуль-точки полагается равной $q + 1 = \nu + 2$.

Аналогичным образом вводятся определения кратностей для концевых нуль-точек отрезка $[a, b]$, только тогда "δ-соседство" должно пониматься в соответствующем «одностороннем смысле».

В. Поскольку аннулирование сплайна $S(t) \in S_n^\nu(\Omega_N[a, b])$ может происходить не только в отдельных точках, но и на целых промежутках вида $[t_i, t_j]$ ($0 < i < j < N + 1$), то необходимы, также, дополнительные определения в отношении изолированных нуль-промежутков.

Замечание 9. На произвольном промежутке вида

$$[\alpha, \beta] \subset [a, b] \quad (t_i < \alpha < t_{i+1} \leq t_{j-1} < \beta) \quad (2.31)$$

аннулирование сплайна $S(t) \in S_n^\nu(\Omega_N[a, b])$ невозможно, поскольку функция $S(t)$ должна быть «склеенной» их элементов пространства $\Phi_n([a, b])$, а никакой гладкий чебышёвский обобщённый полином не может быть «по одну сторону от внутренней точки отрезка тождественно "аннулировать"» и при этом быть нетривиальным по другую сторону от такой точки.

Итак, будем называть промежуток $[t_i, t_j] \subset [a, b]$ ($0 < i < j < N + 1$) внутренним изолированным нуль-промежутком для $S(t) \in S_n^\nu(\Omega_N[a, b])$, если $S(t) \equiv 0$ в $[t_i, t_j]$, но в $[t_{i-1}, t_i]$ и в $[t_j, t_{j+1}]$ функция

$S(t)$ отлична от тождественного нуля. О кратности такого нуль-промежутка имеет смысл говорить лишь таким образом, чтобы происходило обобщение понятия кратности внутренней изолированной нуль-точки, причём реальный интерес здесь имеет лишь ситуация «с соответствующими разрывами» на концах взятого промежутка.

Отсюда, полагая $\nu_i = \nu(t_i) \geq 1$, вводится следующее определение:

если

1) в точке t_i функции

$$S(t), S^{(1)}(t), \dots, S^{(\nu_i)}(t)$$

непрерывны, причём

$$S(t_i) = S^{(1)}(t_i) = \dots = S^{(\nu_i)}(t_i) = 0,$$

а $S^{(\nu_i+1)}(t)$ в точке t_i терпит разрыв ($\nu_i < n$),

2) в точке t_j функции

$$S(t), S^{(1)}(t), \dots, S^{(\nu_j)}(t)$$

непрерывны, причём

$$S(t_j) = S^{(1)}(t_j) = \dots = S^{(\nu_j)}(t_j) = 0,$$

а $S^{(\nu_j+1)}(t)$ в точке t_j терпит разрыв ($\nu_j < n$),

то

кратность данного нуль-промежутка $[t_i, t_j]$ полагается равной $q = \min\{\nu_i, \nu_j\} + 1$ лишь в «нечётном случае», когда

$$S^{(q)}(t_i - 0) \times S^{(q)}(t_j + 0) > 0,$$

и считается равной $q + 1 = \min\{\nu_i, \nu_j\} + 2$ в противоположном, «чётном случае».

Аналогичным образом вводятся определения кратностей для концевых нуль-промежутков отрезка $[a, b]$, только тогда «соседство» должно пониматься в соответствующем «одностороннем смысле».

Замечание 10. В своих истоках понятия изолированных нуль-точек и нуль-промежутков (коротко, изолированных нулей) и их кратностей, рассматриваемых по отношению к классическим (степенным) полиномиальным сплайнам, впервые формализовали, в 1960-ых годах, американские математики Isaac Schoenberg /1903-1990/ и его ученик Robert Shepard Johnson /1928-2008/.

Г. Для описания изолированных нулей сплайна $S(t) \in S_n^\nu(\Omega_N[a, b])$ вводятся ещё следующие понятия:

а) счётчик изолированных нулей для сплайна $S(t)$

$$Z(S; [t_i, t_j]) \quad (0 \leq i \leq j \leq N + 1), \quad (2.32)$$

характеризующий количество изолированных нулей сплайна $S(t)$ (с учётом их кратностей), находящихся в промежутке $[t_i, t_j]$, и, соответственно, числовая характеристика

$$Z(S) = Z(S; [t_0, t_{N+1}]) = Z(S; [a, b]), \quad (2.33)$$

характеризующая количество всех изолированных нулей (с учётом их кратностей) у сплайна $S(t)$ на отрезке $[a, b]$;

б) дефекты гладкости $\eta(S; t_j)$ для узлов сетки t_j сплайна $S(t)$, определяемые величинами

$$\eta(S; t_j) = n - \nu(S; t_j), \quad (2.34)$$

и дефекты гладкости $\eta(S; [t_i, t_j])$ для изолированных нуль-промежутков $[t_i, t_j]$ сплайна $S(t)$, определяемые соотношениями

$$\eta(S; [t_i, t_j]) = \begin{cases} (n - \nu_j)^+ + (j-1)(n - \nu)^+ \\ (\text{при } i = 0, j < N+1); \\ (n - \min\{\nu_i, \nu_j\})^+ + \\ (j-i-1)(n - \nu)^+ \\ (\text{при } i > 0, j < N+1); \\ (n - \nu_i)^+ + (N-i)(n - \nu)^+ \\ (\text{при } i > 0, j = N+1), \end{cases} \quad (2.35)$$

а также полный дефект гладкости $\eta(S) = \eta(S; [a, b])$ сплайна $S(t)$, определяемый суммой дефектов гладкости всех изолированных нуль-промежутков функции $S(t)$ на отрезке $[a, b]$.

8. На базе указанных понятий устанавливается такое утверждение:

Теорема 2. Для всякого нетривиального сплайна $S(t) \in S_n^\nu(\Omega_N[a, b])$ число его изолированных нулей на $[a, b]$ с учётом их кратностей $Z(S) := Z(S; [a, b])$ сложенное с полным дефектом гладкости рассматриваемого сплайна по его изолированным нуль-промежуткам $\eta(S) := \eta(S; [a, b])$ не может превышать суммарного порядка $m := n + N(n - \nu)^+$ пространства $S_n^\nu(\Omega_N[a, b])$, то есть имеет место неравенство

$$m \geq Z(S) + \eta(S). \quad (2.36)$$

Эскиз доказательства.

Поскольку при $\nu \geq n$ неравенство (2.36), очевидным образом, справедливо, то далее будем считать, что

$$\nu < n, \quad m = n + N(n - \nu). \quad (2.37)$$

В этой ситуации доказательство неравенства (2.36) можно провести с помощью «индукцией по N ». Поясним её «два шага».

Шаг 1. Пусть $N = 1$, так что $m = 2n - \nu$. Согласно Теореме 1, сплайн $S(t) \in S_n^\nu(\Omega_N[a, b])$ можно описать выражением

$$S(t) = \sum_{i=0}^n c_{i0} K_i(t; a) + \sum_{i=\nu+1}^n c_{i1} K_i^+(t; t_1). \quad (2.38)$$

Вводя два соответствующих обобщённых полинома $\varphi_1(t), \varphi_2(t) \in \Phi_n([a, b])$, где

$$\varphi_1(t) = \sum_{i=0}^n c_{i0} K_i(t; a), \quad (2.39)$$

$$\varphi_2(t) = \sum_{i=\nu+1}^n c_{i1} K_i(t; t_1),$$

закключаем что

$$S(t) = \begin{cases} \varphi_1(t) & \text{в } [t_0, t_1] \\ \varphi_1(t) + \varphi_2(t) & \text{в } [t_1, t_2] \end{cases}. \quad (2.40)$$

Отсюда нетривиальность исходного сплайна возможна лишь в следующих случаях:

а) полином $\varphi_1(t)$ тождественно равен 0,
а) полином $\varphi_2(t)$ не есть тождественный 0;

б) полином $\varphi_1(t)$ не есть тождественный 0,
б) полином $\varphi_2(t)$ тождественно равен 0;

полином $\varphi_1(t)$ не есть тождественный 0,

в') полином $\varphi_2(t)$ не есть тождественный 0,
и при этом полином $\varphi_1(t) + \varphi_2(t)$
не есть тождественный 0;

полином $\varphi_1(t)$ не есть тождественный 0,

в'') полином $\varphi_2(t)$ не есть тождественный 0,
но при этом полином $\varphi_1(t) + \varphi_2(t)$
есть тождественный 0.

(2.41)

И в каждом из этих случаев можно убедиться в справедливости неравенства (2.36).

Например, в случае а) отрезок $[t_0, t_1]$ для $S(t)$ является его «лево-концевым» изолированным нуль-промежутком. Отсюда

$$\eta(S;[a,b]) = \eta(S;[t_0,t_1]) = (n-\nu_1)^+ \leq n-\nu, \quad (2.42)$$

и, соответственно,

$$m - \eta(S;[a,b]) \geq [2n-\nu] - [n-\nu] = n. \quad (2.43)$$

Но при $\nu_1 = -1$ в рассматриваемой ситуации нуль-промежуток $[t_0, t_1]$ будет характеризоваться нуль-кратностью, а правая ветвь сплайна $S(t)$ может иметь не более n корней с учётом их кратностей, так что здесь справедлива оценка

$$Z(S;[a,b]) \leq n. \quad (2.44')$$

Аналогично для $0 \leq \nu_1 < n$, при учёте потери корней у правой ветви сплайна $S(t)$, в условиях «склейки» в точке t_1 с нуль-промежуток $[t_0, t_1]$ «вплоть до ν_1 -ой производной» и одновременном изменении кратности такого нуль-промежутка, находим оценку вида

$$Z(S;[a,b]) \leq (\nu_1 + 1) + [n - (\nu_1 + 1)] = n. \quad (2.44'')$$

Так как рассматриваемая ситуация в условиях $\nu_1 \geq n$ просто исключена, то, на основании (2.44')-(2.44'') и (2.43), имеем общую оценку

$$Z(S;[a,b]) \leq n \leq m - \eta(S;[a,b]), \quad (2.45)$$

верную при любом допустимом $\nu_1 \geq \nu$. Отсюда вытекает неравенство (2.36), так что в случае (2.41 - а) эта оценка проверена.

Аналогичным способом можно убедиться в справедливости неравенства (2.36) и в остальных случаях из (2.41).

Шаг 2. При индукционном переходе «от $N-1$ к N », когда

$$N > 1, m = n + N(n-\nu),$$

сплайн $S(t) \in S_n^\nu(\Omega_N[a,b])$ описывается уже выражением вида

$$S(t) = \sum_{i=0}^n c_{i0} K_i(t; a) + \sum_{j=1}^N \sum_{i=\nu+1}^n c_{ij} K_i^+(t; t_j). \quad (2.46)$$

Вводя две его «составляющие функции» $S_1(t)$, $S_2(t)$, где

$$S_1(t) = \sum_{i=0}^n c_{i0} K_i(t; a) + \sum_{j=1}^{N-1} \sum_{i=\nu+1}^n c_{ij} K_i^+(t; t_j),$$

$$S_2(t) = \sum_{i=\nu+1}^n c_{iN} K_i(t; t_1), \quad (2.47)$$

так что «первая составляющая» является сплайном из пространства

$S(t) \in S_n^\nu(\Omega_{N-1}[t_0, t_N])$, а «вторая составляющая» служит обобщённым полиномом из пространства $\Phi_n([a,b])$, получаем, что

$$S(t) = \begin{cases} S_1(t) & \text{в } [t_0, t_N] \\ S_1(t) + S_2(t) & \text{в } [t_N, t_{N+1}] \end{cases}. \quad (2.48)$$

Отсюда нетривиальность исходного сплайна $S(t)$ возможна лишь в следующих случаях:

А) сплайн $S_1(t)$ тождественно равен 0, полином $S_2(t)$ не есть тождественный 0;

Б) сплайн $S_1(t)$ не есть тождественный 0, полином $S_2(t)$ тождественно равен 0;

В*) сплайн $S_1(t)$ не есть тождественный 0, полином $S_2(t)$ не есть тождественный 0, и при этом сплайн $S_1(t) + S_2(t)$ не есть тождественный 0;

В**) сплайн $S_1(t)$ не есть тождественный 0, полином $S_2(t)$ не есть тождественный 0, но при этом сплайн $S_1(t) + S_2(t)$ есть тождественный 0.

(2.49)

И в каждом из этих случаев, рассуждая как и на 1-ом шаге, снова можно убедиться в справедливости неравенства (2.36).

Тем самым сформулированное утверждение будет обосновано.

Следствие. Предложенным доказательством, по- существу, обосновывается тот факт, что для всякого нетривиального гладкого чебышёвского обобщённо-полиномиального сплайна $S(t)$ порядка $n \geq 0$ ($n \geq 0$ – целое), характеризуемого гладкостями $\{\nu_j\}_1^N$ ($-1 \leq \nu_j < n$ – целые) в узлах $\{t_j\}_1^N$ разбиения $\Omega_N[a,b]$, справедливо неравенство

$$m(S) \geq Z(S) + \eta(S), \quad (2.50)$$

где $m(S) =: n + \sum_{j=1}^N (n-\nu_j)^+$ – суммарный порядок рассматриваемого сплайна, $Z(S)$ – число его изолированных нулей на $[a,b]$ с учётом их кратностей, а $\eta(S)$ – полный дефект гладкости сплайна по его изолированным нуль-промежуткам.

Отсюда, в частности, следует, что суммарный порядок всякого нетривиального гладкого чебышёвского обобщённо-

полиномиального сплайна всегда не меньше числа его изолированных нулей, подсчитываемых с учётом их кратностей.

Замечание 11. Сформулированный в данном **Следствии** вывод в виде неравенства (2.50) служит, в некотором смысле, обобщением (на *гладкие чебышёвские обобщённо-полиномиальные сплайны*) известного в теории *непрерывных чебышёвских обобщённых полиномов* критерия аннулируемости (сформулированного Марком Григорьевичем Крейном /1907-1989/), согласно которому для того чтобы на заданном наборе $(p + q)$ различных узлов отрезка $[a, b]$, состоящем из поднабора точек $\{\alpha_1, \alpha_2, \dots, \alpha_p\} \in (a, b)$ и, соответственно, поднабора точек $\{\beta_1, \beta_2, \dots, \beta_q\} \in [a, b]$, среди чебышёвских обобщённых полиномов n -го порядка существовал бы аннулирующий, для которого 1-ый поднабор служил бы совокупностью чётных корней (корень непрерывного чебышёвского обобщённого полинома называют чётным, если при переходе через него полином знака не меняет), а 2-ой поднабор – совокупностью нечётных корней (в свою очередь, корень непрерывного чебышёвского обобщённого полинома называют нечётным, если он либо совпадает с одним из концов отрезка $[a, b]$, либо находится строго внутри $[a, b]$, но при переходе через него полином изменяет свой знак), необходимо и достаточно выполнения неравенства $n \geq 2p + q$ (см., например, [А-9, стр. 40-49]).

9. Перейдём теперь к анализу (поставленной в пункте 6) задачи интерполирования функций гладкими чебышёвскими обобщённо-полиномиальными сплайнами. Здесь справедливо такое утверждение:

Теорема 3. Для задачи интерполирования функции

$$x(t) \in C^{(n)}([a, b]) \quad (2.51)$$

гладкими чебышёвскими обобщённо-полиномиальными сплайнами, когда на заданном наборе узлов

$$\{\xi_0 < \dots < \xi_p\} \in [a, b] \quad (2.52)$$

кратностей

$$\{n \geq q_k \geq 0\}_{k=0}^p \quad (2.53)$$

нужно построить в $(m + 1)$ -мерном пространстве

$$S_n^v(\Omega_N[a, b]) \quad (m = \sum_{k=0}^p (q_k + 1) - 1) \quad (2.54)$$

интерполяционный гладкий чебышёвский обобщённо-полиномиальный сплайн $S_m(t; x)$ с реализацией равенств

$$S_m^{(l)}(\xi_k; x) = x^{(l)}(\xi_k) \quad (k = 0, 1, 2, \dots, p; \quad l = 0, 1, \dots, q_k), \quad (2.55)$$

критерием её однозначной разрешимости (при любом выборе функции $x(t) \in C^{(n)}([a, b])$) служат равенства

$$\Gamma[t_i, t_j] \leq (n+1) + (j-i-1)^+(n-v)^+ \quad (0 \leq i \leq j \leq N+1), \quad (2.56)$$

где $\Gamma[t_i, t_j]$ – соответствующие счётчики узлов интерполирования (ведущие подсчёт их, напомним, с учётом их кратностей) на промежутках $[t_i, t_j]$.

Эскиз доказательства.

Необходимость условий (2.56) можно обосновать «от противного». Действительно, пусть задача однозначно разрешима для любой интерполируемой функции, и, в то же время, пусть некоторый промежуток $[t_i, t_j]$ содержит точки $\{\xi_{k_s}\}_{s=0}^r$ ($0 \leq r \leq p$) из набора (2.52) с кратностями $\{q_{k_s}\}_{s=0}^r$ ($0 \leq q_{k_s} \leq n$), для которых (вопреки требованиям (2.56)) имеет место неравенство

$$\sum_{s=0}^r (q_{k_s} + 1) \geq (n+2) - (j-i-1)^+(n-v)^+. \quad (2.57)$$

Тогда система линейных алгебраических уравнений для $S_m(t; x)$, принимающая вид

$$S_m^{(l)}(\xi_{k_s}; x) = x^{(l)}(\xi_{k_s}) \quad (s = 0, 1, 2, \dots, r; \quad l = 0, 1, \dots, q_{k_s}), \quad (2.58)$$

также должна быть однозначно разрешимой для всякой функции (2.51), то есть при любой своей правой части. Но в промежутке $[t_i, t_j]$, где по-существу и рассматриваются соотношения (2.58), сплайн $S_m(t; x)$, принадлежащий пространству $S_n^v(\Omega_N[a, b])$, на основании общего структурного представления сплайнов в этом пространстве и свойств участвующих в нём срезов, может быть выражен в виде суммы

$$S_m(t) = S_{m_1}(t) + S_{m_2}(t) \quad (v [t_i, t_j]) \quad (2.59)$$

где

$$S_{m_1}(t) = \sum_{\alpha=0}^n c_{\alpha,0} K_{\alpha}(t; a) + \sum_{\beta=1}^i \sum_{\alpha=v+1}^n c_{\alpha,\beta} K_{\alpha}(t; t_{\beta}), \quad (2.60)$$

$$S_{m_2}(t) = \sum_{\beta=i+1}^{j-1} \sum_{\alpha=v+1}^n c_{\alpha,\beta} K_{\alpha}^+(t; t_{\beta}).$$

Отсюда, учитывая, что $K_\alpha(t; t_\beta)$ являются некоторыми гладкими чебышёвскими обобщёнными полиномами порядка α ($\alpha \leq n$), заключаем, что $S_{m_1}(t)$ характеризуется «не более чем $(n+1)$ свободными параметрами». В свою очередь, $S_{m_2}(t)$ характеризуется «не более чем $(j-i-1)^+(n-\nu)^+$ свободными параметрами». Следовательно общее число «свободных параметров» в системе (2.58) будет «не более чем $[(n+1) + (j-i-1)^+(n-\nu)^+]$ », и мы приходим к ситуации, когда система *линейных алгебраических уравнений с одной стороны* должна быть *однозначно разрешимой* при любой своей правой части, а *с другой стороны* в ней *число уравнений больше числа её неизвестных*. Поскольку указанная ситуация *противоречива*, то тем самым *необходимость условий* (2.56) обоснована.

Достаточность условий (2.56) поясняется тем обстоятельством, что при их выполнении соответствующая система *линейных однородных алгебраических уравнений*

$$S_m^{(l)}(\xi_k; x) = 0 \quad (k = 0, 1, 2, \dots, p; \quad l = 0, 1, \dots, q_k) \quad (2.61)$$

допускает только *тривиальное решение*. Поясним справедливость этого высказывания «от противного».

Именно, предположим, что условия (2.56) выполнены, а, тем не менее, система (2.61) имеет *нетривиальное решение*, то есть существует *нетривиальный сплайн* $\tilde{S}(t) \in S_n^v(\Omega_N[a, b])$, обращающийся (вместе с соответствующими своими производными) «изолированным образом» в узлах интерполяции (2.52) в нуль. Покажем, что в каждой (из двух допустимых здесь) ситуации данное предположение *приводит к противоречию*.

Пусть, сначала, сплайн $\tilde{S}(t)$ *не имеет нуль-промежутков* на отрезке $[a, b]$. В этой (*первой*) ситуации, согласно (2.61), нетривиальный сплайн $\tilde{S}(t)$ должен иметь не менее $\sum_{k=0}^p (q_k + 1) = m + 1$ нуль-точек (с учётом их кратностей), причём, благодаря (2.53), у них $\{n \geq q_k \geq 0\}_{k=0}^p$. В то же время, согласно Теореме 2, число таких нуль-точек не должно превышать m . Следовательно в данной ситуации мы, действительно, приходим к противоречию.

Пусть, теперь, нетривиальный сплайн $\tilde{S}(t)$ *обладает (изолированными) нуль-промежутками*, и пусть (при упорядоченной

перенумерации «относительно соответствующих узлов разбиения $\Omega_N[a, b]$ ») это будут промежутки

$$[t_{i_k}, t_{j_k}] \quad (i = 1, 2, \dots, r) \quad (2.62)$$

(нетрудно понять, что для r - числа таких промежутков - должна быть справедлива оценка

$$r \leq \begin{cases} \frac{N+1}{2}, & \text{если } N - \text{нечётно} \\ \frac{N+2}{2}, & \text{если } N - \text{чётно} \end{cases},$$

а общая их длина, заведомо, должна быть меньше длины отрезка $[a, b]$). Такая (*вторая*) ситуация возможна лишь в случае $\nu < n$, причём значения *гладкостей* сплайна $\tilde{S}(t)$ в *концах* промежутков (2.62), совпадающих с *внутренними узлами* разбиения $\Omega_N[a, b]$, также должны быть *строго меньше* n . Поэтому в рассматриваемой ситуации вместо необходимых «срезок разности» мы имеем право писать «сами эти разности».

