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

ТРУДЫ НИИСИ РАН

ТОМ 9 № 1

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ СЛОЖНЫХ СИСТЕМ:

ТЕОРЕТИЧЕСКИЕ И ПРИКЛАДНЫЕ АСПЕКТЫ

МОСКВА
2019

Тематика номера:
Моделирование физических процессов, математическое моделирование и визуализация, вопросы программирования, математические исследования и вопросы численного анализа.

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 наноэлектроника, математические исследования и вопросы численного анализа, история науки и техники.

The topic of the issue:
Modelling of physical processes, mathematical modeling and visualization, Programming research, Mathematical studies and Numerical analysis problems.

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, mathematical researches and 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. МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ

II. МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ВИЗУАЛИЗАЦИЯ

III. ВОПРОСЫ ПРОГРАММИРОВАНИЯ

IV. МАТЕМАТИЧЕСКИЕ ИССЛЕДОВАНИЯ И ВОПРОСЫ ЧИСЛЕННОГО АНАЛИЗА

Повышение точности гидродинамических моделей и контроль разработки нефтяных месторождений по данным гидропрослушивания

И.В. Афанаскин[1], А.В. Королев[2], П.В. Крыганов[3]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия

E-mail's: [1]ivan@afanaskin.ru, [2]alexandre.korolev@mail.ru, [3]kryganov@mail.ru

Аннотация: Рассмотрены подходы к повышению точности гидродинамических моделей и контроль разработки нефтяных месторождений по данным гидропрослушивания. Предлагается сравнивать определенные по результатам гидропрослушивания и рассчитанные на гидродинамической модели гидропроводность и пьезопроводность. Предложен подход к определению по результатам гидропрослушивания скоростей фильтрации жидкости, нефти и воды, а также фронта вытеснения.

Ключевые слова: гидропрослушивание, гидродинамическое моделирование, контроль разработки.

Введение

В современной практике разработки нефтяных месторождений активно применяется численное моделирование. Создаваемые гидродинамические модели позволяют решать множество задач, в том числе построение всевозможных карт, анализ, контроль и проектирование разработки, выбор скважин для проведения геолого-технических мероприятий (бурение боковых стволов, проведение ремонтно-изоляционных работ, гидроразрыв пласта и др.), обоснование бурения новых скважин и пр. Однако такие модели требуют большого количества исходных данных. Практически всегда при построении моделей специалисты сталкиваются с нехваткой данных, что часто приводит к невозможности достаточно точного для корректного решения промысловых задач восстановления строения межскважинных зон пласта. Поэтому гидродинамические модели адаптируют к истории разработки, т.е. подгоняют изначально заданные параметры модели для совмещения фактических и расчетных показателей разработки (дебитов флюидов, забойных и пластовых давлений и пр.). Кроме того, существует проблема разномасштабности исходных данных и проблема интеграции результатов исследований пластов и скважин в гидродинамическую модель. Поэтому проблема повышения точности гидродинамических моделей, которой посвящена эта статья, является актуальной. В настоящей работе приблизится к решению этой проблемы предлагается с использованием результатов гидропрослушивания. Кроме того, ниже будет рассмотрен оригинальный подход к контролю разработки нефтяных месторождений по данным гидропрослушивания.

1. Исследование межскважинного пространства методом гидропрослушивания

Метод гидропрослушивания является одним из немногих методов изучения межскважинного, а не околоскважинного пространства. Особенно важным является тот факт, что информацию о фильтрационно-емкостных параметрах и строении пласта в этом случае получают на основании анализа динамики давления в пласте и дебита скважин – параметров, непосредственно характеризующих процессы в пласте, а не по изучению косвенных полей (электрических магнитных, акустических) при геофизических исследованиях скважин или при сейсмических исследованиях.

В классическом варианте в процессе гидропрослушивания участвуют две скважины: возмущающая и реагирующая. В возмущающей скважине по специально разработанному алгоритму изменяют режим работы, а в реагирующей регистрируют изменение забойного давления, возникающее в результате такого изменения режима.

Желательно, чтобы реагирующая скважина простаивала, либо чтобы она длительное время стабильно работала с постоянным дебитом. В любом случае – забойное давление в реагирующей скважине на момент начала проведения исследования должно быть постоянным или плавно меняющимся по устоявшемуся, а потому известному по истории наблюдений закону. Современный подход к интерпретации результатов гидропрослушивания состоит в совмещении путем подбора параметров модели реальной кривой реагирования (забойного давления в реагирующей скважине) с расчетной кривой, полученной по формуле вида:

$$P_w(t) = P_0 - \frac{B}{4\pi\varepsilon}\sum_{j=1}^{N}\left[-(q_j - q_{j-1})Ei\left(-\frac{L^2}{4\chi(t-t_{j-1})}\right)\right], \quad (1)$$

где $P_w(t)$ - забойное давление в реагирующей скважине, P_0 - пластовое давление, B - объемный коэффициент пластовой жидкости, ε - гидропроводность (способность пласта пропускать жидкость), j - номер режима работы возмущающей скважины, N - количество режимов работы возмущающей скважины, q_j и q_{j-1} - дебит возмущающей скважины в поверхностных условиях на j-ом и j-1-ом режимах работы, L - расстояние между возмущающей и реагирующей скважинами, χ - пьезопроводность (характеризует скорость перераспределения давления в пласте), название введено В.Н. Щелкачевым по аналогии с температуропроводностью), t - время с начала исследования, t_{j-1} - время начала j-1-ого режима работы возмущающей скважины, $-Ei(-x)$ - интегральная показательная функция.

Такой подход к интерпретации называется методом наилучшего совмещения [5].

Формула (1) описывает поведение забойного давления для случая вертикальных скважин в однородном бесконечном пласте.

Интегральная показательная функция $-Ei(-x)$, использованная в уравнении (1), является табулированной функцией и представляет собой интеграл вида:

$$-Ei(-x) = \int_x^{\infty}\frac{\exp(-u)}{u}du. \quad (2)$$

Интегральная показательная функция может быть представлена в виде ряда:

$$-Ei(-x) = \ln\left(\frac{1}{x}\right) - 0{,}5772 + \sum_{n=1}^{\infty}\frac{(-1)^{n+1}}{nn!}x^n, \quad (3)$$

который сходится при всех значениях x: $0<x<\infty$ [1]. При изменении x от 0 до ∞ функция $-Ei(-x)$ быстро убывает от ∞ до 0 [1].

При интерпретации результатов гидропрослушивания и решении обратной задачи гидродинамики неизвестными в уравнении (1) являются параметры ε (гидропроводность) и χ (пьезопроводность). Совмещение расчетных и фактических значений давления осуществляется с помощью метода наименьших квадратов. Основным алгоритмом для решения данной задачи является алгоритм нелинейной регрессии. Наиболее часто используемым алгоритмом нелинейной регрессии в этом случае является алгоритм Левенберга-Марквардта, который в свою очередь является улучшенным методом Гаусса-Ньютона [6].

2. Приемы использования результатов гидропрослушивания для повышения точности гидродинамических моделей и контроля разработки нефтяных месторождений

Ниже приведены некоторые вопросы в рамках повышения точности гидродинамических моделей и контроля разработки нефтяных месторождений, которые можно решить с помощью результатов гидропрослушивания.

Первый вопрос - определение наличия или отсутствия гидродинамической связи между скважинами и контроль непроводящих тектонических нарушений по результатам гидропрослушивания для уточнения сообщаемости скважин в фильтрационной модели.

Второй вопрос - сравнение расчетной пьезопроводности по фильтрационной модели $\chi_{ФМ}$ и гидропроводности по результатам гидропрослушивания $\chi_{ГП}$.

По результатам гидропрослушивания пьезопроводность определяется либо напрямую, либо как [2-4]:

$$\chi_{ГП} = \frac{k}{m\mu C_t}, \quad (4)$$

где k - эффективная проницаемость по жидкости, m - пористость, μ - эффективная вязкость жидкости, C_t - суммарная сжимаемость. Эти параметры являются средними в межскважинном пространстве.

Проницаемость пласта может быть определена либо из пьезопроводности при известной пористости, либо из

гидропроводности при известной работающей толщине. Так как точность определения пористости больше, чем точность определения работающей толщины и поскольку пьезопроводность (в отличие от гидропроводности) не зависит напрямую от толщины пласта, ее рекомендуется использовать для контроля проницаемости фильтрационной модели.

Пьезопроводность блока фильтрационной модели определяется как:

$$\left(\chi_{\text{ФМ}}\right)_{ijk} = \left[\frac{k}{mC_t}\left(\frac{k_{ro}}{\mu_o}+\frac{k_{rw}}{\mu_w}\right)\right]_{ijk}, \quad (5)$$

где i,j,k - номера ячейки по осям x,y,z; k - абсолютная проницаемость ячейки; m - пористость ячейки; $C_t=C_r+C_oS_o+C_wS_w$; C_r - сжимаемость пласта; C_oC_w - сжимаемость нефти и воды; S_o,S_w - насыщенность по нефти и воде в ячейке; k_{ro},k_{rw} - текущая относительная фазовая проницаемость по нефти и воде в ячейке; μ_o,μ_w - вязкость нефти и воды.

По результатам гидропрослушивания определяется средние параметры в объеме пласта, охваченном исследованиями. Поскольку пьезопроводность не является мультипликативным параметром – для сравнения с результатом исследований в модели необходимо рассчитывать среднюю пьезопроводность в наборе ячеек между исследуемыми скважинами.

Третий вопрос - сравнение расчетной гидропроводности по фильтрационной модели $\varepsilon_{ФМ}$ и гидропроводности по результатам гидропрослушивания $\varepsilon_{ГП}$.

По результатам гидропрослушивания гидропроводность определяется либо напрямую, либо как [2-4]:

$$\varepsilon_{ГП} = \frac{kh}{\mu}, \quad (6)$$

где h - эффективная толщина пласта, μ - вязкость жидкости. Эти параметры являются средними в межскважинном пространстве.

Чтобы сравнивать гидропроводности по фильтрационной модели с результатами гидропрослушивания ее нужно рассчитывать для столбца ячеек:

$$\left(\varepsilon_{ФМ}\right)_{ij} = \sum_{k=1}^{N}\left[k\cdot DZ\cdot NTG\cdot\left(\frac{k_{ro}}{\mu_o}+\frac{k_{rw}}{\mu_w}\right)\right]_{ijk}, \quad (7)$$

где DZ - толщина ячейки; NTG - доля коллектора в ячейке. Имеет смысл рассчитывать гидропроводность не просто для столбца ячеек, а для какого-либо сечения между исследуемыми скважинами.

Поскольку проницаемость пласта уже была скорректирована с помощью пьезопроводности, с помощью гидропроводности можно корректировать работающую толщину.

Четвертый вопрос - по результатам проведения гидродинамических исследований нефтяных пластов методом гидропрослушивания можно определить скорость фильтрации жидкости между возмущающей и реагирующей скважинами, а также (при наличии кривых относительных фазовых проницаемостей) оценить скорости фильтрации нефти и воды, скорость фронта вытеснения (если в качестве возмущающей и реагирующей скважин используются нагнетательная и добывающая скважины).

Полученные результаты можно использовать как для контроля и регулирования разработки, так и для повышения точности гидродинамической модели. Например, для выгрузки значений скорости фильтрации для положительных *I, J, K* направлений в блоке модели по нефти и воде в симуляторе Eclipse Schlumberger используются мнемоники BVELO и BVELW. Выгруженные значения скоростей можно сравнить с определенными по результатам гидропрослушивания и оценить качество модели, провести ее адаптацию.

Подчеркнем, что гидропроводность ε и пьезопроводность χ, определяемые по результатам гидропрослушивания, зависят от фильтрационно-емкостных и PVT-свойств:

$$\varepsilon = \frac{k\cdot h}{\mu_{eff}}, \quad (8)$$

$$\chi = \frac{k}{m\cdot\mu_{eff}\cdot C_t}, \quad (9)$$

где μ_{eff} - эффективная вязкость (жидкости), формула для вычисления которой будет приведена ниже.

Из соотношений (8) и (9) можно следующим образом выразить подвижность жидкости:

$$\frac{k}{\mu_{eff}} = \frac{\varepsilon}{h}, \quad (10)$$

$$\frac{k}{\mu_{eff}} = \chi\cdot m\cdot C_t. \quad (11)$$

Использование соотношения (11) является предпочтительным, так как пористость обычно известна с большей точностью, чем работающая толщина пласта.

Скорость фильтрации жидкости, согласно закону Дарси, выражается, как [1, 7]:

$$w_l = \frac{k}{\mu_{eff}} \cdot \frac{\Delta P}{L}, \qquad (12)$$

где ΔP - перепад давления, L - расстояние между скважинами.

Соотношение для скорости фильтрации (12) с учетом (10) и (11) можно записать в виде:

$$w_l = \frac{\varepsilon}{h} \cdot \frac{P_{\text{inj}} - P_{prod}}{L}, \qquad (13)$$

$$w_l = \chi \cdot m \cdot C_t \cdot \frac{P_{\text{inj}} - P_{prod}}{L}, \qquad (14)$$

где P_{inj} - забойное давление в нагнетательной скважине, P_{prod} - забойное давление в добывающей скважине.

По указанным выше причинам для определения скорости фильтрации жидкости рекомендуется использовать выражение (14).

Скорость фильтрации нефти, согласно обобщенному закону Дарси, выражается, как [1]:

$$w_o = \frac{k \cdot k_{ro}}{\mu_o} \cdot \frac{\Delta P}{L}. \qquad (15)$$

Комбинируя выражения (14) и (15) можно получить следующую формулу для скорости фильтрации нефти:

$$w_o = \frac{k_{ro}}{\mu_o} \cdot \mu_{eff} \cdot w_l, \qquad (16)$$

где эффективная вязкость жидкости определяется, как:

$$\mu_{eff} = \frac{1}{\frac{k_{ro}}{\mu_o} + \frac{k_{rw}}{\mu_w}}. \qquad (17)$$

Скорости фильтрации связаны соотношением:

$$w_l = w_o + w_w. \qquad (18)$$

Тогда, при известных скоростях фильтрации жидкости и нефти скорость фильтрации воды определяется, как:

$$w_w = w_l - w_o. \qquad (19)$$

Согласно теории Баклея-Леверетта, скорость фронта вытеснения определяется, как [1]:

$$w_c = \frac{dx_c}{dt} = \frac{w_l}{m} f'(S_c), \qquad (20)$$

где $f'(S_c)$ - производная функции Баклея-Леверетта при водонасыщенности, равной насыщенности на фронте вытеснения S_c.

Функция Баклея-Леверетта определяется, как [1]:

$$f(S) = \frac{k_{rw}(S)}{k_{rw}(S) + k_{ro}(S)\frac{\mu_w}{\mu_o}}. \qquad (21)$$

Насыщенность на фронте вытеснения определяется из выражения [1]:

$$f'(S_c) = \frac{f(S_c) - f(S_{wcr})}{S_c - S_{wcr}}, \qquad (22)$$

где S_{wcr} - насыщенность связанной водой. Этому выражению может быть дана простая графическая интерпретация (см. рис. 2).

Текущая водонасыщенность и относительные фазовые проницаемости по нефти и воде оцениваются обратным счетом на основании текущей обводненности, определяемой, как:

$$WWCT(S) = \frac{k_{rw}(S)}{k_{rw}(S) + k_{ro}(S)\frac{\mu_w b_w}{\mu_o b_o}}, \qquad (23)$$

где b_o и b_w - объемные коэффициенты нефти и воды.

Так как водонасыщенность и относительные фазовые проницаемости именно оцениваются, а не определяются, то можно говорить только об оценке скорости фильтрации нефти и воды, а также скорости фронта вытеснения, а не об их определении.

Поскольку относительные фазовые проницаемости являются однозначными функциями насыщенности, используя формулу (23) можно, решая обратную задачу при известной обводненности, оценить водонасыщенность (на стенке скважины), а, следовательно, и функции относительных фазовых проницаемостей. Для этого необходимо задать функциональные зависимости относительных фазовых проницаемостей от водонасыщенности:

$$k_{rw}(S) = A_w \left(\frac{S - S_{wcr}}{1 - S_{wcr}} \right)^{\alpha_w}, \qquad (24)$$

$$k_{ro}(S) = A_o \left(\frac{1 - S_{owcr} - S}{1 - S_{owcr} - S_{wcr}} \right)^{\alpha_o}, \qquad (25)$$

где S_{owcr} - насыщенность остаточной нефтью; A_o, A_w, α_o, α_w - коэффициенты аппроксимации.

3. Пример использования результатов гидропрослушивания для контроля разработки нефтяного месторождения

Коллектор представлен органогенно-обломочным известняком. Залежь нефти массивного типа. В гидропрослушивании участвовало 13 скважин. Из них 3 нагнетательных (№№ 34, 58, 84) и 10 добывающих (№№ 21, 45, 46, 47, 59, 70, 71, 72, 83, 96). Планировалось проведение работ в 3 этапа - регистрация отклика забойного давления в добывающих скважинах на многократное ступенчатое изменение режима работы по очереди каждой нагнетательной скважины. В процессе анализа промысловых данных было обнаружено, что по техническим причинам останавливалась добывающая скважина № 59 и ее остановка повлияла на забойное давление в добывающих скважинах №№ 21 и 46. Таким образом, был осуществлен незапланированный четвертый этап гидропрослушивания. В результате всех работ была получена 21 кривая забойного давления в реагирующих скважинах (21 кривая реагирования). Все кривые были обработаны методом наилучшего совмещения с помощью ПО Saphir Kappa Engineering [8].

На рис. 1 приведены экспериментальные функции относительных фазовых проницаемостей (ОФП), принятые при расчетах скоростей, и их аппроксимация. Видно хорошее качество совмещения экспериментальных точек с расчетными кривыми.

На рис. 2. приведен графический метод определения насыщенности на фронте вытеснения, которая составила 0,5 д.ед. Производная от функции Баклея-Леверетта в этой точке равна 2,5 б/р. В табл. 1 приведены расчетные насыщенности и текущие значения ОФП.

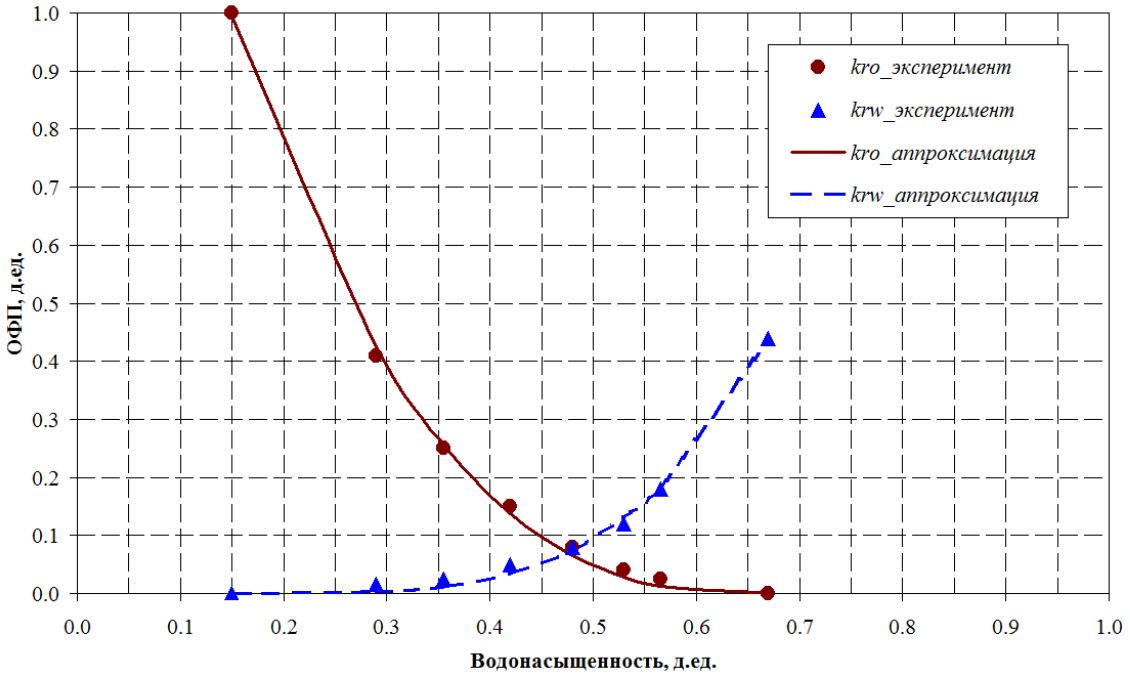


Рис. 1. Экспериментальные функции относительных фазовых проницаемостей (ОФП), принятые при расчетах скоростей, и их аппроксимация

Определенные скорости фильтрации жидкости между возмущающей и реагирующей скважинами, а также оцененные скорости фильтрации нефти и воды, скорости фронта вытеснения приведены на рис. 3-6 и в табл. 2. На картах (рис. 3-6) скорости разбиты на три группы, отмеченные разными цветами. Для каждой карты предельные значения внутри групп индивидуальны и выбирались таким образом, чтобы во все группы входило примерно одинаковое количество картируемых значений скоростей.

В результате анализа данных гидропрослушиваний определены следующие скорости фильтрации жидкости по направлениям между возмущающими и реагирующими скважинами; оценены следующие скорости фильтрации нефти и

воды, скорости фронта вытеснения (приводятся средние результаты по всем реагирующим скважинам, подробные результаты для каждой пары скважин даны в табл. 2 и на рис. 3-6):

1. Этап 1. Возмущающая скважина 58, нагнетательная.
   1.1. Скорость фильтрации жидкости – 146 см/сут.
   1.2. Скорость фронта вытеснения – 25,3 м/сут.
   1.3. Скорость фильтрации нефти – 38 см/сут.
   1.4. Скорость фильтрации воды – 108 см/сут.
2. Этап 2. Возмущающая скважина 84, нагнетательная.
   2.1. Скорость фильтрации жидкости – 159 см/сут.
   2.2. Скорость фронта вытеснения – 27,6 м/сут.
   2.3. Скорость фильтрации нефти – 41 см/сут.
   2.4. Скорость фильтрации воды – 118 см/сут.
3. Этап 3. Возмущающая скважина 34, нагнетательная.
   3.1. Скорость фильтрации жидкости – 23 см/сут.
   3.2. Скорость фронта вытеснения – 3,9 м/сут.
   3.3. Скорость фильтрации нефти – 8 см/сут.
   3.4. Скорость фильтрации воды – 15 см/сут.
4. Этап 4 (незапланированный). Возмущающая скважина 59, добывающая.
   4.1. Скорость фильтрации жидкости – 4,7 см/сут.
   4.2. Оценка скорости фронта вытеснения не представляется возможной, потому, что и возмущающая и реагирующие скважины являются добывающими.
   4.3. Скорость фильтрации нефти – 4,5 см/сут.
   4.4. Скорость фильтрации воды – 0,2 см/сут.

На основании полученной информации можно судить о преимущественном направлении фильтрационных потоков, степени охвата пласта заводнением и о взаимовлиянии скважин.

Полученные скорости можно использовать для повышения точности гидродинамической модели.