Поскольку, на основании (2.56), в *нуль-промежутки* (2.62) может попасть *не более, чем*

$$r(n+1) + (n-\nu) \sum_{k=1}^r (j_k - i_k - 1)$$

узлов интерполяции (с учётом их кратностей), то *вне этих промежутков* найдётся не менее чем

$$r_1 =: [m+1] - [r(n+1) + (n-\nu) \sum_{k=1}^r (j_k - i_k - 1)] \quad (2.63)$$

таких узлов, где, согласно (2.61), происходит «изолированное аннулирование» сплайна $\tilde{S}(t)$.

Среди нуль-промежутков (2.62) могут быть как «концевые» промежутки, так и «внутренние» промежутки. Поэтому введём обозначение

$$\nu[t_{i_k}; t_{j_k}] =: \begin{cases} \nu[t_{j_k}], & \text{если } i_k = 0, j_k < N+1; \\ \min\{\nu[t_{i_k}], \nu[t_{j_k}]\}, & \text{если } i_k > 0, j_k < N+1; \\ \nu[t_{i_k}], & \text{если } i_k > 0, j_k = N+1. \end{cases} \quad (2.64)$$

Тогда можно проверить, что «*полное число*» *искомых нуль-промежутков* у сплайна $\tilde{S}(t)$ *должно быть не менее чем*

$$r_2 =: \sum_{k=1}^r \{\nu[t_{i_k}; t_{j_k}] + 1\} = r + \sum_{k=1}^r \nu[t_{i_k}; t_{j_k}], \quad (2.65)$$

а в отношении соответствующего *полного дефекта* $\eta(\tilde{S};[a,b])$ будет иметь место равенство

$$\begin{aligned}\eta(\tilde{S};[a,b]) &= \\ &= \sum_{k=1}^r \{(n-v[t_{i_k};t_{j_k}]) + (j_k - i_k - 1)(n-v)\} = \\ &= rn + (n-v) \sum_{k=1}^r (j_k - i_k - 1) - \sum_{k=1}^r v[t_{i_k};t_{j_k}].\end{aligned}\quad (2.66)$$

Отсюда, с одной стороны, для числа *изолированных нулей* (с учётом их кратностей) у сплайна $\tilde{S}(t)$ на отрезке $[a,b]$, учитывая (2.63) и (2.65), должна выполняться оценка *снизу*

$$\begin{aligned}Z(\tilde{S};[a,b]) &\geq r_1 + r_2 = \\ &= \{[m+1] - [r(n+1) + (n-v) \sum_{k=1}^r (j_k - i_k - 1)]\} + \\ &+ \{r + \sum_{k=1}^r v[t_{i_k};t_{j_k}]\} = [m+1] + \sum_{k=1}^r v[t_{i_k};t_{j_k}] - \\ &- \{rn + (n-v) \sum_{k=1}^r (j_k - i_k - 1)\},\end{aligned}\quad (2.67')$$

а с другой стороны, согласно Теореме 2 при учёте (2.66), здесь должна быть справедливой оценка *сверху*

$$\begin{aligned}Z(\tilde{S};[a,b]) &\leq m - \eta(\tilde{S};[a,b]) = \\ &= m + \sum_{k=1}^r v[t_{i_k};t_{j_k}] - \\ &- \{rn + (n-v) \sum_{k=1}^r (j_k - i_k - 1)\}.\end{aligned}\quad (2.67'')$$

Противоречивость неравенств (2.67')-(2.67'') *исключает* возможность существования нетривиального сплайна $\tilde{S}(t)$ и во *второй ситуации*. Тем самым *достаточность* условий (2.56) также обоснована.

Следствие 1. Условия (2.56) можно заменить на *два неравенства* вида

$$\begin{aligned}\Gamma[t_0, t_j] &\leq (n+1) + (j-1)^+(n-v)^+ \\ (0 \leq j \leq N+1),\end{aligned}\quad (2.68')$$

$$\begin{aligned}\Gamma[t_i, t_{N+1}] &\leq (n+1) + (N-i)^+(n-v)^+ \\ (0 \leq i \leq N+1).\end{aligned}\quad (2.68'')$$

В самом деле, пусть, *сначала*, $i \leq j$. Обозначим за k_1 количество узлов интерполяции

(с учётом их кратностей), лежащих *левее* t_i , а за k_2 - количество узлов интерполяции (с учётом их кратностей), находящихся *правее* t_j . Тогда, согласно (2.67'), должна быть справедлива оценка

$$\begin{aligned}k_2 &\geq (m+1) - [(n+1) + (j-1)^+(n-v)^+] = \\ &= [n + N(n-v)^+ + 1] - \\ &- [(n+1) + (j-1)^+(n-v)^+] = \\ &= [N - (j-1)^+](n-v)^+, \end{aligned}\quad (2.69')$$

а, в силу (2.68''), соответственно, оценка

$$\begin{aligned}k_1 &\geq (m+1) - [(n+1) + (N-i)^+(n-v)^+] = \\ &= [n + N(n-v)^+ + 1] - \\ &- [(n+1) + (N-i)^+(n-v)^+] = \\ &= [N - (N-i)^+](n-v)^+.\end{aligned}\quad (2.69'')$$

Отсюда, для числа узлов интерполяции (с учётом их кратностей), *расположенных в самом промежутке* $[t_i, t_j]$, приходим к равенству

$$\begin{aligned}\Gamma[t_i, t_j] &= (m+1) - (k_1 + k_2) \leq \\ &\leq [n + N(n-v)^+ + 1] - \\ &- \{[N - (j-1)^+](n-v)^+ + \\ &+ [N - (N-i)^+](n-v)^+\} = \\ &= (n+1) + \\ &+ [(N-i)^+ + (j-1)^+ - N](n-v)^+.\end{aligned}\quad (2.70)$$

Положим, *далее*, что $i < j$. Тогда

$$N-i \geq N-(j-1) = (N+1) - j \geq 0$$

и, следовательно,

$$(N-i)^+ = (N-i). \quad (2.71')$$

Аналогично из $j-1 \geq i \geq 0$, тогда имеем

$$(j-1)^+ = (j-i). \quad (2.71'')$$

Поэтому в этой ситуации неравенство (2.70) обращается в оценку вида

$$\begin{aligned}\Gamma[t_i, t_j] &\leq (n+1) + \\ &+ [(N-i) + (j-i) - N](n-v)^+ = \\ &= (n+1) + (j-i-1)(n-v)^+.\end{aligned}\quad (2.72)$$

Положим, *теперь*, что $i = j$. Тогда справедливо

$$[(N-i)^+ + (j-1)^+ - N] =: \begin{cases} N + 0 - N = 0, \\ \text{если } j = 0; \\ (N-j) + (j-1) - N = -1, \\ \text{если } j = 1, 2, \dots, N; \\ 0 + N - N = 0, \\ \text{если } j = N+1. \end{cases} \quad (2.73)$$

Тем самым, при $i = j$ неравенство (2.70) удовлетворяет оценке

$$\Gamma[t_i, t_j] \leq (n+1), \quad (2.74)$$

являющейся *точной* в случаях $j=0$ и $j=n+1$, а также *заведомо выполнимой* в случае $1 \leq j \leq N$.

Сравнивая (2.72), (2.74) с (2.56), приходим к выводу, что при всех $i \leq j$ из неравенств (2.68')-(2.68'') вытекают неравенства (2.56).

А поскольку, в свою очередь, неравенства (2.56), расписанные по отношению к промежуткам вида $[t_0, t_j]$ ($0 \leq j \leq N+1$) и вида $[t_i, t_{N+1}]$ ($0 \leq i \leq N+1$), как раз приводят нас к неравенствам (2.68')-(2.68''), то, тем самым, эквивалентность условий (2.68')-(2.68'') условиям (2.56) обоснована.

Следствие 2. Если исходный набор узлов интерполяции (2.52) перенумеровать в виде естественного *кратного набора*

$$\{a \leq \zeta_0 \leq \zeta_1 \leq \dots \leq \zeta_m \leq b\}, \quad (2.75)$$

то данный критерий можно переформулировать в виде неравенств

$$\zeta_{i(n-\nu)^+-1} < t_i < \zeta_{n+(i-1)(n-\nu)^++1} \quad (i = 1, \dots, N). \quad (2.76)$$

В самом деле, на основании (2.68') *первой точкой* из набора (2.75), *расположенной правее* t_1 , может быть лишь точка ζ_{n+1} , *первой точкой* из данного набора, *расположенной правее* t_2 , может быть лишь точка $\zeta_{n+(n-\nu)^++1}$, и вообще, *первой точкой* из этого набора, *расположенной правее* t_i ($1 \leq i \leq N$), может быть лишь точка $\zeta_{n+(i-1)(n-\nu)^++1}$. Отсюда получаем неравенства

$$t_i < \zeta_{n+(i-1)(n-\nu)^++1} \quad (i = 1, \dots, N). \quad (2.77')$$

Аналогично, на основании (2.68'') *первой точкой* из набора (2.75), *расположенной левее* t_N , может быть лишь

точка $\zeta_{N(n-\nu)^+-1}$, *первой точкой* из данного набора, *расположенной левее* t_{N-1} , может быть лишь точка $\zeta_{(n-1)(n-\nu)^+-1}$, и вообще, *первой точкой* из этого набора, *расположенной левее* t_i ($N \geq i \geq 1$), может быть лишь точка $\zeta_{i(n-\nu)^+-1}$. Отсюда получаем неравенства

$$\zeta_{i(n-\nu)^+-1} < t_i \quad (i = 1, \dots, N). \quad (2.77'')$$

Объединяя неравенства (2.77')-(2.77'') в единое соотношение, мы и приходим к неравенствам (2.76).

Замечание 12. Если в рассматриваемой интерполяционной задаче (2.51)-(2.55) информация об интерполируемой функции задана «в неполном виде», когда, по отношению к узлам интерполяции, справедливо строгое неравенство

$$m > \sum_{k=0}^p (q_k + 1) - 1, \quad (2.78)$$

то в пространстве $S_n^{\vee}(\Omega_N[a, b])$ данным критерием можно гарантировать лишь *существование* соответствующего *интерполяционного сплайна*.

Не вдаваясь в подробности, поясним, что это так, поскольку «недостающую информацию» всегда можно «искусственно организовать» за счёт «нужного пополнения узлов интерполяции *недостающими точками соответствующего кратного интерполирования*». Конечно же, такое «пополнение» должно *производиться разумным образом, не нарушающим условия* (2.56), причём, даже при таком ограничении, оно не будет обладать *единственностью*. Но зато «неоднозначность» пополнения может быть использована для реализации каких-нибудь *дополнительных требований* к интерполяционному сплайну.

В заключение отметим, что в классическом *степенном* случае исследования (подобные выше рассмотренным) для соответствующих сплайнов в США проводили уже упоминавшийся Исаак Шёнберг со своей ученицей Anne Marie Whitney /1921-2008/, а в России - Василий Николаевич Малозёмов /р. 1939/ со своим учеником Александром Борисовичем Певным /р. 1947/.

В общей чебышёвской ситуации близкие исследования в отношении *существования* сплайнов, имеющих (*с заданными кратностями*) *нуль точки и нуль промежутки*, а также *условий однозначной разрешимости* соответствующей *задачи интерполирования*, можно найти в работах (также уже упоминавшегося) американского учёного Самуила Карлина и Валерия Сергеевича Романова (ученика В.Н.Малозёмова, 1957-го года рождения и ныне покойного, но год его смерти мне выяснить не удалось).

On interpolation of functions by generalized-polynomial splines, constructed on the basis of smooth Tchebycheff systems

V.B.Demidovich

Abstract: It was considered the interpolation problem for functions by means of generalized-polynomial splines constructed on the basis of smooth Tchebycheff systems. It was proposed the structural formulas for the respective interpolation splines and were formulated the conditions of their unique existence .

Keywords: smooth Tchebycheff systems, generalized polynomials, generalized-polynomial splines, interpolation.

Литература

А. Книги:

- [А-1] С.Карлин, В.Стадден «Чебышёвские системы и их применение в анализе и статистике». Москва, "Наука", 1976.
- [А-2] М.Г.Крейн, А.А.Нудельман «Проблемы моментов Маркова и экстремальные задачи». Москва, "Наука", 1973.
- [А-3] Н.П.Корнейчук «Сплаины в теории приближения». Москва, "Наука", 1984.
- [А-4] С.Б.Стечкин, Ю.Н.Субботин «Сплаины в вычислительной математике». Москва, "Наука", 1976.
- [А-5] Ю.С.Завьялов, Б.И.Квасов, В.Л.Мирошниченко «Методы сплайн-функций». Москва, "Наука", 1980.
- [А-6] В.Н.Малозёмов, А.Б.Певный «Полиномиальные сплайны». Ленинград, "Изд-во ЛГУ", 1986.
- [А-7] G.Nürnbergger «Approximation by Spline Functions». Berlin, "Springer", 1989.
- [А-8] В.М.Тихомиров «Некоторые вопросы теории приближений». Москва, "Изд -во Моск. ун -та", 1976.
- [А-9] В.Б.Демидович «Приближённые вычисления с помощью обобщённых полиномов из чебышёвских пространств: чебышёвские обобщённые полиномы». Москва, "Изд -во Моск. ун -та", 1990.
- [А-10] В.Б.Демидович «Приближённые вычисления с помощью обобщённых полиномов из чебышёвских пространств: простое интерполирование, кратное интерполирование, формулы тейлоровского типа». Москва, "Изд -во Моск. ун -та", 1994.

Б. Статьи:

- [Б-1] I.Schoenberg, A.Whitney «On Polya frequency functions: the positivity of translation determinants with an application to the interpolation problem by spline curves». Trans. Am. Math. Soc., v. 74 (1953), № 2, 246-259.
- [Б-2] В.С.Романов «Кратная интерполяция чебышёвскими сплайнами». Методы вычислений (Ленинград), т. 13 (1983), 186-202.
- [Б-3] А.А.Женсыкбаев «Моносплаины минимальной нормы и наилучшие квадратурные формулы». Успехи математических наук, т. 36 (1981), № 4, 107-159.
- [Б-4] В.Б.Демидович, А.С.Кочуров «О некоторых экстремальных задачах в конечно-мерных чебышёвских пространствах». Труды НИИСИ РАН, т. 4 (2014), № 2, 132-141.
- [Б-5] В.Б.Демидович «Об интерполяции функций обобщёнными полиномами, построенными на базе гладких чебышёвских систем». Труды НИИСИ РАН, т. 6 (2016), № 1, 102-114.

Быстрое вычисление значений многочлена на «мультимножестве» точек

О.Ю. Сударева

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»,
Москва, Россия, E-mail: sudareva@niisi.msk.ru

Аннотация. В статье рассматривается вычисление значений многочлена на заданном «мультимножестве» точек, т.е., на наборе точек с натуральными кратностями. Назовём значением многочлена в точке с кратностью набор, состоящий из значений в этой точке самого многочлена и всех его производных порядков меньше кратности точки. Пусть n — число геометрически различных точек, N — сумма всех кратностей. Для многочлена степени не более $N-1$ и мультимножества, состоящего из единственной точки либо только из точек кратности 1, известны алгоритмы вычисления, которые требуют $O(N \log_2(N+1) \log_2(n+1))$ операций. Под операциями понимаются четыре арифметических операции над комплексными числами и операции сравнения натуральных чисел. В статье предлагается алгоритм вычисления в общем случае, когда на количество и кратности точек не накладывается никаких ограничений. Доказывается, что для этого алгоритма справедлива та же по порядку оценка количества операций, что и в частных случаях, и даётся формула для оценки количества операций с точно подсчитанной мультипликативной константой. Автором была разработана вычислительная процедура для микропроцессора КОМДИВ128-РНО, которая реализует описанный алгоритм. Результаты тестирования этой процедуры приведены в статье.

Ключевые слова: алгоритм, многочлен, сложность

1. Постановка задачи

Рассматривается следующая общая задача. Имеется $n \in \mathbb{N}$ произвольных точек из $\mathbb{C}$: $x_0, x_1, \dots, x_{n-1}$. Каждой точке приписано некоторое натуральное число — кратность точки: $m_0, m_1, \dots, m_{n-1}$. Положим

$$(1) \quad N = \sum_{j=0}^{n-1} m_j.$$

Пусть дан произвольный многочлен $P(x) \in \mathbb{C}[x]$ степени $N-1$. Задача: для всех $j = \overline{0, n-1}$ вычислить значение многочлена P и всех его производных до порядка $m_j - 1$ включительно в точке x_j .

Требуется построить алгоритм, который решал бы поставленную задачу за $O(N \log(N+1) \log(n+1))$ арифметических операций над $\mathbb{C}$ (здесь и далее $\log z$ означает $\log_2 z$) и операций сравнения над $\mathbb{N}$. Для этого алгоритма требуется также получить конструктивную оценку вида: количество операций не превышает $T(n, N) = \text{const} \cdot N \log(N+1) \log(n+1)$, где константа явно вычислена. К аргументам логарифмов в формуле добавлена единица, для того чтобы формула была справедлива и для случаев $N = 1$ и $n = 1$.

Известен ряд частных случаев, в которых данная оценка справедлива. Например, если $N = n$, т.е., имеется n точек кратности 1 и вычисляются значения многочлена в этих точках, количество операций составляет по порядку $O(n \log^2(n+1))$ (доказано в [1]).

Если же $n = 1$, т.е., имеется одна точка кратности N , задача представляет собой вычисление производных

многочлена в одной точке, которое может быть выполнено за $O(N \log(N+1))$ операций, как показано в [2].

Для случаев, когда точки располагаются специальным образом, существуют алгоритмы, требующие меньшее по порядку количество операций. Например, если необходимо вычислить значения многочлена в корнях степени n из 1, где $n = 2^q$, $q \in \mathbb{N}$, это можно сделать, применив алгоритм БПФ к вектору коэффициентов, с использованием $O(n \log n)$ операций. Столько же по порядку операций достаточно, если точки образуют геометрическую прогрессию (см. [3]).

Оценок для числа операций снизу на настоящий момент нам не известно.

Можно поставить и двойственную в каком-то смысле задачу: для мультимножества с суммой кратностей N найти многочлен P степени $N-1$, который принимает на мультимножестве заданное значение.

Докажем, что такой многочлен существует и единственен.

Рассмотрим пространство X многочленов из $\mathbb{C}[x]$ степени не более $N-1$. На этом пространстве рассмотрим функционалы

$$(2) \quad \delta_{x_j}^l : P(x) \mapsto \left(\frac{\partial}{\partial x} \right)^l (P)(x_j), \quad l = \overline{0, m_j - 1}.$$

Мы покажем, что эти функционалы линейно независимы, а значит, образуют базис X^* . Тогда значение любой из искоемых производных нельзя выразить через значения других, и наоборот: по заданным значениям производных можно однозначно восстановить многочлен $P(x)$.

Для доказательства построим такой набор многочленов $\{P_{x_j}^l\}$, где $j = \overline{0, n-1}$, $l = \overline{0, m_j-1}$, что матрица, составленная из значений функционалов на этих многочленах, верхняя треугольная, и её определитель отличен от 0. Пусть в матрице различные столбцы соответствуют различным функционалам, в следующем порядке: $\delta_{x_{n-1}}^0, \delta_{x_{n-1}}^1, \dots, \delta_{x_{n-1}}^{m_{n-1}-1}, \delta_{x_{n-2}}^0, \dots, \delta_{x_{n-2}}^{m_{n-2}-1}, \dots, \delta_{x_0}^0, \dots, \delta_{x_0}^{m_0-1}$; строкам соответствуют различные многочлены в том же порядке. Тогда условие на матрицу можно переформулировать:

- при всех допустимых p : $\delta_{x_j}^{m_j-p}(P_{x_j}^{m_j-p}) \neq 0$,
- при $l > j$ и всех допустимых q и p :
 $\delta_{x_l}^{m_l-q}(P_{x_j}^{m_j-p}) = 0$,
- при любом j и $p < r \leq m_j$: $\delta_{x_j}^{m_j-r}(P_{x_j}^{m_j-p}) = 0$.

Рассмотрим многочлены

$$(3) \quad \begin{aligned} P_{x_j}^{m_j-p} &= (x - x_0)^{m_0} \dots (x - x_{j-1})^{m_{j-1}} \cdot \\ &\cdot (x - x_j)^{m_j-p} (x - x_{j+1})^{m_{j+1}} \dots (x - x_{n-1})^{m_{n-1}} = \\ &= (x - x_j)^{m_j-p} \prod_{i=0, i \neq j}^{n-1} (x - x_i)^{m_i}. \end{aligned}$$

и покажем, что они являются искомыми.

Действительно,

$$(4) \quad \delta_{x_j}^{m_j-p}(P_{x_j}^{m_j-p}) = \left(P_{x_j}^{m_j-p}\right)^{(m_j-p)}(x_j) \neq 0,$$

т.к. все слагаемые, полученные при дифференцировании многочлена $m_j - p$ раз, кроме одного, будут содержать сомножитель $(x - x_j)$ (и поэтому обнулятся в точке x_j), а все точки $\{x_j\}$ различны (поэтому оставшееся слагаемое в точке x_j отлично от нуля).

При $1 \leq p < r$

$$(5) \quad \delta_{x_j}^{m_j-r}(P_{x_j}^{m_j-p}) = \left(P_{x_j}^{m_j-p}\right)^{(m_j-r)}(x_j) = 0,$$

т.к. $m_j - r < m_j - p$, а значит, во всех слагаемых будет сомножитель $(x - x_j)$, и они обнулятся.

При $l > j$

$$(6) \quad \delta_{x_l}^{m_l-q}(P_{x_j}^{m_j-p}) = \left(P_{x_j}^{m_j-p}\right)^{(m_l-q)}(x_l) = 0,$$

т.к. $m_l - q < m_l$, и во всех слагаемых будет сомножитель $(x - x_l)$.

Построенные многочлены удовлетворяют всем условиям. Теорема доказана. $\square$

2. Используемые факты

Алгоритм, который будет построен в данной работе, опирается на ряд известных более простых алгоритмов. Ниже приводится краткое описание этих алгоритмов, а также формулы для оценки требуемого числа операций.

2.1. Вспомогательная оценка

Для функции $\varphi(n) = An \log n + Bn$, где $A \in \mathbb{N}$, $B \in \mathbb{N} \cup \{0\}$, справедливо соотношение: $\forall n_1, n_2 \in \mathbb{N} \quad \varphi(n_1) + \varphi(n_2) \leq \varphi(n_1 + n_2)$.

Эта оценка далее будет использоваться в работе для произвольного количества слагаемых.

2.2. Умножение многочленов

Как показано в [4] и [5], умножение двух многочленов степени $n-1$, где $n = 2^q$, сводится к вычислению свёртки векторов их коэффициентов. Свёртка, в свою очередь, может быть вычислена при помощи быстрого преобразования Фурье (БПФ): вектора дополняются нулями до длины $2n$, после чего выполняется БПФ каждого из них, поэлементное умножение результатов и обратное БПФ полученного вектора; результат поэлементно домножается на $1/2n$.

БПФ вычисляется по классическому алгоритму Кули-Тьюки ([6]). Однако, поскольку в данном случае половина элементов дополненных до длины $2n$ векторов коэффициентов — нули, на первом проходе алгоритма все операции «бабочка Фурье» фактически не требуют арифметических операций. Таким образом, можно сэкономить по $3n$ операций для каждого вектора.

Общее количество операций, необходимое для вычисления свёртки, составляет

$$(7) \quad 3 \cdot \frac{3}{2} \cdot 2n \log 2n - 2 \cdot 3n + 2n + 2n = 9n \log n + 7n.$$

Обозначим эту величину за $\tilde{M}(n)$: она будет использоваться в дальнейших выкладках. Также отметим, что при возведении многочлена в квадрат достаточно выполнить только одно БПФ исходного вектора, поэтому число операций в этом случае можно оценить как

$$(8) \quad S(n) = 2 \cdot \frac{3}{2} \cdot 2n \log 2n - 3n + 2n + 2n = 6n \log n + 7n.$$

Оценим теперь количество операций, необходимое для умножения двух многочленов произвольной натуральной степени n . Пусть $2^{q-1} \leq n < 2^q$, тогда многочлены имеют по $n+1$ коэффициенту, а их произведение — $2n+1$ коэффициент, причём $2^q < 2n+1 \leq 2^{q+1}$.

Вектора коэффициентов дополняются нулями до длины 2^{q+1} , их свёртка вычисляется описанным выше способом. На каждое БПФ требуется

$$(9) \quad \frac{3}{2}(q+1)2^{q+1}$$

операций, при этом для каждого БПФ исходных векторов можно сэкономить по $3 \cdot 2^q$ операций; на перемножение и домножение на $1/2^{q+1}$ необходимо по 2^{q+1} операций. Поскольку $2^{q-1} \leq n$, справедлива оценка $q \leq 1 + \log n$. Таким образом, общее количество операций на умножение многочленов — не более, чем

$$(10) \quad \begin{aligned} & 3 \cdot \frac{3}{2}(q+1)2^{q+1} - 2 \cdot 3 \cdot 2^q + 2 \cdot 2^{q+1} = \\ & = \frac{9}{2}q \cdot 2^{q+1} + 7 \cdot 2^q \leq \frac{9}{2}(\log n + 1) \cdot 4n + 7 \cdot 2n = \\ & = 18n \log n + 32n = M(n). \end{aligned}$$

Все три величины: $\tilde{M}(n)$, $S(n)$ и $M(n)$, — по-рядку составляют $O(n \log n)$.

2.3. Деление многочленов

2.3.1. Вычисление обратного многочлена.

Обратным к многочлену $p(x) = \sum_{i=0}^{n-1} a_i x^i$ степени ровно $n-1$ называется многочлен $q(x)$, такой, что

$$(11) \quad \begin{aligned} x^{2n-2} &= p(x)q(x) + r(x), \\ \deg q(x) &= n-1, \deg r(x) < n-1. \end{aligned}$$

Мы будем использовать приведённый в [4] алгоритм вычисления многочлена, обратного к $p(x)$, для $n = 2^q$.

1. Если $n = 1$, то $q(x) = 1/a_0$.
2. Если $n > 1$, рекурсивно вычисляется многочлен $s(x)$, обратный к многочлену, составленному из $n/2$ старших членов $p(x)$, делённых на $x^{n/2}$: $\sum_{i=n/2}^{n-1} a_i x^{i-n/2}$. Далее строится

$$(12) \quad r(x) = 2s(x)x^{3/2 \cdot n - 2} - (s(x))^2 p(x).$$

Этот многочлен можно переписать в виде $r(x) = r_1(x)x^{n-2} + r_2(x)$, где $\deg r_2(x) < n-2$.

Тогда $q(x) = r_1(x)$ — искомый обратный многочлен.

Обозначим количество необходимых на эти вычисления операций за $\tilde{R}(n)$ и оценим его. На нахождение $s(x)$ требуется $\tilde{R}(n/2)$ операций, $S(n/2)$ на возведение $s(x)$ в квадрат, $\tilde{M}(n)$ на умножение на $p(x)$. Умножение на степень x операций не требует. При вычитании достаточно найти только коэффициенты $r_1(x)$ — на это требуется $n-1$ операций. Итого получаем:

$$(13) \quad \begin{aligned} \tilde{R}(n) &= \tilde{R}(n/2) + S(n/2) + \tilde{M}(n) + n - 1 = \\ &= \tilde{R}(2^{q-1}) + S(2^{q-1}) + \tilde{M}(2^q) + 2^q - 1 = \\ &= \tilde{R}(2^{q-2}) + S(2^{q-2}) + \tilde{M}(2^{q-1}) + S(2^{q-1}) + \\ &\quad + \tilde{M}(2^q) + 2^q - 1 + 2^{q-1} - 1 = \dots = \\ &= \tilde{R}(2^0) + \sum_{j=0}^{q-1} S(2^j) + \sum_{j=1}^q \tilde{M}(2^j) + \sum_{j=1}^q 2^j - q \leq \\ &\leq 1 + S\left(\sum_{j=0}^{q-1} 2^j\right) + \tilde{M}\left(\sum_{j=1}^q 2^j\right) + \sum_{j=1}^q 2^j = \\ &= 1 + S(2^q - 1) + \tilde{M}(2(2^q - 1)) + 2(2^q - 1) \leq \\ &\leq S(n) + \tilde{M}(2n) + 2n = \\ &= 6n \log n + 7n + 9 \cdot 2n(\log n + 1) + 7 \cdot 2n + 2n = \\ &= 24n \log n + 41n \end{aligned}$$

(здесь в первом неравенстве использована оценка из п. 2.1).

Пусть теперь n произвольное, тогда существует такое натуральное q , что $2^{q-1} \leq n < 2^q$, то есть, $n = 2^q - 1 - m$, где $0 \leq m < 2^{q-1}$. Многочлен, обратный к $p(x)$ степени n , можно найти следующим образом. Положим $\tilde{p}(x) = x^m p(x)$ и построим $\tilde{q}(x)$, обратный к нему, по описанному выше алгоритму. Степень многочлена $\tilde{p}$ и многочлена $\tilde{q}$ — $n + m$. Можем записать:

$$(14) \quad \begin{aligned} x^{2 \cdot 2^q - 2} &= \tilde{p}(x)\tilde{q}(x) + \tilde{r}(x) \\ (\deg \tilde{r}(x) &< n + m), \end{aligned}$$

$$(15) \quad x^{2(n+m)} = x^m p(x)\tilde{q}(x) + \tilde{r}(x),$$

$$(16) \quad x^{2n} = p(x) \frac{\tilde{q}(x)}{x^m} + \frac{\tilde{r}(x)}{x^{2m}} \quad (\text{всюду, кроме } 0),$$

$$(17) \quad \frac{\tilde{q}(x)}{x^m} = q(x) + \sum_{j=1}^m \frac{\tilde{q}_{m-j}}{x^j} = q(x) + f(x),$$

где $q(x)$ — многочлен степени $n-1$. Поскольку в левой части равенства (16) стоит многочлен, слагаемые в правой части с отрицательными степенями x взаимно уничтожаются:

$$(18) \quad p(x)f(x) + \frac{\tilde{r}(x)}{x^{2m}} = r(x),$$

где $\deg r(x) < n$. Таким образом, справедливо соотношение

$$(19) \quad x^{2n} = p(x)q(x) + r(x),$$

откуда $q(x)$ — искомый многочлен, обратный к $p(x)$.

На определение коэффициентов q по коэффициентам $\tilde{q}$ операций не требуется, поэтому общее количество операций на нахождение обратного многочлена не превышает

$$(20) \quad \begin{aligned} \tilde{R}(2^n) &\leq \tilde{R}(2n) = 24 \cdot 2n(\log n + 1) + 41 \cdot 2n = \\ &= 48n \log n + 130n = R(n). \end{aligned}$$

Обе величины — $\tilde{R}(n)$ и $R(n)$ — составляют по порядку $O(n \log n)$.

2.3.2. Деление многочлена степени не более $2n$ на многочлен степени n .

Пусть $\deg s(x) \leq 2n$, $\deg p(x) = n$, и пусть по алгоритму из п. 2.3.1 найден многочлен $q(x)$, обратный к $p(x)$. Тогда справедливо соотношение (19).

Разделив многочлены $q(x)$ и $r(x)$ на x^{2n} , получим функции, определённые всюду, кроме 0:

$$(21) \quad \begin{aligned} \hat{q}(x) &= \frac{q_0}{x^{2n}} + \frac{q_1}{x^{2n-1}} + \dots + \frac{q_{n-1}}{x^{n+1}} + \frac{q_n}{x^n}, \\ \hat{r}(x) &= \frac{r_0}{x^{2n}} + \frac{r_1}{x^{2n-1}} + \dots + \frac{r_{n-1}}{x^{n+1}}. \end{aligned}$$

Разделив обе части (19) на x^{2n} , получаем:

$$(22) \quad 1 = p(x)\hat{q}(x) + \hat{r}(x),$$

$$(23) \quad s(x) = p(x)\hat{q}(x)s(x) + \hat{r}(x)s(x).$$

Старшая степень x , содержащаяся в $\hat{r}(x)s(x)$, не превышает $2n - n - 1 = n - 1$, значит, $\hat{r}(x)s(x)$ представляет собой сумму некоторого многочлена степени меньше n и, возможно, ряда слагаемых с x в отрицательной степени. Далее,

$$(24) \quad \begin{aligned} \hat{q}(x)s(x) &= (q_0 + q_1x + \dots + q_nx^n)s_{2n} + \\ &+ \frac{q_0s_{2n-1}}{x} + (q_1 + q_2x + \dots + q_nx^{n-1})s_{2n-1} + \\ &+ \frac{q_0s_{2n-2}}{x^2} + \frac{q_1s_{2n-2}}{x} + \\ &+ (q_2 + q_3x + \dots + q_nx^{n-2})s_{2n-2} + \dots + q_ns_n + \dots = \\ &= q(x)s_{2n} + q^1(x)s_{2n-1} + \dots + q^n(x)s_n + \sum_{j=1}^{2n} \frac{\alpha_j}{x^j} = \\ &= \tilde{q}(x) + \sum_{j=1}^{2n} \frac{\alpha_j}{x^j}. \end{aligned}$$

В левой части равенства (23) стоит многочлен, а значит, при подстановке (24) в правой части члены с x в отрицательных степенях взаимно уничтожаются. Поскольку $\deg p(x) = n$, в произведение слагаемых из (24) с x в отрицательных степенях и $p(x)$ переменная x входит в степенях меньше n . Таким образом, после подстановки получаем:

$$(25) \quad s(x) = p(x)\tilde{q}(x) + \tilde{r}(x), \quad \deg \tilde{q} \leq n, \deg \tilde{r} < n,$$

откуда $\tilde{q}(x)$ — частное от деления $s(x)$ на $p(x)$ с остатком. Чтобы найти его коэффициенты, достаточно вычислить

$$(26) \quad q(x)s(x) = x^{2n}\hat{q}(x)s(x) = x^{2n}\tilde{q}(x) + t(x),$$

где $\deg t(x) < 2n$. Тогда коэффициенты $\tilde{q}(x)$ — это коэффициенты в $q(x)s(x)$ при x в степенях $2n$ и выше. Вычислив $\tilde{q}(x)$, из равенства (25) находим и $\tilde{r}(x)$ — искомый остаток.

Оценим количество операций на эти вычисления. На нахождение $q(x)$ требуется, по доказанному, не более $R(n)$ операций. Для вычисления произведения $q(x)s(x)$ достаточно $M(2n)$ операций, и ещё $M(n) + n$ операций на вычисление $s - pq$. Всего требуется не более, чем

$$(27) \quad \begin{aligned} &48n \log n + 130n + 18 \cdot 2n \log 2n + 32 \cdot 2n + \\ &+ 18n \log n + 32n + n = 102n \log n + 263n = D(n) \end{aligned}$$

арифметических операций, что составляет по порядку $O(n \log n)$.

2.3.3. Деление многочленов произвольных степеней.

Пусть теперь $\deg s(x) \leq n$, $\deg p(x) = m$ (m — произвольное натуральное число, меньше либо равное n). Пусть $k \in \mathbb{N}$ таково, что $2^{k-1}m \leq n < 2^k m$.

Вычислим последовательно:

1) p^2 — на это потребуется не более, чем $M(m)$ операций;

2) p^4 — $M(2m)$ операций;

...

k-1) $p^{2^{k-1}}$ — $M(2^{k-2}m)$ операций.

Затем последовательно, в обратном порядке, по алгоритму, приведённому в 2.3.2, разделим с остатком $s(x)$ на полученные многочлены:

k) $s \bmod p^{2^{k-1}} = r_{k-1}$ — на это потребуется не более, чем $D(2^{k-1}m)$ операций;

k-1) $r_{k-1} \bmod p^{2^{k-2}} = r_{k-2}$ — $D(2^{k-2}m)$ операций;

...

1) $r_1 \bmod p = r$ — $D(m)$ операций.

Для многочленов справедливо следующее утверждение: если $a \bmod b_1 b_2 = c$ и $c \bmod b_1 = d$, то $a \bmod b_1 = d$. Отсюда следует, что найденный многочлен $r(x)$ является искомым остатком от деления $s(x)$ на $p(x)$.

Количество арифметических операций на описанные вычисления составляет не более, чем

$$\begin{aligned}
& \sum_{j=1}^{k-1} M(2^{j-1}m) + \sum_{j=1}^k D(2^{j-1}m) \leq \\
& \leq M\left(m \sum_{j=0}^{k-2} 2^j\right) + D\left(m \sum_{j=0}^{k-1} 2^j\right) \leq \\
(28) \quad & \leq M(n) + D(2n) = 18n \log n + 32n + \\
& + 102 \cdot 2n \log 2n + 263 \cdot 2n = \\
& = 222n \log n + 762n = \tilde{D}(n)
\end{aligned}$$

— также $O(n \log n)$ по порядку.

2.4. Вычисление производных в одной точке

Как показано в [2], нахождение производных многочлена степени $n-1$ до порядка $n-1$ включительно в заданной точке сводится, как и умножение многочленов степени $n-1$, к вычислению свёртки двух векторов длины n . На вычисление элементов этих векторов требуется не более $4n$ арифметических операций. Алгоритм вычисления свёртки описан в п. 2.2. Общее количество арифметических операций не превышает:

$$\begin{aligned}
(29) \quad & 4n + M(n-1) \leq 4n + M(n) = \\
& = 18n \log n + 36n = C(n).
\end{aligned}$$

Таким образом, все описанные вспомогательные алгоритмы требуют $O(n \log n)$ арифметических операций. Более того, все полученные оценки имеют вид $\phi(n)$ (см. п. 2.1).

3. Алгоритм вычисления

В алгоритме, который будет построен, развивается идея вычисления значений при помощи дерева, которая предложена в [1] для частного случая. Эта идея используется также для быстрых вычислений в [4] и [5].

Алгоритм подразделяется на три этапа: построение дерева произведений для заданного мультимножества, построение соответствующего ему дерева остатков для многочлена и, наконец, вычисление значений в каждой точке.

3.1. Дерево произведений

Определим дерево произведений в общем случае.

Для заданного набора многочленов $q_0, q_1, \dots, q_{k-1} \in \mathbb{C}[x]$ деревом произведений называется пара (T, f) , где T — бинарное дерево (см., напр., [7], разд. 2.3), f — отображение множества вершин этого дерева в пространство многочленов $\mathbb{C}[x]$, и выполненные следующие условия:

- все вершины T , кроме концевых, имеют ровно по 2 сына;
- T имеет k концевых вершин, причём f задаёт взаимно-однозначное соответствие между

множеством концевых вершин T и множеством многочленов $\{q_j\}_{j=0, k-1}$;

- для любых трёх вершин α, β_1, β_2 , где β_1 и β_2 — сыновья α , выполнено соотношение $f(\alpha) = f(\beta_1)f(\beta_2)$.

В частности, f отображает корень дерева T в

многочлен $\prod_{j=0}^{k-1} q_j$. Поскольку выбор бинарного дерева

T , а также соответствия между многочленами и концевыми вершинами, не однозначен, для заданного набора многочленов дерево произведений может быть не единственным. Кроме того, уровни различных концевых вершин дерева могут не совпадать.

Дерево произведений может быть построено индуктивно, от концевых вершин к корню, следующим образом. В начале полагаем $q_j^0 = q_j$. На шаге s , $1 \leq s \leq k$, имеем набор многочленов:

$$(30) \quad q_0^{s-1}, q_1^{s-1}, \dots, q_{k-s}^{s-1}.$$

Из этого набора выбирается любая пара многочленов: q_l^{s-1}, q_m^{s-1} ; она удаляется из набора, вместо неё добавляется их произведение. Получается набор

$$(31) \quad q_0^s, q_1^s, \dots, q_{k-s-1}^s,$$

в котором многочлены переобозначены в произвольном порядке, причём для некоторого $p \in \{0, 1, \dots, k-s-1\}$ выполнено: $q_p^s = q_l^{s-1} q_m^{s-1}$. Соответственно, в дерево T будет входить вершина α , сыновьями которой являются вершины β_1 и β_2 , такие, что $f(\beta_1) = q_l^{s-1}$ и $f(\beta_2) = q_m^{s-1}$. При этом полагаем $f(\alpha) = q_p^s$. В результате, после k шагов, получим последовательность наборов многочленов и соответствующее ей дерево произведений.

Рассмотрим пример. Для 5 многочленов $q_0, \dots, q_4$ один из возможных вариантов дерева произведений выглядит следующим образом:

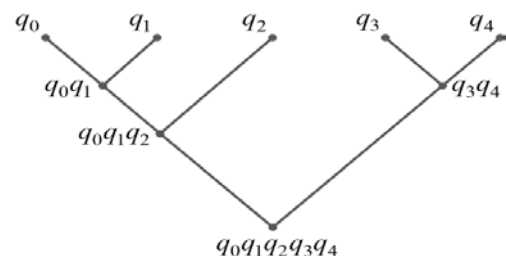


Рис. 1. Пример дерева произведений

Соответствующую последовательность наборов многочленов можно записать так:

$$\begin{aligned}
(32) \quad & q_0, q_1, q_2, q_3, q_4; \\
& q_0q_1, q_2, q_3, q_4; \\
& q_0q_1q_2, q_3, q_4; \\
& q_0q_1q_2, q_3q_4; \\
& q_0q_1q_2q_3q_4.
\end{aligned}$$

Построим дерево произведений, которое потребуется для вычисления значений многочлена на множестве точек.

По условию, имеется набор точек $x_0, x_1, \dots, x_{n-1}$ и соответствующий им набор кратностей $m_0, m_1, \dots, m_{n-1}$. Расположим точки в порядке неубывания кратностей и переобозначим эти кратности:

$$(33) \quad m_0^0 \leq m_1^0 \leq m_2^0 \leq \dots \leq m_{n-1}^0;$$

$\{x_j\}$ — переобозначены в соответствии с этим порядком.

Концевым вершинам дерева произведений ставятся в соответствие многочлены

$$(34) \quad (x - x_j)^{m_j}, \quad j = \overline{0, n-1}.$$

Дальнейшее построение выполняется пошагово, за $n-1$ шагов.

Перед k -м шагом, $k = \overline{1, n-2}$, имеется частично построенное дерево и соответствующий набор многочленов (см. (30)), степени которых образуют упорядоченный список: $m_0^{k-1} \leq m_1^{k-1} \leq m_2^{k-1} \leq \dots \leq m_{n-k}^{k-1}$. Соединяем вершины, соответствующие двум многочленам наименьших степеней — m_0^{k-1} и m_1^{k-1} — и перемножаем эти многочлены. Получаем новую вершину и многочлен степени $m_*^k = m_0^{k-1} + m_1^{k-1}$.

Переупорядочиваем новый набор степеней $\{m_*, m_2^{k-1}, m_3^{k-1}, \dots, m_{n-k}^{k-1}\}$ в порядке неубывания — после переобозначения получаем список $m_0^k \leq m_1^k \leq \dots \leq m_{n-k-1}^k$. Нетрудно убедиться, что в этом списке

$$(35) \quad \begin{aligned} m_0^k &= \min\{m_*^k, m_2^{k-1}\} = \min\{m_0^{k-1} + m_1^{k-1}, m_2^{k-1}\}, \\ m_1^k &= \text{mid}\{m_*^k, m_2^{k-1}, m_3^{k-1}\} = \\ &= \text{mid}\{m_0^{k-1} + m_1^{k-1}, m_2^{k-1}, m_3^{k-1}\} \end{aligned}$$

(здесь $\text{mid}\{a, b, c\}$ обозначает среднее из трёх чисел).

Наконец, перед шагом $n-1$ имеется список из двух элементов: $m_0^{n-2} \leq m_1^{n-2}$, — в котором $m_1^{n-2} = \max\{m_0^{n-3} + m_1^{n-3}, m_2^{n-3}\}$. Перемножаем соответствующие многочлены и получаем многочлен степени $m_*^{n-1} = m_0^{n-2} + m_1^{n-2} = N$ в корне дерева.

3.2. Дерево остатков

Для заданного дерева произведений и некоторого многочлена можно определить дерево остатков.