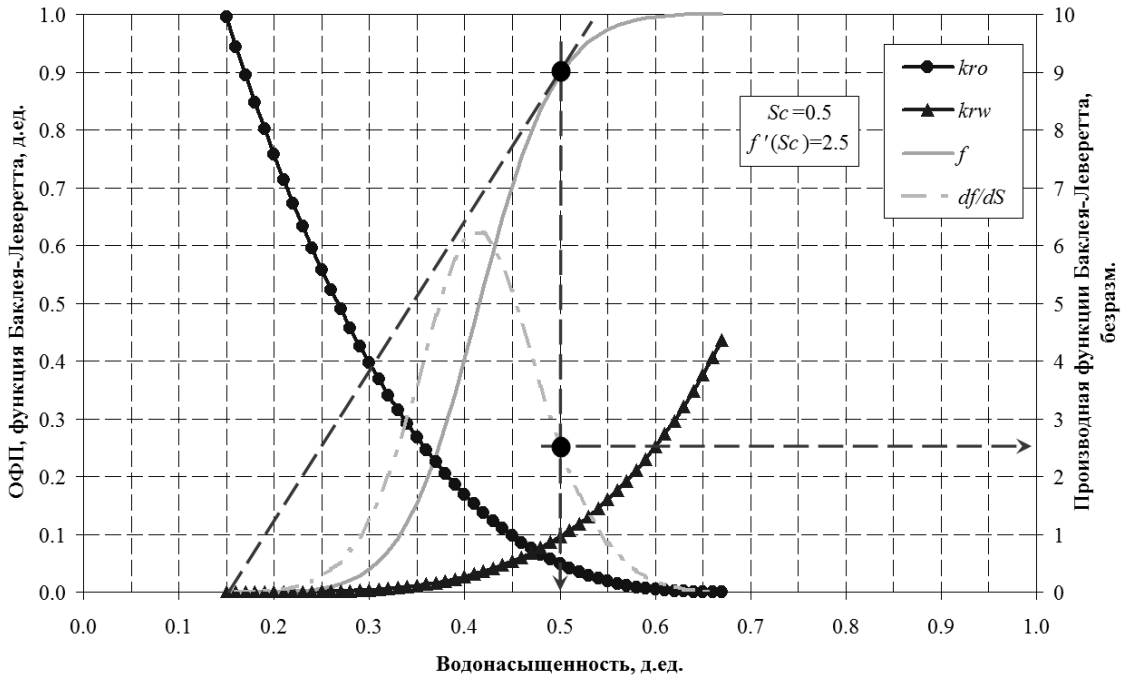


Рис. 2. Определение насыщенности на фронте вытеснения

Табл. 1

Определение насыщенности и текущих значений ОФП

Возмущающая скважина	Реагирующая скважина	Расстояние, м.	Обводненность, %	В зоне отбора / закачки			В межскважинной зоне		
				Sw, д.ед.	*kro*, д.ед.	*krw*, д.ед.	*Sw*, д.ед.	*kro*, д.ед.	*krw*, д.ед.
58	**1-ый этап**			**0.670**	**0.000**	**0.440**			
58	21	1835	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	45	820	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	46	808	5.6	0.312	0.364	0.005	0.491	0.056	0.087
	59	1174	18.0	0.357	0.252	0.013	0.514	0.038	0.111
	70	849	28.0	0.379	0.208	0.019	0.524	0.032	0.124
	71	861	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	72	1846	0.0	0.150	1.000	0.000	0.410	0.153	0.031
84	**2-ой этап**			**0.670**	**0.000**	**0.440**			
84	21	3059	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	46	1935	5.6	0.312	0.364	0.005	0.491	0.056	0.087
	59	1245	18.0	0.357	0.252	0.013	0.514	0.038	0.111
	70	1901	28.0	0.379	0.208	0.019	0.524	0.032	0.124
	71	869	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	72	873	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	83	1178	0.4	0.240	0.595	0.001	0.455	0.091	0.057
	96	1062	5.2	0.309	0.371	0.005	0.490	0.056	0.086
34	**3-ий этап**			**0.670**	**0.000**	**0.440**			
34	21	811	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	46	849	0.0	0.150	1.000	0.000	0.410	0.153	0.031
	47	863	1.0	0.262	0.516	0.001	0.466	0.079	0.650
	59	1191	18.0	0.357	0.252	0.013	0.514	0.038	0.111
59	**4-ый этап**		**18.0**	**0.357**	**0.252**	**0.013**			
59	21	1851	0.0	0.150	1.000	0.000	0.254	0.544	0.001
	46	844	5.6	0.312	0.364	0.005	0.334	0.305	0.008

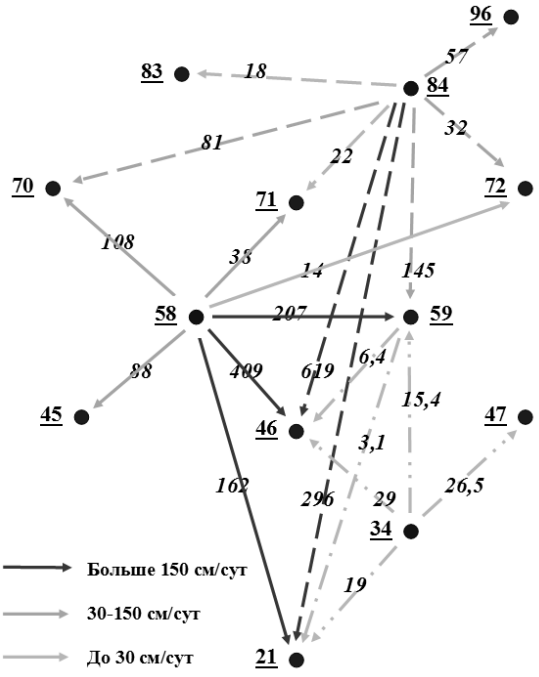


Рисунок 3. Скорость фильтрации жидкости, см/сут

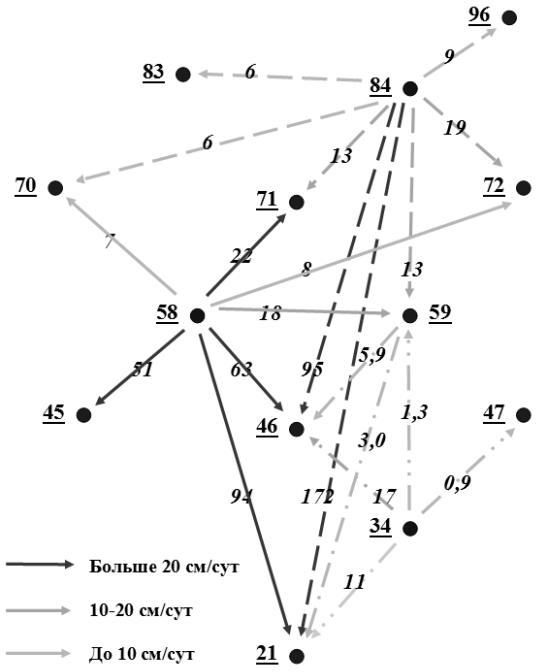


Рисунок 4. Оценочная скорость фильтрации нефти, см/сут

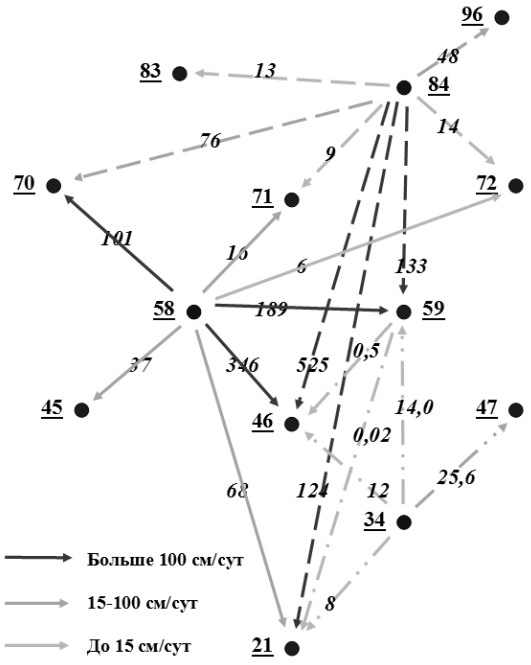


Рисунок 5. Оценочная скорость фильтрации воды, см/сут

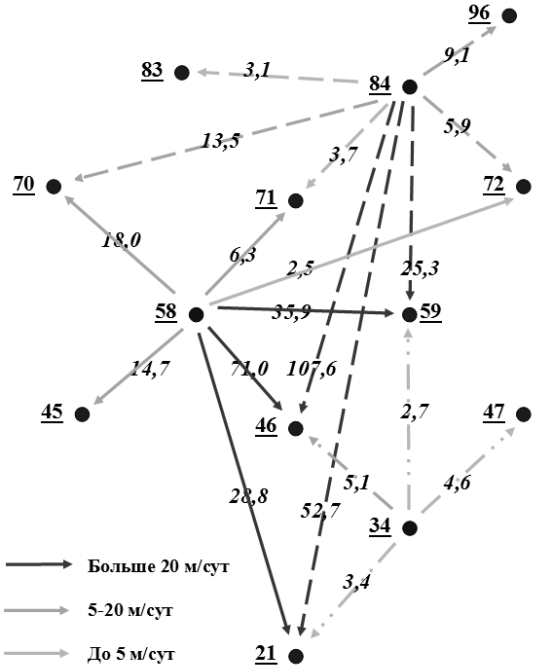


Рисунок 6. Оценочная скорость фронта вытеснения, м/сут

Таблица 2

Определение скоростей фильтрации и скорости фронта вытеснения

Возмущающая скважина	Реагирующая скважина	Расстояние, м.	Гидропроводность, Д*м/сПз	*Рзаб* на опорной глубине, МПа	*m*, %	*h*, м	Пьезопроводность, тыс.см^2/с	Скорость фильтрации, см/сут			Скорость фронта, м/сут
								Жидкости	Нефти	Воды	
58	**1-ый этап**			**24.8**	**0.198**						
58	21	1835	81.4	16.5	0.194	20	492	162	94	68	28.8
	45	820	70.5	16.5	0.206	72	111	88	51	37	14.7
	46	808	69.1	16.6	0.199	15	543	409	63	346	71.0
	59	1174	58.6	16.8	0.199	17	406	207	18	189	35.9
	70	849	61.8	16.8	0.208	47	148	108	7	101	18.0
	71	861	48.1	16.1	0.208	113	48	38	22	16	6.3
	72	1846	29.8	15.6	0.191	96	38	14	8	6	2.5
84	**2-ой этап**			**25.3**	**0.203**						
84	21	3059	278.9	16.5	0.194	24	1405	296	172	124	52.7
	46	1935	218.9	16.6	0.199	14	1843	619	95	525	107.6
	59	1245	170.5	16.8	0.199	71	283	145	13	133	25.3
	70	1901	66.3	16.8	0.208	32	234	81	6	76	13.5
	71	869	43.5	16.1	0.208	183	27	22	13	9	3.7
	72	873	50.2	15.6	0.191	152	41	32	19	14	5.9
	83	1178	64.9	16.2	0.200	245	31	18	6	13	3.1
	96	1062	56.1	16.2	0.216	75	81	57	9	48	9.1
34	**3-ий этап**			**29.4**	**0.207**						
34	21	811	20.6	16.5	0.194	151	16	19	11	8	3.4
	46	849	49.8	16.5	0.199	228	26	29	17	12	5.1
	47	863	21.3	15.4	0.198	115	22	26.5	0.9	25.6	4.6
	59	1191	16.9	16.8	0.199	103	19	15.4	1.3	14.0	2.7
59	**4-ый этап**			**16.8**	**0.199**						
59	21	1851	35.1	16.5	0.194	16	265	3.1	3.0	0.02	
	46	844	66.0	16.6	0.199	17	457	6.4	5.9	0.5	

Опорная глубина 2230 м.

Заключение

Предложены подходы к повышению точности гидродинамических моделей и контроля разработки нефтяных месторождений по данным гидропрослушивания. В том числе – подходы к определению скорости фильтрации жидкости между возмущающей и реагирующей скважинами, а также к оценке скорости фильтрации нефти и воды, скорости фронта вытеснения.

На реальном промысловом примере по результатам проведенных гидродинамических исследований методом гидропрослушивания определены скорости фильтрации жидкости по направлениям между возмущающими и реагирующими скважинами; оценены скорости фильтрации нефти и воды, скорости фронтов вытеснения нефти водой.

Полученные скорости можно использовать для контроля и анализа разработки данного участка нефтяного месторождения, а также для повышения точности гидродинамической модели. Например, для выгрузки значений скорости фильтрации для положительных *I, J, K* направлений в блоке модели по нефти и воде в симуляторе Eclipse Schlumberger используются мнемоники BVELO и BVELW. Путем сравнения модельных и рассчитанных по данным гидропрослушивания скоростей фильтрации можно оценить качество гидродинамической модели, провести ее адаптацию.

Кроме фильтрационно-емкостных свойств скорость фильтрации жидкости зависит от перепада давления между возмущающей и реагирующей скважинами. Причем зависимость имеет линейный характер. Поэтому при существенном изменении забойного давления в скважинах скорость фильтрации жидкости также изменится. Это нужно учитывать при использовании полученных скоростей фильтрации в процессе анализа разработки.

Скорости фильтрации нефти и воды, скорость фронта вытеснения зависят от скорости фильтрации жидкости, а также от функций относительных фазовых проницаемостей. Поскольку эти функции могут иметь высокую степень изменчивости по площади и по толщине, а их определение носит существенную долю субъективизма, скорости фильтрации нефти и воды, а также скорость фронта вытеснения, определенные по результатам гидропрослушивания, являются оценочными. Они могут быть пересчитаны с другими функциями относительных фазовых проницаемостей для нефти и воды.

Работа выполнена при поддержке Программы ФНИ государственных академий наук на 2013-2020 гг., проект № 0065-2019-0019.

Hydrodynamic Modeling Accuracy and Oil Fields Development Monitoring Improvement Based on Well Interference Tests Data

I.V. Afanaskin, A.V. Korolev, P.V. Kryganov

Abstract: Some approaches to improve the accuracy of hydrodynamic modeling and oil field development monitoring based on well interference tests data are considered. It is proposed to compare the conductivity and piezoconductivity values estimated using well interference tests and hydrodynamic modeling. Estimation method for liquid, oil and water rates and displacement front propagation velocity based on well interference tests data is also described.

Key words: well interference tests, hydrodynamic modeling, development monitoring.

Литература

1. Басниев К.С., Кочина И.Н., Максимов В.М. Подземная гидромеханика. М: Недра, 1993. – 416 с.

2. Деева Т.А., Камартдинов М.Р., Кулагина Т.Е. и др. Гидродинамические исследования скважин: анализ и интерпретация данных. Томск: ЦППС НД ТПУ. – 2009. – 243 с.
3. Ипатов А.И., Кременецкий М.И. Геофизический и гидродинамический контроль разработки месторождений углеводородов. Ижевск: Регулярная и хаотическая механика, 2010. – 780 с.
4. Кременецкий М.И., Ипатов А.И., Гуляев Д.Н. Информационное обеспечение и технологии гидродинамического моделирования нефтяных и газовых залежей. М.-Ижевск: Издательство «ИКИ», 2012. – 869 с.
5. Кульпин Л.Г., Мясников Ю.А. Гидродинамические методы исследования нефтегазоводоносных пластов. М., Недра, 1974. - 200 с.
6. Курочкин В.И., Санников В.А. Теоретические основы и анализ гидродинамических исследований скважин. М.-Ижевск: Институт компьютерных исследований, 2015. - 372 с.
7. Роберт Эрлагер мл. Гидродинамические методы исследования скважин. М.-Ижевск: Институт компьютерных исследований. - 2007. – 512 с.
8. Olivier Houze, Didier Viturat, Ole S. Fjaere. Dynamic Data Analysis. V 5.12. Kappa Engineering, 2017. – 743 p.

ПОВЫШЕНИЕ ДОСТОВЕРНОСТИ ОПРЕДЕЛЕНИЯ ФИЛЬТРАЦИОННЫХ ПАРАМЕТРОВ ПЛАСТА ПРИ РЕШЕНИИ ЗАДАЧИ НЕСТАЦИОНАРНОЙ ФИЛЬТРАЦИИ ПУТЁМ УЧЁТА ПРЕДЫСТОРИИ РАБОТЫ СКВАЖИНЫ

П.В. Крыганов[1], Н.П. Ефимова[2], И.В. Афанаскин[3], С.Г. Вольпин[4], Ю.А. Мясников[5], А.В. Свалов[6]

[1, 2, 3, 4]ФГУ ФНЦ НИИСИ РАН, Москва, Россия
[5, 6]ОАО «ВНИИнефть», Москва, Россия

E-mail's: [1]kryganov@mail.ru, [2]efinatka@gmail.com, [3]ivan@afanaskin.ru, [4]sergvolpin@gmail.com

Аннотация: Рассмотрено влияние длительности учитываемого изменения дебита нефтяной скважины на результаты интерпретации материалов гидродинамических исследований скважин. Предложен подход для восстановления предыстории дебита, предшествующего известной истории работы скважины, если он не был замерен, либо его замеры вызывают сомнения. Разработан и апробирован алгоритм восстановления предыстории работы скважины.

Ключевые слова: гидродинамические исследования скважин, история работы скважины.

Введение

Достоверность результатов интерпретации гидродинамических исследований скважин значительно влияет на точность информационного обеспечения проектирования разработки нефтяных месторождений и контроля за её осуществлением. В частности, ошибки результатов интерпретации, как качественные, так и количественные, важны при их использовании в геолого-гидродинамическом моделировании. В работах [1-4] было отмечено, что при интерпретации материалов исследований на качество определяемой информации значительно влияет история работы скважины.

При наиболее информативных гидродинамических исследованиях методом восстановления или падения давления под историей подразумевается подробная информация о дебите (или расходе) скважины в течение достаточного длительного времени, много большем, чем длительность кривой восстановления давления (КВД). Настоящая работа посвящена анализу влияния истории работы скважины на фильтрационные параметры пласта, определяемые по кривой восстановления давления, и способу увеличения их достоверности.

Вопросы, изученные в работах [1-4], позволили сделать вывод, что необходимо иметь всю динамику дебита (учитывая кратковременные колебания) за период, предшествующий началу исследований. И даже осреднение дебита, рассчитанного по накопленному объему за весь период перед исследованиями, не является достаточным условием точности интерпретации. На практике чаще всего добиться постоянного контроля за добычей жидкости очень непросто, если процесс замера не автоматизирован. В работе [4] на основе численного эксперимента выявлено значительное влияние истории работы на вид диагностического графика, а значит и на выбор интерпретационной модели, и на точность определения фильтрационных параметров.

1. Анализ влияния длительности учитываемого изменения дебита скважины на результаты интерпретации гидродинамических исследований

Чтобы провести обработку кривой восстановления давления, предварительно

необходимо определить по какой интерпретационной модели следует вести обработку. С этой целью строится диагностический график Бурде, позволяющий определить интерпретационную модель. График представляет собой семейство двух кривых, рис. 1б. Одна кривая (на графике верхняя) – это изменение депрессии на пласт во времени, а вторая кривая (на графике нижняя) – это изменение во времени производной от депрессии. Диагностический график строится в билогарифмических координатах. Аргументом, по которому производится дифференцирование кривой давления, является функция суперпозиции, учитывающая историю работы скважины.

В развитие изучения влияния истории работы на выбор модели пласта для последующей интерпретации было проведено численное гидродинамическое моделирование для ряда случаев с разной длительностью работы скважины с постоянным дебитом. В первом случае скважина работала в течение 9 суток (КВД приведена на рис. 1-1а в декартовых координатах и в билогарифмических координатах на рис. 1-1б), после чего скважина остановлена для регистрации КВД. Во втором случае - история работы скважины до КВД составляла 2 суток (КВД в декартовых координатах на рис. 1-2а и в билогарифмических координатах на 1-2б, 1-2в). В третьем случае - скважина работала до остановки на КВД в течение 30 суток (КВД в декартовых координатах на рис. 1-3а и в билогарифмических координатах на 1-3б, 1-3в).

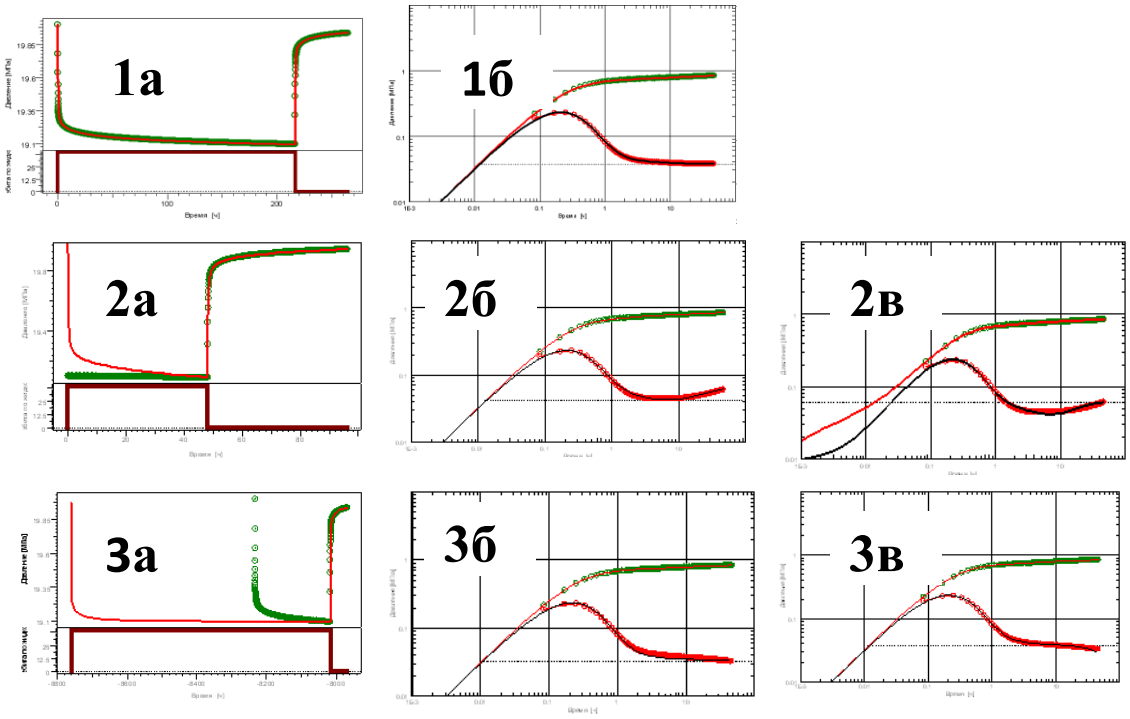


Рис. 1. Интерпретация методом наилучшего совмещения смоделированных (сплошные кривые) кривых восстановления давления в декартовых координатах (а), в билогарифмических координатах (б, в) с дебитом продолжительностью 9 сут (1), 2 сут (2), 30 сут (3) с фактическими точками

Рис. 1-1а и 1-1б отражают исходную принятую технологию исследований. На остальных рисунках показано, как изменяется вид производной в том случае, если используется дебит скважины с уменьшенной и увеличенной продолжительностью работы и на что это влияет. Так, из рис. 1-2б, 1-2в видно, что при уменьшенной продолжительности производная начинает расти в конце КВД. Рост производной может быть характерен для разных моделей, в том числе модели однородного пласта с непроницаемой границей или модели двойной проницаемости. На рис. 1-2б приведена интерпретация методом наилучшего совмещения по модели однородного пласта с непроницаемой границей и на рис. 1-2в по модели пласта с двойной проницаемостью. Видно, что в обоих случаях получено хорошее совмещение. Соответственно, в первом случае ошибка в выборе модели пласта приводит к неверному выявлению наличия границы. Во втором случае

ошибка в выборе модели приведет к заниженному значению проницаемости.

Из рис. 1-3б, 1-3в видно, при используемой завышенной продолжительности работы производная начинает падать в конце КВД. При падении производной интерпретировать КВД можно, например, при использовании модели однородного пласта без выхода на радиальную фильтрацию и тогда в результате интерпретации с хорошим совмещением (рис. 1-3б) будет получена завышенное значение проницаемости. Либо выбрать модель однородного пласта с границей постоянного давления и в результате интерпретации получить хорошее совмещение (рис. 1-3в) и определить ошибочно границу и расстояние до нее.

Таким образом, ошибки в задании истории работы скважины дебита сказываются как на количественной оценке проницаемости, так и качественно в выборе интерпретационной модели пласта. Рассмотренные ошибки в задании истории работы скважины возникают, в первую очередь, при отсутствии полной и достоверной информации о характере работы скважины до начала её исследований. С целью повышения достоверности определения фильтрационных параметров пласта необходимо разработать методику воссоздания предыстории - истории работы скважины, происходившей до начала её исследований.

2. Разработка и апробация алгоритма восстановления предыстории работы скважины

Существует несколько методических приемов, позволяющих в ряде случаев уменьшить влияние неполной информации об истории дебитов на результаты интерпретации. В частности, в работе [5] предложен подход с использованием метода деконволюции, который позволяет значительно увеличить точность результатов интерпретации при недостатке информации о дебите скважины до КВД.

В настоящей работе предложен принципиально другой подход, при котором режим работы скважины делится на два периода:

– на историю работы скважины, т.е. отрезок времени работы скважины с известными параметрами дебита непосредственно перед началом анализируемой кривой давления;

– на предысторию, т.е. период работы скважины в предшествующее истории время с неизвестными параметрами дебита.

Идея данного подхода в том, чтобы каким-то образом восстановить часть истории работы скважины, если она неизвестна. При этом в уравнении интерпретационной модели в число неизвестных параметров (наряду с проницаемостью, скин-фактором, пластовым давлением) добавляются дополнительные параметры – эквивалентные время и дебит предыстории. Эти неизвестные параметры так же, как и фильтрационные характеристики, определяются методом наилучшего совмещения (нелинейной регрессии). Реализация данного подхода осуществлена в специально созданном программном обеспечении (СПО).

Расчетное давление определяется с учетом принципа суперпозиции дебита. Для этого весь период исследования делится на интервалы, как показано на рис. 2, с тем, чтобы на границе каждого интервала дебит изменялся скачками. На рис. 2 показан случай произвольного числа дебитов.

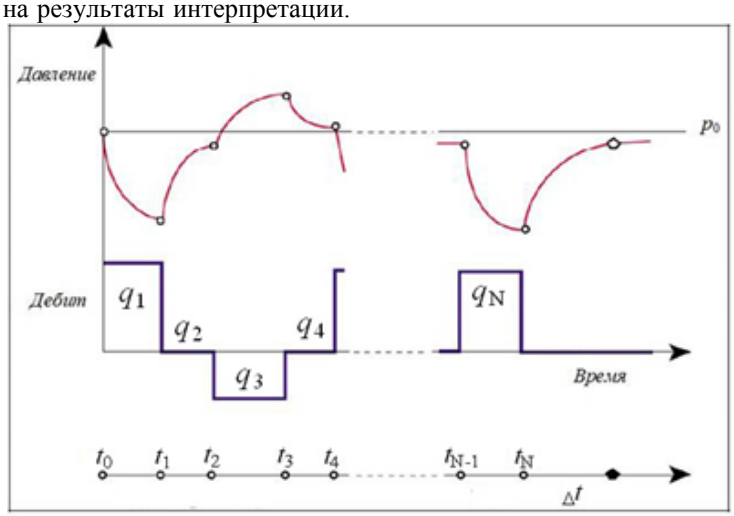


Рис. 2. Работа скважины с переменным дебитом при произвольном количестве режимов [5]

Давление на забое скважины в произвольный момент времени описывается уравнением (1):

$$p(t) = p_0 - \frac{B\mu}{2\pi kh}\sum_{j=1}^{N} q_j\left[p_D\left(t_N - t_{j-1} + \Delta t\right)_D - \right.$$
$$\left. - p_D\left(t_N - t_{j-1} + \Delta t\right)_D\right], \qquad (1)$$

где t_j, q_j, $(j = 1, 2, ..., N)$ - время и дебит конца j - го интервала;

B – объемный коэффициент жидкости;

μ - вязкость жидкости;

k – проницаемость пласта;

h – эффективная толщина пласта;

P_D – безразмерное давление;

t_D – безразмерное время;

Безразмерное давление - функция безразмерного времени и параметров, характеризующих условия на скважине, в пласте и на границе. Для учета истории работы скважины используется принцип суперпозиции дебитов. Формула для давления записана с использованием безразмерного давления для удобства. Аналитическое выражение для безразмерного давления для разных моделей имеет свой вид. На рис. 3 показан гипотетический пример работы скважины на семи разных режимах перед остановкой.

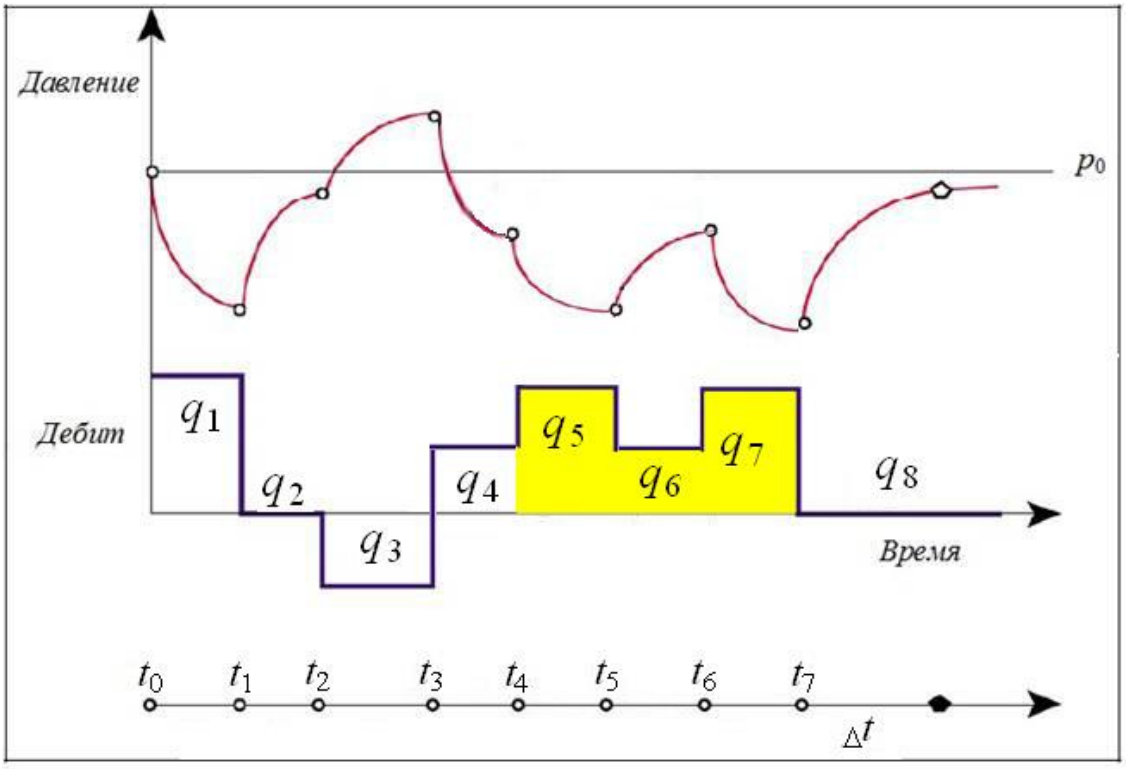


Рис. 3. Работа скважины с переменным дебитом на 7-ми режимах

Выражение для давления на КВД для 7-ми ступенчатого дебита с учетом суперпозиции дебитов выглядит следующим образом:

$$p(t) = p_0 - \frac{B\mu}{2\pi kh}\{q_1\left[p_D\left(t_7 + \Delta t\right)_D - p_D\left(t_7 - t_1 + \Delta t\right)_D\right] +$$
$$+ q_2\left[p_D\left(t_7 - t_1 + \Delta t\right)_D - p_D\left(t_7 - t_2 + \Delta t\right)_D\right] + ... +$$
$$+ q_5\left[p_D\left(t_7 - t_4 + \Delta t\right)_D - p_D\left(t_7 - t_5 + \Delta t\right)_D\right] +$$
$$+ q_6\left[p_D\left(t_7 - t_5 + \Delta t\right)_D - p_D\left(t_7 - t_6 + \Delta t\right)_D\right] +$$
$$+ q_7\left[p_D\left(t_7 - t_6 + \Delta t\right)_D\right]\} \qquad (2)$$

Если известна только часть информации о дебитах, которые необходимо использовать при интерпретации (т.е. мы имеем основание считать, что учет этих дебитов влияет на интерпретацию), то рассмотрим понятия эквивалентного времени и эквивалентного дебита предыстории (рис. 4). Например, если на рис. 3 известны дебиты на 5-7 режимах (выделены затемнением) и неизвестны на 1-4 режимах, то на рис. 4 неизвестная часть обозначается как эквивалентная с характеристиками $q_э$ и $t_э$, а известная – с характеристиками $q_1^{'}, q_2^{'}, q_3^{'}$ и $t_1^{'}, t_2^{'}, t_3^{'}$.

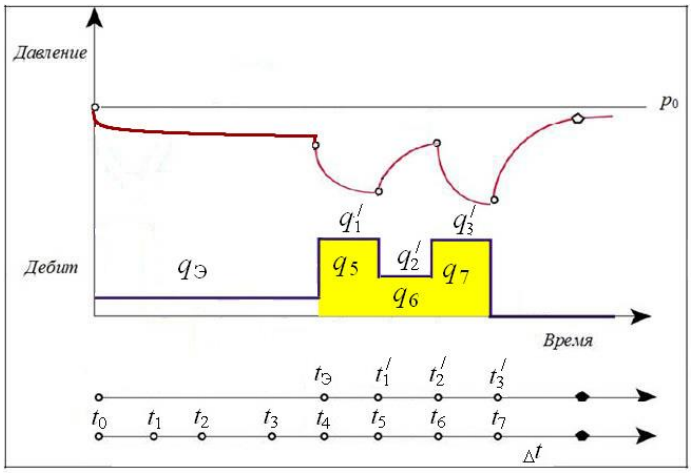


Рис. 4. Работа скважины с переменным дебитом
на 7-ми режимах с добавлением эквивалентного времени и дебита

Соответственно, выражение для давления с учетом суперпозиции дебитов примет вид:

$$p(t) = p_0 - \frac{B\mu}{2\pi kh}\{q_э\left[p_D\left(t_э + t_3' + \Delta t\right)_D - p_D\left(t_3' + \Delta t\right)_D\right] +$$
$$+ q_1'\left[p_D\left(t_3' + \Delta t\right)_D - p_D\left(t_э + t_3' - t_1' + \Delta t\right)_D\right] +$$
$$+ q_2'\left[p_D\left(t_э + t_3' - t_1' + \Delta t\right)_D - p_D\left(t_э + t_3' - t_2' + \Delta t\right)_D\right] +$$
$$+ q_3'\left[p_D\left(t_э + t_3' - t_2' + \Delta t\right)_D - p_D\left(t_э + \Delta t\right)_D\right]\} \qquad (3)$$

В выражении (3) помимо параметров интерпретационной модели, характеризующих условия на скважине, в пласте и на границе, появляются 2 дополнительных неизвестных параметра $q_э$ и $t_э$. Эти параметры так же, как и фильтрационные параметры, определяются методом наилучшего совмещения (нелинейной регрессии).

На кривую забойного давления влияет накопленная добыча жидкости. В случае предыстории она определяется, как произведение эквивалентного времени $t_э$ на эквивалентный дебит $q_э$. Поэтому при решении обратной задачи один из этих двух параметров необходимо зафиксировать, а другой при этом можно определить. Поскольку с точностью до константы определяющим решение параметром является произведение $q_э t_э$, то одну и ту же кривую забойного давления можно получить, используя различные сочетания параметров $q_э$ и $t_э$, дающих одно и то же произведение $q_э t_э$

Эффективность данного подхода была проверена с использованием программ «Saphir» и СПО и проверка пошагово показана в табл. 1. Рассмотрим каждый шаг подробно:

Шаг 1. В программе «Saphir» заданы исходные данные для численного моделирования (время истории 7 сут проницаемость 50x10^{-3}мкм^2, скин-фактор 2, пластовое давление 20 МПа, влияние ствола скважины 0.5 м^3/МПа).

Шаг 2. В программе «Saphir» смоделирована кривая изменения давления (время истории 7 сут).

Шаг 3. В СПО проведена обработка смоделированной КВД методом наилучшего совмещения с полной историей (время истории 7 сут).

Шаг 4. В СПО получены результаты обработки КВД с полной историей работы 7 сут (проницаемость 49.3x10^{-3}мкм^2, скин-фактор 2.1, пластовое давление 20 МПа). Результаты обработки практически идентичны с исходными данными для моделирования.

Шаг 5. В СПО задана неполная история работы 2 суток (исключены из расчетов 5 первых ступеней дебитов и оставлены только последние 2) и проведена обработка той же КВД методом наилучшего совмещения с неполной историей (время истории 2 сут).

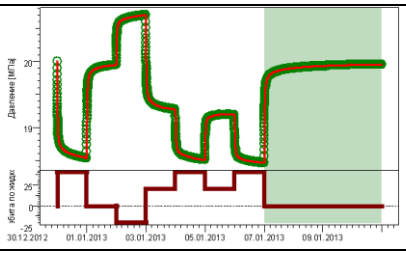
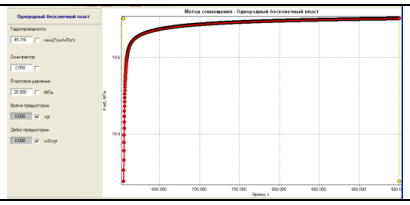
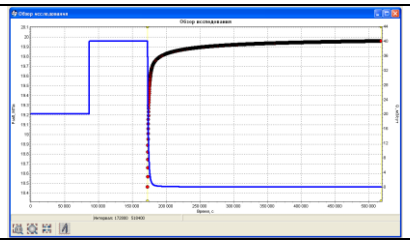
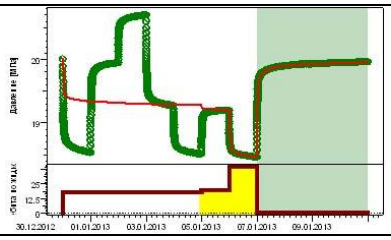
Шаг 6. В СПО получены результаты обработки КВД с неполной историей работы 2 сут (проницаемость 49x10^{-3}мкм^2, скин-фактор 1.9, пластовое давление 20 МПа, зафиксированное $t_э$ = 5 сут, рассчитанный $q_э$ = 17.987 м^3/сут).

Шаг 7. В программе «Saphir» обработана смоделированная КВД методом наилучшего совмещения с восстановленной историей скважины 7сут (рассчитанное $q_э$ = 17.987 м^3/сут).

Шаг 8. В программе «Saphir» получены результаты обработки КВД с восстановленной историей работы скважины 7сут (проницаемость 50x10^{-3}мкм^2, скин-фактор 2.0, пластовое давление 20 МПа). Результаты обработки идентичны с исходными данными для моделирования, что подтверждает эффективность нового подхода.

Табл. 1.

Проверка эффективности методики восстановления дебита

		Saphir	СПО
Шаг 1	Исходные параметры для численного моделирования	время истории 7 сут, однородный пласт, проницаемость 50x10^{-3}мкм^2, скин-фактор 2, пластовое давление 20 МПа, влияние ствола скважины 0.5 м^3/МПа	
Шаг 2	Смоделированная кривая изменения давления (время истории 7 сут)		
Шаг 3	Обработка смоделированной КВД с полной историей (время истории 7 сут)		
Шаг 4	Результаты обработки КВД с полной историей работы (время истории 7 сут)		проницаемость 49.3x10^{-3}мкм^2 скин-фактор 2.1 пластовое давление 20 МПа
Шаг 5	Обработка смоделированной КВД с неполной историей (время истории 2сут)		
Шаг 6	Результаты обработки КВД с неполной историей работы (время истории 2сут)		проницаемость 49x10^{-3}мкм^2 скин-фактор 1.9 пластовое давление 20 МПа $t_э$ = 5 сут (зафиксировано) $q_э$ = 17.987 м^3/сут (рассчитано)
Шаг 7	Обработка смоделированной КВД с восстановленной историей (время истории 7сут)		
Шаг 8	Результаты обработки КВД с восстановленной историей работы (время истории 7сут)	проницаемость 50x10^{-3}мкм^2 скин-фактор 2.0 пластовое давление 20 МПа	

На рис. 5 проиллюстрированы другие случаи кривых, смоделированных на основе однородного бесконечного пласта при различных технологических особенностях истории работы скважины, для которых эффективность новой методики также была доказана. Данный анализ позволил сделать вывод о том, что, если искомый дебит и время – в виде одной ступени, то с помощью данного подхода его можно

определить точно. Для случая нескольких ступеней дебита данный подход позволяет определить эквивалентный дебит и эквивалентное время.

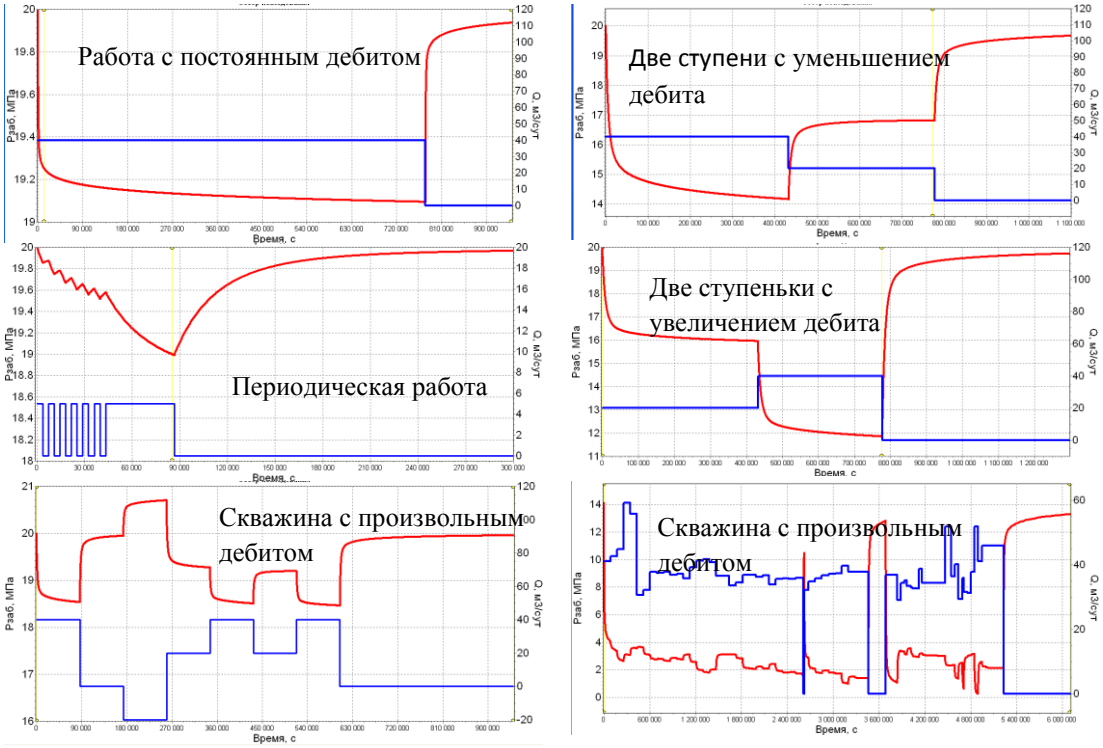


Рис. 5. Примеры различных технологий исследований для проверки эффективности методики

На рис. 6 представлена технологическая схема фактических исследований, проведенных на одном из месторождений Западной Сибири. Перед регистрацией КВД (около 6 сут) скважина работала и были известны замеры дебита скважины около 10 сут. На рис. 7 приведены результаты интерпретации КВД методом наилучшего совмещения по соответствующей пласту модели с дебитом продолжительностью 10 сут. Из рис. 7 видно, что расчетная кривая плохо совмещена с фактическими точками давления.

Для того, чтобы улучшить совмещение с той же моделью пласта КВД была обработана повторно с целью оценки $q_э$ при фиксированном $t_э$=10 сут. Затем обработка КВД была произведена с восстановленным дебитом предыстории (рис. 8). Видно, что данный подход позволил добиться хорошего совмещения красной расчетной кривой и синих фактических точек.

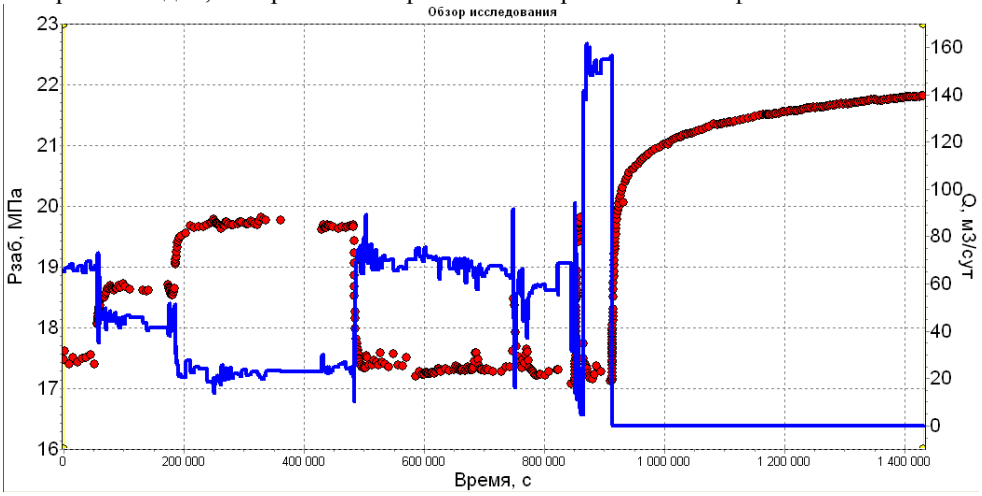


Рис. 6. Технологическая схема фактических исследований на примере одного из месторождений Западной Сибири

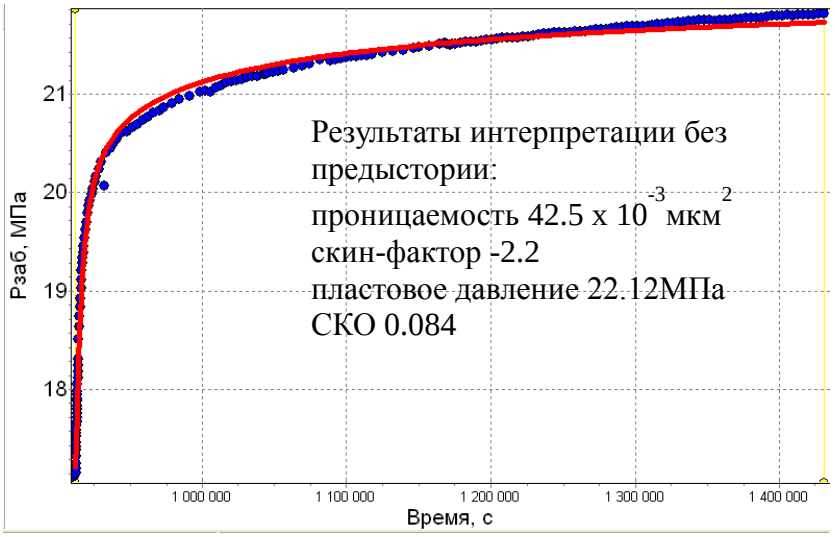


Рис. 7. Результаты интерпретации КВД без предыстории
СКО – среднеквадратичное отклонение

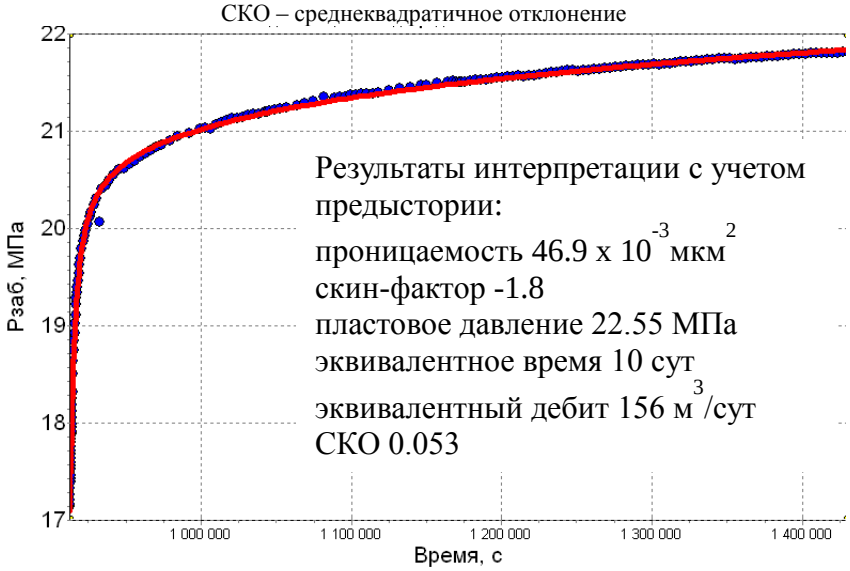


Рис. 8. Результаты интерпретации КВД с учетом предыстории
СКО – среднеквадратичное отклонение

Заключение

Новый подход, рассмотренный в данной работе, позволяет:

- восстановить время и дебит, предшествующий известной истории работы скважины, если он не был замерен либо его замеры вызывают сомнения;
- определить эквивалентные время и дебит предыстории, создающие такое же возмущение в пласте, как и то, которое обусловлено реальными, но неизвестными дебитами;
- повысить качество интерпретации: более достоверно выбрать интерпретационную модель и с большей точностью оценить фильтрационно-емкостные свойства и геометрию пласта.

Практическое применение предложенного подхода зависит от возможности включения дополнительных неизвестных параметров в некоторые интерпретационные модели программного обеспечения для интерпретации ГДИС. Такое включение, по нашему мнению, возможно.

Работа выполнена при поддержке гранта РФФИ 18-07-00503 А.

RESEARCH OF WELL PRODUCTION HISTORY AT SUBSURFACE HYDRODYNAMICS INVERSE PROBLEM SOLUTION

P.V. Kryganov, N.P. Efimova, I.V. Afanaskin, S.G. Volpin, Y.A. Myasnikov, A.V. Svalov

Abstract: Oil well rate duration changes influence on quality and quantitative well test interpretation results described. Preceding rate history reconstruction approach applied for unknown or non-correct rate history. Oil well rate history reconstruction algorithm developed and tested.

Key words: welltest, well production history.

Литература

1. Акрам Х. Исследование малодебитных скважин в России / Х. Акрам, С. Г. Вольпин, Ю. А. Мясников, И. Р. Дияшев, У. Д. Ли, А. Н. Шандрыгин // Нефтегазовое обозрение – Шлюмберже. - Весна 1999. - Том 4. №1. – С. 4-13.
2. Вольпин С. Г. Гидродинамические исследования низкопроницаемых коллекторов / С. Г. Вольпин, Ю. А. Мясников, А. В. Свалов // Нефтяное хозяйство. - 2000. – № 12. - С. 8-10.
3. Эрлагер Р. Гидродинамические методы исследования скважин / Р. Эрлагер. Перевод с англ. под ред. М. М. Хасанова // Москва - Ижевск: НИЦ «Регулярная и хаотическая динамика». - Институт компьютерных исследований. - 2006. – 512 с.
4. Крыганов П.В. Методы повышения достоверности результатов гидродинамических исследований нефтяных пластов и скважин // Диссертация кандидата техн. наук. – М.: ОАО «ВНИИнефть». - 2012. – 133 с.
5. Olivier Houze, Didier Viturat, Ole S. Fjaere. Dynamic Data Analysis. V 5.20. Kappa Engineering, 2019. – 757 p.

К ВОПРОСУ О РАЗРАБОТКЕ ТРУДНОИЗВЛЕКАЕМЫХ ЗАПАСОВ

К.Д. Ашмян[1], С.Г. Вольпин[2], О.В. Ковалева[3]
ФГУ ФНЦ НИИСИ РАН. Москва. Россия
E-mail's: kdashmyan@yandex.ru, sergvolpin@gmail.com, olgakovaleva57@ mail.ru

Аннотация: В статье проанализировано состояние запасов нефти в РФ. Показано, что в **настоящее** время большая часть их относится к разряду трудноизвлекаемых. Показаны два вида трудноизвлекаемых запасов: 1 – природные и 2 – техногенные. Рассмотрены основные проблемы при разработке этих месторождений. Показана необходимость создания обобщающей программы исследования месторождений нефти с трудноизвлекаемыми запасами.

Ключевые слова: трудноизвлекаемые запасы, коллекторские свойства, аномальные свойства флюидов и пород, остаточные нефти, исследования скважин и пластов, физико-химические исследования, технология доизвлечения нефти.

Российская Федерация достаточно богата запасами нефти. На территории страны, составляющей 12,8% территории планеты Земля, сосредоточено более 12% прогнозных ресурсов и около 12% разведанных запасов нефти. По оценке зарубежных экспертов по состоянию на 2015 год доказанные запасы нефти России составляли примерно 8% от мировых, и по этому показателю Россия находится на четвёртом месте в мире.

Однако, несмотря на это, в нефтедобывающей промышленности России наблюдается невысокая обеспеченность запасами, что вытекает из отсутствия опережающей разведки новых нефтяных месторождений и подтверждения выявленных запасов. В течение длительного периода наблюдается так же ухудшение качества имеющихся разведанных запасов.

В России извлекаемые запасы нефти составляют около 20 млрд. тонн категории АВС1 + С2, приблизительно 55 % вовлечены в разработку, из них 42 % - это традиционные запасы и 13% - это трудноизвлекаемые. Однако порядка 45% извлекаемых запасов в настоящее время не вовлечены в разработку, это и неразрабатываемые залежи разрабатываемых месторождений и месторождения, еще не введенные в разработку.

По оценке министерства Природных ресурсов и экологии РФ, существуют объективные причины, по которым почти половина открытых запасов не разрабатывается. Во многих случаях это трудноизвлекаемые запасы, для которых пока не существует технологий их эффективной разработки. В других случаях отсутствует транспортная инфраструктура. И, в-третьих, это недостаток экономических стимулов при существующей высокой налоговой нагрузке».

Кроме того, необходимо учитывать то, что промышленная добыча природных углеводородов (жидких и газообразных) ведется уже более 100 лет, и при этом по мере развития техники и технологии добычи в разработку поэтапно вводились сначала более доступные месторождения, а затем и более сложные по добыче. В настоящее время таких месторождений становится все меньше и меньше, не говоря о том, что крупных месторождений, подобных Самотлорскому, в перспективе ожидать не приходится. Сохранить объемы добычи нефти, газа и газоконденсата в РФ очень важно для развития собственной экономики и поддержания влияния на мировой арене.