Пусть дан многочлен $P(x) \in \mathbb{C}[x]$ и дерево произведений (T, f) . Тогда деревом остатков называется пара (T, g) , где T — то же бинарное дерево, что и в дереве произведений, g — отображение множества

вершин T в пространство $\mathbb{C}[x]$, и для любой вершины α выполнено условие: $g(\alpha) = P \bmod f(\alpha)$.

В частности, g задает взаимно-однозначное соответствие между множеством концевых вершин дерева T и множеством остатков $\{P \bmod q_j\}_{j=\overline{0, k-1}}$. Если

$$\deg P < \sum_{j=0}^{k-1} \deg q_j, \text{ то корню дерева соответствует сам}$$

многочлен P . Для фиксированного многочлена и дерева произведений дерево остатков определено однозначно.

Дерево остатков также можно строить индуктивно, от корня к концевым вершинам. Вершины T обходятся в ширину или в глубину, при этом если β_1 и β_2 — сыновья очередной вершины α , полагаем:

$$(36) \quad \begin{aligned} g(\beta_1) &= g(\alpha) \bmod f(\beta_1), \\ g(\beta_2) &= g(\alpha) \bmod f(\beta_2). \end{aligned}$$

Например, для дерева произведений, приведённого на рисунке 1, дерево остатков будет выглядеть следующим образом:

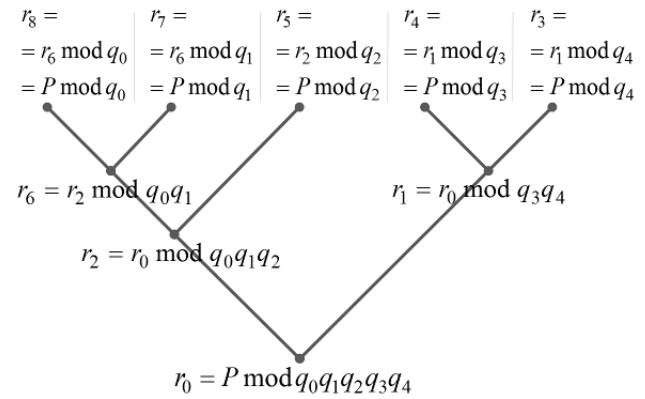


Рис. 2. Пример дерева остатков

Второй этап вычисления значений многочлена на множестве точек — построение дерева остатков, для построенного выше дерева произведений и заданного многочлена. По условию, $\deg P = N-1$, а в корне дерева произведений находится многочлен степени N , поэтому в корень дерева остатков помещается сам многочлен $P(x)$. Дальнейшее построение снова выполняется пошагово.

На первом шаге вычисляются остатки от деления $P(x)$ на многочлены степеней m_0^{n-2} и m_1^{n-2} — получаются два многочлена степеней $m_0^{n-2}-1$ и $m_1^{n-2}-1$ соответственно.

Перед шагом k степень одного из полученных на предыдущем шаге многочленов равна $m_*^{n-k}-1$. Для определённости, очередное деление производится для этого многочлена — таким образом, вершины дерева обходятся в порядке, обратном тому, в котором они добавлялись в дерево произведений. Вычисляются ос-

татки от деления этого многочлена на многочлены степеней m_0^{n-k-1} и m_1^{n-k-1} .

После $n-1$ шагов дерево остатков построено.

3.3. Вычисление значений

Концевым вершинам построенного дерева остатков соответствуют многочлены $r_j = P \bmod (x - x_j)^{m_j}$:

$$(37) \quad P(x) = q_j(x)(x - x_j)^{m_j} + r_j(x).$$

Поскольку для всех $k = \overline{0, m_j - 1}$ $P^{(k)}(x_j) = r_j^{(k)}(x_j)$, для решения поставленной задачи достаточно вычислить необходимые производные остатка в каждой точке, так, как это описано в 2.4.

4. Оценка количества операций

4.1. Операции на построение дерева произведений

Первое действие, которое выполняется при построении дерева произведений, описанном в разделе 3.1. — переупорядочение кратностей точек. Его можно выполнить, применив, например, алгоритм сортировки слияниями (см. [4]). На это потребуется не более $4n \log n$ операций сравнения.

Далее, вычисляются коэффициенты многочленов $(x - x_j)^{m_j}$. Биномиальные коэффициенты не зависят от входных точек и могут быть вычислены заранее. Степени x_j от 2 до m_j могут быть вычислены индуктивно, за $m_j - 1$ операцию. Ещё $m_j - 1$ операций требуется на домножение этих степеней на биномиальные коэффициенты — таким образом, всего на вычисление коэффициентов многочленов в концевых вершинах требуется

$$(38) \quad \sum_{j=0}^{n-1} 2(m_j - 1) = 2N - 2n$$

арифметических операций с комплексными числами.

В ходе построения на шаге k переупорядочение нового набора степеней $\{m_*^k, m_2^{k-1}, m_3^{k-1}, \dots, m_{n-k}^{k-1}\}$ представляет собой вставку нового элемента m_*^k в упорядоченный список длины $n - k - 1$. Эту вставку можно выполнить при помощи бинарного поиска, за не более чем $\log(n - k - 1)$ операций сравнения.

Таким образом, общее количество операций сравнения на построение дерева произведений не превышает

$$(39) \quad 4n \log n + \sum_{k=1}^{n-2} \log(n - k - 1) \leq 4n \log n + \log n! < 5n \log n.$$

Для оценки количества арифметических операций докажем два простых вспомогательных факта.

Утверждение 1. Для любых $a, b, c \in \mathbb{N}$ справедливо соотношение $\text{mid}\{a, b, c\} \leq (a + b + c)/2$.

Доказательство. Не ограничивая общности, перенумеруем числа так, что $a \leq b \leq c$; тогда $\text{mid}\{a, b, c\} = b$. Из неравенства $b \leq c < a + c$ следует, что $b < (a + b + c)/2$, что и требовалось доказать. $\square$

Утверждение 2. Пусть $a_0 \leq a_1 \leq \dots \leq a_{k-1}$, где $a_j \in \mathbb{N}$ для $j = \overline{0, k-1}$. Тогда для всякого $p \leq k$ имеет место неравенство:

$$(40) \quad a_0 + a_1 + \dots + a_{p-1} \leq \frac{p}{k} A, \quad \text{где } A = \sum_{j=0}^{k-1} a_j.$$

Доказательство. Если $p = k$, то левая часть равна правой по определению. Если $p < k$, то возможны два случая.

1) $\forall j \geq p \quad a_j \geq A/k$. Тогда

$$(41) \quad \begin{aligned} a_0 + \dots + a_{p-1} &= A - (a_p + \dots + a_{k-1}) \leq \\ &\leq A - \frac{k-p}{k} A = \frac{p}{k} A. \end{aligned}$$

2) $\exists j \geq p: a_j < A/k$. Тогда $\forall j < p \quad a_j < A/k$, откуда и следует доказываемое соотношение. $\square$

Из утверждения 1, для $k = \overline{1, n-3}$ справедливо:

$$(42) \quad \begin{aligned} m_1^k &= \text{mid}\{m_0^{k-1} + m_1^{k-1}, m_2^{k-1}, m_3^{k-1}\} \leq \\ &\leq (m_0^{k-1} + m_1^{k-1} + m_2^{k-1} + m_3^{k-1})/2. \end{aligned}$$

Заметим, что на каждом шаге построения дерева произведений сумма всех текущих степеней многочленов равна N :

$$(43) \quad \forall k = \overline{0, n-1} \quad N = \sum_{j=0}^{n-k-1} m_j^k.$$

Отсюда и из утверждения 2 следует общая формула: для $k = \overline{0, n-1}$ и $p = \overline{1, n-k}$

$$(44) \quad \sum_{j=0}^{p-1} m_j^k \leq \frac{p}{n-k} N,$$

— в частности, для $k = \overline{0, n-4}$

$$(45) \quad m_0^k + m_1^k + m_2^k + m_3^k \leq \frac{4}{n-k} N.$$

Теперь можно вывести оценку количества арифметических операций на умножение многочленов в ходе построения дерева. На шаге k на умножение двух многочленов достаточно $M(m_1^{k-1})$ или менее арифметических операций. При этом, поскольку при дальнейшем

построении дерева остатков делить $P(x)$ на многочлен в корне не требуется, умножение на последнем шаге можно опустить. В выкладках используется соотношение из п. 2.1, а также оценки (42) и (45). Считаем, что при $k < 0$ $m_j^k = 0$ и соответствующие им слагаемые пропадают.

$$\begin{aligned}
 \sum_{k=0}^{n-3} M(m_1^k) &\leq M\left(\sum_{k=0}^{n-3} m_1^k\right) \leq \\
 &\leq M\left(m_1^0 + \sum_{j=1}^{n-3} (m_0^{k-1} + m_1^{k-1} + m_2^{k-1} + m_3^{k-1})/2\right) \leq \\
 (46) \quad &\leq M\left(N + \frac{1}{2} \sum_{k=0}^{n-4} (m_0^k + m_1^k + m_2^k + m_3^k)\right) \leq \\
 &\leq M\left(N + \frac{1}{2} \cdot 4N \sum_{l=4}^n \frac{1}{l}\right).
 \end{aligned}$$

Известно ([7], раздел 1.2.7), что для $q \in \mathbb{N}$

$$(47) \quad 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{q} = C + \ln q + \varepsilon_q,$$

где $0,57 < C < 0,58$, $\varepsilon_q \xrightarrow{q \rightarrow \infty} 0$, причём $\varepsilon_q < 0,2$ при $q \geq 4$. Отсюда, неравенство (46) можно продолжить:

$$\begin{aligned}
 \sum_{k=0}^{n-3} M(m_1^k) &\leq \\
 (48) \quad &\leq M\left(N\left(1 + 2\left(C + \ln n + \varepsilon_n - \frac{11}{6}\right)\right)\right) \leq \\
 &\leq M\left(N\left(1 + 1,16 + 2 \ln n + 0,4 - \frac{22}{6}\right)\right) \leq \\
 &\leq M(N \cdot 2 \ln 2 \log_2 n) \leq M(3/2 \cdot N \log n).
 \end{aligned}$$

Итого, количество операций (сравнения и арифметических) на построение дерева произведений не превышает

$$(49) \quad 2N - 2n + 5n \log n + M(3/2 \cdot N \log n).$$

4.2. Операции на построение дерева остатков

На первом шаге многочлен степени $N-1$ делится на многочлены степеней: 1) m_0^{n-2} и 2) m_1^{n-2} . Поскольку $m_0^{n-2} + m_1^{n-2} = N$ и $m_0^{n-2} \leq m_1^{n-2}$, справедлива оценка $m_1^{n-2} \geq N/2$. Отсюда, на нахождение остатка от деления 2 требуется $D(m_1^{n-2})$ или менее арифметических операций, а на нахождение остатка от деления 1 — $\tilde{D}(N)$ или менее.

Аналогичная оценка применяется и для каждого k -го шага. Для нахождения остатка отделения многочлена степени $m_*^{n-k} - 1$ на многочлен степени m_1^{n-k-1} достаточно $D(m_1^{n-k-1})$ операций, а на многочлен степени m_0^{n-k-1} — $\tilde{D}(m_*^{n-k})$ операций.

Заметим, что $N = m_*^{n-1}$, а $m_*^l \leq 2m_1^{l-1}$. Общее количество арифметических операций на построение дерева остатков не превышает

$$\begin{aligned}
 \sum_{k=1}^{n-1} (D(m_1^{n-k-1}) + \tilde{D}(m_*^{n-k})) &\leq \\
 (50) \quad &\leq D\left(\sum_{l=0}^{n-2} m_1^l\right) + \tilde{D}\left(\sum_{l=1}^{n-1} m_*^l\right) \leq \\
 &\leq D\left(\sum_{l=0}^{n-2} m_1^l\right) + \tilde{D}\left(2 \sum_{l=0}^{n-2} m_1^l\right) \leq \\
 &\leq D(3/2 \cdot N \log n) + \tilde{D}(3N \log n).
 \end{aligned}$$

Здесь к аргументам D и $\tilde{D}$ применена та же оценка, что и в п. 4.1.

4.3. Операции на вычисление значений

Для каждого $j = \overline{0, n-1}$ на вычисление $m_j - 1$ производных $r_j(x)$ в точке x_j требуется $C(m_j)$ арифметических операций. Всего требуется

$$(51) \quad \sum_{j=0}^{n-1} C(m_j) \leq C\left(\sum_{j=0}^{n-1} m_j\right) = C(N)$$

операций.

4.4. Общее количество операций

Суммируя (49), (50) и (51), получаем, что описанный алгоритм требует на все вычисления не более, чем

$$\begin{aligned}
 &2N - 2n + 5n \log n + M(3/2 \cdot N \log n) + \\
 &+ D(3/2 \cdot N \log n) + \tilde{D}(3N \log n) + C(N) = \\
 &= 2N - 2n + 5n \log n + 18N \log N + 36N + \\
 &+ (18 + 102) \cdot 3/2 \cdot N \log n \log(3/2 \cdot N \log n) + \\
 &+ (32 + 263) \cdot 3/2 \cdot N \log n + \\
 (52) \quad &+ 222 \cdot 3N \log n \log(3N \log n) + 762 \cdot 3N \log n \leq \\
 &\leq 38N + 5n \log n + 18N \log N + \\
 &+ 180N \log n(1 + \log N + \log \log n) + \\
 &+ 666N \log n(2 + \log N + \log \log n) + \\
 &+ 2729N \log n = 38N + 5n \log n + 18N \log N + \\
 &+ 4241N \log n + 846N \log N \log n + \\
 &+ 846N \log n \log \log n
 \end{aligned}$$

операций. Эту оценку можно упростить, если пожертвовать точностью и принять во внимание соотношение:

$$\begin{aligned}
 (53) \quad & 1 \leq \log N \leq \log(N+1), \\
 & 1 \leq \log n \leq \log(n+1), \\
 & n \leq N, \\
 & \log \log n \leq \log N.
 \end{aligned}$$

Продолжая выкладки (52), получаем:

$$\begin{aligned}
 (54) \quad & (38 + 5 + 18 + 4241 + 2 \cdot 846)N \log(N+1) \log(n+1) = \\
 & = 5994 \cdot N \log(N+1) \log(n+1) = T(n, N).
 \end{aligned}$$

Искомая оценка доказана для значения константы 5994.

5. Реализация на микропроцессоре КОМДИВ128-РИО

5.1. Особенности реализации

Автором была разработана программная процедура, реализующая описанный алгоритм на микропроцессоре КОМДИВ128-РИО (K128, см. [8]), с использованием математического сопроцессора CP2. Данная процедура вычисляет значения многочлена на мультимножестве в более общем случае: степень входного многочлена может быть произвольной. Учитывая, что CP2 является SIMD-сoproцессором и ориентирован на потоковые вычисления, для эффективного использования аппаратных возможностей процессора была реализована множественная процедура: вычисление значений целого набора многочленов одинаковой степени на одном и том же мультимножестве точек.

На вход процедуре передаются:

- количество точек мультимножества (n);
- набор этих точек;
- набор их кратностей;
- сумма кратностей (M);
- степень многочленов (N);
- количество многочленов;
- набор коэффициентов многочленов.

На выходе процедура выдаёт набор значений многочленов на мультимножестве.

Верхняя логика алгоритма: построение дерева, сортировка и т.д. — включает рекурсию и большое количество условных переходов. Такого рода операции выгоднее производить на управляющем процессоре (CPU), чем на сопроцессоре CP2. Однако весь алгоритм сводится в конечном итоге к умножению многочленов, которое вычисляется через свёртку. Имеется оптимизированная процедура множественной свёртки, через БПФ, для CP2 (см. [9]).

Исходя из этих соображений, была разработана процедура, которая представляет собой код на языке C для управляющего процессора; в этом коде для умножения многочленов вызывается процедура свёртки на CP2. Перед началом вычислений выделяются массивы в системной памяти для сохранения коэффициентов промежуточных многочленов, которые встречаются в алгоритме. Все массивы выравнены согласно требованиям DMA (см. [8]) — это позволяет избежать много-

кратного выделения и освобождения памяти в ходе вычислений.

Была разработана также версия процедуры без использования CP2, в которой умножение многочленов выполняется на управляющем процессоре.

5.2. Результаты замеров

Готовая процедура была верифицирована и протестирована на эмуляторе и на реальном процессоре K128, а также на ПЛИС-прототипе перспективного микропроцессора BM9.

Замеры производились на случайных наборах кратностей, точек и коэффициентов многочленов, при заданной сумме кратностей и степени многочленов. При этом рассматривались только точки и коэффициенты многочленов, по модулю меньшие 1. Тестирование показало, что несмотря на это ограничение, в среднем если количество точек больше нескольких десятков, или, когда точек больше одной, кратность хотя бы одной из них больше 5, то в ходе вычислений очень быстро накапливается критическая погрешность (поскольку производные высоких порядков включают биномиальные коэффициенты). Отсюда следует, что для данного алгоритма необходимо выполнять вычисления с двойной точностью; на текущей версии сопроцессора CP2 они не поддерживаются.

Ниже приводятся некоторые результаты замеров производительности, для тех случаев, когда погрешность результата не превышает 10^{-4} . На K128 замеры производились на рабочей частоте 200 МГц. На ПЛИС BM9 замеры производились на частоте 62,5 МГц — на графиках приведены результаты, промасштабированные до 200 МГц, для удобства сопоставления с результатами на K128. Количество многочленов при тестировании — 100 для процедуры на CPU и 1000 для процедуры с использованием CP2. Во всех случаях приводится среднее значение производительности по 10 запускам.

На рисунке 3 представлены производительности процедуры для случая, когда вычисляются производные многочленов в одной точке. Независимо от кратности точки, рассматриваются многочлены степени 7.

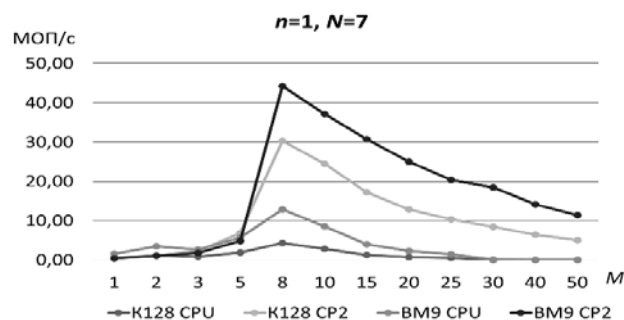


Рис. 3. Замеры на K128 и BM9: вычисление производных в одной точке

В данном случае «дерево произведений» состоит из одного узла, и существенная доля времени работы приходится на вычисление свёртки. За счёт этого использование CP2 даёт выигрыш в производительности в несколько раз, по сравнению с вычислениями на CPU.

Пик достигается при $M = 8$: поскольку $N \leq M - 1$, не требуется выполнять деление входных многочленов на многочлен в узле дерева. Дальнейшее падение производительности связано с избыточностью: для вычисления свёртки наборы коэффициентов входных многочленов приходится дополнять нулями до длины $2M$. Однако даже максимальная производительность составляет менее 1% от пиковой (8 ГОП/с) — это объясняется большими накладными расходами на подготовительные действия и вычисления на CPU (например, вычисление коэффициентов многочлена в узле), даже в случае одной точки.

На рисунке 4 приводятся результаты замеров для случая, когда вычисляются значения многочленов в наборе точек. Степень многочленов также фиксирована и равна 39.

Здесь вычисления по алгоритму, в отличие от рассмотренного выше случая, включают построение дерева, которое имеет симметричную структуру. На графике можно увидеть, что явный выигрыш от использования CP2 имеет место на K128, но не на BM9, который, согласно документации, обладает более производительным управляющим процессором.

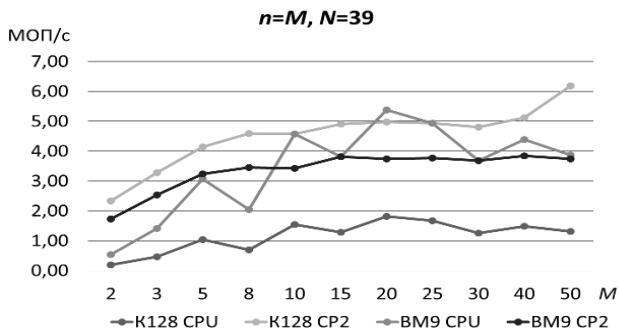


Рис. 4. Замеры на K128 и BM9: вычисление значений в n точках

На рисунке 5 представлены результаты для общего случая, когда точки имеют различную кратность. Здесь, для определённости, рассматриваются только многочлены степени $M - 1$.

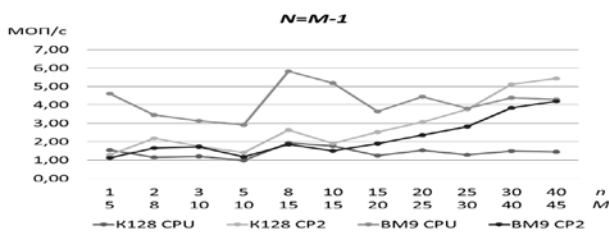


Рис. 5. Замеры на K128 и BM9: вычисление на наборе точек со случайными кратностями

Согласно приведённым данным, на малых множествах выгода от использования CP2 нивелируется за счёт сложной верхней логики построения дерева произведений.

На BM9 целесообразно производить все вычисления на CPU. С ростом количества точек и суммы их кратностей производительность вычислений с использованием CP2 возрастает, но при этом становится неудовлетворительной точность результата.

6. Выводы

Построен алгоритм, решающий поставленную общую задачу вычисления значений многочлена на множестве точек.

Доказано, что на все вычисления по этому алгоритму достаточно

$$\begin{aligned}
 T(n, N) &= 5994 \cdot N \log(N+1) \log(n+1) = \\
 (55) \quad &= O(N \log(N+1) \log(n+1))
 \end{aligned}$$

арифметических операций над комплексными числами и операций сравнения натуральных чисел.

Разработана программная процедура, реализующая множественную версию алгоритма на микропроцессорах НИИСИ РАН линейки КОМДИВ, в частности, с использованием сопроцессора CP2.

Результаты тестирования этой процедуры на доступном оборудовании показали, что для обеспечения возможности расчётов по данному и аналогичным ему алгоритмам в реальных задачах необходима доработка аппаратной базы — главным образом, поддержка вычислений с двойной точностью.

Использование математического сопроцессора CP2 позволяет получить заметный выигрыш в производительности в ряде частных случаев, когда существенная доля времени работы процедуры приходится на вычисление свёртки.

В данной публикации развиты результаты дипломной работы автора, выполненной на механико-математическом факультете МГУ под руководством А.Г.Кушниренко.

Благодарю А.Г.Кушниренко за интерес к работе и плодотворные обсуждения.

Fast polynomial evaluation on a multiset of points

O.J. Sudareva

Abstract The article considers the evaluation of a polynomial on a given “multiset” of points, i.e., a set of points with natural multiplicities. We denote by the value of a polynomial at the point with a given multiplicity a set consisting of values at this point of the polynomial and all its derivatives of orders less than the point multiplicity. Let n denote the number of geometrically different points, N — the sum of all the point multiplicities. For a polynomial of degree $N - 1$ or less and a multiset consisting of one point or only of points with multiplicities equal to 1 there are well-known evaluation algorithms that require $O(N \log_2(N + 1) \log_2(n + 1))$ operations. An operation refers to the four arithmetic operations with complex numbers and operations of comparison of natural numbers. In the article an algorithm is proposed for a general case, when there are no restrictions on multiplicities and the number of points. The same asymptotic estimate of the operations number as in the special cases is proven to be valid for this algorithm, and a formula for the operations number estimate is given with a precisely calculated multiplicative constant. An implementation of this algorithm for the KOMDIV128-RIO microprocessor has been devised by the author. The results of its tests are provided in the article.

Keywords: algorithm, polynomial, complexity

Литература

1. V. Y. Pan. Structured Matrices and Polynomials: Unified Superfast Algorithms — Boston: Birkhäuser/Springer, 2001 — 278 p.
2. A. G. Akritas, S. D. Danielopoulos. On the complexity of algorithms for the translation of polynomials // Computing, 1980 — no. 24 — pp. 51-60.
3. A. Bostan, E. Schost. Polynomial evaluation and interpolation on special sets of points // Journal of Complexity, 2005 — no. 21(4) — pp. 420–446.
4. A. V. Aho, J. E. Hopcroft, J. D. Ullman. The Design and Analysis of Computer Algorithms — Addison-Wesley, Reading, Mass, 1974 — 470 p.
5. D. J. Bernstein. Fast multiplication and its applications // Algorithmic number theory / Ed. J. Buhler, P. Stevenhagen — Cambridge: Cambridge University Press, 2008 — pp. 325–384.
6. J. W. Cooley, J. W. Tukey. An algorithm for the machine calculation of complex Fourier series // Math. Comput., 1965 — no. 19 — pp. 297–301.
7. Д. Кнут. Искусство программирования, том 1. Основные алгоритмы: Пер. с англ. — 3-е изд. — М.: Вильямс, 2006. — 720 с.
8. Микросхема интегральная 1890ВМ7Я (КОМДИВ128-РИО), указания по применению. // Москва: НИИСИ РАН, 2009.
9. О.Ю. Сударева. Эффективная реализация алгоритмов быстрого преобразования Фурье и свёртки на микропроцессоре КОМДИВ128-РИО / под ред. академика РАН В.Б. Бетелина — Москва: НИИСИ РАН, 2014 — 266 с.

Ускорение степенного метода нахождения максимальных собственных значений и их векторов для сверхбольших заполненных матриц

М.Л.Бахмутский

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН», Москва, Россия,
E-mail: mbakhmut@mail.ru

Аннотация: В работе предложен новый вариант степенного метода для нахождения нескольких максимальных собственных значений и, соответствующих, собственных векторов сверхбольших (порядка тысяч и десятков тысяч строк и столбцов) заполненных матриц. Предлагаемый метод заключается в использовании вейвлет – преобразования Хаара и теории возмущений для построения вектора близкого к искомому собственному в качестве начального приближения.

Ключевые слова: степенной метод, начальное приближение

1. Введение

Рассмотрим классический степенной метод [1]. Пусть мы имеем квадратную положительно определенную матрицу с собственными значениями $\lambda_1 > \lambda_2 > \dots > \lambda_N$. Нашей целью является нахождение максимального собственного значения λ_1 и, соответствующего собственного вектора.

Возьмем произвольный вектор $y^{(0)}$, как правило со случайным набором компонент, и построим итерационный процесс:

$$\begin{aligned} x^{(0)} &= y^{(0)} / \|y^{(0)}\| \\ y^{(k+1)} &= Ax^{(k)} \\ x^{(k+1)} &= y^{(k+1)} / \|y^{(k+1)}\| \end{aligned} \quad (1)$$

В результате

$$\lambda_1 = \lim_{k \rightarrow \infty} (x^{(k)}, y^{(k+1)}) \quad (2)$$

И для достаточно больших k , вектор $x^{(k)}$ приближенно равен соответствующему собственному вектору.

Действительно, предположим что нам известны все собственные значения λ_i и все собственные вектора u_i матрицы A . Разложим произвольный вектор $y^{(i)}$ по векторам базиса u_i :

$$y^{(0)} = \sum_{i=1}^N c_i u_i \quad (3)$$

При этом полагаем, что c_1 – проекция начального вектора на искомый, не равна нулю. У нас нет гарантии этого, однако даже в случае нулевой проекции, в ходе итераций появляется

ненулевая проекция, в силу конечности разрядной сетки компьютера. Этот вектор с ненулевой компонентой на направление собственного вектора может рассматриваться как начальный в процессе (1). В ходе итерационного процесса (1) получаем последовательность векторов

$$\begin{aligned} x^{(0)} &= \frac{u_1 + \sum_{i=2}^N \frac{c_i}{c_1} u_i}{\sqrt{1 + \sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2}} \\ x^{(1)} &= \frac{\left(u_1 + \sum_{i=2}^N \frac{c_i}{c_1} \frac{\lambda_i}{\lambda_1} u_i\right)}{\sqrt{\left(1 + \sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2 \left(\frac{\lambda_i}{\lambda_1}\right)^2\right)}} \\ x^{(2)} &= \frac{u_1 + \sum_{i=2}^N \frac{c_i}{c_1} \left(\frac{\lambda_i}{\lambda_1}\right)^2 u_i}{\sqrt{\left(1 + \sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2 \left(\frac{\lambda_i}{\lambda_1}\right)^4\right)}} \\ &\dots\dots\dots \\ x^{(k)} &= \frac{u_1 + \sum_{i=2}^N \frac{c_i}{c_1} \left(\frac{\lambda_i}{\lambda_1}\right)^k u_i}{\sqrt{1 + \sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2 \left(\frac{\lambda_i}{\lambda_1}\right)^{2k}}} \end{aligned} \quad (4)$$

и приближенные собственные значения матрицы A :

$$\begin{aligned} \lambda_1^k &= (x^{(k)}, y^{(k+1)}) \\ \lambda_1^k &= \lambda_1 \frac{1 + \sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2 \left(\frac{\lambda_i}{\lambda_1}\right)^{2k+1}}{1 + \sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2 \left(\frac{\lambda_i}{\lambda_1}\right)^{2k}} \end{aligned} \quad (5)$$

В результате k итераций получаем собственную пару $\lambda_1 \approx \lambda_1^k$ и $u_1 \approx x^{(k)}$. Относительная ошибка при этом

$$\delta \equiv \left| \frac{\lambda_1 - \lambda_1^k}{\lambda_1} \right|$$

$$\delta = \frac{\sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2 \left(\frac{\lambda_i}{\lambda_1}\right)^{2k} \left(1 - \frac{\lambda_i}{\lambda_1}\right)}{1 + \sum_{i=2}^N \left(\frac{c_i}{c_1}\right)^2 \left(\frac{\lambda_i}{\lambda_1}\right)^{2k}} \quad (6)$$

Асимптотически скорость сходимости определяется только отношением λ_i / λ_1 , т.е. только свойствами матрицы и не зависит от компонент C_i , т.е. от начального приближения $x^{(0)}$. Однако, на практике, для достижения конечной заданной точности решения спектральной проблемы число итераций будет зависеть от подбора начального приближения, т.е. от соотношений C_i / C_1 . Поэтому, в данной работе, для ускорения степенного метода предлагается подбирать вектор начального приближения близкий к искомому вектору.

2. Описание алгоритма

2.1 Преобразование матрицы

В данной работе мы обсуждаем матрицы, которые можно рассматривать как дискретную аппроксимацию функций Грина некоторых операторов Штурма – Лиувилля.

Предположим, что $\mu_1 < \dots < \mu_n$

собственные значения, а ψ_n – собственные функции некоторого оператора Штурма – Лиувилля. Согласно осциляционной теореме [2], собственная функция соответствующая n -му собственному значению имеет $n-1$ нулей. Тогда функция Грина этого оператора имеет вид

$$\sum_k \frac{\psi_k(\vec{r}) \psi_k(\vec{r}_0)}{\mu_k}.$$

Таким образом, чем больше собственное значение матрицы, тем менее осцилирующий вектор ей соответствует.

Рассмотрим задачу

$$Ax = \lambda x. \quad (7)$$

Симметричная матрица A имеет размерность $N \times N$. Фактически не теряя общности, полагаем

$$N = 2^k N_1.$$

Сформируем матрицу $W_{(N)}$ с элементами

$$(W_{(N)})_{i,j} = \left\{ \frac{\delta_{2i-1,j} + \delta_{2i,j}}{\sqrt{2}} \right\}; 1 \leq i \leq \frac{N}{2}$$

$$(W_{(N)})_{i,j} = \left\{ \frac{\delta_{2i-1-N,j} - \delta_{2i-N,j}}{\sqrt{2}} \right\};$$

$$\frac{N}{2} < i \leq N$$

И транспонированную к ней матрицу с элементами:

$$(W_{(N)}^T)_{i,j} = \left\{ \frac{\delta_{i,2j-1} + \delta_{i,2j}}{\sqrt{2}} \right\}; 1 \leq j \leq \frac{N}{2}$$

$$(W_{(N)}^T)_{i,j} = \left\{ \frac{\delta_{i,2j-1-N} - \delta_{i,2j-N}}{\sqrt{2}} \right\}$$

$$\frac{N}{2} < j \leq N \quad (8)$$

Здесь δ – символ Кронекера.

Эта матрица соответствует одному уровню вейвлет – преобразованию Хаара [3]. Легко проверить, что

$$W_{(N)} W_{(N)}^T = W_{(N)}^T W_{(N)} = E_{(N)}$$

т.е. матрица W ортогональная с единичной нормой.

Пусть $E_{(N)}$ – единичная матрица размерности N . Обозначим

$$\underset{(1)}{W} = W_{(N)}$$

и умножим слева уравнение (7) на матрицу

$$\underset{(1)}{W}$$

Получим:

$$\underset{(1)}{W} A \underset{(1)}{W}^T \underset{(1)}{W} x = \lambda \underset{(1)}{W} x$$

Обозначим далее :

$$\underset{(0)}{\hat{A}} \equiv A,$$

$$\underset{(1)}{\hat{A}} = \underset{(1)}{W} \underset{(0)}{\hat{A}} \underset{(1)}{W}^T,$$

$$\underset{(1)}{\hat{x}} \equiv \underset{(1)}{W} x.$$

Тогда задача (7) переходит в задачу

$$\underset{(1)}{\hat{A}} \underset{(1)}{\hat{x}} = \lambda \underset{(1)}{\hat{x}} \quad (9)$$

Матрица

$$\underset{(1)}{\hat{A}}$$

имеет следующую структуру:

$$\underset{(1)}{\hat{A}} = \begin{pmatrix} A_{S,S}^1 & A_{S,D}^1 \\ A_{D,S}^1 & A_{D,D}^1 \end{pmatrix}$$

Блок $A_{S,S}^{(1)}$ соответствует сглаживанию строк и столбцов. Ему отвечают номера строк и столбцов

$$1 \leq i \leq \frac{N}{2}, 1 \leq j \leq \frac{N}{2}$$

Элементы блока имеют вид:

$$a_{i,j}^1 = \frac{1}{2}(a_{2i-1,2j-1}^0 + a_{2i-1,2j}^0 + a_{2i,2j-1}^0 + a_{2i,2j}^0)$$

Блок $A_{S,D}^1$ соответствует сглаживанию строк и вычитанию столбцов. Ему отвечают номера строк и столбцов

$$1 \leq i \leq \frac{N}{2}, \frac{N}{2} + 1 \leq j \leq N$$

Элементы блока имеют вид:

$$a_{i,j}^1 = \frac{1}{2}(a_{2i-1,2j-1-N}^0 - a_{2i-1,2j-N}^0 + a_{2i,2j-1-N}^0 - a_{2i,2j-N}^0)$$

Блок $A_{D,S}^1$ соответствует сглаживанию столбцов и вычитанию строк. Ему отвечают номера строк и столбцов

$$\frac{N}{2} < i \leq N, 1 \leq j \leq \frac{N}{2}$$

Элементы блока имеют вид:

$$a_{i,j}^1 = \frac{1}{2}(a_{2i-1-N,2j-1}^0 + a_{2i-1-N,2j}^0 - a_{2i-N,2j-1}^0 - a_{2i-N,2j}^0)$$

Блок $A_{D,D}^1$ соответствует вычитанию столбцов и вычитанию строк. Ему отвечают номера строк и столбцов

$$\frac{N}{2} < i \leq N, \frac{N}{2} < j \leq N$$

Элементы блока имеют вид:

$$a_{i,j}^1 = \frac{1}{2}(a_{2i-1-N,2j-1-N}^0 - a_{2i-1-N,2j-N}^0 - a_{2i-N,2j-1-N}^0 + a_{2i-N,2j-N}^0). \quad (10)$$

Легко видеть, что матрица $\hat{A}^{(1)}$ остается симметричной. Также легко видеть, что матричные элементы блока $A_{S,S}^1$ являются сглаживанием элементов матрицы A. Заметим, что непосредственно вычисляются только элементы верхнего левого блока $A_{S,S}^1$, вычислять блоки $A_{S,D}^1, A_{D,S}^1, A_{D,D}^1$ не надо. В качестве

следующего шага, сформируем матрицу $\hat{W}^{(2)}$:

$$\hat{W}^{(2)} = \begin{pmatrix} W^{(N/2)} & O^{(N/2)} \\ O^{(N/2)} & E^{(N/2)} \end{pmatrix} \quad (11)$$

Здесь $E^{(N/2)}$ – единичная матрица размерности N/2, а $O^{(N/2)}$ – нулевая матрица этой же размерности. Матрица $W^{(N/2)}$ определяется формулами (8) с заменой N на N/2.

Легко проверить, что эта матрица является ортогональной.

Умножим уравнение (9) слева на

матрицу $\hat{W}^{(2)}$. Получим:

$$\hat{W}^{(2)} \hat{A}^{(1)} \hat{W}^{(2)T} \hat{W}^{(2)} \hat{x}^{(1)} = \lambda \hat{W}^{(2)} \hat{x}^{(1)}$$

Обозначим

$$\hat{\hat{A}}^{(2)} = \hat{W}^{(2)} \hat{A}^{(1)} \hat{W}^{(2)T}, \quad \hat{\hat{x}}^{(2)} = \hat{W}^{(2)} \hat{x}^{(1)}$$

Тогда задача (9) переходит в задачу:

$$\hat{\hat{A}}^{(2)} \hat{\hat{x}}^{(2)} = \lambda \hat{\hat{x}}^{(2)} \quad (12)$$

Матрица $\hat{\hat{A}}^{(2)}$ имеет следующую структуру:

$$\hat{\hat{A}}^{(2)} = \begin{pmatrix} \begin{pmatrix} A_{S,S}^2 & A_{S,D}^2 \\ A_{D,S}^2 & A_{D,D}^2 \end{pmatrix} & \begin{pmatrix} W^{(N/2)} A_{S,D}^1 \\ A_{D,S}^1 W^{(N/2)} \end{pmatrix} \\ \begin{pmatrix} A_{D,S}^1 W^{(N/2)} \\ A_{D,D}^1 \end{pmatrix} & \begin{pmatrix} W^{(N/2)} A_{S,D}^1 \\ A_{D,S}^1 W^{(N/2)} \end{pmatrix} \end{pmatrix} \quad (13)$$

Легко убедиться непосредственно, что матрица

$\hat{\hat{A}}^{(2)}$ является симметричной. Для дальнейшей работы необходимо вычислить только матрицу $A_{S,S}^2$. Элементы этой матрицы вычисляются по формулам аналогичным (10). В формулах (10) размерность N заменяется на N/2, верхний индекс 0 заменяется на 1, а верхний индекс 1 заменяется на 2. Вычислять же матрицы

$$A_{S,D}^2, A_{D,S}^2, A_{D,D}^2, W^{(N/2)} A_{S,D}^1, A_{D,S}^1 W^{(N/2)}$$

не надо. Если k в уравнении (7) больше 2, то

сформируем теперь матрицу $\hat{W}^{(3)}$:

$$\hat{W}^{(3)} = \begin{pmatrix} \begin{pmatrix} W^{(N/4)} & O^{(N/4)} \\ O^{(N/4)} & E^{(N/4)} \end{pmatrix} & \begin{pmatrix} O^{(N/2)} \\ E^{(N/2)} \end{pmatrix} \\ \begin{pmatrix} O^{(N/2)} \\ E^{(N/2)} \end{pmatrix} & \begin{pmatrix} O^{(N/2)} \\ E^{(N/2)} \end{pmatrix} \end{pmatrix} \quad (14)$$

Матрица $\hat{W}^{(3)}$ является ортогональной.

Обозначим

$$\hat{W}^{(1)} = W^{(N)}$$

Матрица $\underline{W}_{(2)}$ получается подстановкой в $\underline{W}_{(1)}$ вместо матрицы $\underline{W}_{(N)}$ матрицы

$$\begin{pmatrix} \underline{W}_{(\frac{N}{2})} & \underline{O}_{(\frac{N}{2})} \\ \underline{O}_{(\frac{N}{2})} & \underline{E}_{(\frac{N}{2})} \end{pmatrix}.$$

Матрица $\underline{W}_{(3)}$ получается заменой в $\underline{W}_{(2)}$ матрицы $\underline{W}_{(\frac{N}{2})}$ на матрицу

$$\begin{pmatrix} \underline{W}_{(\frac{N}{4})} & \underline{O}_{(\frac{N}{4})} \\ \underline{O}_{(\frac{N}{4})} & \underline{E}_{(\frac{N}{4})} \end{pmatrix}$$

и т.д.

Формируя последовательность матриц

$$\underline{W}_{(1)}, \underline{W}_{(2)}, \underline{W}_{(3)}, \dots, \underline{W}_{(k)}$$

и умножая исходное уравнение (7) на матрицу

$$\underline{W} = \underline{W}_{(k)} \dots \underline{W}_{(2)} \underline{W}_{(1)},$$

получим

$$\hat{\underline{A}}^{(k)} \hat{\underline{x}}^{(k)} = \lambda \hat{\underline{x}}^{(k)},$$

где

$$\hat{\underline{A}}^{(k)} = \underline{W} \underline{A} \underline{W}^T; \quad \hat{\underline{x}}^{(k)} = \underline{W} \underline{x}. \quad (15)$$

Далее нас будет интересовать только матрица $\hat{\underline{A}}^{(k)}$ $\underline{A}_{S,S}^k$ - наиболее гладкая часть матрицы $\hat{\underline{A}}^{(k)}$, размерность которой равна $N_1 \times N_1$. Заметим, что полностью вычислять преобразованные матрицы не надо, вычисляется только самая гладкая часть матриц $\underline{A}_{S,S}^i$.

2.2 Спектральная задача

Рассмотрим спектральную задачу

$$\underline{A}_{S,S}^k \underline{x}_0^k = \lambda^0 \underline{x}_0^k \quad (16)$$

Решение частичной спектральной проблемы для этой задачи возможно традиционными методами, а именно, приведением матрицы к трехдиагональной форме при помощи ортогональных отражений, поиском отдельных собственных значений, например, методом бисекций с применением теоремы Штурма и методом обратной итерации для поиска отдельных собственных векторов [4], [5]. На рис. 1 показано сопоставление поведения 55 собственных значений модельной матрицы и матрицы самой гладкой ее части. В качестве модельной матрицы взята матрица с элементами

$$a_{i,j} = -\frac{2}{N+1} \sum_{k=1}^N \frac{\sin\left(\frac{\pi k i}{N+1}\right) \sin\left(\frac{\pi k j}{N+1}\right)}{\left(\sin\left(\frac{\pi k}{2(N+1)}\right)\right)^2}$$

и собственными значениями

$$\lambda_m = \frac{1}{\left(\sin\left(\frac{\pi m}{2(N+1)}\right)\right)^2}$$

Здесь $N=6528$, $k=7$, $N_1=51$.

На графике видно, что кривые подобны. Найдя максимальное собственное значение задачи (16) и определив по каким – либо критериям число искоемых собственных пар задачи (16), получаем нужное число собственных пар основной матрицы и приближение к собственным парам основной матрицы.