К трудноизвлекаемым запасам углеводородов относятся:

1. Низкопроницаемые карбонатные отложения.
2. Низкопроницаемые терригенные отложения.
3. Нефть с высокой вязкостью, битумы.
4. Остаточная нефть в месторождениях на поздней стадии разработки.
5. Нетрадиционные запасы - так называемая «сланцевая» нефть (а точнее «керогеновая»), запасы которой в России во много раз больше, чем в США.

Комплексная задача разработки трудноизвлекаемых запасов углеводородов

В настоящее время доля трудноизвлекаемых запасов нефти в России и странах СНГ постоянно увеличивается ввиду того, что идет интенсивная разработка месторождений, разведанных ранее. Большое число имеющихся на госбалансе запасов относятся к трудноизвлекаемым, а разработка таких месторождений априори ведет к естественному удорожанию добычи, снижению проектных коэффициентов извлечения нефти (КИН), усложнению применяемых технологий повышения нефтеотдачи. Кроме того, подбор новой технологии разработки трудноизвлекаемых запасов процесс неоднозначный, т.к. отсутствует необходимая информация о пласте, скважинах, пластовых флюидах, нефтенасыщенности породы и т.д. Именно по этой причине многие ранее хорошо зарекомендовавшие себя технологии не являются сейчас эффективными, что и приводит к отказу от их применения и поиску новых решений: неапробированных, экзотических, а значит дорогих.

Решение проблемы грамотного подбора технологии добычи нефти, отнесенной по той или иной причине к категории трудноизвлекаемых, основывается на детальном изучении физико-химических параметров и свойств пластовых флюидов, а также исследований пластов, скважин, призабойной зоны и нефтенасыщенного коллектора.

Создание экспериментального и аналитического комплекса, ориентированного на подбор технологий воздействия и управления разработкой, а также системы хранения и обработки полученных данных в условиях необходимости проведения импортозамещения, становится для нефтедобычи приоритетной задачей.

Такой комплекс включает в себя экспериментальные (лабораторные) исследования пластовых флюидов и нефтенасыщенных пород, непрерывный процесс скважинных измерений забойных давлений и дебитов, передача информации и её накопление в базе данных. Обработка непрерывных скважинных измерений и анализ результатов обработки гидродинамических и геофизических исследований в комплексе с численным моделированием разработки нефтяного месторождения, создание Банков данных для хранения аналитических обработок информации.

Структура комплекса:

I. Физико-химические исследования пластовой нефти, попутного газа, пластовой воды, керновые исследования.

II. Банк данных лабораторных и промысловых исследований, включая построение карт распределения параметров и свойств трудноизвлекаемых нефтей, включая остаточные нефти по залежи в целом.

III. Анализ и обобщение гидродинамических исследований скважин и пластов.

IV. Анализ и обобщение геофизических исследований.

V. Анализ непрерывных скважинных измерений давлений и дебитов.

Разработка залежей природных углеводородов, причисляемых к трудноизвлекаемым запасам, а именно, требующим применения новых методов и технологий воздействия на нефтенасыщенный пласт, в настоящее время стало уже практически самостоятельным направлением. Так, в работе [1] приводятся методические рекомендации по применению методов повышения нефтеотдачи и обработки призабойной зоны пластов, позволяющие в зависимости от вида остаточной в пласте нефти, выбрать способ воздействия, рабочий агент (табл. 1, рис. 1). Исходя из геолого-физических свойств пласта и физико-химических свойств пластовых остаточных нефтей и других пластовых флюидов, возможно оценить эффективность применения того или иного метода воздействия [1].

Известно, что залежи с трудноизвлекаемыми запасами, как правило, подразделяются на два вида: природные и техногенные.

Природные залежи с трудноизвлекаемыми запасами пластовых углеводородов.

К залежам такого вида относятся залежи со сложными геолого- физическими свойствами пластов - коллекторов, аномальной термодинамической обстановкой в залежах и энергетической особенностью пластовых систем, а также аномальными свойствами флюидов и насыщающих пород.

Таблица 1. Методы повышения нефтеотдачи и обработки призабойной зоны

Цель	Способ воздействия	Рабочий агент
Воздействие на нефть, оставшуюся в пласте в макромасштабе – повышение охвата вытеснением	Повышение вязкости вытесняющего агента, снижение вязкости нефти	Полимеры Мицеллярные растворы Пар Воздух + вода (горение) Углекислый газ
	Увеличение (расширение) нефти	То же Пар Воздух + вода (горение)
	Увеличение дренируемой (работающей) толщины пласта	ПАВ Полимеры Водогазовые смеси Щелочи Вода (циклическое заводнение)
Воздействие на нефть, оставшуюся в пласте в микромасштабе – вытеснение рассеянной остаточной нефти	Достижение смешиваемости нефти и вытесняющего агента	Углекислый газ Газ высокого давления
	Снижение межфазного натяжения	Мицеллярные растворы Щелочи
	Повышение смачиваемости пластовой водой	Водорастворимые ПАВ Щелочи
	Повышение фазовой проницаемости для нефти и снижение для воды	Водорастворимые ПАВ Водогазовые смеси

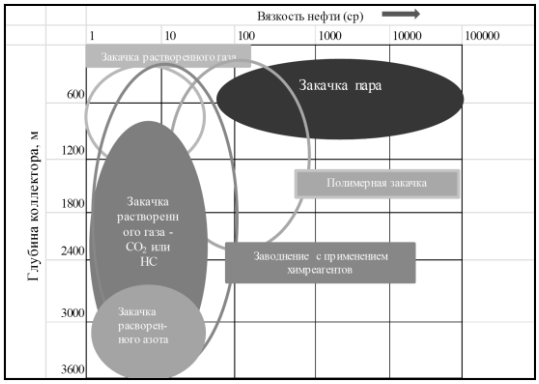


Рис. 1 Методы повышения нефтеотдачи и обработки призабойной зоны

Трудноизвлекаемые запасы природных углеводородов техногенного происхождения образуются в результате выработки запасов при помощи традиционных методов или же в результате нарушения правил разработки с применением этих методов.

В результате применения первичных методов разработки в пласте остается существенная доля не вытесненной из пласта нефти, получившая название «остаточная нефть». Проблема повышения коэффициента нефтеотдачи этих месторождений напрямую связана с уменьшением доли «остаточной нефти».

Однако, технические и экономические затраты, связанные с увеличением доли добычи остаточной нефти, кратно превосходят аналогичные затраты на начальных стадиях разработки данного месторождения.

Недостатками в развитии и применении известных и новых методов вовлечения «остаточных нефтей» в разработку с целью повышения нефтеотдачи является:

1. Плохая изученность условий формирования и распределения в залежи «остаточных нефтей»;
2. Отсутствие необходимой информации по изменению коллекторских свойств пласта и изменения компонентного состава и физико-химических свойств флюидов.

Исходя из вышесказанного, следует, что все исследования необходимо проводить систематически, регистрируя динамику изменения параметров, характеризующих остаточные нефти. На основании этих данных необходимо создавать **адекватные** модели залежи, а не ограничиваться одной моделью, созданной в период проведения подсчета запасов. Так как к трудноизвлекаемым запасам причисляются все неразрабатываемые традиционными методами залежи углеводородного сырья (нефти, газа и газоконденсата), то, в первую очередь, необходимо учитывать условия и историю их образования.

Как известно, от качества геологического и технологического обеспечения проектирования разработки зависит эффективность (включая стоимость) применяемых в данных условиях технологий, причем, не только первичных методов воздействия на пласт, но и последующих методов воздействия на остаточные нефти.

Создание новых методов и технологий по вовлечению в разработку залежей с трудноизвлекаемыми запасами, как природных, так и техногенного происхождения, основывается на подробном изучении нефтенасыщенной залежи с точки зрения получения традиционного комплекса параметров, необходимых при проектировании разработки. Однако здесь и происходит информационная нестыковка, т.к. традиционные методы разработки располагают математическими моделями для проектирования именно традиционными методами разработки, основанными на лабораторных и промысловых исследованиях, выполняемых с применением стандартных методов исследований, имеющих большое количество ограничений по видам и диапазону исследований.

Таким образом, принятые методики исследования и соответственно используемая аппаратура (как лабораторные приборы, так и промысловые глубинные приборы) не всегда могут быть правильно использованы в исследовательских работах. Такое положение в исследованиях залежей трудноизвлекаемых пластовых углеводородов приводит к получению неоднозначных по трактовке результатам или же недостоверных данных, ошибочно полученных по методикам, неприменимым к данным объектам исследования.

Отсутствие грамотного и обоснованного подхода к комплексному изучению трудноизвлекаемых залежей нефти приводит к тому, что каждая группа исследователей выполняет свою задачу не в контакте с другими специалистами, а как узконаправленную задачу.

В настоящее время отсутствует единая программа исследований, которая позволит всеми доступными современными средствами изучать параметры трудноизвлекаемых нефтей из нефтенасыщенной залежи и получать комплексное описание продуктивного пласта.

Представленный в качестве иллюстрации рисунок «слона», рис. 2, которого старательно изучают незрячие специалисты различного профиля, не только показывают отсутствие контактов между исследователями, но и взаимосвязи между проводимыми ими исследованиями.

Пятеро слепых мудрецов безудержно кричали друг на друга. Каждый думал, что только он прав и знает, на что похож слон. Никто не хотел слушать то, что говорят ему другие.

Рис. 2 Трудноизучаемые/ Трудноизвлекаемые запасы природных углеводородов

Как уже было сказано выше, изучение объекта (скважины, пласта, залежи, месторождения) необходимо проводить на протяжении всего периода разработки в динамическом режиме, используя современные методы получения и хранения информации. Так, вся исходная промысловая информация и полученная при экспериментальных исследованиях нефти, должна быть занесена в Банк данных для оперативного использования [2, 3].

В рамках принятой в декабре 2018 года «**Стратегии развития минерально-сырьевой базы Российской Федерации до 2035 года»,** указано на необходимость создания «Информационного обеспечения развития минерально-сырьевой базы Российской Федерации». Это подразумевает, что первичная и интерпретированная информация собирается и накапливается в единой системе фондов геологической информации, а в цифровом виде – в федеральной государственной информационной системе и т.д. Доступ в режиме он-лайн к информационным геологическим ресурсам различного уровня предусматривает переход на цифровое управление недропользованием в соответствии с требованиями экономики: для анализа массивов данных с целью обеспечения мониторинга изменения параметров условий залегания, степени выработанности месторождений, изменения физико-химических свойств пластовых флюидов.

Повышение нефтеотдачи месторождений, эксплуатируемых традиционными методами, – перспективный путь развития нефтяной отрасли

Проблема повышения нефтеотдачи стоит на повестке дня отечественной нефтедобывающей отрасли уже более 40 лет. В рамках этой проблемы был выделен ряд направлений, учитывающий основные (апробированные) и создание новых технологий повышения нефтеотдачи, которые можно отнести к физическим, химическим и синтетическим при совместном воздействии различных методов (таблица 1).

Все вышеназванные методы разделяются также на конкретные технологические методики, которые созданы, исходя из типа воздействия, а именно, на какой нефтенасыщенный объект направлено воздействие. Фактически их три:

Первый тип - довытеснение остаточной нефти из «промытых зон»;

Второй тип – вовлечение в разработку зон, участков пласта или пропластков по тем или иным причинам не участвующим в фильтрации.

Третий тип – разработка месторождений высоковязких и битуминозных нефтей, а также керогеновых нефтей, требующих создания оригинальных типовых методов.

По первому типу объектов

С точки зрения качества дополнительно добытой нефти, следует отметить, что она всегда будет отличаться по составу и физико-химическим свойствам от первоначальной, полученной из одной и той же залежи. Происходит это, прежде всего, в связи с длительным воздействием на нефть различных факторов техногенного характера:

1) изменения термобарических условий эксплуатируемого пласта и, как следствие, изменение фазового равновесия пластовой нефти, приводящему к образованию 2-х или 3-х фазового состояния жидкая нефть + газ или жидкая нефть + газ + твердая фаза. Твёрдой фазой являются асфальто-смоло-парафиновые отложения АСПО (в нефтяной практике используется термин «парафин»).

2) изменения в связи с высоким уровнем обводнения, т.к. при этом происходит перераспределение компонентов, содержащихся в растворенном виде в нефти и пластовой воде. (Учитывая большие объемы контактирующей с остаточной нефтью пластовой воды, эффект весьма значительный, и, кроме того, в связи с высоким обводнением активно происходят окислительные процессы в нефти с участием микробов и с образование сероводорода).

3) В обводненных залежах активно применяются химические методы воздействия на пласт, что также приводит к изменению состава и физико-химических свойств дополнительно добытой нефти.

Данное направление развития нефтяной отрасли является, несомненно, очень важным и необходимым на сегодняшний день, тем более что повышение КИН является для всех нефтяных компаний трудно достигаемым результатом.

По второму типу объектов –

С остаточными запасами, ранее не вовлеченными в разработку, в основном применяются физические технологии повышения нефтеотдачи (новые участки, пропластки, не включенные в фильтрацию). Это гидроразрыв, приводящий к протеканию нефти по заводненному пласту, или же тепловые методы, применяемые для высоковязких нефтей или же в малопроницаемых пластах.

Развитие этого направления является **«экстенсивным путем развития»**, т.е. оно сводится к наращиванию производственных мощностей на прежней сырьевой базе.

Ввиду отсутствия перспективы быстрого увеличения сырьевой базы, уже сейчас возможно создание комплексной методики повышения нефтеотдачи на эксплуатируемых месторождениях через внедрение современных технологий, основанных на лабораторных и промысловых гидродинамических исследованиях, позволяющих выявлять зоны пласта с не включёнными в разработку пластами.

При экстенсивном пути развития рост добычи нефти достигается за счет количественного увеличения производства на уже существующей технической базе. Т.е. увеличение производства обеспечивается за счет новых технологий.

На практике развитие нефтяной отрасли в целом не планируется на преобладание одного из указанных выше путей развития, т.е. нефтяная промышленность в различные периоды проходит и интенсивный, и экстенсивный путь, или же эти два пути сочетаются в различных соотношениях.

В настоящее время экстенсивный путь без активного поиска и введения в эксплуатацию новых месторождений оценивается как наиболее реализуемый, из-за отсутствия перспективы открытия новых крупных месторождений.

По третьему типу объектов вопросы реализации добычи трудноизвлекаемых запасов термическими методами для высоковязких и битуминозных нефтей, запасов керогеновых нефтей в данной работе не рассматриваются, т.к. они априори являются трудноизвлекаемыми и составляют самостоятельное направление.

Работа выполнена при поддержке Программы ФНИ государственных академий наук на 2013 – 2020 гг., проект № 0065-2019-0019.

TO THE QUESTION ABOUT THE DEVELOPMENT OF STRANDED

K.D. Ashmyan, S.G. Volpin, O.V. Kovaleva

Abstract: The article analyzes the state of oil reserves in Russia. It is shown that at present most of them belong to the category of hard-to-recover. Two types of hard – to – recover reserves are shown: 1-natural and 2-technogenic. The main problems in the development of these fields are considered. The necessity of creation of the generalizing program of research of oil fields with hard-to-recover reserves is shown.

Keywor: hard-to-recover reserves, reservoir properties, abnormal properties of fluids and rocks, residual oil, well and reservoir studies, physical and chemical studies, oil recovery technology.

Литература

1. Сургучев М.Л. Вторичные и третичные методы увеличения нефтеотдачи. М. Недра, 1985.- 308с.

2. Закономерности распределения начальных свойств нефти на основе анализа данных «Оперативного банка по физико-химическим свойствам нефти» и картопостроения на примере двух месторождений// Повышение эффективности разработки месторождений трудноизвлекаемыми запасами нефти. Сб. научных трудов ОАО «ВНИИнефть». // Выпуск №153. 2015. - С. 47 - 64

3. Оптимизация исследования пластовых флюидов. // Сб. тр. АО "ВНИИнефть", «Технологии разработки трудноизвлекаемых запасов углеводородов» Вып. 155. 2016, с.68-85

4. Стратегии развития минерально-сырьевой базы Российской Федерации до 2035 года. Утверждена распоряжением Правительства Российской Федерации от 22 декабря 2018 года., № 2914-р. Глава X.

Средства контроля энергоэффективности систем на кристалле для приложений реального времени

К.А. Петров[1], С.А. Сидоров[2]

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»,
e-mail's: [1]petrovk@cs.niisi.ras.ru, [2]sidorov@niisi.msk.ru

Аннотация: Приведен обзор методов и средств контроля энергоэффективности разрабатываемых электронных компонентов, прежде всего сложных систем на кристалле, предназначенных для работы в качестве управляющих в системах реального времени.

Ключевые слова: энергоэффективность, система на кристалле

Введение

В настоящее время системы на кристалле (СнК), представляющие собой сверхбольшие интегральные схемы (СБИС), за счет высокой степени интеграции являются основным типом вычислительной системы, предназначенным для выполнения широкого диапазона задач. Наибольшее применение СнК имеют в области управляющих систем, функционирующих в реальном масштабе времени. Огромная емкость рынка систем на кристалле привлекает множество разработчиков и производителей. Обеспечение конкурентоспособности электронной компонентной базы (ЭКБ), прежде всего для специальных применений, делает необходимым для разработчиков уделять внимание как технологическим, так и экономическим аспектам микросхемы, куда входят надежность и технологичность, низкая себестоимость производства, требуемая производительность, широкий набор выполняемых функций, низкое энергопотребление и др. Еще 15-20 лет назад для потребителей ЭКБ основными параметрами являлись производительность, набор выполняемых функций и надежность. Нынешние технологии проектирования и изготовления микросхем в целом обеспечивают требуемый уровень этих характеристик для большинства применений. Но растущие степень интеграции и производительность СБИС СнК приводят к росту рассеиваемой ими мощности, как общей, так и удельной. В настоящее время рассеиваемая мощность является основным ограничителем возможностей СнК и устройств на их основе. Например, в работе [1], отмечается, что повышение тактовой частоты центрального процессора персонального компьютера со 100 МГц в 1994 до 4200 МГц в 2007 привело к возрастанию рассеиваемой им мощности с 10 Вт до 90 Вт, а удельной рассеиваемой мощности с 0.09 Вт/мм до 0.62 Вт/мм. Такие показатели приводят к росту рабочей температуры СнК, который влияет на вероятность отказа СБИС в среднем в 2 раза на каждые 10° С [2], а также увеличивают токи утечек и снижают быстродействие активных элементов.

Охлаждение микросхемы при применении специально спроектированных корпусов и систем охлаждения существенно повышает весовые и габаритные параметры электронных устройств, составной частью которых являются СнК, зачастую сводя к нулю все преимущества от повышения производительности.

В связи с этим одной из актуальных задач развития методологии проектирования современных цифровых СБИС сложных микропроцессорных систем является разработка комплексных мер повышения их энергоэффективности при сохранении других технологических и экономических параметров. СнК, выполненные по суб100-нм технологиям, содержат в себе микропроцессорные ядра, запоминающие устройства, а также значительное количество периферийных и вспомогательных устройств. В зависимости от функционального назначения СнК, на сложно-функциональные блоки может приходиться более 70% энергопотребления, а именно на запоминающие и арифметико-логические устройства, элементы управляющей логики, устройства синхронизации и др. [3].

Одним из основных направлений развития методологии снижения энергопотребления является применения технологий с несколькими пороговыми напряжениями, варьированием технологических параметров и др., что позво-

ляет снизить напряжение питания СБИС, а также токи утечки как активных, так и паразитных элементов.

Снижение энергопотребления СБИС является сложной комплексной задачей, имеющей решения на каждом уровне маршрута проектирования микросхемы. Эти решения делятся на несколько типов в зависимости от уровня проектирования СБИС: для архитектурного и системного уровней проектирования это алгоритмические методы, а для функционального уровня – схемотехнические и топологические.

Согласно данным, приведённым в работе [4], энергопотребление СБИС может отличаться в 20-100 раз в зависимости от алгоритмических решений, принятых только на системном и архитектурном уровнях. Поэтому так важна методология проектирования, позволяющая на самых ранних этапах разработки проводить исследования энергоэффективности проектов сложных микропроцессорных систем и оценивать полученные результаты. В то же время, разработчику СнК специального назначения, где изначально заданы весьма жесткие требования, остается ограниченный выбор вариантов проектирования, зато возрастает роль исследований энергоэффективности и надежности получаемых образцов.

Понятие энергоэффективности микропроцессорных систем

Сложные микропроцессорные системы обычно строятся в виде СнК, объединяя в одной СБИС как необходимые вычислительные ресурсы – одно или несколько процессорных ядер, а часто и групп (кластеров) ядер, интерконнект, вспомогательные устройства, контроллеры ввода-вывода, встроенную память. Высокая степень интеграции приводит к увеличению энергопотребления, значительная часть которой переходит в тепло, что отрицательно сказывается на производительности и надежности системы.

Важнейшей задачей при разработке ЭКБ, в частности СнК, является достижение высокого быстродействия при низком энергопотреблении. Эти требования являются взаимоисключающими, т.к. повышение быстродействия всей СнК требует увеличения частоты работы составных частей, что приводит к увеличению суммарных энергозатрат, а общее снижение быстродействия СнК, наоборот, уменьшает ее энергопотребление. Снижение энерговыделения СнК СБИС обеспечивается путем ряда методик, которые делятся на конструктивные, схемотехнические и технологические.

Энерговыделение СБИС подразделяют на несколько составных частей, вносящих свой вклад, согласно следующей классификации [5]:

- энерговыделение, обусловленное токами утечек в статическом режиме работы микросхемы;
- энерговыделение, обусловленное протеканием токов в динамическом режиме работы микросхемы;
- энерговыделение, обусловленное формированием логического состояния элемента.

Энерговыделение, обусловленное токами утечек в статическом режиме работы микросхемы, определяется токами утечки транзисторов в отсутствии фронта тактового сигнала. Для микросхем, выполненных по технологиям более 65 нм, вклад токов утечки в статическом режиме работы намного меньше общей потребляемой микросхемой мощности и их вклад незначителен [6]. Для микросхем, выполненных по технологиям 65 нм и менее, вклад в энерговыделение токов утечки в статическом состоянии становится уже заметным. А для микросхем, выполненных по технологиям 45 нм и ниже вклад токов утечки становятся значительными.

Эффективным топологическим методом снижения статического энерговыделения, обусловленного токами утечки транзисторов, является создание библиотеки цифровых логических элементов на базе транзисторов с несколькими пороговыми напряжениями. Но такие транзисторы обладают меньшей нагрузочной способностью. Как следствие, их использование увеличит время прохождения сигналов в логических элементах и, соответственно, отрицательно скажется на производительности СБИС в целом.

Уменьшение статического энерговыделения схемотехническими методами достигается последовательным включением в стоковую цепь логических элементов дополнительных КМОП-транзисторов. Это ограничивает величину тока стока и, следовательно, снижает токи утечек логических элементов. При частоте переключения логических элементов не более нескольких сотен МГц этот метод имеют наибольшую эффективность – уменьшает статическое энерговыделение в 2–5 раз [7].

В момент переключения логического элемента происходит смена его внутренних состояний и перезаряд внутренних емкостей. В процессе перезаряда возникают динамические токи, приводящие к динамическому энерговыделению. Значения этих внутренних емкостей определяются размерами элементов и их межсоединений [8, 9].

Классическая формула расчета энергопотребления выглядит так:

$$P \sim K * C * V^2 * f \qquad (1)$$

где P – мощность системы, C – суммарная емкость схемы, V – напряжение питания, f – частота, а коэффициент K представляет активность системы и отражает степень нагрузки (количество переключений элементов) в зависимости от типа выполняемой задачи.

Основным принципом любого метода снижения энергопотребления СнК является уменьшение значения одного или нескольких параметров, входящих в качестве сомножителей в выражение (1). И для каждого из представленных сомножителей необходимо найти оптимальное значение с учетом потерь в областях прочих её параметров или даже в системных характеристиках, возникающих в результате применения методов снижения энергопотребления.

Снижение проектных норм производства микросхем приводит к уменьшению топологических размеров элементов, величин их емкостей и, как следствие, динамического энергопотребления. Однако в секвенциальной логике это приводит к уменьшению сбоеустойчивости, что является важным параметром СБИС, предназначенных для применения при воздействии внешних факторов. Определение оптимальных топологических размеров для каждого конкретного приложения библиотечных элементов обеспечивается исследованием предполагаемых условий функционирования данной СБИС.

Зависимость динамического энергопотребления от квадрата напряжения питания V определяет высокую эффективность снижения V. Это является определяющим фактором существующей тенденции снижения напряжения питания микросхем от классических значений в 5 В до более современных 1-1,2 В и ниже. У такого решения есть ограничение, а именно снижение значения порогового напряжения транзисторов. Это вызывает уменьшение помехоустойчивости элементов, что, в свою очередь, снижает диапазон температурных, электромагнитных и прочих условий, в которых возможно функционирование данной СБИС.

Напряжения питания, обеспечивающие исполнение операций с минимальной и максимальной скоростью, могут различаться в разы. Это даёт возможность разработчику применять методики динамического управления значением напряжения питания микросхемы или её отдельных блоков в разных режимах работы с целью уменьшения суммарного динамического энергопотребления. Для повышения быстродействия отдельных блоков при необходимости используют локальное повышение напряжения питания и/или изменяют значение порогового напряжения транзисторов, снижая его до предельно допустимых значений.

Также существует ряд методик, направленных на снижение энергопотребления посредством программного обеспечения [10]. Дополнительно решению этой же задачи способствует набор технологических приемов, таких, как например, использование технологии производства интегральных схем «кремний на изоляторе» [11], которая отличается от технологии объемного КМОП меньшими величинами емкостей топологических элементов при сравнимых геометрических размерах.

Суммируя вышесказанное, методы снижения энергопотребления можно классифицировать как направленные на снижение напряжения питания, снижение тактовой частоты переключения отдельных элементов и снижение суммарной емкости схемы.

Методы оценки и снижения энергопотребления сложных систем

Анализ энергопотребления цифровых схем может быть выполнен с разной степенью детализации. Более высокие уровни абстракции позволяют выполнять оценки значительно быстрее и пригодны для очень больших проектов, но при этом дают менее точные результаты. И наоборот, оценки, производимые на уровнях, близких к физическому, близки к реальности с точностью до нескольких процентов.

Можно выделить следующие подходы:

- оценка энергопотребления на низком уровне - уровне схемы, с использованием систем автоматизации проектирования (САПР) и симуляторов;

- оценка потребления, использующая динамическую потоковую информацию (так называемые «трассы») о работе устройства – задействованные узлы, типы операций и пр., а также основанная на анализе статистических данных;

- оценка потребления в действующих системах и динамическое управление напряжением питания и частотой (DVFS).

Ниже приведен обзор САПР, использующихся в том числе для моделирования энерговыделения СБИС на разных уровнях.

Моделирование на функциональном уровне

Одной из наиболее известных САПР, поддерживающих оценку энергопотребления, можно считать SPICE (Simulation Program with Integrated Circuit Emphasis) [12, 13] – универсальный симулятор цифровых схем, обеспечивающий в том числе и оценку энергопотребления. Выполняет оценку на основе т.н. SPICE-моделей элементов схемы, моделируя NetList. Весьма распространенный инструмент, не в последнюю очередь из-за своей доступности, т.к. относится к свободно распространяемому программному обеспечению. Позволяет получить достаточно точные оценки, однако из-за необходимости моделирования NetList работает крайне медленно, и пригоден к применению лишь на довольно поздней стадии разработки, после того, как выполнено логическое проектирование и логический синтез.

Известный с начала 1970-х годов, симулятор SPICE стал основой для разработки целого ряда подобных САПР, как в академической, так и в промышленной областях применения. SPICE стал популярен благодаря встроенной поддержке анализа и наличию моделей элементов, необходимых для проектирования актуальных на тот момент интегральных схем, и при этом был достаточно быстродействующим при использовании его на вычислительных мощностях того времени. Индустрия разработки интегральных схем довольно рано начала пользовать SPICE, и до момента развития коммерческих реализаций аналогов многие крупные компании-разработчики микросхем имели собственные проприетарные версии этой САПР. В настоящий момент крупные производители микросхем развивают собственные САПР с функцией симуляции на основе SPICE, такие как ADICE у компании Analog Devices, Mica у Freescale Semiconductor, LTspice у Linear Technology и TISPICE у Texas Instruments.

Encounter Power System – унифицированный инструмент для оценки энергопотребления, интегрированный в семейство САПР от Cadence. Работает как с RTL, так и с NetList моделями. Позволяет проводить детальный анализ энергопотребления по дизайну в целом и по его составным частям, с учетом температуры [14]. Система построена на проверенных практикой и качественных алгоритмах, использованных для тысяч успешных проектов. Измеряет динамическое и статическое потребление в сочетании с высокой производительностью, емкостью и точностью.

Synopsys Power Compiler позволяет выполнять оптимизацию по энергопотреблению на уровне RTL и NetList, естественно предоставляя возможности измерения и оценки потребления. Применяется прежде всего для оптимизации на уровне логической модели [15]. Этот инструмент предназначен для анализа, а также схемотехнической оптимизации и синтеза логических схем с пониженным энергопотреблением, эффективных с функциональной точки зрения. Он является основной для виртуальной платформы Eclipse, содержащей в себе готовые инструменты для снижения потребления энергии. Работая в связке с Design Compiler или Physical Compiler, этот инструмент предоставляет возможность оптимизации одновременно по таймингу, площади и энергопотреблению. Оптимизация по энергопотреблению основана на вычислении среднего потребления по активности компонентов дизайна.

Synopsys PowerMill – симулятор энергопотребления уровня транзисторов для CMOS и BiCMOS дизайнов. Дает точные результаты по потреблению и расширенную диагностику, необходимую для оптимизации энергопотребления.

Power Aware Verification – инструмент от Mentor Graphics, предназначенный для функциональной верификации схемотехнических решений для управления энергопотреблением на уровне RTL-моделей [16].

Известны также WattWatcher от Sente [17], PowerPlay, Mars-Rail и Mars Xtalk от Avanti и другие инструменты.

Моделирование на уровне системы

Cadence InCyte Chip Estimator – инструмент, специально разработанный в Cadence для предварительной оценки основных технических и «рыночных» параметров разрабатываемых микросхем: размера кристалла, энергопотребления, производительности и цены. Оценка выполняется на основе спецификации и характеризованной коллекции IP-блоков, на основе которых разрабатывается микросхема. Весьма мощный инструмент, не учитывающий, однако, целевое назначение и предполагаемый класс решаемых задач [18].

Первым известным продуктом для быстрой оценки энергопотребления, площади и

тайминга был CACTI [19], в котором применялись модели устройств на базе индустриального стандарта ITRS [20]. CACTI позволял также находить конфигурации с минимальным энергопотреблением с заданными ограничениями на площадь и тайминг.

Широкое применение нашел Wattch [21]. Этот инструмент оценивает динамическое энергопотребление по событиям, получаемым из архитектурной симуляции и емкостных моделей компонентов микроархитектуры. Wattch позиционировался сообществом компьютерных архитекторов как стандартная опция для дополнения САПР, но его ограничения быстро стали очевидными: не рассматривались площадь кристалла и тайминг, оценивалась только динамическое потребление, применялась линейная модель потребления, ориентированная на технологию 800 нм.

Orion [22] предназначен для моделирования энергопотребления в NoC. Система позволяет рассматривать площадь и динамическое потребление, однако не охватывает тайминг.

Один из наиболее распространенных сегодня инструментов для оценки энергопотребления на уровне системы – McPAT (Multicore Power, Area, and Timing) [23, 24, 25], предназначенный для моделирования энергопотребления, площади кристалла и тайминга. McPAT ориентирован на мультиядерные и многоядерные процессоры, в том числе сгруппированные в кластеры, для проектных норм от 90 нм до 22 нм и ниже (под мультиядерными понимаются процессоры, включающие 2 и более однородных ядер, возможно, сгруппированные в кластеры; под многоядерными же понимаются процессоры, в которых помимо универсальных или управляющих ядер используются также ядра, предназначенные для специализированных вычислений – такие как векторные, графические, вещественные и пр. сопроцессоры [26]). На микроархитектурном уровне McPAT содержит модели для базовых компонентов микропроцессоров. На схемном и технологическом уровне McPAT поддерживает моделирование тайминга на критических путях, моделирование площади, моделирование всех видов энергопотребления. В совокупности с симулятором, позволяющим оценивать производительность, McPAT дает возможность разработчику адекватно оценивать стоимость новых идей и выбирать оптимальные архитектурные решения.

Более простой инструмент, использующий те же исходные данные, что и McPAT, разработан в NTNU – Trindheim, Norvegian University of Science and Technology [27]. Он получил название PET – Power Estimation Tool. Этот программный инструмент способен оценивать энергопотребление за определенное время для заданной нагрузки и конкретной архитектуры. PET использует заданные энергетические метрики и логи симуляции – трассы. Работает очень быстро и эффективно, позволяя оперативно оценивать те или иные архитектурные решения.

DVFS

Динамическое масштабирование напряжения питания и частоты (Dynamic Voltage and Frequency Scaling – DVFS) представляет собой широко используемую технологию управления энергопотреблением, в которой частота процессора или иного узла системы с выделенным тактовым сигналом снижается, чтобы сделать возможным соответствующее понижение питающего напряжения. За счет этого заметно снижается энергопотребление.

Большинство современных систем на кристалле, как мощных серверных, так и встраиваемых в мобильные устройства, и многопроцессорных чипов имеют встроенную аппаратную поддержку технологии DVFS. Операционные системы в свою очередь содержат программное обеспечение для управления этими возможностями.

Применительно к сложным микропроцессорным системам технология DVFS дает весьма ощутимый результат. Эффективность ее тем выше, чем более «гранулярно» ее применение. В пределе, возможность управлять частотой и напряжением питания каждого микропроцессора в многопроцессорном чипе должно дать наилучший результат, однако при значительном числе ядер количество накристальных регуляторов напряжения становится запредельным. Наиболее разумным подходом является кластеризация всего множества ядер с управлением питания для каждого кластера [28, 29, 30].

Множество работ посвящено выбору оптимальных с точки зрения производительности алгоритмов управления напряжением и частотой применительно к различным классам задач, а также адаптивных алгоритмов, учитывающих динамику вычислительного процесса [31, 32, 33, 34].

Комплексное и нагрузочное тестирование

Не менее важно в разработке СнК комплексное тестирование, одним из видов которого можно считать нагрузочное тестирова-

ние. Под комплексным тестированием понимается проверка функционирования сложной СнК в условиях совместной и/или одновременной работы всех или большинства входящих в его состав устройств, как коммуникационных контроллеров, так и вычислительных узлов. Основная задача комплексного тестирования – верификация взаимодействия различных компонентов с точки зрения корректности информационного обмена, достаточности производительности и пропускной способности каналов передачи информации для решения заданного класса задач. Такое тестирование позволяет выявить узкие места в системе и оценить возможности исследуемого устройства.

При наличии готовых образцов и ли макетов комплексное тестирование традиционно выполняется в среде операционной системы, либо Linux, для которой создано великое множество разнообразных тестов, либо целевой, предназначенной для применения с этой СнК. Для сложных многофункциональных систем, таких, например, как мультимедийные процессоры, построение комплексных тестов становится самостоятельной задачей, перекрывая по ресурсоемкости все целевое тестирование, выполненное на этапе разработки.

Стресс-тестирование, или нагрузочное тестирование, предназначено для исследования стабильности и надежности функционирования устройства вне пределов условий нормального функционирования. Стресс-тесты чаще всего направлены на отдельные узлы или подсистемы (вычислитель, коммутатор каналов, дисковая подсистема и т.п.), реже на все устройства целиком [35, 36]. Нагрузочное тестирование можно отнести скорее к этапу испытаний, поскольку выполняется обычно на готовом устройстве и позволяет измерить энергопотребление, рассеиваемую в виде тепла мощность, реальную наработку на отказ в экстремальных условиях, оценить поведение и устойчивость в граничных режимах.

В ряде случаев комплексное тестирование можно в какой-то степени провести и на этапе разработки логической модели, хотя и в гораздо меньшем объеме.

Один из инструментов для комплексного тестирования – Perspec System Verifier от Cadence Design Systems [37]. Perspec – это программный комплекс для тестирования логических моделей СнК. Целью его было создание универсальной платформы для автоматизации процесса тестирования СнК по следующим направлениям:

- моделирование сценариев использования тестируемой системы;
- проверка совместной работы компонентов СнК;
- нагрузочное тестирование;
- генерация наборов случайных тестов.

Perspec дает возможность оперировать аппаратно-независимыми абстракциями устройств из состава СнК, составляя из них различные сценарии поведения системы. Реализация абстракций выполняется на языке SLN, предназначенном для упрощения работы с платформозависимым кодом, и поддерживает описание реализаций компонентов системы на языках Си, ассемблер и Verilog, которые могут различаться от изделия к изделию, при этом сохраняя возможность запуска уже созданных сценариев тестирования. Такой подход позволяет разработчику сценариев не вникать в особенности аппаратуры и сосредоточиться на высокоуровневых аспектах тестирования и моделирования поведения.

В составе Perspec имеется набор инструментов, поддерживающих различные этапы разработки комплексных тестов и собственно тестирования.

SLN – декларативный язык программирования, позволяющий описывать исследуемую систему для задач автоматической генерации платформозависимого кода и автоматизации проверок. Язык описывает компоненты тестируемой системы, определяет методы взаимодействия данных компонентов, после чего данные описания используются для автоматической генерации тестовых сценариев.

Центральная компонента, именуемая собственно Perspec, предназначена для генерации кода из подготовленных описаний на языке SLN и платформозависимых функций, реализующих взаимодействие с конкретной аппаратурой (по сути, драйверов и тестов со стандартизованным системным интерфейсом).

Composer – графическая оболочка, дающая возможность наглядного конструирования множества тестовых сценариев и автоматической генерации тестового кода с учетом перебора параметров, визуализации конечного сценария.

И, наконец, Analyzer – программа-анализатор, вызываемая после запуска сценария, позволяющий узнать, какие действия были выполнены, как часто, в каком порядке, и пр.

Perspec System Verifier применяется и для нагрузочного тестирования, причем механизм варьирования параметров при генерации тестов обеспечивает тестировщику возможность разностороннего исследования устройства. Однако применение нагрузочных тестов не на модели, а на реальной аппаратуре оказывается

весьма нетривиально, поскольку изначально не проектировалось разработчиками.

KMD-Stress

С целью прежде всего нагрузочного тестирования всей СнК в целом разработана подсистема Stress в рамках многоплатформенной тестовой системы KMDTESTKIT [38]. Stress предназначена для работы на электронном модуле, представляющем собой СнК и необходимое окружение (память, приемопередатчики интерфейсов и пр.), т.е. на простых отладочных и исследовательских платах.

Поддерживаются два режима тестирования: одновременная работа различных функциональных узлов и режим максимальной нагрузки. Различие режимов достаточно условное, наиболее существенно то, что в режиме одновременной работы включены некоторые проверки правильности функционирования, а в режиме нагрузки все нацелено на нагружение узлов и никаких проверок не выполняется.

Реализуются оба режима задействованием для одновременной работы максимального числа компонентов СнК – контроллеров периферии, внутренних устройств (таймеры, DMA, и пр.), центрального процессора и сопроцессоров. Информационного взаимодействия компонентов СнК не предполагается, все работают автономно.

Нагрузочное тестирование проверяет одновременную работу аппаратуры, позволяет исследователю контролировать (внешними приборами) температурный режим и энергопотребление.

Подсистема представляет собой несложный генератор, который по заданным в конфигурационном файле параметрам (перечень тестов, уровень проверки правильности функционирования и пр.) генерирует исходный текст теста из имеющихся заготовок-фрагментов для каждого узла и затем просто компилирует его для текущей платформы.

Идея построения стресс-теста состоит в том, что любой тест можно разделить на три части: пролог (инициализация, подготовка), собственно тест и эпилог (выключение, диагностика). Генератор собирает сначала все функции-прологи, затем организует вызов в цикле последовательности тестовых функций, и наконец все эпилоги. Каждая тестовая функция также весьма проста: если тестируемый узел еще не завершил предыдущее действие, то она заканчивается до нового вызова на следующем цикле, уступая место очередной функции; если узел освободился, он запускается снова, и функция также заканчивает работу. По окончании работы, который задаётся либо количеством циклов, либо внешним сигналом, вызываются все эпилоги и тестирование завершается.

Такой способ построения стресс-теста позволяет свести к минимуму накладные расходы на организацию тестирования, которые в случае, например, применения какой-либо операционной системы вырастают на порядки. С целью уменьшения накладных расходов в подсистеме Stress не используются прерывания, поскольку обслуживание прерываний заметно снижает содержательную нагрузку на СнК.

Для получения адекватных результатов необходимо внимательно подходить к выбору набора узлов, компонуемых в тест. Для проверки одновременной работы целесообразно включать все, а для нагрузочного тестирования целесообразно отключать простые медленные устройства, например, UARTили I2C, которые почти ничего не дают для нагрузки, но требуют ресурсов на обслуживание.

Время автономной работы, от запуска до запуска, коммуникационного контроллера, например, Ethernet, может многократно превышать время выполнения основного цикла тестирования нагрузочного теста. Это означает, что цикл будет крутиться вхолостую, занимаясь только проверкой готовности контроллеров. Для более полного нагружения СнК в цикл вставляется модуль, выполняющий вычисления с плавающей точкой. Его размер подбирается опытным путем на основе статистического профилирования выполнения стресс-теста.

Заключение

Приведенный обзор иллюстрирует важность и многогранность проблемы создания энергоэффективных электронных компонентов. Большое разнообразие инструментов, предназначенных для оценки и измерения мощности СнК, моделирования и оптимизации энергопотребления, а также управления энергопотреблением в процессе работы вычислительной или управляющей системы в реальном масштабе времени предоставляет как разработчику ЭКБ, так и архитектору конечной системы множество возможностей для построения энергоэффективных продуктов. И разработка новых инструментов, как универсальных, так и специализированных, как сложных и мощных, так и простых, не прекращается.

Публикация выполнена в рамках государственного задания по проведению фундамен-

тальных научных исследований по теме «Архитектурные и схемотехнические методы снижения энергопотребления и повышения сбоеустойчивости микропроцессоров и коммуникационных контроллеров высокопроизводительных ЭВМ» (No 0065-2019-0008).

Power efficiency tools for system-on-chip real-time applications

K.A. Petrov, S.A. Sidorov

Abstract: Described methods and tools for power efficiency analysis in complex system-on-chip and real-time applications

Keywords: power efficiency, system-on-chip.

Литература

1. P.E. Gronowski et al. High performance microprocessor design // IEEE J.Solid-State Circuits. -vol.33. -№5. -pp.676-686.

2. JLRabaey, M.Pedran. Low power design methodologies // Kluwer academic publishers. Thud printing. -1997. -368c.

3. В.Немудров Г.Мартин. Системы-на-кристалле. Проектирование и развитие // Техносфера Москва -2004. -216с.

4. A. Krishnamoorthy. Minimize 1C power without sacrificing performance // EEdisign. -2004. -№5.

5. Бобков С.Г., Мадера А.Г. Энергетические затраты, быстродействие и проблема теплоотвода в микропроцессорах // Программные продукты и системы. 2013. № 4.

6. Адамов Д. Учет особенностей микроэлектронных нанотехнологий при проектировании СБИС // Электроника: Наука. Технология. Бизнес. 2007. № 7. С. 98–105; 2007. № 8. С. 114–118.

7. Abdollahi A., Fallah F., Pedram M. Leakage Current Reduction in CMOS VLSI Circuits by Input Vector Control, Proc. ISLPED, 2002.

8. Chandrakasan A., Sheng S., Broersen R. Low-power CMOS Digital Design, IEEE Journ. of Solid-State Circuits, 1999, vol. 27 (4), pp. 473–484.

9. Chatzigeorgiou A. and Stephanides G. Energy Issues in Software Design of Embedded Systems, 2nd WSEAS Intern. Conf. on Applied Informatics, Rethymnon, Crete, Greece, 2002, July 7–14.

10. Kougia S., Chatzigeorgiou A., Nikolaidis S. Evaluating Power Efficient Data-Reuse Decisions for Embedded Multimedia Applications: An Analytical Approach. Journ. of Circuits, Systems and Computers. February, 2004, vol. 13, no. 1.

11. Boroshko S.I., Ivanov S.G., Ivlev A.A., Ilyaguev V.N., Kalashnikov O.A., Osipenko P.N., Pekach Yu.V. Ways of reduction in absorbed current KMOP SBIS. 7 konf. “Radiatsionnaya stoykost elektronnykh sistem. Stoykost-2004” [7th conf. “Resistance to radiation for electric systems Stoykost-2004”]. Report thesis, Moscow, SPELS, 2004, iss. 7, pp. 77–78 (in Russ.).

12. SPICE Simulation or other CAD Tools // http://jas.eng.buffalo.edu/education/mos/mosfet/mosfetCAD.html

13. Vladimirescu, Andrei, SPICE — The Third Decade, Proc. 1990 IEEE Bipolar Circuits and Technology Meeting, Minneapolis, Sept. 1990, pp. 96-101

14. Encounter Power System // http://www.cadence.com/products/di/eps/Pages/default.aspx

15. Synopsys Power Compiler // http://www.synopsys.com/TOOLS/IMPLEMENTATION/RTLSYNTHESIS/Pages/PowerCompiler.aspx

16. Power Aware Verification // http://www.mentor.com/solutions/low-power/functional-verification

17. WattWatcher manual, Sente Inc. (http://www.senteinc.com)

18. Cadence InCyte Chip Estimator // http://www.cadence.com/rl/Resources/datasheets/incyte_chip_estimator_ds.pdf

19. S. Thoziyoor, J. Ahn, M. Monchiero, J. Brockman, and N. Jouppi, A Comprehensive Memory Modeling Tool and its Application to the Design and Analysis of Future Memory Hierarchies // ISCA, 2008.

20. Semiconductor Industries Association, “Model for Assessment of CMOS Technologies and Roadmaps (MASTAR),” 2007, http://www.itrs.net/models.html.

21. D. Brooks, V. Tiwari, and M. Martonosi, “Wattch: a framework for architectural-level power analysis and optimizations,” // ISCA, 2000.

22. A. Kahng, B. Li, L.-S. Peh, and K. Samadi, “ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration,” // DATE, 2009.

23. Sheng Li, Jung Ho Ahn, Richard D. Strong, Jay B. Brockman, Dean M. Tullsen, Norman P. Jouppi. McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures // Micro, 2009

24. Wooseok Lee, Youngchun Kim, Jee Ho Ryoo, Dam Sunwoo, Andreas Gerstlauer, Lizy K. John. PowerTrain: A Learning-based Calibration of McPAT Power Models // http://lca.ece.utexas.edu/pubs/wooseok_islped.pdf

25. Jieun Lim, Nagesh B. Lakshminarayana, Hyesoon Kim, William Song, Sudhakar Yalamanchili, Wonyong Sung. Power Modeling for GPU Architecture using McPAT // http://www.cercs.gatech.edu/tech-reports/tr2013/git-cercs-13-10.pdf [26] Multi-Core vs. Many-Core, или Зачем нужны многоядерные микропроцессоры? // http://netler.ru/pc/multi-core.htm

26. Stian Hvatum,Terje Runde. Power Profiling: From Measurements to Simulation Models // http://brage.bibsys.no/xmlui/handle/11250/253833

28. Tejaswini Kolpe, Antonia Zhai, Sachin S. Sapatnekar. Enabling Improved Power Management in Multicore Processors through Clustered DVFS // http://www.ece.umn.edu/~sachin/conf/date11tk.pdf

29. Etienne Le Sueur, Gernot Heiser. Dynamic Voltage and Frequency Scaling: The Laws of Diminishing Returns // http://ssrg.nicta.com.au/publications/papers/LeSueur_Heiser_10.pdf

30. Saugata Ghose, Jonathan Tse. Memory-Aware DVFS for CMP Systems // http://www.jontse.com/papers/files/2009_memory-aware_dvfs.pdf

31. Robert Hesse, Natalie Enright Jerger. Improving DVFS in NoCs with Coherence Prediction // http://www.eecg.toronto.edu/~enright/72-Hesse.pdf

32. Nathaniel Gaskin, Amlan Ghosh, Spencer Kellis. Dynamic Voltage and Frequency Scaling in an Embedded Microcontroller SoC // http://www.eng.utah.edu/~kstevens/6770/reports/06-dvfs-report.pdf

33. Qingyang Wang, Yasuhiko Kanemasa, Jack Li, Chien An Lai, Masazumi Matsubara, Calton Pu. Impact of DVFS on n-Tier Application Performance // http://www.cc.gatech.edu/~qywang/papers/TRIOS13-Qingyang.pdf

34. С.В. Минухин, М.И. Сухонос. Алгоритмы оптимизации энергопотребления и повышения эффективности процессоров с масштабированием частоты и напряжения гетерогенного кластера // ISSN 2079 - 0023. Вісник НТУ «ХПІ». 2013. No 62 (1035)

35. Phoronix Test Suit // http://www.phoronix-test-suite.com

36. Stress Testing // https://wiki.archlinux.org/index.php/Stress_testing

37. Perspec System Verifier. Cadence Design Systems // https://www.cadence.com/content/dam/cadence-www/global/en_US/documents/tools/system-design-verification/perspec-system-verifier-ds.pdf

38. Сидоров С.А., Слепов А.Б. Система тестирования логических моделей KMDTESTKIT. // Труды НИИСИ РАН, Том 8 № 1. Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты. М.: ФГУ ФНЦ НИИСИ РАН, 2018, с.37-43 (7 стр.) ISSN 2225-7349.

2D аналитическая модель распределения потенциала полностью обедненного КМОП нанотранзистора с полностью охватывающим затвором

Н. В. Масальский

ФГУ ФНЦ НИИСИ РАН, г. Москва, Россия,

E-mail: volkov@niisi.ras.ru

Аннотация. Эффективная 2D аналитическая модель распределения потенциала в рабочей области коротко-канального полностью обедненного КМОП транзистора с полностью охватывающим затвором разработана. Модель сформулирована на основе аналитического решения уравнения Пуассона в цилиндрических координатах. Выполнены компьютерные расчеты распределения потенциала для широкого диапазона топологических параметров. Сопоставление результатов расчетов с данными моделирования программного пакета ATLAS^{TM}, доказывает приемлемое соответствие между ними.

Ключевые слова: полностью обедненный КМОП нанотранзистор, полностью охватывающий затвор, уравнение Пуассона, 2D распределение потенциала

Введение

Среди различных направлений развития транзисторных архитектур конструкция КМОП транзистора с полностью охватывающим затвором отличается наиболее эффективным электростатическим управлением канала [1]. Структура рассматриваемого транзистора показана на рис 1.

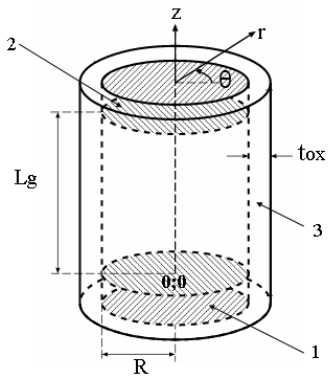


Рис. 1. Структурная схема КМОП транзистора с полностью охватывающим затвором, где 1 – исток, 2 – сток, 3 –рабочая область, 4 – подзатворный диэлектрик, обозначения: Lg – длина затвора, R – радиус рабочей области, tox - толщина подзатворного диэлектрика

Хотя данная архитектура была предложена еще в конце прошлого века, большая часть исследований была направлена на длинно-канальные устройства. Совсем недавно интерес к такой архитектуре проявился вновь в связи с тем, что для них характерно двукратное превосходство по подавлению коротко-канальных эффектов (ККЭ) по сравнению с традиционными планарными структурами. Этот интерес обусловлен наличием современной технологической базой и гигантскими вычислительными возможностями для приборно-технологического моделирования [2].