Итак в результате решения задачи (16), найдены g собственных пар этой задачи. Пусть u_i один из собственных векторов задачи (16). Его размерность равна N_1 . В пространстве R^N ему будет соответствовать вектор

$$\underline{u}_i^{(k)T} = (\underline{u}_i^T, 0, \dots, 0).$$

Проделав обратные вейвлет – преобразования получим соответствующий ему вектор

$$\underline{u}_i^{(0)} = \underline{W}_1^T \underline{W}_2^T \dots \underline{W}_k^T \underline{u}_i^{(k)} \quad (17)$$

В результате в исходном пространстве R^N получаем некоторую матрицу $\tilde{\underline{A}}$ (явно вычислять не придется, хотя это и не трудно), для которой справедливо:

$$\tilde{\underline{A}} \underline{u}_i^{(0)} = \lambda_i^0 \underline{u}_i^{(0)}. \quad (18)$$

Здесь λ_i^0 - собственное значение задачи (16), а собственный вектор определен в (17). Исходную матрицу $\underline{A}$ представим в виде:

$$\underline{A} = \tilde{\underline{A}} + \underline{B}, \quad \text{где } \underline{B} \equiv \underline{A} - \tilde{\underline{A}}.$$

Запишем уравнение (7) в виде

$$(\tilde{\underline{A}} + \varepsilon \underline{B}) \underline{x} = \lambda \underline{x}. \quad (19)$$

Здесь малый параметр ε введен формально, потом будем полагать $\varepsilon = 1$. Будем искать n – ю собственную пару задачи ($n \leq r$) путем формального разложения в степенной ряд теории возмущений Релея-Шредингера [6]:

$$\begin{aligned}\lambda_n &= \lambda_n^0 + \varepsilon \lambda_n^1 + \varepsilon^2 \lambda_n^2 + \dots \\ x_n &= u_n^{(0)} + \varepsilon x_n^1 + \varepsilon^2 x_n^2 + \dots\end{aligned}$$

Следуя [6], после простых выкладок получим в первом приближении:

$$x_n \approx u_n^{(0)} + \varepsilon \sum_{m=1, m \neq n}^N \frac{(u_m^{(0)}, B u_n^{(0)})}{\lambda_n^0 - \lambda_m^0} u_m^{(0)} \quad (20)$$

Так как ряд теории возмущений в общем случае является асимптотическим, то нет смысла вычислять высшие приближения для того, чтобы найти начальное приближение степенного метода.

Итак, в качестве начального приближения для вычисления n – й собственной пары степенным методом (1) полагаем:

$$y_n^{(0)} = u_n^{(0)} + \sum_{m=1, m \neq n}^r \frac{(u_m^{(0)}, B u_n^{(0)})}{\lambda_n^0 - \lambda_m^0} u_m^{(0)} \quad (21)$$

Эта формула есть формула (20), в которой $\varepsilon = 1$ и N заменен на r . Здесь, N заменен на размерность подпространства вычисленных собственных векторов r , в предположении что вклад остальных выгодней вычислять в ходе итераций степенного метода.

2.3. Вычисление матричных элементов

Прежде всего заметим, что в силу симметрии исходной матрицы справедливо равенство

$$(u_m^{(0)}, B u_n^{(0)}) = (u_n^{(0)}, B u_m^{(0)})$$

Для $m \neq n$ получаем

$$\begin{aligned}(u_m^{(0)}, B u_n^{(0)}) &= (u_m^{(0)}, (A - \tilde{A}) u_n^{(0)}) = (u_m^{(0)}, A u_n^{(0)} - \tilde{A} u_n^{(0)}) = (u_m^{(0)}, A u_n^{(0)} - \lambda_n^0 u_n^{(0)}) \\ &= (u_m^{(0)}, A u_n^{(0)}) - \lambda_n^0 (u_m^{(0)}, u_n^{(0)}) = (u_m^{(0)}, A u_n^{(0)})\end{aligned}$$

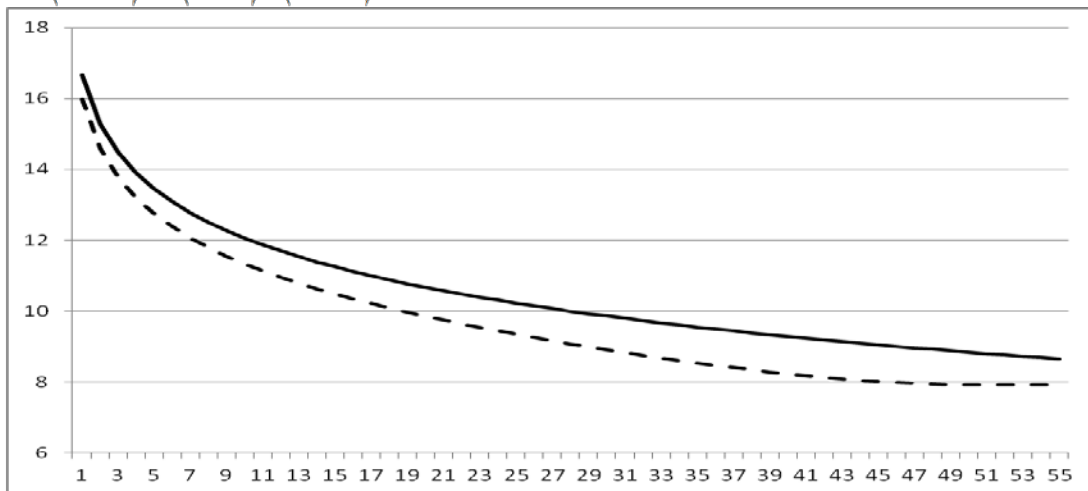


Рис.1 Натуральные логарифмы (ось ординат) точных и начальных приближений собственных значений модельной матрицы. По оси x номера собственных значений.

Поэтому окончательно получаем начальное приближение для вычисления степенным методом n – го собственного вектора:

$$y_n^{(0)} = u_n^{(0)} + \sum_{m=1, m \neq n}^r \frac{(u_m^{(0)}, A u_n^{(0)})}{\lambda_n^0 - \lambda_m^0} u_m^{(0)} \quad (22)$$

2.4. Исключение ранее вычисленных собственных пар

Предположим, что при помощи степенного метода вычислены r первых собственных пар. Для вычисления $r+1$ – й собственной пары степенной метод применяется к матрице:

$$\bar{A} = A - \sum_{s=1}^r \lambda_s x_s \otimes x_s$$

В этом случае, итерационный процесс принимает вид:

$$\begin{aligned}x^{(0)} &= \frac{y^{(0)}}{\|y^{(0)}\|} \\ y^{(k+1)} &= A x^{(k)} - \sum_{s=1}^r \lambda_s x_s (x_s, x^{(k)}) \\ x^{(k+1)} &= \frac{y^{(k+1)}}{\|y^{(k+1)}\|}\end{aligned}$$

легко убедиться непосредственно, что если в итерируемом векторе появляются компоненты в направлении векторов x_s , то они удаляются на каждой итерации.

3. Заключение.

Предлагаемый алгоритм степенного метода позволяет резко ускорить вычисление нескольких максимальных собственных пар для широкого класса сверхбольших заполненных матриц. Этот

алгоритм может быть обобщен для применения степенного метода при нахождении максимальных сингулярных чисел и соответствующих сингулярных векторов сверхбольших заполненных матриц общего типа.

Acceleration Power Method For Solving Maximal Eigenvalue And Eigenvector Very Large Dense Matrix.

M.L. Bakhmutsky

Abstract. New variant power method for determination several maximal eigenvalues and its eigenvectors for large dense matrix provided. The main idea of this method is application of wavelet transformation and perturbation theory for eigenvector initial approximation.

Keywords: power method, initial approximation

Литература.

1. Д.К. Фаддеев, В.Н. Фаддеева, Вычислительные методы линейной алгебры М., 1950
2. Дж.Сансоне, Обыкновенные дифференциальные уравнения, т.1, Изд-во «Иностранной литературы», М., 1953
3. Э.Столниц, Т.ДеРоуз, Д.Салезин, Вейвлеты в компьютерной графике: Пер. с англ.- Ижевск: НИЦ «Регулярная и хаотическая динамика», 2002
4. С.К.Годунов, Решение систем линейных уравнений, Изд-во «Наука», Новосибирск, 1980г.
5. Б.Парлетт, Симметричная проблема собственных значений, М., «Мир», 1983
6. Л.Д.Ландау, Е.М.Лифшиц, Квантовая механика, «Физматлит», М., 1963

К 70-ЛЕТИЮ ВЛАДИМИРА БОРИСОВИЧА БЕТЕЛИНА

В.Б. Демидович

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН» -
ФГБОУ ВО «Московский государственный университет имени М.В.Ломоносова»,
Механико-математический факультет, Москва, Россия, E-mail : vasdem@mech.math.msu.su



Аннотация: Приводится интервью автора с научным руководителем ФГУ ФНЦ НИИСИ РАН, академиком РАН, Владимиром Борисовичем Бетелиным по случаю его 70-летия.

Ключевые слова: системное программирование, Мехмат МГУ, НИИСИ РАН.

В конце июня 2016 года, в связи с приближающимся 70-летним юбилеем Владимира Борисовича Бетелина, я предложил ему провести со мной Интервью для публикации его в журнале «Труды НИИСИ РАН». При этом я сразу же передал Владимиру Борисовичу и *вопросы предполагаемого нашего Интервью*, как я всегда поступаю со своими собеседниками. Бегло ознакомившись с моими

вопросами, Владимир Борисович согласился на них мне ответить. Но предупредил, что выделить время для такого Интервью он сможет лишь «недельки через три», Так всё и произошло: двадцатого июля, в кабинете Владимира Борисовича, состоялось, под диктофон, наше (полтора часовое) Интервью, расшифрованный текст которого приводится ниже.

ИНТЕРВЬЮ С ВЛАДИМИРОМ БОРИСОВИЧЕМ БЕТЕЛИНЫМ

В.Б.Д.: Дорогой Владимир Борисович! Восьмого августа 2016 года Вам исполнится 70 лет. В связи с этим было бы интересно, если бы Вы поделились своими воспоминаниями о Вашей жизни, в

частности, вспомнили бы о своей молодости, о своих университетских учителях и коллегах и, конечно же, о становлении по существу Вашего детища *НИИСИ РАН*.

Из Книги «НИИСИ РАН: от Бесты до суперкомпьютера», выпущенной по случаю 25-летнего юбилея Института, я узнал, что родились Вы в Кинешме. Там же, в семье Вашего отца, инженера Бориса Алексеевича Бетелина, (в Книге сказано, что «в Вашу семью он вошёл в 1950 году») прошло и Ваше детство. Он уже умер, как мне говорили. Кстати, в каком году это произошло?

В.Б.Б.: В 2001-ом.

В.Б.Д.: Ясно.

Я знаю, также, что Ваша мама, Муза Владимировна, и поныне здравствующая – известный врач-стоматолог. Наконец, знаю, что у Вас есть младший брат, Алексей Бетелин, который, как и Вы, *мехматянин* по образованию. Так вот, первый вопрос у меня такой: кто же всё-таки у Вас пробудил интерес к математике с прикладным уклоном? Борис Алексеевич? Ваш какой-нибудь школьный учитель или же школьный приятель? А может быть это произошло у Вас «самостоятельно»?

В.Б.Б.: Ну, на самом деле, здесь должен быть *многоплановый* ответ... Если говорить про математику, то я с ней довольно долго «не очень дружил» ...

В.Б.Д.: В школьные годы?

В.Б.Б.: Да, в школе. И это было связано, вообще говоря, с учителями. Скажем, про преподавание моей учительницы математики в 6-7 классе у меня сохранилось такое мнение – «тоска зелёная» (*смеются*).

В.Б.Д.: Это в Кинешме было?:

В.Б.Б.: Нет, это уже было в Москве.

Потом я перешёл в *математическую школу*, поскольку она мне казалась «ближе к физике», которой я уже заинтересовался – космос и всё такое ...

В.Б.Д.: А какой номер школы?

В.Б.Б.: ... 320-я школа

И там «притягательной частью» стали уже *программирование* и *электронно-вычислительные машины*.

В.Б.Д.: Тогда это было очень модно – программирование.

В.Б.Б.: Модно, да... И таинственно!

А поскольку школа была с *математическим уклоном*, то там были учителя совершенно другого уровня. В частности, математику мне там преподавала Шлыкова Татьяна Максимовна. Так вот она, я не могу сказать «привила любовь к математике», но, во всяком случае, уже вызвала к ней интерес...

В.Б.Д.: *Разбудила* у Вас интерес к математике.

В.Б.Б.: Да, именно так.

А дальше вот что. По окончании этой школы я хотел поступить в МИФИ. Физика меня больше интересовала, чем математика, хотя математическое преподавание у нас в школе было поставлено гораздо выше, чем преподавание физики...

В.Б.Д.: Как всегда, кстати...

В.Б.Б.: Да. Но я, всё-таки, до последнего, хотел пойти в МИФИ. И лишь в июне месяце (1964-го года), когда кончался срок подачи вступительных документов, я понял, что меня не возьмут туда просто по зрению. И возник вопрос: если не в МИФИ, то куда тогда пойти?

В.Б.Д.: Оставался лишь Мехмат МГУ.

В.Б.Б.: Да, оставался Мехмат. Хотя один из моих взрослых друзей сказал: «Ты, конечно, не математик, но иди на Мехмат. Там и с таким зрением берут» ...

В.Б.Д.: Понятно.

И кто у Вас устный экзамен по математике принимал?

В.Б.Б.: Принимал устный экзамен? Или адис принимал.

В.Б.Д.: Ставрос Илиадис? Я его хорошо знаю. Значит, в 1964-ом году Вы поступили на Мехмат МГУ ...

В.Б.Б.: Точнее в 1964-ом году я, не пройдя по конкурсу на *дневной* Мехмата МГУ, поступил лишь на *вечерний* Мехмата МГУ. Два года там отучился, и уже в 1966-ом году, *с потерей курса*, меня, наконец-таки, перевели на *дневной Мехмат*. Причём, переводили *с вечернего на дневной* Мехмат, тогда только на *Отделение вычислительной математики*.

В.Б.Д.: Раз Вы учились на вечернем Мехмате, то, значит, Вам где-то пришлось ещё и работать?

В.Б.Б.: Да, я работал в ФИАНе

В.Б.Д.: Поближе к физике.

В.Б.Б.: Да, поближе к физике.

В.Б.Д.: Понял. А уже тогда в ФИАНе приходилось что-нибудь программировать?

В.Б.Б.: Конечно, я работал в вычислительном центре ФИАНа. Там была М-20 – тогда это была ещё полужакрывающаяся машина. Я там уже программировал на АЛГОЛе ...

В.Б.Д.: АЛГОЛ уже был?

В.Б.Б.: Да, был.

Там был такой совершенно замечательный человек, перебежчик. Я сидел в одной комнате с этим

перебежчиком из... *Агенства национальной безопасности США*, шифровальщиком Уильямом Гамильтоном Мартиным. Потом он стал называться Владиславом Антоновичем Соколовским.

Он известный человек. Можно сказать, что он меня там, в каком-то смысле, сильно образовал. Потому что АЛГОЛ там читал. Потом Шура-Бура, со своей командой, сделали из «его АЛГОЛа» «свой» *полный АЛГОЛ* - ТА-2.

Вот всем этим я в ФИАНЕ и занимался.

В.Б.Д.: Хорошо. Я вспомнил - была даже такая книжка «АЛГОЛ-60».

В.Б.Б.: Да, это классическая книжка, по которой, собственно, учились.

В.Б.Д.: Учась на Мехмате, посещали ли Вы какие-нибудь спецсеминары «не программистского типа»? Или спецкурсы?

В.Б.Б.: Да, конечно, посещал семинар *по логике*.

В.Б.Д.: Это чей же спецсеминар по логике?

В.Б.Б.: Классика нашего.

В.Б.Д.: Андрея Андреевича Маркова?

В.Б.Б.: Нет, Успенского Владимира Андреевича. Я и его спецкурс ему сдавал. С удовольствием к нему ходил.

В.Б.Д.: Думаю, он зажигательно всё рассказывал. К сожалению, сейчас он сильно болеет и, говорят, не выходит из квартиры.

А на какую кафедру, обучаясь уже на «дневном Мехмате», Вы распределились?

В.Б.Б.: На кафедру *вычислительной математики*.

В.Б.Д.: Которой заведовал тогда Андрей Николаевич Тихонов.

В.Б.Б.: Да, тогда ещё Тихонов. Я выбрал эту кафедру, поскольку у меня уже нацеленность была, «на программирование». Пошёл я к Николаю Павловичу Трифонову.

В.Б.Д.: Ясно. Он жив. Но я не сказал бы, что он здоров.

В.Б.Б.: Да. Тем не менее, я видел его несколько лет тому назад, и он выглядел ещё «вполне ничего» ...

А жена его жива?

В.Б.Д.: Валерия Васильевна? Она умерла, помнится, в 2014-м году.

В.Б.Б.: Ну, ладно, поработал я в ФИАНе где-то полгода и меня отправили в ВЦ.

В.Б.Д.: Уже в ВЦ МГУ?

В.Б.Б.: Да, в ВЦ МГУ. На самом деле, отправили писать свою курсовую работу. Но, честно говоря, мне там было просто скучно...

А потом произошла встреча с Кронродом...

В.Б.Д.: Вот! Очень интересно! Когда Вы встретились с Александром Семёновичем - на третьем курсе?

В.Б.Б.: На третьем курсе. У него был семинар по программированию, самый мощный.

В.Б.Д.: Этот семинар был на Мехмате, или куда-то ездить приходилось?

В.Б.Б.: Семинар, по-моему, всё-таки был... Нет, сейчас я уже не помню.

В.Б.Д.: Мне почему-то говорили, что Кронрод на Мехмате МГУ никогда не работал.

В.Б.Б.: Нет, Кронрод работал на Мехмате. Он работал на кафедре вычислительной математики – это точно. Но потом, в какой-то момент, он, аккуратно выражая, умудрился ...

В.Б.Д.: ... не поладить с Тихоновым ...

В.Б.Б.: Точнее, он КОГО-ТО КЕМ-ТО КАК-ТО назвал ... (*смеются*).

В.Б.Д.: То, что Александр Семёнович был «не сдержан на язык» - это мне известно. Я учился с его сыном Димой.

В.Б.Б.: Понятно.

Так вот я, собственно, ходил «на Кронрода».

В.Б.Д.: И это было, конечно, веселее, чем в ВЦ МГУ.

В.Б.Б.: Конечно! И стал уже писать свою курсовую у Кунина Петра Ефимовича.

В.Б.Д.: Я не слышал такой фамилии.

В.Б.Б.: Он тоже из этой же команды...

В.Б.Д.: То есть, из «кронродовцев»?

В.Б.Б.: Да. Он заинтересовался *медицинской диагностикой*, в частности, онкологической диагностикой. Математики этими вещами тогда только-только начали заниматься. И меня в эти дела тоже «потянуло» - ведь у меня же имеются и «медицинские корни» ... А он, со своими учениками, уже всю работу над такими проблемами.

У них даже была организована своя Лаборатория в Московском научно-исследовательском онкологическом институте имени П.А.Герцена. Там они и осуществляли свои попытки научиться, на раннем периоде, диагностировать раковые заболевания.

(Примеч. В.Б.Д.: Напомним, что, ещё до Великой Октябрьской социалистической революции ставший «доктором медицины», Пётр Александрович Герцен (1871-1947) был хирургом и одним из основателей онкологии.)

В.Б.Д.: Да-а-а, цель серьёзная!

В.Б.Б.: Вот-вот. И я, собственно говоря, всем этим тоже увлёкся... Стал работать с ними, с большим, так сказать, вдохновением и удовольствием...

Но потом мне стало не то что скучно, а просто не по себе. Я понял, что там было уже очень много таких ... однообразных приёмов ... Грубо говоря, происходило вот что: *признаки болезни*, которые давали врачи, как-то *кодировались*, *«затихивались»* в М-20, а затем, в соответствующем многомерном пространстве, *«проводилась»* *разделяющая гиперплоскость, отделяющая больных от здоровых.* Таким образом, процесс компьютерной диагностики сводился, в основном, к вычислению параметров такой гиперплоскости. Хотя, конечно, там иногда возникали и ... «нестандартные ситуации».

К тому же я заметил, что врачи нам поставляли, в основном, выборки «известных больных», быть может, даже «уже умерших» ... И постоянно в предоставляемой нам информации «здоровые попадали в больные» ... Вот тогда-то я понял, что главная-то проблема *реальной* эффективной диагностики не в алгоритмах, а в том, что мы «работаем с неинформативными признаками». *Что-то, по чему-то*, мы строим, но это, на самом деле, не то, что реально характеризует болезнь. А это, как Вы понимаете, уже *совсем другая история* ...

В.Б.Д.: Ну, да...

В.Б.Б.: И мне стало «не по себе»... Хотя Пётр Ефимович, и все в его «команде», верили, что, в конце концов, «всё наладится»... Они были математики, и всё ... «преодолеют» ...

В.Б.Д.: Но от медицины они же были, всё-таки, далеки ...

В.Б.Б.: Да, это так. И им были интересны, в первую очередь, *алгоритмы*... А для меня это уже ... потеряло смысл ...

В.Б.Д.: Поскольку не ясно, насколько такая диагностика отражает действительность?

В.Б.Б.: Именно. Алгоритмы строить важно, да, - но для чего? Ведь здесь можно придумывать совершенно красивые вещи, но они, в данном случае,

будут совершенно бесполезными. И я, написав свой диплом, покончил со всей этой тематикой ...

В.Б.Д.: Но диплом по этой тематике, всё-таки, получился?

В.Б.Б.: Да-да, у меня диплом получился именно по этой части.

В.Б.Д.: А руководителем на титуле дипломной работы кто «стоял» - Кронрод?

В.Б.Б.: Нет, не Кронрод. Как раз Кунин.

Я-то с ним точно работал. И, более того, он мне нравился гораздо больше, чем другие «кронродовцы». Скажем, чем Джеф (Игорь) Фараджев ... Были там такие ... «чапаевцы».

В.Б.Д.: Я слышал, что такое *семинар Кронрода* – ведь мой старший брат, Коля, в нём участвовал «в давние годы».

В.Б.Б.: Но Кунин был совсем другой, глубоко интеллигентный человек, очень образованный. И поэтому я, в основном, работал в его группе. Однако ходил и на семинар Кронрода.

В.Б.Д.: В той же книжке об Институте упоминается, что Вас запомнил Иван Георгиевич Петровский ещё по семинару, где рассказывалось об *искусственном интеллекте*. Это было на чьём же семинаре? Там не сказано.

В.Б.Б.: На «кронродовском». Петровский уважал Александра Семёновича, и иногда «захаживал» на заседания его семинара ...

В.Б.Д.: Теперь чуть-чуть отвлечёмся от математики на комсомольскую работу. Я, будучи в 1969-1970 годах первым секретарём комитета ВЛКСМ Мехмата МГУ, хорошо помню, что Вы активно участвовали в университетской общественной жизни, фактически возглавляя СТЭМ - Студенческий театр эстрадных миниатюр.