Математические модели транзисторов рассматриваемой архитектуры были предложены достаточно давно, но большая часть их часть описывает длинно-канальные транзисторы (см. библиографию [1]). Модель коротко-канального транзистора была предложена Hamid [3]. Она отличается высокой точностью, даже для структур с длиной канала Lg до 30 нм. Однако эта модель требует значительных вычислительных ресурсов. Другая модель была создана Tsormpatzoglou [4]. Эта модель сформулирована на основе квазианалитического решения уравнения Пуассона и отличается вычислительной экономичностью. Однако она применима только в режиме низкой инверсии. Поэтому, в данной работе исследуется возможность совместить эти подходы для разработки коротко-канальной модели нанотранзистора, которая применима в широком диапазоне длин затворов, и во всем допустимом диапазоне затворных напряжений.

В настоящей статье рассматривается 2D аналитическая модель распределения потенциала полностью обедненного коротко-канального КМОП нанотранзистора с полностью охватывающим затвором. Она базируется на аналитическом решении 2D уравнение Пуассона. Результаты моделирования сопоставляются с данными полученными помощи программного пакета ATLAS™ [5]

1. 2D модель распределения потенциала

Распределение потенциала $\varphi(r,z)$ в рабочей области транзистора с полностью охватывающим затвором будет являться решением уравнения Пуассона [1]. Это уравнение можно представить в 2D форме следующим образом [6]:

$$\frac{1}{r}\frac{\partial}{\partial r}(r\frac{\partial}{\partial r}\varphi(r,z))+\frac{\partial^2}{\partial z^2}\varphi(r,z)=-\frac{qN_A}{\varepsilon_S}, \quad (1)$$

где q – элементарный заряд, ε_S, N_A - диэлектрическая проницаемость и концентрация примеси в рабочей области транзистора.

Ввиду классического условия, что длина рабочей области L_g всегда в несколько раз больше «толщины» рабочей области (2R), то потенциал можно представить как суперпозицию двух: радиального и латерального, например, в виде [7]:

$$\varphi(r,z)=\varphi_{1D}(r)+\varphi_{2D}(r,z) \quad (2)$$

В данном случае 1D уравнение Пуассона для радиального направления аналогично случаю длинного канала, которое может быть выражено следующим образом:

$$\frac{1}{r}\frac{\partial}{\partial r}\varphi_{1D}+\frac{\partial^2}{\partial r^2}\varphi_{1D}=\frac{qN_A}{\varepsilon_S}(1+(\frac{n_i}{N_A})^2 e^{\beta(\varphi_{1D}-(V_n-V_p))}) \quad (3)$$

где n_i – собственная концентрация носителей, β - термический потенциал, V_n, V_p - положение уровня Ферми для электронов и дырок, соответственно. Граничные условия для (3) можно представить так:

$$\frac{\partial}{\partial r}\varphi_{1D}(r,z)=0$$

$$\varphi_{1D}(r,z)\big|_{r=R}=\varphi_f(z)$$

$$\frac{\varepsilon_{ox}}{t_{ox}}(U_g-U_{FB}-\varphi_{1D}(R,z))=\varepsilon_S\frac{\partial\varphi_{1D}(r,z)}{\partial r}\Big|_{r=R},$$

где $\varphi_f(z)$ - поверхностный потенциал, ε_{ox} - диэлектрическая проницаемость подзатворного окисла, U_g - напряжение на затворе (см. рис. 1, поз. 3), U_{FB} - напряжение плоских зон.

2D уравнение Лапласа для случая короткого канала можно представить так:

$$\frac{1}{r}\frac{\partial}{\partial r}\varphi_{2D}(r,z)+\frac{\partial^2}{\partial r^2}\varphi_{2D}(r,z)+\frac{\partial^2}{\partial z^2}\varphi_{2D}(r,z)=0, \quad (4)$$

с граничными условиями

$$\varphi_{2D}(r,0)=U_{bi}-\varphi_{1D}(r)$$

$$\varphi_{2D}(r,L_g)=U_{bi}+U_{ds}+\varphi_{1D}(r),$$

где U_{bi} - контактная разность потенциалов, U_{ds} - напряжение между истоком и стоком (см. рис. 1, поз. 1 и поз. 2).

Для того чтобы найти решение уравнения (4), необходимо использовать дизъюнкцию переменных. Однако это приводит к более сложным математическим вычислениям. Поэтому, в данном случае используется параболическое приближение для представления $\varphi_{2D}(r,z)$, которое довольно близко к дизъюнктивному подходу. Тогда выражение для $\varphi_{2D}(r,z)$ можно записать в виде [7]:

$$\psi^{2D}(z,r)=a_0(z)+a_1(z)r+a_2(z)r^2 \quad (5)$$

В силу симметрии транзистора, коэффициент $a_1(z)$ можно представить следующим образом:

$$\frac{\partial}{\partial r}\psi^{2D}(z,r)\Big|_{r=0}=a_1(z)=0 \quad (6)$$

При применении только к 2D потенциалу теоремы Гаусса, получим следующие соотношения:

$$C_{ox}\psi^{2D}(z,r) = -\varepsilon_S \frac{\partial \psi^{(2D)}(z,r)}{\partial r}\Big|_{r=R}$$

(7)

$$a_2(z) = -\frac{C_{ox}}{C_{ox}R^2 - 2\varepsilon_S R} a_0(z) r^2$$

где $C_{ox} = -\frac{\varepsilon_{ox}}{t_{ox}}$ - емкость подзатворного диэлектрика.

Из (5, 7) 2D потенциал можно представить в виде:

$$\psi^{(2D)}(z,r) = a_0(z) - \frac{C_{ox}}{C_{ox}R^2 + 2\varepsilon_S R} a_0(z) r^2$$

(8)

Если подставить (8) в (4), то можно перейти к следующему уравнению:

$$l^2 \frac{\partial^2}{\partial z^2} a_0(z) - a_0(z) = 0, \quad (9)$$

где $l = R\sqrt{\frac{\varepsilon_S \ln(1+\frac{t_{ox}}{R})}{2\varepsilon_{ox}}}$ - характеристическая длина

и его решение запишем так:

$$a_0(z) = A\exp(\frac{z}{l}) + B\exp(-\frac{z}{l})$$

Граничные условия для данного уравнения следуют из граничных условий для уравнения (4):

$$a_0(0) = U_{bi} - \psi^{(1D)}(0)$$

$$a_0(L_g) = U_{bi} + U_{ds} - \psi^{(1D)}(0)$$

Тогда выражения для A и B можно записать следующим образом:

$$A = -\frac{a_0(0)\exp(-\frac{L_g}{l}) - a_0(L_g)}{\exp(\frac{L_g}{l}) - \exp(-\frac{L_g}{l})}$$

(10)

$$B = \frac{a_0(0)\exp(\frac{L_g}{l}) - a_0(L_g)}{\exp(\frac{L_g}{l}) - \exp(-\frac{L_g}{l})}$$

Финальное выражение для $\psi^{(2D)}(z,r)$ представим следующим образом:

$$\psi^{2D}(z,r) = \left[a_0(0)\left\{\frac{\sinh(\frac{L_g - z}{l})}{\sinh(\frac{L_g}{l})}\right\} + a_0(L_g)\left\{\frac{\sinh(\frac{z}{l})}{\sinh(\frac{L_g}{l})}\right\} \right] \times \left(1 - \frac{C_{ox}r^2}{C_{ox}R^2 + 2\varepsilon_S R}\right)$$

(11)

Полученное выражение пригодное для дальнейшего и аналитического и численного анализа. С его помощью можно легко получить выражение для минимума потенциала и определить один из главных параметров транзистора - пороговое напряжение. Также с помощью (11) можно вычислить ток утечки и крутизну подпороговой характеристики. Мы сосредоточимся на исследовании поведения поверхностного потенциала.

2. Результаты моделирования и обсуждение

Рассмотрим прототип полностью обедненного КМОП нанотранзистора с полностью охватывающим затвором, топологические параметры которого варьируются в широком диапазоне. Фиксированный параметр - максимальный уровень легирования стока/истока $0{,}5\text{x}10^{20}$ см$^{-3}$. Из результатов расчетов характеристической длины l, которые получены при помощи (9), для заданного значения Lg можно определить диапазон параметров R и tox, при которых выполняется условие полного подавления ККЭ [2]. Диапазоны параметров R и tox необходимо выбирать с учетом ограничений из-за проявления квантово-механических эффектов, характерных для устройств с топологическими нормами близкими к границе масштабирования. Если ориентироваться на минимальные топологические нормы исходя из ограничений для R и tox, то для найденной области значений R и tox транзисторные структуры можно с полной уверенностью отнести к полностью обедненным [2-4, 7, 8]

На рис 2. представлены результаты моделирования распределения поверхностного потенциала вдоль рабочей области для двух прототипов с разными топологическими параметрами, но при одинаковых напряжениях на контрактах. А также приведены данные моделирования, полученные по моделям [3] и [4] и при помощи программного пакета ATLAS^{TM} .

Параметры прототипов приведены в таблице 1.

Таблица 1. Параметры прототипов транзисторов

Параметр	Прототип 1	Прототип 2
Lg, нм	25	50
tox, нм	1.2	1.6
R, нм	3.5	7,5
N_A,см$^{-3}$	$1.0x10^{17}$	$1.0x10^{17}$

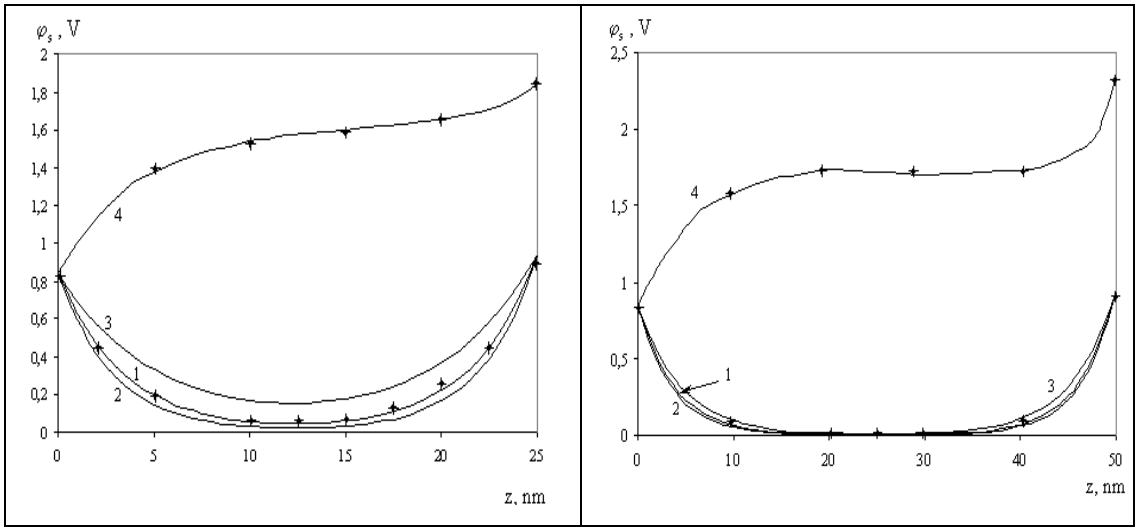


Рис. 2. Распределение поверхностного потенциала вдоль рабочей области: а (левый) прототип 1, где 1 – новая модель, 2 - модель [3], 3 - модель [4] при Uds=0.1 В и Ug= 0, 4 – новая модель при при Uds=Ug= 1.0 В и б (правый) прототип 2 , где 1 – новая модель, 2 - модель [3], 3 - модель [4] при Uds=0.1 В и Ug= 0, 4 – новая модель при при Uds=Ug= 1.5 В.
Звездочкой обозначены даны моделирования программы ATLAS^{TM}

Из представленных данных следует, что хорошее (до 5%) совпадение результатов моделирования в рассмотренных случаях для всех моделей реализуется около областей стока и истока из-за жесткой привязки к граничным условиям, которые одинаковы для всех моделей. Отметим существенные различия для прототипов вдоль длины канала. Так для прототипа 1 различие нашей модели, модели [3] от данных программного пакета ATLAS^{TM} составляет в точке минимума потенциала 8 и 6 %, соответственно. Ошибка модели [4] - более 200%. Для прототипа 2 ошибки составляют 4, 4 % и более 20%.

Из сопоставления результатов расчетов по разработанной модели с данными моделирования ATLAS^{TM}, позволяет сделать вывод о приемлемом соответствии между ними во всем диапазоне затворных напряжений. Разработанная модель распределения потенциала характеризуется приемлемой точностью и высокой вычислительной эффективностью.

Для прототипа 1 на рис. 3а представлены результаты моделирования распределения потенциала при разных R. Показано, что варьированием R можно эффективно управлять распределением потенциала. При этом рост минимума потенциала опережает рост радиуса в примерно полтора раза.

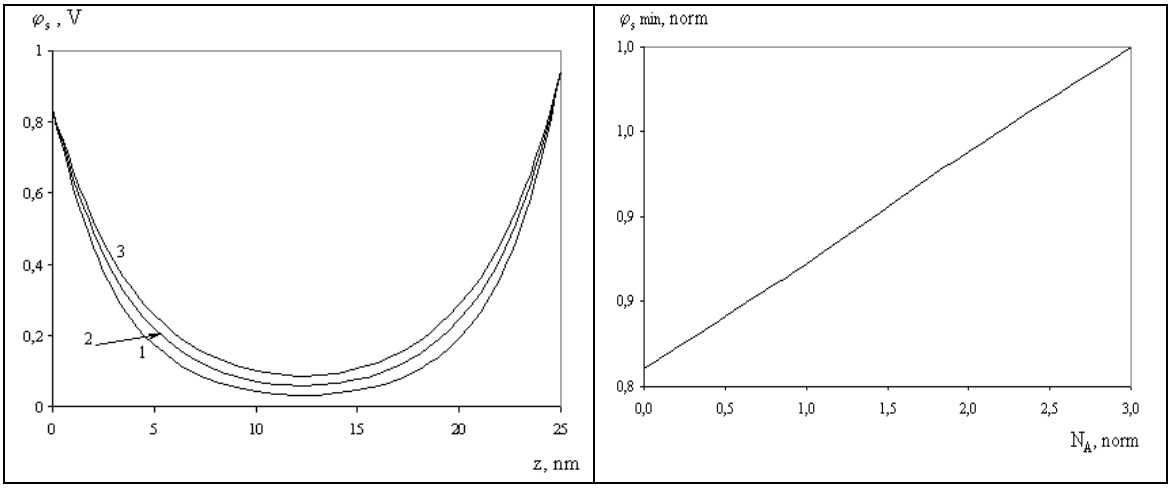


Рис. 3а. (левый). Распределение поверхностного потенциал вдоль рабочей области прототипа 1, где 1 – R=3 нм, 2 - R=4 нм, 3 – R=5 нм при Uds=0.1 В и Ug= 0;

Рис. 3б. (правый). Зависимость $\varphi_{s\min}(N_A)$.

Зависимость величины минимума поверхностного потенциала от уровня N_A представлена на рис. 3б. В логарифмических координатах по концентрации в диапазоне от 10^{+15} до 10^{+18} $см^{-3}$ эта зависимость линейна. Ее вид одинаков для всех рассмотренных выше структур. Отличие заключается в значениях минимума поверхностного потенциала. В общем случае для повышения потенциала на 25% необходимо увеличить концентрацию на три порядка.

Заключение

Разработаны теоретические основы математической модели суб 50 нм полностью обедненного КМОП транзистора с полностью охватывающим затвором. Применение такой архитектуры перспективно и для более весомого подавления ККЭ и для улучшения транспорта носителей в канале.

В частности, разработана 2D аналитическая модель распределения потенциала в рабочей области транзистора. Она сформулирована на основе решения уравнения Пуассона в цилиндрических координатах. Численно исследованы распределения потенциала. Исследована зависимость величины потенциала в каждой точке канала от топологических параметров транзистора. Сопоставление результатов расчетов с данными моделирования, полученными при помощи программного пакета ATLAS™, позволяет сделать вывод о приемлемом соответствии между ними.

Показано, что варьированием R можно эффективно управлять распределением потенциала. При этом рост минимума потенциала опережает рост радиуса примерно в полтора раза. Увеличение концентрации легирования также приводит к росту величины минимума поверхностного потенциала. При этом для повышения потенциала на 25% необходимо увеличить концентрацию на три порядка.

Работа выполнена в рамках Государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) N 0065-2019-0001

2D analytical model of potential distribution for fully depleted surrounding gate CMOS nanotransistor

N. Masalsky

Abstract. Effective 2D analytical distribution model of potential in work area of short - channel fully depleted surrounding gate CMOS transistor is developed. The model is formulated on the basis of an analytical solution of Poisson equation in cylindrical coordinates. Computer calculations of potential distribution for the broad range of topological parameters are executed. Comparison of calculation results to data of modeling of the ATLAS™ software package, proves acceptable compliance between them.

Keywords: fully depleted CMOS nanotransistor, surrounding gate, Poisson equation, 2D potential distribution

Литература

1. J.-P. Colinge. FinFETs and Other Multi-Gate Transistor. NewYork: Springer-Verlag, 2008.
2. I. Ferain, C. A. Colinge, J. Colinge. Multigate transistors as the future of classical metal–oxide–semiconductor field-effect transistors. Nature. 2011. vol. 479, p. 310–316.
3. H. A. E. Hamid, B. Iñıacuteguez, J. R. Guitart. Analytical model of the threshold voltage and subthreshold swing of undoped cylindrical gate-all-around-based MOSFETs. IEEE Trans. Electron Devices. 2007. vol. 54, № 3, p. 572–579.
4. A. Tsormpatzoglou, D. H. Tassis, C. A. Dimitriadis, G. Ghibaudo, G. Pananakakis, R. Clerc. A compact drain current model of short-channel cylindrical gate-all-around MOSFETs. Semicond. Sci. Technol. 2009. vol. 24, № 7, p. 75017-75028.
5. URL: http://www.silvaco.com/ Silvaco Int. 2004: ATLAS User's Manual A 2D numerical device simulator (дата обращения 5.12.2018).
6. J. He, M. Chan, X. Zhang, Y. Wang. A carrier-based analytic model for the undoped (lightly doped) cylindrical surrounding-gate MOSFETs. Solid State Electron. 2006. vol. 50, № 3, p. 416–421.
7. D. Sharma, S. K. Vishvakarma. Precise analytical model for short channel cylindrical gate (CylG) gate-all-around (GAA) MOSFET. Solid. State. Electron. 2013. vol. 86, № 1, p. 68–74.
8. M. R. Kumar, S. K. Mohapatra, K. P. Pradhan, P. K. Sahu. A simple analytical center potential model for cylindrical gate all around (CGAA) MOSFET. J. Electron Devices. 2014. vol. 19, p. 1648–1653.

Метод вычисления множества ячеек двумерной квадратной сетки, пересекаемых заданным лучом

М.В. Михайлюк[1], Д.А. Кононов, Д.М. Логинов

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail: [1]mix@niisi.ras.ru

Аннотация: В статье описывается метод и алгоритм определения множества ячеек двумерной квадратной сетки, которые пересекает заданный луч. Этот метод может быть обобщен и использован в задачах трассировки лучей, воксельной визуализации, реализации проекции максимальной интенсивности и др.

Ключевые слова: 2D квадратная сетка, ячейки пересечения с лучом, максимальная интенсивность.

Введение

Во многих задачах возникает необходимость выделить из регулярной квадратной сетки множество ячеек, которые пересекает заданный луч. К таким задачам относятся реализация трассировки лучей [1], метод проекции максимальной интенсивности [2], воксельная визуализация [3] и др. В базовой работе [4] приводится алгоритм последовательного вычисления ячеек вдоль луча. В дальнейшем различными авторами [5, 6] этот алгоритм модифицировался и улучшался. В работе [7] предложена модификация, в которой формулы вычисления следующей ячейки не меняются в зависимости от исходной точки и направления луча.

В данной работе рассматривается двумерная квадратная сетка, в каждой ячейке которой записано некоторое число, и произвольный направленный в сторону сетки луч с началом вне ее. Здесь предлагается алгоритм определения множества ячеек, пересекаемых лучом, и на его основе рассматривается задача вычисления максимальной ячейки (т.е. ячейки с максимальным значением) этого множества.

Метод вычисления максимальной ячейки

Определим квадратную сетку как двумерный массив вещественных чисел размером $N \times N$. Визуально ее можно представить в виде квадрата в объектной системе координат с центром в его левом нижнем углу и осями X и Y вдоль сторон. Длина и высота каждой ячейки сетки равна единице. В этой же объектной системе рассмотрим луч $P_0 + \vec{v}t$ (где $t \geq 0$), исходящий из некоторой точки P_0 с направляющим вектором $\vec{v}$. На рис. 1 показан пример такого квадрата для $N = 4$. Уточним понятие пересечения луча с квадратом. Если луч пересекает квадрат только в одной точке (в вершине квадрата) или луч следует вдоль стороны квадрата, мы будем считать, что пересечения нет. В противном случае будет ровно две точки пересечения, т.е. луч входит в квадрат в некоторой точке P_1 и выходит из него в некоторой точке P_2. Вычислим значения t_1 и t_2 параметра t,

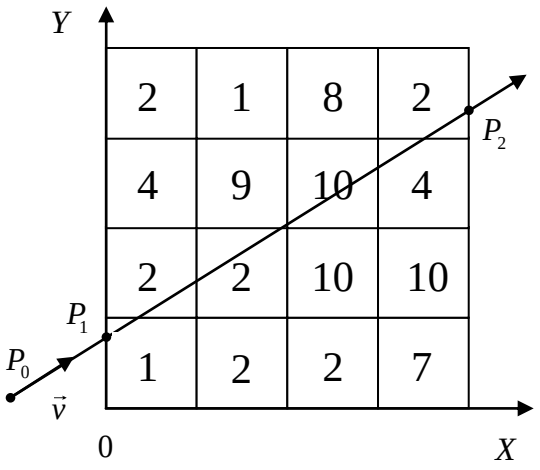


Рис. 1. Квадрат и луч в объектной системе координат

соответствующие этим точкам. Не зная заранее, как расположен луч относительно квадрата, мы вынуждены рассмотреть его пересечения со всеми ребрами.

Вычисление точек пересечения луча с квадратом. Создадим массив E ребер квадрата из четырех элементов, в котором E[0] и E[1] – ребра, соответствующие значениям x = 0 и x = N, а E[2] и E[3] – ребра, соответствующие значениям y = 0 и y = N. Определим массив $T[\mathrm{j}]$ из двух элементов, в которые будут записаны значения параметра t для точек пересечения P_1 и P_2. Начальное значение числа j точек пересечения зададим равным 0. Тогда получим следующий

Алгоритм вычисления точек пересечения луча с квадратом

Начало.
$j = 0;\ t_1 = 0$.
Цикл по i от 0 до 3 // по всем ребрам
// вычисляем параметр t точки
// пересечения $\vec{v}$ и $E[i]$
$t = IsIntersect(\vec{v}, i)$;
Если $t > 0\ u\ t_1 \neq t\ u\ j \leq 1$, то
// луч пересекает $E[i]$
$T[j] = t;\ \ j{+}{+};\ t_1 = t$.
Конец если.
Конец цикла.
Если $j > 1$, то
Если $T[0] > T[1]$, то меняем их местами.
Вычисляем
$P_1 = P_0 + \vec{v}\,T[0];\ P_2 = P_0 + \vec{v}\,T[1]$.
Если $P_1\ u\ P_2$ лежат на одной стороне квадрата, то return(0).
return(1).
Конец если
return(0).
Конец.

Вычисление точки пересечения луча с ребром квадрата. Рассмотрим подробнее задачу определения пересечения луча и отдельного ребра $E[i]$ (реализацию функции $IsIntersect(\vec{v}, i)$). Если луч параллелен $E[i]$, то пересечения нет, и функция возвращает 0. В противном случае луч пересекает линию ребра при некотором значении параметра t. Если найденная точка лежит внутри ребра, то функция возвращает значение t, в противном случае возвращается 0. Таким образом, получаем следующий алгоритм функции $IsIntersect(\vec{v}, i)$:

Алгоритм определения пересечения луча и ребра $E[i]$

Начало.
// Если $E[i]$ - вертикальное ребро
Если $(i < 2$ и $v_x \neq 0)$, то
$$t = \frac{E[i]_x - P_{0x}}{v_x};$$
$$Q = P_{0y} + v_y t;$$
Если $0 \leq Q \leq N$, то return(t).
// Если $E[i]$ - горизонтальное ребро
иначе если $(2 \leq i \leq 3$ и $v_y \neq 0)$, то
$$t = \frac{E[i]_y - P_{0y}}{v_y};$$
$$Q = P_{0x} + v_x t$$
Если $0 \leq Q \leq N$, то return(t).
Конец если
return(0);
Конец.

Естественно, что в этих алгоритмах сравнение действительных чисел необходимо выполнять с точностью до некоторого малого ε.

Теперь нашей задачей будет, начиная с точки P_1, определить все ячейки квадрата моделирования, которые пересекаются с лучом, вплоть до точки P_2. Рассмотрим несколько случаев.

Вычисление максимального значения в столбце ячеек. Пусть луч проходит некоторое множество ячеек столбика (см. рис. 2), задаваемых значениями $x = i,\ \ j$ изменяется от начальной ячейки $j_н$ до конечной ячейки $j_к$ (при этом может оказаться, что $j_н \leq j_к$ и наоборот). Введем глобальные переменные $m, i_{\max}, j_{\max}$, в которые будут записываться соответственно максимальное значение в вычисленном множестве ячеек и координаты максимальной ячейки.

Рассмотрим функцию
$Max\_j\ (i, j_н, j_к, s)$

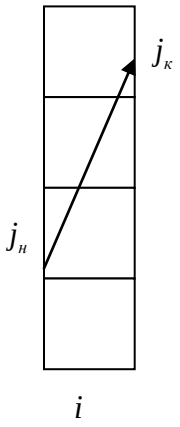


Рис. 2. Столбик ячеек

вычисления максимальной ячейки в выделенной части столбика вдоль луча. Алгоритм этой функции имеет простой вид:

Начало.