Довелось ли Вам впоследствии видаться с кем-нибудь из артистов того СТЭМа? И пригодился ли Вам опыт этой общественной деятельности в Вашей дальнейшей жизни?

Я даже вспомнил, что мехматовским СТЭМом руководил Игорь Левятов, но не помню, в каком году он родился. Он жив-здоров?

(Примеч. В.Б.Д.: Впоследствии я выяснил, что Игорь Давидович Левятов жив, и родился он в 1944-ом году.)

В.Б.Б.: Насколько я знаю, он жив-здоров ... Он сейчас возглавляет где-то какую-то автомобильную ассоциацию - что-то в таком духе.

В.Б.Д.: То есть, с театральной деятельностью он уже не связан?

В.Б.Б.: Думаю, не связан. Видимо, увлёкся бизнесом.

Он всегда был ... весьма глубоким автомобилистом. Поэтому он, по этой линии, и пошёл. А все остальные там ... были ребята на два или три года младше меня.

В.Б.Д.: Там, помнится, был племянник доцента Мехмата МГУ Исаака Ароновича Вайнштейна – не помню, как его звали. Запомнился мне ещё Рузер – кажется, его звали Серёжа ...

В.Б.Б.: ... Вайнштейн был, да. Был и Рузер...

В.Б.Д.: ... Какие-то две девочки были, которые Володе Гендугову нравились.

В.Б.Б.: Да, в этой компании были и девочки ... Позднее, в 1971-м году, у нас была создана ещё театральная студия «Публицист», которая «вобрала в себя» остатки тех, кто ранее играл в СТЭМе.

И мы даже поставили в ней свой спектакль ...

В.Б.Д.: Эта театр-студия была при Мехмате?

В.Б.Б.: Не совсем так – там были студенты и из других факультетов...

В.Б.Д.: Но из МГУ?

В.Б.Б.: Это да.

Так вот, с ней мы поставили, на смотре университетской самодеятельности, спектакль «Открытие Америки». Но это был первый, и последний, наш спектакль – через какое-то время, когда уже все «студийцы» стали «взрослыми», всё это как-то «исчезло»... Поэтому мы её, потом, назвали «студией одного спектакля» (*смеются*).

А про Рузера я случайно узнал – не помню, кто мне сказал – что он теперь в Израиле...

В.Б.Д.: И Вайнштейн, думаю, тоже там. Во всяком случае, его дядя, Исаак Аронович, туда уехал ... Дяди уже нет в живых – он скончался на 92-ом году жизни ...

В.Б.Б.: Понятно.

А Рузера как-то увидели, по телевизору, в какой-то телепередаче...

В.Б.Д.: По российскому или израильскому телевидению?

В.Б.Б.: По израильскому. Но в телепередаче что-то он говорил о России ...

В.Б.Д.: Любопытно.

А вообще, весёлое было то время на Мехмате.

В.Б.Б.: Да, было замечательно. С удовольствием то время вспоминаю.

В.Б.Д.: Я тоже. И о своей общественной деятельности я вспоминаю, не смотря «на некоторые неприятные моменты», *с теплотой*.

В.Б.Б.: Да-да-да.

В.Б.Д.: Но пойдём далее.

Мехмат Вы окончили в 1970-м году и распределились на работу в («закрытый») Центр – я не буду точно его называть, а скажу «по простому» – в *Центр Пилюгина*.

В.Б.Б.: Да, это так

В.Б.Д.: Проработав у Пилюгина около года, в 1971-м году, при поддержке Ивана Георгиевича Петровского, Вы вернулись в наш университет.

В.Б.Б.: И это верно.

В.Б.Д.: На Мехмате МГУ Вы стали научным сотрудником в Лаборатории вычислительных методов.

В.Б.Б.: Но сначала стал инженером.

В.Б.Д.: Понятно.

В.Б.Б.: История была такова.

Когда было решено взять меня в Лабораторию, я пришёл к её заведующему Абраму Александровичу Файнлейбу ...

В.Б.Д.: Да, в 1971-ом году её возглавлял ещё Файнлейб...

В.Б.Б.: Но вскоре Владимир Федотович Дьяченко его сменил в качестве заведующего. А после него заведующим Лабораторией стал Виктор Яковлевич Шкадов ...

В.Б.Д.: И уж потом заведующим Лабораторией стал Александр Васильевич Михалёв. Правда, пару месяцев, между Шкадовым и Михалёвым, Лабораторией «покомандовал» ещё Дмитрий Алексеевич Силаев....

В.Б.Б.: Да-да-да.

Ну так вот, пришёл я к Файнлейбу.. А ему, понятно, подозрительно, кого там «сверху» к нему направили в «его Лабораторию». И кто это такой? Он и говорит мне: «Сто рублей и инженером». Я в ответ: «Ну, ладно, сто так сто, и инженером так инженером».

Тогда вся Лаборатория занималась «запуском» машины М-3000. И Петровский мне сказал: «Занимайтесь этой машиной, потому что на Мехмате должна быть машина». Это была его такая концептуальная мысль. А ребята из Лаборатории на меня накинулись: «Ты что? Такие деньги грошовые! Мы тут боремся за зарплату!»

В.Б.Д.: А ты легко соглашаешься...

В.Б.Б.: На копейки.

Такая диковинная история была.

В.Б.Д.: И тогда же мы встретились снова на Мехмате. Я имею в виду, снова – это после комсомольской работы.

В.Б.Б.: Да.

В.Б.Д.: Вспоминая 1970-й год, напомним, что как раз в том году в МГУ был создан факультет *вычислительной математики и кибернетики* (ВМиК). Практически все мехматовские *вычислители* и *программисты* перешли на этот новый факультет. Поэтому возникла необходимость кому-то на Мехмате читать курсы *вычислительной математики* и *программирования*. Было решено передать чтение этих курсов нашей кафедре – кафедре *общих проблем управления* (ОПУ). В результате на кафедре ОПУ появились, в частности, три специалиста по программированию: Бонгард Михаил Моисеевич, Адельсон-Вельский Георгий Максимович и Штаркман Всеволод Серафимович. К сожалению, все они уже покойники.

Вы с ними, видимо, сразу же тогда познакомились. Лёгкое ли было для Вас общение с ними? Или для Вас это были «разные» люди?

В.Б.Б.: Разные, конечно.

С Адельсоном-Вельским я был связан, ещё когда учился, по «кронродовским» спецкурсу и спецсеминару, но более детально с ним не работал.

С Бонгардом та же история.

А вот Штаркман был тем человеком, который, собственно, и определил мою дальнейшую жизнь. Так сказать, «программистскую». Потому что именно он поставил на Мехмате МГУ и оригинальный курс программирования, и практическую работу на АСВТ М 3000. Потом он стал моим научным руководителем по кандидатской диссертации.

В.Б.Д.: Про это я, позже, Вам соответствующий вопрос ещё задам.

Ну, и тогда же Вы сразу начали на Мехмате преподавать?

В.Б.Б.: Да, я тогда сразу начал, в группе, вести занятия по программированию.

В.Б.Д.: Иногда, грубо говоря, «под сухую», без машины, учили студентов АЛГОЛу? Или, всё-таки, с выходом на машину?

В.Б.Б.: Не-не-не. Был Ассемблер, и работа была «на Ассемблере». Всё было «по настоящему».

Собственно говоря, тогда у меня и появились первые студенты, которые, в последствии, у меня писали свои «курсовые»...

В.Б.Д.: В связи с этим следующий вопрос у меня таков.

К середине 1970-х годов, в ходе развития научно-исследовательской деятельности в Лаборатории, обусловленной появлением на

Мехмате МГУ новой электронно-вычислительной техники с большим количеством терминальных устройств, вокруг Вас сложилась неформальная группа, из творчески увлеченной системным программированием, молодежи (там были и сотрудники, и аспиранты, и студенты), где обсуждались новейшие результаты в данной области. Наиболее активными студентами в этой группе, мне помнится, были Гена Лебедев, Гена Маслов, Владимир Груздев, Алексей Веденов, Михаил Исаев (этого я хуже помню), Виктор Левитин (тоже хуже помню). Видимо, эти фамилии справедливо, в истории Мехмата МГУ, напомним. В результате деятельности этой группы, пользующейся поддержкой Александра Васильевича Михалёва, в 1979-ом году уже возглавившего Лабораторию ...

В.Б.Б.: Тут, Василий Борисович, надо разделить перечисленные фамилии *по периодам*.

Витя Левитин, а ещё Саша Смирнов, Коля Осадченко и Светлана Базеева - это первая команда, активно функционирующая до середины 1970-ых годов ... Это – *АСВТ М-3000* и потом *ЕС*. Напомню, что аббревиатура «АСВТ» означает сокращённое название появившейся в стране «Агрегатной системы средств вычислительной техники», предназначенной для построения управляющих вычислительных комплексов при автоматическом управлении производством.

В.Б.Д.: Значит Лебедев, Маслов, Веденов и Груздев – это позже?

В.Б.Б.: Да, это «костяк» более поздней команды ... Примерно 1979-1980 годов. И связано это уже было с *автоматической обработкой студенческих практикумов*. Мы, действительно, сотворили тогда, своего рода, «ренессанс» в указанном направлении. Способствовало же этому то, что тогда у меня «завязались» отношения с *Советом по автоматизации научных исследований* при АН СССР.

Учёным секретарём *Совета* был Александр Петрович Иванов. Вот с его помощью у нас и возникли творческие отношения с целой (опять же, *неформальной*) группой специалистов из Академии наук, занимавшихся автоматизацией научных исследований. Естественно, и программированием тоже... А главное, Александр Петрович *своей властью* (он, кстати, потом за это пострадал) выделил мне четыре машины (точнее, выдал мне «четыре фонда на ИВК», то есть, на «*Информационно-вычислительные комплексы*») для Мехмата МГУ: *две СМ-3, и две СМ-4*. Правда, под давлением Андрея Николаевича Тихонова, две машины от Мехмата ректорат отнял, но обе СМ-4 нам удалось у себя «отстоять». В этом, конечно, большая заслуга Александра Васильевича Михалёва – к тому времени уже он возглавлял нашу Лабораторию.

В.Б.Д.: Кто возглавлял этот Совет по автоматизации?

В.Б.Б.: Председателем там был Дородницын. Его заместителем - Куклин. Но Александр Петрович

Иванов, являясь Учёным секретарём Совета, фактически вёл «все дела» самостоятельно.

Александр Петрович пришёл в Совет из Института космических исследований. Хотя должность была, конечно, чиновничья, но он очень *правильным* был. И очень многим помогал, не только нам....

Когда я пришёл с этими четырьмя «бумагами», *в университете не было ни одной СМ, ни одной!* А это, на самом деле, были не просто СМ, это были *ИБК!* То есть, эти машины были снабжены *блоками связи с внешними приборами* ... Они были, как бы, «нацелены» на автоматизацию. Вот и возник скандал: что, мол, на Мехмате нужно автоматизировать, ведь никаких приборов там нет. И от Мехмата забрали две машины: одну отдали в Институт механики МГУ, а другую - на Физфак МГУ.

А суть-то была вот в чём: такие машины нам были нужны, поскольку мы, к тому времени, уже поняли, как из них сделать «многотерминалки»! И на их основе, вскоре, мы сделали на Мехмате МГУ первый *дисплейный класс*.

В.Б.Д.: Понятно.

В.Б.Б.: И сделали мы это с нашей программой... Вот, собственно, в её разработке и участвовали Лебедев, Маслов, Веденов и Груздев.

В.Б.Д.: То есть, первая группа – это Левитин, Смирнов, Базаева и Осадченко. Вторая – Лебедев, Маслов, Веденов и Груздев. А Исаева к какой группе отнести?

В.Б.Б.: Он, как бы, из промежуточной группы, занимавшейся развитием деятельности первой группы.

В.Б.Д.: Значит, он «ближе» к «лебедевцам»?

В.Б.Б.: Да, можно и так сказать.

Левитин потом распределился в Пущино. В НИИВЦ, где директором был Альберт Макарьевич Молчанов... «По следам Левитина», туда, также, распределился Исаев и кто-то ещё - я уже не помню. Веденов, Маслов и Груздев, как москвичей, взяли на работу в университетские лаборатории ...

В.Б.Д.: А Исаев?.

В.Б.Б.: И он тоже устроился в Москве.

В.Б.Д.: Но мы забыли упомянуть ещё молодого сотрудника Лаборатории - Лёшу Васильева. Где он сейчас?

В.Б.Б.: Лёша относится к первой группе - это АСВТ М-3000. Потом он перешёл на работу в НИИ Высшей школы.

В.Б.Д.: Так вот оно что - он с Мехмата ушёл в НИИ Высшей школы!

В.Б.Б.: С Мехмата МГУ он ушёл, потому что ему никак не давали должность *старшего научного сотрудника*.

В.Б.Д.: И вскоре он уехал из страны.

В.Б.Б.: Ну-у, это произошло позже, в 1990-ые годы ...

В.Б.Д.: Теперь вопрос такой: с кем-нибудь из вышеперечисленных учеников Вы продолжаете контактировать? Лебедев умер давно, Маслов – недавно, в прошлом году. Но остальные, насколько мне известно, ведь живы...

В.Б.Б.: Вы понимаете, тут дело вот в чём. Все они стали очень *сильными* специалистами. Но они привыкли работать лишь «в одиночку». И в этом их *слабость*.

В.Б.Д.: Ну, это типично для программистов: все они «волки-одиночки».

В.Б.Б.: Нет, есть разные программисты. У меня, например, был потом ученик Шура Мурашкин. Тоже личность выдающаяся, не менее, чем все они. Вот он спокойно может *работать в коллективе*. А они все поражены как бы одной болезнью - *индивидуализмом*. Они очень сильные, но не способные вокруг себя никого организовать. Да и сами в команде не могут работать ... То есть, они готовы в команде работать, но независимо и самостоятельно.

В.Б.Д.: И, наверное, делать *лишь то, что им интересно* – а не то, что нужно ...

В.Б.Б.: Точнее, *что они считали интересным, то считали и нужным*.

Вот мы, с большинством из них, и «разошлись» в каком-то смысле - они от меня как бы отделились. Но не со всеми.

Скажем, братья Юра и Алик Прилипко, условно принадлежащие *ко второй группе*, успешно работают в нашем Институте. Они из того набора студентов на кафедру ОПУ, который мы, с Анатолием Георгиевичем Кушниренко, набрали чуть ли не в количестве *сорока человек на двоих!* Помните такое?

В.Б.Д.: Конечно помню. Тогда на кафедру ОПУ было набрано около 80 человек - четыре группы! Из них две группы - «Ваши с Толей Кушниренко».

В.Б.Б.: Вот братья Прилипко, а также Мурашкин, собственно и были из того набора.

В.Б.Д.: Хорошо. Но пойдём дальше.

В 1975-м году Вы перешли на работу в ИПМ. Вернее, тогда ещё в ОПМ – лишь позднее «отделение» стало «институтом». Там Вы стали работать в отделе Штаркмана. И в 1981-м году Вы, под руководством Всеволода Серафимовича, защитили там свою кандидатскую диссертацию. Так?

В.Б.Б.: Нет, защитил не там. Ведь мои «защиты диссертации» всегда были «политическими событиями» (*смеются*).

В 1979-м году я вернулся на Мехмат МГУ. Причём, у меня кандидатская диссертация была уже практически готова. И встал вопрос: «Где защищаться?» А, поскольку, понятно, защита на Мехмате по программистской тематике было ...

В.Б.Д.: ... «дело тёмное» ...

В.Б.Б.: ... да-да... то возникли проблемы. И Лупанов, который мог бы представлять мою диссертацию на Совете «к защите», занял ... *осторожную позицию...*

В.Б.Д.: Деканом Мехмата МГУ тогда ещё был Алексей Иванович Кострикин.

В.Б.Б.: Именно так. Вот я и пришёл к Кострикину. Но и с ним ... «ситуация с возможностью защищаться мне на Мехмате МГУ не прояснилась» ... В общем, в конце концов, я защищался в Вычислительном центре.

В.Б.Д.: У Дородницына – в ВЦ АН СССР?

В.Б.Б.: Да-да.

В.Б.Д.: Лупанов содействовал Вашей защите?

В.Б.Б.: Нет. Просто специалисты меня там уже знали и мне посодействовали. Ну, собственно, что там ещё говорить - всё было пристойно и хорошо.

В.Б.Д.: И кто у Вас были оппоненты?

В.Б.Б.: Первым оппонентом был Михаил Романович Шура-Бура. Он «поругал» меня немного, но, в целом, «похвалил». Потому что, действительно, диссертация была не плохой ...

В.Б.Д.: А вторым оппонентом кто был?

В.Б.Б.: Вот второго оппонента я уже не вспомню.

В.Б.Д.: Ладно.

Тогда же, в 1979-м году или чуть позже, у Вас наладилось творческое сотрудничество с ЗИЛом, для которого потребовалось решать практические задачи. Как я понял из Книги, ведущую роль в этом сотрудничестве играл Александр Ильич Ставицкий, возглавлявший тогда в ЗИЛе *бюро инженерных расчётов*, а в «новые времена» занявшийся самостоятельным бизнесом.

Это сотрудничество многому Вас научило? И к Ставицкому у Вас благодарность есть?

В.Б.Б.: Безусловно, есть. К Ставицкому, и, я бы сказал, вообще к ЗИЛу. Потому что это сотрудничество приучило меня к тому, чего, я считаю, не хватает сейчас многим учёным – к реальному

участию в серийном производстве. Понимаете? Одно дело – сделать *один опытный образец*, который стоит и работает, и абсолютно другое дело – сделать так, чтобы этот образец стал *серийным*, выпускался бы *стабильно* и показывал бы бесперебойную *работу* ...

Вот это, на мой взгляд, самое важное для производства ...

В.Б.Д.: Ставицкий это чётко понимал?

В.Б.Б.: Ну, не только Ставицкий ... На самом деле, «движителем» нашего сотрудничества с ЗИЛом был, прежде всего, зам. генерального директора по экономике Бужинский Александр Иванович, совершенно выдающаяся личность. Про него тоже можно посмотреть в нашей Книге. Можно вспомнить ещё о Вениамине Давидовиче Кальнере, зам. главного инженера по науке ...

Но главным, всё-таки, был Бужинский.

В.Б.Д.: Картина восстанавливается, как всё происходило...

В.Б.Б.: Ставицкий совсем недавно умер, буквально в июне.

В.Б.Д.: Да, я уже знаю. При этом он умер... в Австрии, кажется.

В.Б.Б.: Нет, в Германии.

В.Б.Д.: В 1986-м году Вы защитили свою докторскую диссертацию. В том же году был основан *Институт автоматизации проектирования*, директором которого был назначен Олег Михайлович Белоцерковский - в прошлом году он тоже умер. Вы тогда стали его заместителем. Там, под Вашим руководством, были созданы и внедрены на производственном объединении ЗИЛ основные программные компоненты для машиностроительной *системы автоматизированного проектирования* (сокращённо, для САПРа, и принято эту аббревиатуру считать *мужского рода*), а также для семейства рабочих станций «Беста» (как я понимаю, последнее название происходит от слияния двух фамилий: «Бе» – Бетелин, «Ста» – Ставицкий).

В.Б.Б.: Правильно.

В.Б.Д.: Эти разработки продолжают функционировать и модернизироваться? Или они уже забыты?

В.Б.Б.: Ну, понятно ведь, что в 1991-м году «многое кончилось» ... Скажем, ЗИЛ уже практически «умирает», и смотреть на это очень грустно ... Для меня это просто *непрерывная душевная боль* глядеть на то, что там сейчас происходит. И если говорить про САПР, для которого мы тогда всё делали, то да, конечно, тут многое устарело.

В.Б.Д.: Кстати, САПРы сейчас уже не модны? Или теперь просто «не говорят так»? Раньше, я помню,

чуть ли не везде слышалось: «САПР, САПР, САПР ...».

В.Б.Б.: Тогда это, действительно, было «передним краем» производственной науки. А сейчас это обычная история, норма жизни, повседневная действительность.

В.Б.Д.: То есть, на любом нормальном производстве имеется свой САПР.

В.Б.Б.: Конечно... А тогда всё это мы только начинали.

Но вот что касается «Бест», то они безотказно работают до сих пор! Ведь одной из основных наших задач было сделать такие станции *надёжными*.

В.Б.Д.: То есть, и через тридцать лет такая станция должна работать?

В.Б.Б.: Ну, не тридцать, а хотя бы двадцать пять лет. И у нас такие станции есть.

Скажем, в нашем Институте стоят две такие станции, и они работают до сих пор. В Челябинске, в администрации города, вот уже 22 года также работает наша «Беста» - год назад к нам приезжал человек, который на ней работает, с просьбой лишь заменить на ней один диск. Есть и другие примеры.

В.Б.Д.: Замечательно.

В.Б.Б.: А главное, и с точки зрения электроники, и с точки зрения программирования, разработка такой станции позволила нам перейти *на следующий уровень*. Вот это важно! То есть, всё, что мы делали *потом*, было уже *тогда* заложено и отработано.

В.Б.Д.: То есть, все идеи *надёжности* и *серийности* были уже тогда предусмотрены и реализованы?

В.Б.Б.: Да. И в этом смысле я счастливый человек. Я никогда ничего «не бросал» и, как бы, каждый предыдущий шаг ...

В.Б.Д.: ... был необходимой ступенькой к шагу следующему

В.Б.Б.: Точнее он был *основой* для следующего шага.

В.Б.Д.: Ясно. Но идём дальше.

В 1989-м году по предложению, по-моему, Евгения Павловича Велихова, был создан НИИСИ РАН, и директором этого института именно Евгений Павлович рекомендовал сделать Вас. Ваша кандидатура была поддержана, также, Израилем Моисеевичем Гельфандом. Правильно я понимаю ситуацию?

В.Б.Б.: Не совсем точно ... Дело в том, что НИИСИ, как Институт, уже был. Он был создан ещё в

1986-м году. Инициатором его создания являлся Рональд Зиннурович Сагдеев, который хотел заниматься *высокопроизводительными вычислениями*. Но в 1989-м году у него жизненные планы изменились, и он уехал в США. А «вывеска» у Института осталась. Вот тогда Евгений Павлович обратился ко мне – ведь я уже руководил в нашем Институте огромным отделением (более сотни человек), которое *конкретно* занималось этими делами.