Цикл по j от $j_н$ с шагом s пока $j \neq j_к + s$

Закрашивание квадратика (i, j);

Если $c_{ij} > m$, то

$m = c_{ij},\ i_{\max} = i,\ j_{\max} = j$.

Конец цикла

Конец

В этом алгоритме параметр (шаг цикла) $s = \pm 1$ в зависимости от того в какую сторону направлен луч (т.е. в каком порядке надо перебирать ячейки).

Аналогичную функцию $Max\_i\ (j, i_н, i_к, s)$ можно написать для вычисления максимальной ячейки для подмножества ячеек i-й строки.

Вычисление максимальной ячейки квадрата вдоль луча. В общем случае алгоритм определения множества ячеек вдоль луча зависит от направления этого луча (от знаков величин $v_x\ и\ v_y$). Для того, чтобы избежать большого числа условий в алгоритме, введем флаги s_x, s_y, s_{1x}, s_{1y}, которые вычисляются следующим образом:

Если $v_x > 0$, то $s_x = 1$, иначе $s_x = -1$;

Если $v_y > 0$, то $s_y = 1$, иначе $s_y = -1$;

Если $v_x > 0$, то $s_{1x} = 1$, иначе $s_{1x} = 0$;

Если $v_y > 0$, то $s_{1y} = 1$, иначе $s_{1y} = 0$.

Зная точки $P_1\ и\ P_2$ входа и выхода луча, можно вычислить номера $(i_н, j_н)$ начальной и $(i_к, j_к)$ конечной ячеек:

$i_1 = [P_{1x}];\ j_1 = [P_{1y}];\ i_2 = [P_{2x}];\ j_2 = [P_{2y}];$

Тогда алгоритм функции $MaxValue(i_1, j_1, i_2, j_2)$ поиска максимальной ячейки вдоль этого луча будет иметь следующий вид:

Начало.

1. Если $v_x = 0$ и $i_1 < N$, то

$Max\_j\ (i_1, j_1 + s_{1y} - 1, j_2 - s_{1y}, s_y)$;

2. иначе, если $v_y = 0$ и $j_1 < N$, то

$Max\_i\ (j_1, i_1 + s_{1x} - 1, i_2 - s_{1x}, s_x)$;

3. иначе // $v_x \neq 0$ и $v_y \neq 0$

$j_m = j_0$.

Если $i_1 = P_{1x}$, то $i_1 = i_1 + s_{1x} - 1$;

Если $j_1 = P_{1y}$, то $j_1 = j_1 + s_{1y} - 1$;

Если $i_2 = P_{2x}$, то $i_2 = i_2 - s_{1x}$;

Если $j_2 = P_{2y}$, то $j_2 = j_2 - s_{1y}$;

Цикл по i от i_1 с шагом s_x пока $i \neq i_2 + s_x$

$y_{сл} = P_{0y} + v_y \dfrac{i + s_{1x} - P_{0x}}{v_x},\ j_{сл} = [y_{сл}]$;

Если $j_{сл} \geq N$, то $j_{сл} = N - 1$;

Если $j_{сл} < 0$, то $j_{сл} = 0$;

$s_n = 0$;

Если $v_y > 0\ u\ j_{сл} = y_{сл}$, то $s_n = -1$;

$Max\_j\ (i, j_m, j_{сл} + s_n, s_y)$;

Если $v_y < 0\ u\ j_{сл} = y_{сл}$, то $j_{сл} = j_{сл} - 1$;

$j_m = j_{сл}$.

Конец цикла

Конец если

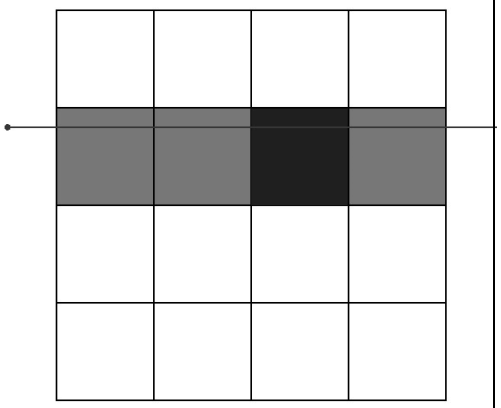
Конец.

На рис. 3 показаны примеры выделения множества ячеек и максимальной ячейки вдоль лучей различного направления. Под каждым рисунком указаны координаты начальной точки P_0 луча и вектор $\vec{v}$ его направления.

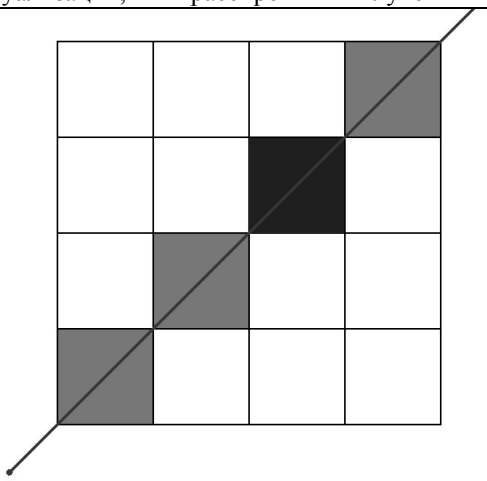
Заключение

В данной работе описан метод вычисления множества ячеек двумерной квадратной сетки, пересекаемых заданным лучом. Этот метод использует вычисляемые флаги, с помощью

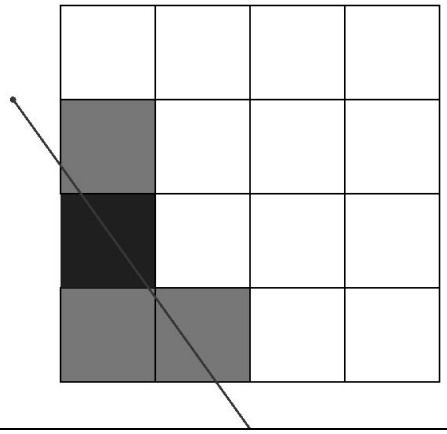
которых можно уменьшить число условий, обусловленных различными возможными направлениями луча, т.е. упростить алгоритм решения поставленной задачи. Метод может стать основой для решения задач воксельной визуализации, трассировки лучей в



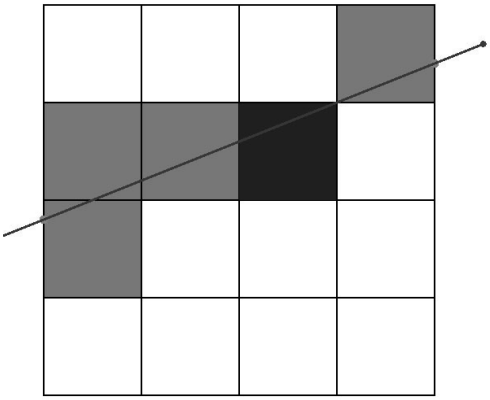
$P_0 = (-0.5, 2.8);\ \vec{v} = (1, 0)$



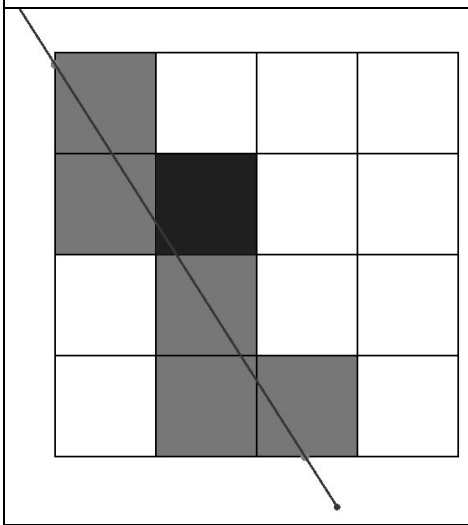
$P_0 = (-0.5, -0.5);\ \vec{v} = (1, 1)$



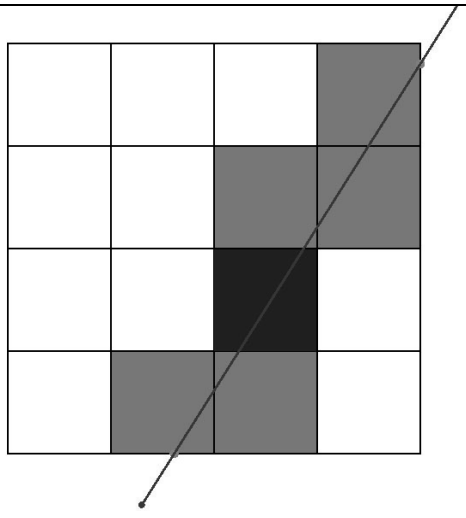
$P_0 = (-0.5, 3);\ \vec{v} = (1, -1.4)$



$P_0 = (4.5, 3.6);\ \vec{v} = (-1.3, -0.52)$



$P_0 = (2.8, -0.5);\ \vec{v} = (-2.3, 3.6)$



$P_0 = (1.3, -0.5);\ \vec{v} = (2.7, 4.3)$

Рис. 3. Примеры выделения ячеек для разных лучей

компьютерной графике, визуализации скалярных полей и др.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) «34.9. Системы виртуального окружения: технологии, методы и алгоритмы математического моделирования и визуализации» (0065-2019-0012).

The method for calculating the set of cells in 2D square grid intersected by a given ray

M.V. Mikhaylyuk, D.A. Kononov, D.M. Loginov

Abstract: The paper describes the method and algorithm for calculating the set of cells in 2D square grid intersected by a given ray. This method can be generalized and used in the ray tracing, voxel visualization, volume rendering etc.

Keywords: 2D square grid, maximum intensity projection, volume rendering, grid traversal, OpenGL

Литература

1. Wald, T. Ize, A. Kensler, A. Knoll, and S. G. Parker, “Ray tracing animated scenes using coherent grid traversal,” ACM Transactions on Graphics, pp. 485–493, 2006.
2. Hashemi, R.H.; Bradley, W.G.; Lisanti, C.J. MRI: The Basics. — Wolters Kluwer Health, 2012.
3. Szymon Jabło´ nski, Tomasz Martyn. Real-time voxel rendering algorithm based on Screen Space Billboard Voxel Buffer with Sparse Lookup Textures. WSCG 2016 - 24th Conference on Computer Graphics, Visualization and Computer Vision 2016, pp. 27- 36.
4. Amanatides, J., and Woo, A. A fast voxel traversal algorithm for ray tracing. In In Eurographics 87, 3–10.
5. Lukas Mroz, Andreas Konig and Eduard Groller Real-Time Maximum Intensity Projection Proceedings of the Joint EUROGRAPHICS and IEEE TCVG Symposium on Visualization in Vienna, Austria, May 26–28, 1999 Data Visualization ’99 pp 135-144.
6. Colin Braley, Robert Hagan, Yong Cao, Denis Gracanin GPU Accelerated Voxel Traversal using the Prediction Buffer. https://pdfs.semanticscholar.org/ba66/05343a30154006b3bd8a44ba2b2af5500e5a.pdf?_ga=2.190798955.1716594299.1549894210-659905767.1549894210
7. А.В. Мальцев, М.В. Михайлюк. Регулярные сетки для высоко реалистичной визуализации 3D сцен в реальном времени. Информационные технологии и вычислительные системы № 4, 2012, стр. 49-58.

Технология создания виртуальных моделей квадрокоптера и пульта управления

Е.В. Страшнов[1], Л.А. Финагин, И.Н. Мироненко

ФГУ «ФНЦ Научно-исследовательский институт системных исследований РАН»
E-mails: [1]strashnov_evg@mail.ru

Аннотация: В статье рассматривается задача создания виртуальных моделей квадрокоптера и пульта управления для систем виртуального окружения. Для этого предлагается технология разработки трехмерной модели летательного аппарата в системе моделирования 3ds Max с помощью конструктора из набора базовых элементов. С помощью предлагаемого подхода геометрическая модель квадрокоптера дополняется новыми объектами, такими как винты и датчики, для которых можно задавать свои параметры. В свою очередь создание виртуального пульта основывается на применении разработанного редактора, в котором каждому органу управления (кнопка, джойстик, тумблер и т.д.) соответствует свой виртуальный элемент с возможностью задания для него визуального представления и набора параметров. Апробация созданных моделей квадрокоптера и пульта выполнялась с применением тестовых схем управления в рамках комплекса виртуального окружения, разработанного в ФГУ ФНЦ НИИСИ РАН.

Ключевые слова: система виртуального окружения, беспилотные летательные аппараты, модель квадрокоптера, виртуальный пульт управления, датчики.

Введение

Развитие систем связи и навигации, в частности инерциальных навигационных систем (ИНС), привело к широкому распространению беспилотных летательных аппаратов (БПЛА). К таким устройствам также относятся мультикоптеры – винтокрылые летательные аппараты с произвольным количеством несущих винтов. Квадрокоптер [1] является частным случаем мультикоптера и представляет собой летательный аппарат с четырьмя несущими винтами. В настоящее время подобные аппараты нашли широкое применение [2] в различных сферах и областях, включая аэрофото- и киносъемку, инспектирование местности и помещений, перенос грузов и т.д. Отдельным направлением является использование квадрокоптеров в условиях, опасных или вредных для здоровья человека. Примером служит их применение для мониторинга пожаров [3] с целью обнаружения очагов возгорания, изучения степени разрушений при землетрясениях, исследования уровня радиации местности [4], анализа химического заражения при авариях на шахтах [5] и т.д.

Управление движением квадрокоптера [6] осуществляется путем изменения скоростей вращения винтов, создающих подъемную силу. Например, если синхронно увеличить скорости всех винтов, то будет осуществляться подъем коптера, а при уменьшении – опускание. Такое управление может осуществляться автономно или дистанционно с помощью пульта под управлением оператора. Обучение навыкам управления квадрокоптером более целесообразно проводить в виртуальной среде с применением виртуальных моделей аппарата и пульта. Это позволит избежать поломки дорогостоящего оборудования, отработать различные режимы движения коптера, уменьшить время и повысить качество обучения. В связи с этим виртуальные модели квадрокоптера и пульта должны соответствовать их реальным прототипам. К виртуальной модели коптера предъявляются требования, согласно которым необходимо задавать аэродинамические параметры и измерять текущее состояние квадрокоптера с помощью датчиков. В свою очередь виртуальный пульт управления должен обладать функциональностью реального пульта и быть таким, что его элементы (джойстики, тумблеры, регуляторы и т.д.) будут адекватно реагировать на задаваемое пользователем действие (например, перемещение джойстика вверх). Поэтому разработка виртуальных моделей квадрокоптера и пульта является важной и актуальной задачей.

В данной работе предлагается технология создания виртуальной модели и пульта управления квадрокоптером, предназначенного для

исследования уровня радиоактивного загрязнения. Предлагаемое решение состоит в создании модели летательного аппарата в системе трехмерного моделирования 3ds Max с применением разработанного конструктора новых объектов, который дополняет стандартный набор объектов этой системы (геометрические примитивы, источники освещения, камеры и т.д.). С помощью такого подхода аэродинамические параметры квадрокоптера задаются в новом объекте «Винт», а состояние коптера определяется с помощью разработанных виртуальных датчиков [7], измеряющих параметры его движения, расстояния до ближайших объектов, интенсивность радиоактивного излучения и т.д. В свою очередь создание пульта управления квадрокоптером осуществляется в разработанном редакторе двумерных пультов и функциональных схем [8, 9], позволяющем создавать органы управления виртуального пульта (джойстики, тумблеры, регулятора) из набора визуальных элементов и задавать для них параметры.

1. Трехмерная модель квадрокоптера

В данной работе рассмотрена виртуальная модель квадрокоптера, предназначенного для радиационного мониторинга местности и находящихся на ней объектов. Трехмерная модель этого аппарата (см. Рис. 1) создана в системе трехмерного моделирования 3ds Max и содержит порядка 20 тысяч полигонов. Конструкция квадрокоптера представляет собой раму типа “X”, на концах лучей которой крепятся моторы. В этой конструкции подвес камеры располагается на передней части коптера, поэтому лучи рамы не попадают в кадр. Камера имеет угол обзора 84°, что позволяет получить панорамное изображение исследуемой местности. Модель аппарата также снабжена тремя фарами, которые установлены на нижней, боковой и задней части робота.

Помимо рассмотренных объектов виртуальная модель дополняется набором новых объектов, для которых можно задавать пара-

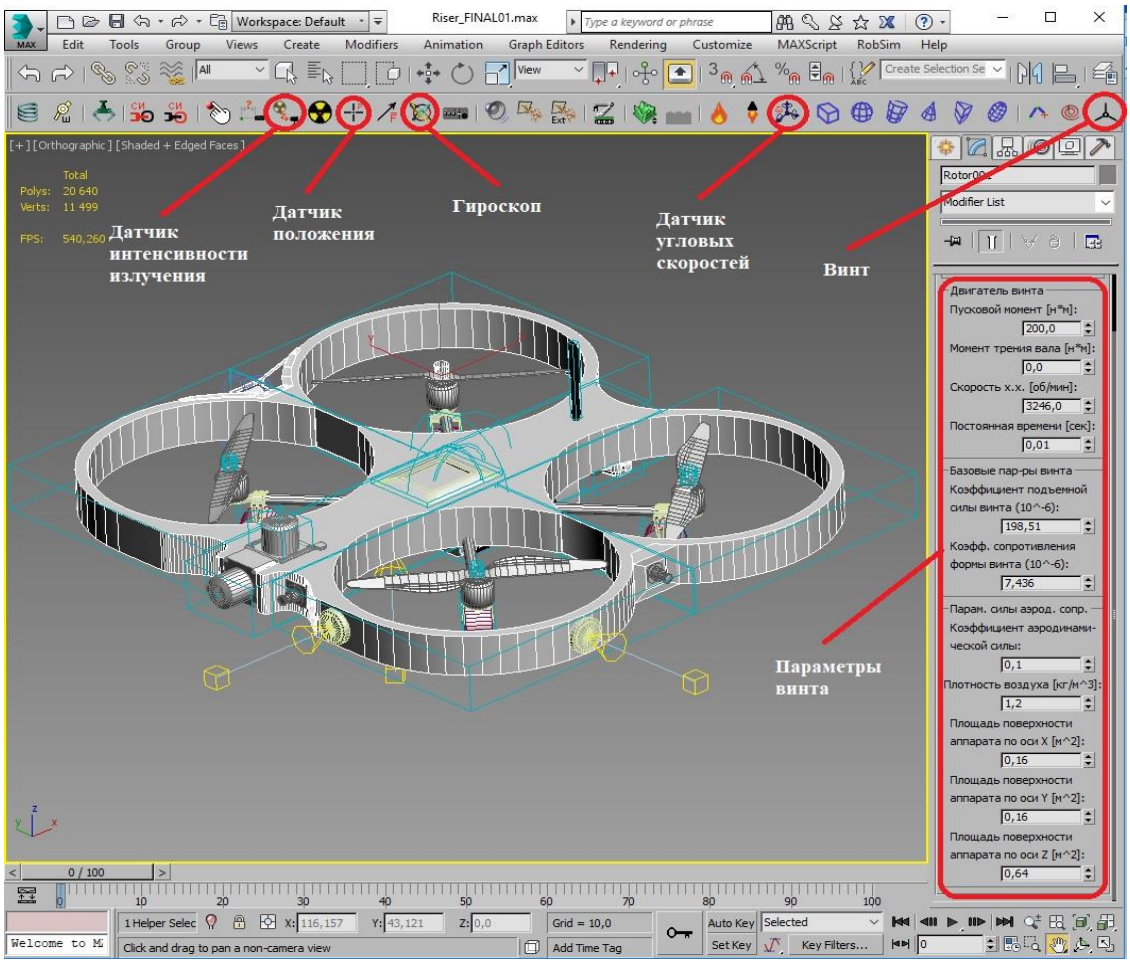


Рис. 1. Трехмерная модель квадрокоптера в системе 3ds Max

метры, необходимые для моделирования динамики и управления квадрокоптером. Это реализуется посредством разработанных плагинов, которые дополняют стандартный набор объектов системы 3ds Max. С помощью такого подхода в конструктор добавлен новый объект «Винт» (см. Рис. 1), содержащий параметры двигателя винта, силы тяги и момента сопротивления винта, а также параметры аэродинамической силы сопротивления. Эти параметры используются для расчета сил и моментов, действующих на квадрокоптер.

В силу того, что в процессе полета квадрокоптер необходимо стабилизировать [10], то он должен иметь набор сенсоров, непрерывно измеряющих его положение и ориентацию. Полетный контроллер квадроптера для этих целей снабжен набором устройств. В этот набор входят датчики микроэлектромеханической системы (МЭМС), такие как интегральный гироскоп, акселерометр, магнитометр и барометр, а также ультразвуковой сонар (дальномер), лидар, систему GPS (Global Positioning System – система глобального позиционирования), гамма-детектор излучения и т.д. В зависимости от различных условий необходимая информация измеряется одним из этих устройств или их комбинацией с применением фильтров для уменьшения ошибок. Так, для удержания высоты могут быть использованы барометр, сонар, GPS и датчик визуального позиционирования. В данной работе использованы виртуальные датчики [7], вычисляющие точную информацию об объекте в рамках моделирующего комплекса. На рис. 1 на панели инструментов конструктора объектов 3ds Max показаны некоторые из этих датчиков. В рассматриваемую виртуальную модель квадрокоптера входят датчик положения, датчик угловых скоростей, акселерометр, дальномеры, гироскоп и датчик интенсивности излучения. Датчик положения измеряет положение квадрокоптера относительно мировой (глобальной) системы координат. В системе управления показания датчика положения используются для вычисления скоростей и ускорений коптера, что позволяет в некоторых случаях заменить работу акселерометра. Датчик угловых скоростей измеряет угловую скорость в локальной системе координат летательного аппарата. Акселерометр определяет сумму собственного ускорения квадрокоптера и ускорения свободного падения в локальной системе координат коптера. Дальномеры расположены на всех сторонах квадрокоптера и моделируют работу лидара. Каждый из этих датчиков вычисляет расстояние до ближайшего объекта вдоль своего заданного направления. С помощью гироскопа определяется ориентация летательного аппарата путем вычисления углов Эйлера, задающих последовательность поворотов сначала вокруг оси Z (рысканье), затем вокруг оси Y (крен) и наконец вокруг оси X (тангаж). Датчик интенсивности излучения определяет уровень интенсивности излучения в точке нахождения квадрокоптера. Показания рассмотренных датчиков используются в системе управления квадрокоптером.

2. Виртуальный пульт управления квадрокоптером

В данной работе рассматривается управ-

Рис. 2. Виртуальный пульт управления квадрокоптером

ление квадрокоптером с помощью пульта. Для этого в разработанном редакторе пультов и функциональных схем [8, 9] был создан виртуальный пульт (см. Рис. 2), который позволяет управлять движением коптера, вращать подвес камеры, включать и выключать фары, а также настраивать различные режимы управления. Интерфейс пульта представляет собой набор визуальных элементов, с которыми может взаимодействовать пользователь посредством нажатия и перемещения компьютерной мыши. К таким элементам относятся джойстики, тумблеры и переключатели. Для элементов задается визуальное представление в различных состояниях и поддерживаемый набор параметров поведения (например, свойство автовозврата джойстика).

Опишем функциональность созданного пульта. Путем изменения положения тумблера “Power” (в центре пульта) осуществляется включение и выключение пульта. Джойстики отвечают за движение коптера и являются двухосевыми, т.е. способны перемещаться горизонтально (вдоль оси X) и вертикально (вдоль оси Y). Правый джойстик имеет свойство автовозврата, то есть, если его отпустить, то он возвращается в свое нейтральное положение. Левый джойстик обладает этим свойством только по оси X. Управление движением квадрокоптера с помощью джойстика осуществляется следующим образом. Смещая левый джойстик вдоль оси Y осуществляется вертикальное перемещение квадрокоптера (подъем и опускание), а при смещении вдоль оси X выполняется его поворот в горизонтальной плоскости. Изменяя положение правого джойстика, осуществляется наклон коптера, что позволяет обеспечить его движение в горизонтальной плоскости. При этом, если двухпозиционный переключатель “B” находится в верхнем положении, то включен режим Headless mode и оси правого джойстика соответствуют направлениям движения квадрокоптера в исходной системе координат (в которой изначально располагался коптер). В противном случае (если переключатель “B” находится в нижнем положении) оси правого джойстика соответствуют направлениям движения квадркоптера в его локальной системе координат. Включение винтов коптера осуществляется при удержании левого джойстика в правом нижнем положении, а выключение – в левом нижнем. С помощью двухпозиционного переключателя “A” осуществляется включение и выключение фар. Переводом двухпозиционного переключателя “D” в нижнее положение выполняется переход на видовой режим с камеры, установленной на квадрокоптере. С помощью регуляторов осуществляется поворот подвеса камеры в горизонтальном (регул. 1) и вертикальном (регул. 2) направлении. Трехпозиционный переключатель “C” позволяет выбрать режим управления квадрокоптером. Верхнее положение переключателя “C” соответствует режиму ATTI со стабилизацией высоты только с помощью барометра, его среднее положение включает режим ATTI с модулем GPS, что позволяет сохранять его текущее положение перед началом движения и, наконец, нижнее положение переключателя запускает принудительно функцию FailSafe (функция отказобезопасности). В разработанном пульте данная функция реализована следующим образом. В режиме ATTI с GPS запускается функция “Возврат домой” и квадрокоптер возвращается в сохраненную в памяти позицию и опускается вниз, в противном случае, в режиме ATTI без GPS квадрокоптер только опускается вниз.