В.Б.Д.: То есть, он предложил: «Возьмите Вы в свои руки этот Институт».

В.Б.Б.: В общем, да.

А Израиль Моисеевич меня *поддержал*, потому что к тому времени мы уже довольно активно сотрудничали. Его семинар я посещал и в 1987-м, и в 1988-ом, и в 1989-м году. И когда я стал директором Института, то Израиль Моисеевич, из ИПМ со всем своим отделом, просто перешёл к нам в Институт.

В.Б.Д.: Ваше сотрудничество с Израилем Моисеевичем было *по направлению медицинской диагностики*?

В.Б.Б.: По всем направлениям, чем он занимался ... Сам и весь его отдел ...

А его *рекомендация* насчёт меня была по другому поводу. Это было уже в 1997-м году: он написал письмо президенту РАН Юрию Сергеевичу Осипову, рекомендуя меня избрать членом-корреспондентом РАН ...

В.Б.Д.: Об избрании Вас в РАН у меня, чуть позже, будет отдельный вопрос.

В Институте был проведен большой цикл исследований, не буду всё перечислять. Но задам общий вопрос: более или менее, *всё из намеченного было реализовано* в коллективе за 25 лет? Мне очень понравилось, что, по Вашим словам, Вы *не любите бросать что-то недоделанное, а стараетесь довести всё до конца*. Так вот, по крупному, *всё задуманное было доделано до конца*?

В.Б.Б.: В общем, да. То, что можно сказать без особых деталей, заключается в следующем: мы, как и обещали, взялись за создание отечественной серии машин и сделали это. Причём, нашли комплексное решение поставленной задачи, то есть, для этой серии машин, сделали и технологическую базу, и микроэлектронную. Говоря в целом, мы создали *серийное* производство машин, обеспечивающих, на данный момент, объём товарной продукции порядка 20-ти миллиардов рублей.

В.Б.Д.: В нынешних ценах?

В.Б.Б.: Да.

В.Б.Д.: Неплохо!

В.Б.Б.: Это суммарно ... Это и есть подтверждение того, что *всё не зря* ...

В.Б.Д.: Да-да, я понял.

Теперь вот о чём. Не секрет, что в трудные времена, после крушения СССР, финансово выжить НИИСИ во многом помогло *сотрудничество с ВПК*. При этом большую поддержку такому сотрудничеству Институту оказал генерал-полковник Анатолий Петрович Ситнёв, как я понимаю. Он жив, слава Богу. Вы согласны с мнением, что без его поддержки Институту было бы трудно выжить?

В.Б.Б.: Ну конечно! Но тут совпало, что для самого Ситнёва наш Институт ... стал удобным инструментом, чтобы решать те проблемы, которые он видел.

В.Б.Д.: То есть, он не просто поддерживал, а ещё и направлял Институт на разработку нужных ему направлений.

В.Б.Б.: Именно так. Не то что он просто брал всё то, что мы сделали ...

В.Б.Д.: ... Иными словами, не то, что он «абстрактно поддерживал» Институт ...

В.Б.Б.: ... а поддерживал лишь разработку того, что он считал «нужным» ... И у нас была своя программа, намеченная на перспективу дальнейшего развития наших разработок.

Кстати, мы с ним до сих пор поддерживаем отношения. Он, всё-таки, был очень грамотным генералом ...

В.Б.Д.: Понятно.

Но ведь не только «на военные дела» был ориентирован НИИСИ. Как я уже понял, в Институте поддерживались, например, новаторские разработки, относящиеся к вопросам математического моделирования в области медицины. В частности, проводились серьёзные исследования, посвященные прогнозированию течения инфаркта миокарда и его осложнений, инициированные Израилем Моисеевичем Гельфандом и известным кардиологом Абрамом Львовичем Сыркиным, тоже, слава Богу, живым.

Медицинская тематика по-прежнему актуальна для НИИСИ? Есть люди, которые и сейчас ею занимаются в Институте?

В.Б.Б.: Конечно. А с Сыркиным я встречался буквально сегодня.

В.Б.Д.: Он бодр?

В.Б.Б.: Да, бодр, всё замечательно.

Сейчас мы, с его сотрудниками, занимаемся, так называемой *неинвазивной диагностикой*. То есть, диагностикой болезни по анализу выдыхаемого воздуха. И, действительно, совместно с ними, мы получили очень интересные результаты ... В связи с этим нам присылаются приглашения на международные конференции. Так что эта тема даже стала одной из приоритетной для Института.

В.Б.Д.: Хорошо: репутация Института поддерживается на международном уровне.

В.Б.Б.: Во всех этих исследованиях возникает *проблема ранней диагностики*.

В.Б.Д.: Даже я, далёкий от медицины, понимаю важность *ранней диагностики*..

Но продолжим. С 1997-го года Вы ещё стали директором *Института микротехнологий* Российского научного центра «Курчатовский институт», а с 2005 года – вице-директором всего этого РНЦ.

Эта организаторская деятельность, дело, безусловно, очень важное, сохраняется за Вами и поныне?

В.Б.Б.: Только не вице-директором, а *вице-президентом по информационным технологиям* там я был, да. Сейчас я вице-президентом там уже не являюсь.

Вся моя деятельность в РНЦ «Курчатовский институт» была связана только с «микроэлектронной мини-фабрикой». Мы арендовали там, для неё, площади, на которых расположили необходимое нам оборудование, и на нём работали наши сотрудники.

В.Б.Д.: Это осталось и по сей день?

В.Б.Б.: Осталось, да-да-да. Только наше «тамошнее» *Отделение микротехнологий*. теперь называется *Институтом микротехнологий* (кратко, ИМТ).

В.Б.Д.: И Вы являетесь директором ИМТ?

В.Б.Б.: Да, находящегося в составе РНЦ «Курчатовский институт».

Институт небольшой, человек тридцать. А по существу он состоит из сотрудников *Отделения микротехнологий НИИСИ РАН*: здесь они на ставке, а там они на полставки, или, наоборот, там они на ставке, а у нас на полставки. И научная тема у них одна ...

В.Б.Д.: Понятно.

Теперь про академию. В 1997-м году Вы были избраны членом-корреспондентом РАН, в 2003-м году – академиком РАН, по *Отделению нанотехнологий и информационных технологий* (сокращённо – ОНИТ), академиком-секретарем которого является Евгений Павлович Велихов. При этом, Вы член бюро ОНИТ и входите в *Секцию информационных технологий и автоматизации ОНИТ*. Кроме того, в РАН Вы возглавляете *Совет по координации научных исследований по направлению «Стратегические информационные технологии»*.

Так вот, в указанном Совете, как я понимаю, рассматриваются, в частности, вопросы *создания суперкомпьютеров и разработки для них программного обеспечения*. И Учёным секретарем Совета является Алексей Борисович Жижченко.

С Жижченко сотрудничество остаётся?

В.Б.Б.: Сотрудничество остаётся, но уже не в той форме. Ведь сейчас пересматриваются все Советы, меняется вся их структура. И в этом смысле *устанавливается*, скажем так, *более тесное их сотрудничество...* Ведь НИИСИ, как Вы знаете, теперь – Федеральный научный центр. В состав этого Центра входит и *Межведомственный суперкомпьютерный центр Российской Академии наук* (МСЦ РАН) целиком, где Жижченко является *заместителем директора*.

Так что произошло наше *организационно устроенное* единение.

В.Б.Д.: Ясно.

Кстати, в связи с Вашей огромной нагрузкой вопрос: как Вы расслабляетесь? Спорт? Теннис? Пешие прогулки? Плавание?

В.Б.Б.: Многие вещи мне, по здоровью, противопоказаны. Остаются, пожалуй, лишь пешие прогулки – и то когда есть возможность ...

В.Б.Д.: Значит, пешком Вы тоже ходите, не только на машине ездите?

В.Б.Б.: Хожу-хожу.

В.Б.Д.: С 2010 года Вы – член Консультативного научного совета Фонда «Сколково». Немножко об этой деятельности можете сказать? Или это всё ещё «в зачаточном состоянии»?

В.Б.Б.: Дело вот в чём. «Сколково» нацелено на так называемую «посевную технологию», то есть, на малые компании. Основная задача «Сколково» – создавать малые компании, которые КОГДА-НИБУДЬ, ГДЕ-НИБУДЬ, ЧТО-НИБУДЬ да сделают.

В.Б.Д.: Сделают какой-нибудь «прорыв», да?

В.Б.Б.: Да. Так что, парадигма такая: они там *работают на будущее*, возможно даже *на удалённое будущее*. Поэтому там я, являясь членом Совета, стараюсь *отслеживать* научные перспективы ...

В.Б.Д.: Это же хорошо для нашего Института – входить в такого сорта Советы. Хотя бы для того, чтобы *быть в курсе*, в каком русле развивается научно-техническая мысль ...

В.Б.Б.: Ну, да, следить за тем, что происходит. И я прекрасно понимаю, что происходит.

В.Б.Д.: И последний мой традиционный вопрос. Его я задаю всем своим собеседникам в конце Интервью, хотя, честно говоря, он мне уже немножко наскучил.

Довольны ли Вы своей судьбой и ничего не хотели бы в ней изменить? Вот один профессор математики сказал мне, что, быть может, лучше было бы, если бы он стал физиком ...

В.Б.Б.: Понимаете, я, действительно, доволен своей судьбой. Ведь, вообще-то говоря, по духу, по внутреннему устройству, я и не физик, и не математик, а инженер.

В.Б.Д.: Я видел Вас в одной телевизионной передаче с Алексеем Львовичем Семёновым. Там Вы говорили, что в нашей стране есть математические школы, но пора создавать и *инженерные школы*.

Мне эта мысль понравилась: ведь не все умные школьники хотят только доказывать теоремы ...

В.Б.Б.: Да-да! Хотя теоремы-то доказывать, конечно, полезно, но сейчас нам нужно много хороших инженеров! ...

В юности, в детстве, я зачитывался книгой «Будни инженера Бережкова». Это, по существу, художественный вариант биографии Люльки.

В.Б.Д.: Академика Люльки?

В.Б.Б.: Да-да, . моторостроителя.

Вот это для меня!.. И, в общем и целом, я этим и занимаюсь. Понимаете?

Хотя я считаю, что ещё не всё сделал, чем мы *сейчас* занимаемся. Это - проблемы «нефтянки», одна из ключевых проблем нашей страны.

В.Б.Д.: Этим, в частности, объясняются Ваши регулярные поездки в Сургут?

В.Б.Б.: В общем, да ...

И тут «всё должно быть в комплексе»: и образовательная деятельность, и производственная деятельность и многое другое. Ведь по опыту ЗИЛа я хорошо понимаю, что такое *новые технологии* ... ЗИЛ - это была такая ШКОЛА!

В частности я уже тогда понял, что нельзя что-то «кусочно» автоматизировать, в одном месте, а надо смотреть «всё вместе». Но «всё вместе» – это «целая история», с большим риском для производителей. И понятно, что если *лишь в одном месте* что-то «не так» изменишь, то это сразу же *отразится везде* ...

А по тем временам, я помню, производственный контроль был жесточайшим. Это же КОНВЕЙЕР! Остановка его на *три* минуты – *ч.п.* для *главного инженера*, на *пять* минут – для *директора*, а на *больше* – уже для *министерства*. Вот как было жёстко выстроена эта система!

В.Б.Д.: В советское время?

В.Б.Б.: Да, в советское время.

Но сейчас ЗИЛа нет, и остаётся лишь вспоминать об этом. Сейчас *новые технологии* нужны для *нефтяных компаний*.

А их «конвейер» - это «баррели–баррели–баррели». И не дай Бог, если что-то «в этом конвейере остановится» ...

В.Б.Д.: Естественно – возникает «утерянная прибыль»!.

В.Б.Б.: Ну, конечно! Поэтому, понимаете, я благодарен судьбе, что у меня была «ЗИЛовская школа».

Таков мой ответ на Ваш последний вопрос.

В.Б.Д.: Понял Ваш ответ, Владимир Борисович. И я рад нашей беседе.

В.Б.Б.: Взаимно, Василий Борисович ...

СПИСОК УПОМЯНУТЫХ ПЕРСОНАЖЕЙ

АДЕЛЬСОН-ВЕЛЬСКИЙ Георгий Максимович (1922-2014) – математик и прикладной математик (Россия, Израиль), преподавал программирование на кафедре общих проблем управления Мехмата МГУ
 БАЗАЕВА Светлана Евгеньевна (р. 1954) – прикладной математик и программист
 БЕТЕЛИН Владимир Борисович (р. 1946) – прикладной математик и программист, академик РАН, был сотрудником Лаборатории вычислительных методов Мехмата МГУ
 БЕТЕЛИН Борис Алексеевич (1918-2001) – инженер-конструктор
 БЕТЕЛИНА (урожд. Козлова) Муза Владимировна (р. 1921) – врач-стоматолог
 БЕТЕЛИН Алексей Борисович (р. 1960) – прикладной математик и программист
 БЕЛОЦЕРКОВСКИЙ Олег Михайлович (1925-2015) – математик, прикладной математик и механик, академик АН СССР
 БОНГАРД Михаил Моисеевич (1924-1971) – прикладной математик и программист, преподавал на кафедре общих проблем управления Мехмата МГУ
 БУЖИНСКИЙ Александр Иванович (р. 1921) – экономист
 ВАЙНШТЕЙН Исаак Аронович (1917-2008) – математик (Россия, Израиль), был доцентом кафедры математического анализа Мехмата МГУ
 ВАСИЛЬЕВ Алексей Вадимович (р. 1945) – прикладной математик и программист (Россия, США), был сотрудником Лаборатории вычислительных методов Мехмата МГУ
 ВЕДЕНОВ Алексей Александрович (р. 1958) – прикладной математик и программист
 ВЕЛИХОВ Евгений Павлович (р. 1935) – физик, академик АН СССР
 ГЕЛЬФАНД Израиль Моисеевич (1913-2009) – математик (Россия, США), академик РАН, преподавал на кафедре теории функций и функционального анализа Мехмата МГУ
 ГЕНДУГОВ Владимир Михайлович (1945-2016) – механик, ведущий научный сотрудник кафедры газовой и волновой динамики Мехмата МГУ
 ГРУЗДЕВ Владимир Алексеевич (р. 1958) – прикладной математик и программист
 ДОРОДНИЦИН Анатолий Алексеевич (1910-1994) – прикладной математик, механик и геофизик, академик АН СССР
 ДЬЯЧЕНКО Владимир Федотович (р. 1929) – прикладной математик, профессор кафедры общих проблем управления Мехмата МГУ, в 1971-1973 годы был заведующим Лабораторией вычислительных методов Мехмата МГУ
 ЖИЖЧЕНКО Алексей Борисович (р. 1934) – прикладной математик, специалист в области информационных технологий, академик РАН

ИВАНОВ Александр Борисович (р. 1940) – прикладной математик
 ИЛИАДИС Ставрос Дмитриевич /Stavros Iliadis/ (р. 1941) – математик (Россия, Греция), профессор кафедры общей топологии и геометрии Мехмата МГУ
 ИСАЕВ Михаил Авенкиевич (р. 1957) – прикладной математик и программист
 КАЛЬНЕР Вениамин Давидович (р. 1938) – инженер и эколог, доктор технических наук, специалист по «издательскому производству»
 КОСТРИКИН Алексей Иванович (1929-2000) – математик, член-корреспондент АН СССР, заведующий кафедрой высшей алгебры Мехмата МГУ (1972-2000), декан Мехмата МГУ (1977-1980)
 КРОНРОД Александр Семёнович (1921-1986) – математик и прикладной математик, профессор, преподавал на кафедре вычислительной математики Мехмата МГУ
 КРОНРОД Дмитрий Александрович (р. 1943) – математик
 КУКЛИН Герман Николаевич (1934-2008) – прикладной математик
 КУНИН Пётр Ефимович (1921-1976) – прикладной математик и биофизик, доктор физико-математических наук
 КУШНИРЕНКО Анатолий Георгиевич (р. 1944) – математик и прикладной математик, преподавал на кафедре общих проблем управления Мехмата МГУ
 ЛЕБЕДЕВ Геннадий Викторович (1957-2004) – прикладной математик и программист
 ЛЕВИТИН Виктор Валимович (р. 1954) – прикладной математик и программист
 ЛУПАНОВ Олег Борисович (1932-2006) – математик, академик РАН, организатор и первый заведующий кафедрой дискретной математики Мехмата МГУ (1981-2006), декан Мехмата МГУ (1980-2006)
 ЛЮЛЬКА Архип Михайлович (1908-1984) – конструктор авиационных двигателей, академик АН СССР
 МАРКОВ Андрей Андреевич (1903-1979) – математик, член-корреспондент АН СССР, организатор и первый заведующий кафедрой математической логики Мехмата МГУ (1959-1979)
 МАТИН Уильям /William Martin/ (1931-1987) – американский криптолог, сотрудник Агентства национальной безопасности США (вместе со своим коллегой, Берноном Митчеллом /Bernon Mitchell/ (1929-2001), в 1960-м году /через Кубу/ перебежал в СССР)
 МАСЛОВ Геннадий Вениаминович (1957-2015) – прикладной математик и программист
 МИХАЛЁВ Александр Васильевич (р. 1940) – математик, профессор кафедры высшей алгебры Мехмата МГУ, заведующий (с 1979 года) Лабораторией вычислительных методов Мехмата МГУ, основатель и заведующий кафедрой

теоретической информатики Мехмата МГУ (с 2013 года)

МОЛЧАНОВ Альберт Макарьевич (1928-2011) – математик и прикладной математик

МУРАШКИН Александр Иванович (р. 1960) – прикладной математик и программист (Россия, США)

ОСАДЧЕНКО Николай Владимирович (р. 1955) – прикладной математик, механик и программист

ОСИПОВ Юрий Сергеевич (р. 1936) – математик и механик, академик РАН, президент РАН (1991-2013), заведующий кафедрой оптимального управления факультета ВМиК МГУ (с 1990 года)

ПЕТРОВСКИЙ Иван Георгиевич (1901-1973) – математик, академик АН СССР, заведующий кафедрой дифференциальных уравнений Мехмата МГУ (1951-1973), декан Мехмата МГУ (1939-1944), ректор МГУ (1951-1973)

ПИЛЮГИН Николай Алексеевич (1908-1982) – конструктор, специалист в области автоматического управления ракетами и ракетно-космическими комплексами, академик АН СССР

ПРИЛИПКО Алексей Алексеевич (р. 1960) – прикладной математик и программист

ПРИЛИПКО Георгий Алексеевич (р. 1960) – прикладной математик и программист

САГДЕЕВ Роальд Зиннурович (р. 1932) – физик (Россия, США), академик АН СССР

СЕМЁНОВ Алексей Львович (р. 1950) – математик и методист математики, академик РАН, преподавал на кафедре математической логики Мехмата МГУ

СИЛАЕВ Дмитрий Алексеевич (1945-2015) – математик и прикладной математик, доцент кафедры общих проблем управления Мехмата МГУ

СИТНОВ Анатолий Петрович (р. 1944) – генерал-полковник, специалист по проблемам военно-технической политики и развития вооружения в Российской Федерации

СМИРНОВ Александр Сергеевич (р. 1954) – прикладной математик и программист

СТАВИЦКИЙ Александр Ильич (1938-2016) – конструктор и бизнесмен

СЫРКИН Абрам Львович (р. 1930) – медик-кардиолог, доктор медицинских наук

ТИХОНОВ Андрей Николаевич (1906-1993) –

математик, академик АН СССР, преподавал на кафедре дифференциальных уравнений Мехмата МГУ, заведующий кафедрой математики Физфака МГУ (1938-1970), заведующий кафедрой вычислительной математики Мехмата МГУ (1960-1970), организатор и первый декан факультета ВМиК МГУ (1970-1990), организатор и первый заведующий кафедрой вычислительной математики факультета ВМиК МГУ (1970-1981), а потом и кафедры математической физики факультета ВМиК МГУ (1981-1993)

ТРИФОНОВ Николай Павлович (р. 1925) – прикладной математик и программист, профессор, заведовал кафедрой алгоритмических языков факультета ВМиК МГУ (1971-1993), до перехода на факультет ВМиК МГУ был доцентом кафедры вычислительной математики Мехмата МГУ

УСПЕНСКИЙ Владимир Андреевич (р. 1930) – математик, заведующий кафедрой математической логики и теории алгоритмов Мехмата МГУ (с 1993 года)

ФАЙНЛЕЙБ Абрам Александрович (1905-1980) – инженер, полковник, возглавлял, в 1969-ом году созданную на Мехмате МГУ, Лабораторию по проблемам управления, преобразованную, в 1970-м году, в Лабораторию вычислительных методов, и руководил этой Лабораторией до 1971 года

ФАРАДЖЕВ Игорь Александрович (р. 1939) – прикладной математик

ШКАДОВ Виктор Яковлевич (р. 1935) – механик, профессор кафедры аэромеханики и газовой динамики Мехмата МГУ, в 1973-1978 годы был заведующим Лабораторией вычислительных методов Мехмата МГУ

ШЛЫКОВА Татьяна Максимовна (? - ?) – школьная учительница математики

ШТАРКМАН Всеволод Серафимович (1931-2005) – прикладной математик и программист, преподавал на кафедре общих проблем управления Мехмата МГУ

ШУРА-БУРА Михаил Романович (1918-2008) – математик, прикладной математик и программист, до перехода на факультет ВМиК МГУ был профессором кафедры вычислительной математики Мехмата МГУ, организатор и первый заведующий кафедрой системного программирования факультета ВМиК МГУ (1970-1993)

On the 70th anniversary of Vladimir Borisovich Betelin

V.B.Demidovich

Abstract: It's the Interview of the author with the supervisor of the Federal State Institution «Federal Scientific Center Scientific Research Institute for System Analysis of the Russian Academy of Sciences» (FSI FSC SRISA RAS), Academician, Vladimir Borisovich Betelin on the occasion of his 70th birthday.

Keywords: applied Mathematics, system programming, MSU, Mech-Math faculty, SRISA RAS