Виртуальный пульт связан с функциональной схемой управления, состоящей из набора блоков, соединенных линиями. Каждому визуальному элементу пульта в данной схеме соответствует свой блок. Также схема содержит блоки исполнительных устройств (винтов), датчиков и библиотеки редактора функциональных схем (алгебраические, логические, тригонометрические и т.д.). На Рис. 3 представлен фрагмент тестовой схемы для управления вертикальным движением квадрокоптера с помощью пульта. Схема построена таким образом, чтобы определить управляющие сигналы (напряжения двигателей), подаваемые на винты квадрокоптера, которые обеспечат необходимую скорость движения коптера, задаваемую с помощью джойстика. В блоке сумматора со знаком “dZ” вычисляется величина рассогласования между текущей и заданной скоростью движения коптера. Текущая скорость движения коптера вычисляется путем дифференцирования сигнала, поступаемого от блока датчика положения “PositionSensor”, а требуемая скорость задается сигналом от блока джойстика “JoystickLeft” (при его перемещении вдоль оси Y). Далее полученный сигнал передается блоку произведения “T/m”, в котором определяется сила тяги винтов. Затем в блоках произведения “T/4k” и квадратного корня “W” определяется требуемое значение скорости вращения винтов. Полученное значение, поделенное на скорость холостого хода двигателя, используется для вычисления напряжения, подаваемого на двигатели винтов блоков “Rotor1”, “Rotor2”, “Rotor3” и “Rotor4”. При этом для компенсации моментов сопротивления два винта вращаются

по часовой стрелке, а другие два винта – против. Блок тумблера “On/Off” отвечает за работу всей схемы и, в случае, если на выходе блока получен нулевой сигнал, то вычисляемые напряжения равны нулю.

3. Апробация созданных моделей

Апробация созданных моделей квадрокоптера и пульта управления проводилась в рамках комплекса виртуального окружения, разработанного в ФГУ ФНЦ НИИСИ РАН. Этот программный комплекс состоит из подсистем управления, динамики и визуализации. Подсистема управления по значениям сигналов элементов пульта и датчиков вычисляет управляющие сигналы исполнительных устройств, которые передаются в подсистему динамики. В подсистеме динамики осуществляется вычисление всех сил и моментов, действующих на квадрокоптер, включая силу тяги винта, момент сопротивления винта и аэродинамическую силу сопротивления. По этим силам и моментам определяются новые скорости и координаты квадрокоптера. Полученные координаты передаются в подсистему визуализации, которая синтезирует изображение из выбранной оператором виртуальной камеры на мониторе. Показания датчиков рассчитываются в подсистемах динамики и визуализации и передаются в подсистему управления. Весь цикл расчета занимает не более 40 мс, что обеспечивает режим реального времени.

Для апробации была рассмотрена тестовая схема управления, в которой задействована модель вычисления управляющих сигналов с помощью ПИД-регуляторов [11], построенных исходя из точных показаний датчиков. На рисунке 4 показано положение квадрокоптера при его горизонтальном движении на некоторой высоте. Управление коптером осуществляется с помощью пульта путем отклонения левого джойстика вверх для того, чтобы квадрокоптер набрал необходимую высоту полета. При этом переключатель “C” установлен в режиме ATTI со стабилизацией высоты с помощью барометра. Переключатель “B” переведен в нижнее положение для движения коптера относительно его локальной системы координат. На рисунке показано начало движения коптера после отклонения правого джойстика вертикально вниз. Видно, что осуществляется наклон коптера и движение по направлению, заданному с помощью пульта. Апробация виртуальных моделей квадрокоптера и пульта показала их адекватность в рамках систем виртуального окружения и имитационно-тренажерных комплексов.

Заключение

В работе предложена технология создания виртуальных моделей квадрокоптера и пульта управления для систем виртуального окружения. В рамках этой технологии разработаны плагины для создания новых объектов системы 3ds Max, позволяющих задавать аэродинамические параметры квадрокоптера и определять его текущее состояние с помощью набора виртуальных датчиков. С помощью разработанного виртуального пульта осуществляется управление движением модели квадрокоптера, поворотами подвеса камеры и настройка различных режимов, которые соответствуют реальному пульту управления. Отметим, что не-

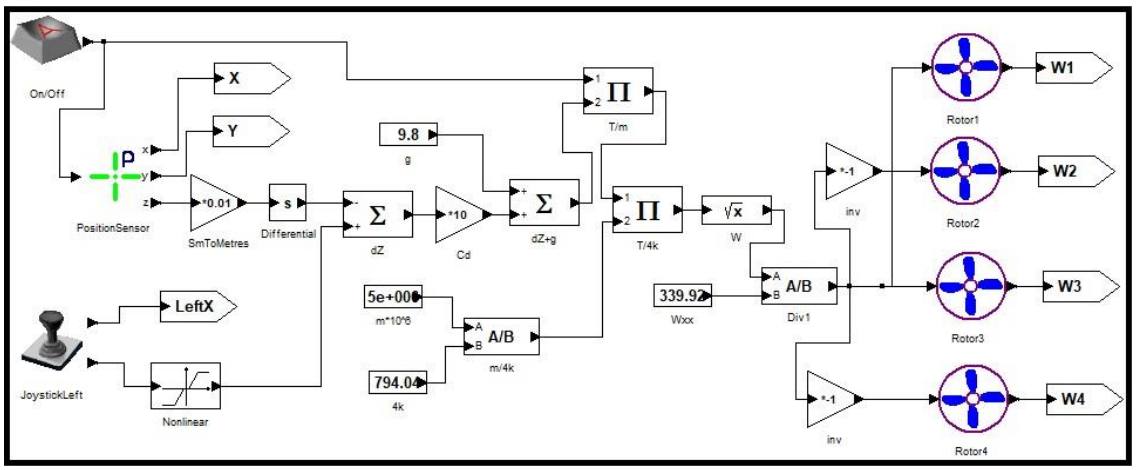


Рис.3. Функциональная схема управления

достатком применения технологии виртуальных пультов является невозможность одновременного управления левым и правым джойстиком с помощью компьютерной мыши. Кроме того, в данной работе рассматривается управление с использованием точных показаний датчиков, хотя в реальности эти показания подвержены шумах и ошибкам интегрирования (например, при вычислении координат путем интегрирования показаний акселерометра). В дальнейшем ожидается, что созданные модели квадрокоптера и пульта будут применены в рамках разработки сложных систем управления с учетом БИНС (бесплатформенная инерциальная навигационная система), стабилизации положения и ориентации с помощью ПИД-регуляторов, траекторного управления и т.д.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) «34.9. Системы виртуального окружения: технологии, методы и алгоритмы математического моделирования и визуализации». (0065-2019-0012).

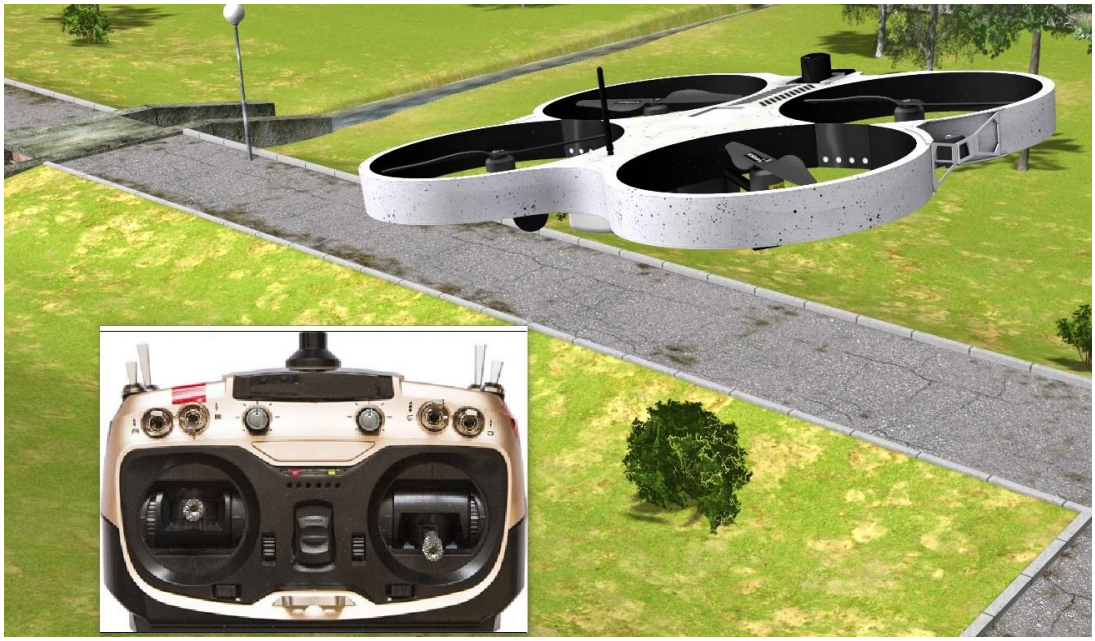


Рис. 4. Управление квадрокоптером с помощью виртуального пульта

A technology to develop virtual models of the quadcopter and its remote controller

E.V. Strashnov, L.A. Finagin, I.N. Mironenko

Abstract: This paper deals with the challenge to create virtual models of quadcopter and its remote controller for virtual environment systems. For this, a technology to develop a 3D model of aerial vehicle in 3ds Max system using the constructor comprised of the set of basic elements is proposed. By means of the approach proposed, the geometric model of the quadcopter is complemented by new objects, such as rotors and sensors, for which own parameters can be set. In turn, to create a virtual remote controller the developed editor is utilized, where every control (button, joystick, toggle switch, and etc.) is corresponded with its own virtual element with the ability to specify its visual representation and a set of parameters. Using the test control schemes, quadcopter and remote controller models created were approbated in virtual environment complex, developed in SRISA RAS.

Keywords: virtual environment systems, unmanned aerial vehicles, quadcopter model, virtual remote controller, sensors.

Литература

1. S. Gupte, P.I.T. Mohandas, and J.M. Conrad. A survey of quadrotor unmanned aerial vehicles. In Proc. IEEE Southeastcon, Orlando, FL, USA, 2012, pp. 1-6.
2. В.У. Зулькарнаев, В.Р. Камалова. Практическое применение беспилотных летательных аппаратов в современном мире // Инновации, 2016, № 56-2, стр. 23-27.
3. И.В. Минин. Противопожарный мониторинг объектов нефтехимической промышленности малыми автономными беспилотными летательными аппаратами // Фундаментальные исследования, 2015, № 10-3, стр. 324-327.
4. Д.Т. Рубин, В.Н. Конев, А.В. Стариковский, А.А. Шептунов, А.С. Смирнова, А.М. Толстая. Разработка квадрокоптеров со специальными свойствами для проведения разведывательных операций // Спецтехника и связь, 2012, № 1, стр. 28-30.
5. М.Л. Ким, А.С. Родичев, Л.Д. Певзнер, А.К. Платонов. О возможности использования робототехнических летательных аппаратов при выполнении оперативного плана ликвидации аварии на шахтах // Уголь, 2018, № 1, стр. 34-38.
6. S. Bouabdallah. Design and control of quadrotors with application to autonomous flying, Ph.D. dissertation, Ecole Polytechnique Federate de Lausanne, 2007.
7. М.В. Михайлюк, Е.В. Страшнов, Д.М. Логинов. Моделирование датчиков в системах виртуального окружения // Труды НИИСИ, т. 8, № 2, стр. 70-76.
8. М.В. Михайлюк. Двумерные виртуальные пульты управления в тренажерных комплексах // Программная инженерия, № 5, 2014, стр. 20-25.
9. М.В. Михайлюк, М.А. Торгашев. Визуальный редактор и модуль расчета функциональных схем для имитационно-тренажерных комплексов // Программные продукты и системы, № 4, 2014, стр. 10-15.
10. В.С. Яценков. Электроника. Твой первый квадрокоптер. Теория и практика // СПб.: БХВ-Петербург, 2017, 256 с.
11. T. Luukkonen. Modelling and control of quadcopter. School of Science, Espoo, August 22, 2011, P. 26.

Составление расписания для неоднородного набора заданий в системе с несколькими приборами и не возобновляемым ресурсом

М.Г. Фуругян

ВЦ ФИЦ ИУ РАН, ВМК МГУ, Москва, Россия, E-mail: rtsccas@yandex.ru

Аннотация. Решается задача составления расписания выполнения неоднородного набора заданий в системе с несколькими идентичными приборами и не возобновляемым ресурсом. Часть заданий являются прерываемыми, а часть – непрерываемыми. Заданы характеристики заданий: время готовности, директивный срок и длительность выполнения, которая зависит от объема выделенного ресурса. Решается задача существования допустимого расписания в зависимости от директивных сроков и суммарного объема ресурса.

Ключевые слова: неоднородный набор заданий, система с несколькими приборами, допустимое расписание, время готовности, директивный срок, распределение ресурсов.

.

1. Введение

При проектировании, испытании и эксплуатации сложных технических объектов возникает необходимость в проведении вычислений и обработке больших массивов информации в строго определенные интервалы времени. Подобные задачи возникают при проведении летных испытаний, управлении производственными комплексами и транспортными сетями, в других областях человеческой деятельности. Для решения этих задач требуются соответствующие алгоритмы планирования вычислений. Результатом работы таких алгоритмов является расписание, показывающее, в какие временные интервалы должно выполняться каждое задание и какие ресурсы при этом будут ему выделены.

По данной тематике известно большое число публикаций, в которых в каждой исследуемой задаче все задания либо допускают прерывания и переключения с одного прибора на другой, либо не допускают. В этих работах дополнительный, помимо приборов, ресурс не рассматривается. Примером является работа [1]. В работе [2] рассмотрена задача со смешанным набором заданий (прерываемых и непрерываемых) без дополнительного ресурса, а в работах [3, 4] исследована задача построения допустимого расписания прерываемых работ с дополнительным ресурсом. В [4] решена задача оптимальной коррекции директивных интервалов для прерываемых работ в случае, когда времена готовности всех заданий совпадают.

В настоящей статье продолжается тематика по составлению расписаний в системах с дополнительным ресурсом, предложенная автором в работах [3 – 5] для прерываемых и непрерываемых работ. Решается задача составления допустимого расписания для неоднородного набора заданий (часть заданий – прерываемые, а часть – непрерываемые) в системе с несколькими идентичными приборами и дополнительным ресурсом не возобновляемого типа. Каждое задание характеризуется фиксированными параметрами – временем готовности и директивным сроком, а также нефиксированным – длительностью выполнения, которая зависит от объема ресурса, выделенного заданию (при увеличении объема ресурса, выделяемого заданию, его длительность выполнения уменьшается). Модельные примеры задач с нефиксированными параметрами рассмотрены в [4]. Решается также вопрос о существовании допустимого расписания в зависимости от директивного срока и объема дополнительного ресурса.

2. Формулировка задачи

Рассматривается набор подлежащих выполнению заданий (работ) $Z = Z_1 \cup Z_2$, Z_1 – прерываемые задания (при их выполнении допускаются прерывания и переключения с одного прибора на другой, временные затраты на обработку прерываний и переключений не учитываются), Z_2 – непрерываемые (при их выполнении прерывания и переключения не

допускаются). Число работ в Z_1 и Z_2 равно соответственно k_1 и k_2, $k_1 + k_2 = k$. Задания выполняются системой P, состоящей из p идентичных приборов, занумерованных от 1 до p, $P = \{1, 2, ..., p\}$, и, кроме того, работам может быть выделено некоторое количество не возобновляемого ресурса, суммарный объем которого составляет S. Параллельное выполнение одного задания несколькими приборами и одновременное выполнение одним прибором нескольких заданий не допускается. Каждое задание $i \in Z$ характеризуется фиксированными параметрами – временем готовности $a_i = 0$ и директивным сроком $b_i = F$ (т.е. все задания должны выполняться в интервале [0, F]), а также нефиксированным параметром – длительностью выполнения τ_i, которая зависит от объема ресурса s_i, выделенного заданию i, и выражается линейным соотношением $\tau_i = \tau_i^0 - c_i s_i$. Здесь

$$c_i \ge 0,\ 0 \le s_i \le s_i^0,\ \tau_i^0 - c_i s_i^0 > 0,\ \tau_i^0 \le F \qquad (1)$$

при всех $i \in Z$, c_i и τ_i^0 – известные величины. Суммарный объем ресурса, выделяемый всем заданиям, не должен превышать величины S:

$$\sum_{i \in Z} s_i \le S\ . \qquad (2)$$

Необходимо определить, можно ли так распределить ресурс между заданиями, чтобы каждое и них полностью успевало выполниться в директивном интервале [0, F]. В случае положительного ответа надо найти это распределение и допустимое расписание выполнения заданий. Требуется также при фиксированном директивном сроке F найти минимально допустимый объем ресурса $S_{\min}(F)$ и при фиксированном объеме ресурса S найти минимально допустимый директивный срок $F_{\min}(S)$.

Рассмотрим сначала два частных случая. В первом из них все работы являются прерываемыми, а во втором – непрерываемыми.

3. Построение допустимого расписания прерываемых работ

В этом разделе будем предполагать, что все работы допускают прерывания и переключения, т.е. $Z_2 = \varnothing,\ Z = Z_1,\ k_2 = 0,\ k_1 = k$. Воспользуемся результатом, полученным в [3], согласно которому для существования допустимого расписания необходимо и достаточно выполнение линейных неравенств (1), (2) и неравенства

$$\sum_{i \in Z} c_i s_i \ge \sum_{i \in Z} \tau_i^0 - pF. \qquad (3)$$

В случае, когда указанная система неравенств совместна, допустимое расписание строится с помощью решений $s_1, s_2, ..., s_k$ этой системы и длительностей $\tau_i = \tau_i^0 - c_i s_i$ выполнения заданий [1].

Для нахождения минимально допустимого объема ресурса $S_{\min}(F)$ при фиксированном директивном сроке F и минимально допустимого директивного срока $F_{\min}(S)$ при фиксированном объеме ресурса S рассмотрим следующие задачи линейного программирования. В первом случае минимизируется величина $S \ge 0$ при ограничениях (1) – (3). В случае существования решения у этой задачи получаем величину $S_{\min}(F)$. В противном случае ни для какого значения S искомого распределения ресурса и, следовательно, допустимого расписания не существует. Во втором случае минимизируется величина $F \ge 0$ при тех же ограничениях. Специфика ограничений (1) – (3) такова, что эта задача всегда имеет решение, которое и определяет величину $F_{\min}(S)$.

Предложим теперь для решения рассмотренных выше задач алгоритм, который будем называть алгоритмом приоритетного распределения ресурса. Этот алгоритм для заданного множества заданий $U \subseteq Z$ и заданных чисел $v > 0$ и $S \ge 0$ находит такое распределение ресурса $s_i,\ i \in U$, при котором $\sum_{i \in U} \tau_i = v$, $\sum_{i \in U} s_i \le S$, или определяет, что такого распределения не существует. При этом суммарная величина потребленного ресурса, т.е. $\sum_{i \in U} s_i$, должна быть минимально возможной.

Приведем описание алгоритма приоритетного распределения ресурса, который будем обозначать как $A(U, v, S, s_i, \tilde{S})$. Входными параметрами алгоритма являются U, v и S, а выходными – искомое распределение ресурса $s_i,\ i \in U$, и остаток дополнительного ресурса $\tilde{S} = S - \sum_{i \in U} s_i$, либо сообщение о том, что искомого распределения ресурсов не существует. В последнем случае распределение ресурса,

полученное в ходе работы алгоритма, также выдается на выходе.

Упорядочим задания $i \in U$ по не возрастанию величин c_i ($c_1 \geq c_2 \geq ...$), которые определяют приоритеты заданий. Будем выделять дополнительный ресурс заданиям поочередно, начиная с первого, т.е. сначала ресурс выделяется заданиям с большим приоритетом. Поскольку единица выделенного ресурса заданию с более высоким приоритетом даст большее уменьшение длительности его выполнения по сравнению с заданием с меньшим приоритетом, то по окончании работы алгоритма величина $\sum_{i \in U} s_i$ будет минимально возможной.

Покажем, как дополнительный ресурс выделяется первой работе. Для остальных работ эта процедура аналогична. Считаем, что сначала работам $i \in U$ дополнительный ресурс не выделен, т.е. $\tau_i = \tau_i^0, \quad i \in U$. Пусть $\sum_{i \in U} \tau_i = \bar{v}, \ \bar{v} > v$. Для того, чтобы после выделения дополнительного ресурса в объеме $\bar{s}_1$ первой работе величина $\bar{v}$ оказалась равной v, требуется выполнение равенства $\bar{v} - c_1 \bar{s}_1 = v$, или $\bar{s}_1 = (\bar{v} - v)/c_1$. В силу ограничений (1), (2) первой работе выделяем ресурс $s_1 = \min(\bar{s}_1, s_1^0, S)$. Если при этом $s_1 = \bar{s}_1$ (т.е. $\bar{v} = v$), то задача решена. Если $s_1 = S, \ s_1 < \bar{s}_1$ (т.е. $\bar{v} < v$, а дополнительный ресурс израсходован), то искомого распределения дополнительного ресурса не существует. Если же $s_1 = s_1^0$ и при этом $s_1 < S, \ s_1 < \bar{s}_1$, то следует перейти ко второй работе (предварительно уменьшив величину S на s_1, а величину $\bar{v}$ – на $c_1 s_1$) и т.д. Если при этом на некотором шаге будет выполнено равенство $\bar{v} = v$, то искомое распределение дополнительного ресурса найдено. Если же на некотором шаге окажется, что $\tilde{S} = 0$, а $\bar{v} > v$, или если всем заданиям $i \in U$ будет выделен максимально допустимый ресурс s_i^0 и при этом $\bar{v} > v$, то искомого распределения дополнительного ресурса не существует. Отметим, что сложность предложенного алгоритма (включая сортировку) составляет $O(n \log n)$.

Применим теперь алгоритм приоритетного распределения ресурса к решению поставленной задачи. Основываясь на условии (3), выполним следующее обращение к алгоритму A: $A(Z, pF, S, s_i, \tilde{S})$. В результате получим искомое распределение ресурса или сообщение о том, что его не существует.

4. Построение допустимого расписания непрерываемых работ

В этом разделе будем предполагать, что ни одна работа не допускает прерываний и переключений, т.е. $Z_1 = \varnothing, \ Z = Z_2, \ k_1 = 0, \ k_2 = k$. Такая задача является *NP*-трудной, т.к. при $S = 0$ к ней сводится по Тьюрингу известная *NP*-полная задача о разбиении.

Обратимся сначала к алгоритму A, с помощью которого получим такое распределение дополнительного ресурса, при котором суммарная длительность всех работ будет минимально возможной. Для этого выполним следующее обращение к алгоритму A: $A(Z, 0, S, s_i, \tilde{S})$. Затем вычислим искомые длительности выполнения заданий: $\tau_i = \tau_i^0 - c_i s_i, \quad i \in Z$. Отметим, что после работы алгоритма A при $v = 0$ длительность τ_i ни одной работы не может быть уменьшена, т.к. либо весь дополнительный ресурс S был исчерпан, либо каждой работе было выделено максимально допустимое количество s_i^0 дополнительного ресурса.

Для построения допустимого расписания воспользуемся приближенным полиномиальным алгоритмом, описанным в [2]. Это “жадный” алгоритм, предназначенный для нахождения расписания минимальной длины. Его сложность составляет $O(mn)$. Отметим, что для построения допустимого расписания можно также воспользоваться точным псевдополиномиальным алгоритмом [5].

Предположим, что отработал один из этих алгоритмов. Введем обозначения: V_j ($j = 1, ..., p$) – задания, приписанные прибору j, $F_j = \sum_{i \in V_j} \tau_i$, $F_0 = \max_{j=1, ..., p} F_j$. Если $F_0 > F$, то допустимое расписание не найдено; если $F_0 \leq F$, то построенное расписание – допустимое. Величину $F_{\min}(S)$ полагаем равной F_0.

Перейдем к вычислению величины $S_{\min}(F)$. Рассмотрим сначала некоторое множество работ V_j, для которого $F_j > F$. Наша цель – добиться выполнения равенства $F_j = F$

при минимальном потреблении дополнительного ресурса. Упорядочим работы в V_j по не возрастанию их приоритетов и будем рассматривать их поочередно, начиная с первой. Выбрав работу $i \in V_j$, путем увеличения выделяемого ей дополнительного ресурса, не нарушая ограничения (2), максимально приблизим величину F_j к F. Указанную процедуру выполняем по аналогии с тем, как это было описано для алгоритма A. При этом уменьшаем F_j и увеличиваем s_i на соответствующие величины. Если будет выполнено равенство $F_j = F$, то переходим к следующему множеству V_j, для которого $F_j > F$. В противном случае выбираем очередную работу $i \in V_j$.

Теперь рассмотрим некоторое множество V_j, для которого $F_j < F$. Наша цель – добиться выполнения равенства $F_j = F$ при максимальном освобождении дополнительного ресурса. Упорядочим работы в V_j по не убыванию приоритетов и будем рассматривать их поочередно, начиная с первой. Выбрав работу $i \in V_j$, путем уменьшения выделяемого ей дополнительного ресурса, не нарушая ограничения (2), максимально приблизим величину F_j к F. Указанную процедуру выполняем по аналогии с тем, как это было описано для алгоритма A. При этом увеличиваем F_j и уменьшаем s_i на соответствующие величины. Если будет выполнено равенство $F_j = F$, то переходим к следующему множеству V_j, для которого $F_j < F$. В противном случае выбираем очередную работу $i \in V_j$. Множества работ V_j, для которых $T_j = T$, оставляем без изменения.

Если после выполнения указанных процедур $\max\limits_{j=1,\ldots,p} F_j > F$, то допустимого расписания не существует ни при каком S. Если же $\max\limits_{j=1,\ldots,p} F_j \le F$, то величину $S_{\min}(F)$ полагаем равной $\sum\limits_{i \in Z} s_i$.

5. Построение допустимого расписания для неоднородного множества работ

Теперь рассмотрим задачу, когда имеются как прерываемые, так и непрерываемые работы. Выполним следующее обращение к алгоритму A: $A(Z, 0, S, s_i, \tilde{S})$ и вычислим длительности выполнения работ: $\tau_i = \tau_i^0 - c_i s_i$. $i \in Z$. При этом суммарная длительность всех работ $\sum\limits_{i \in Z} \tau_i$ будет минимально возможной при объеме S дополнительного ресурса. Далее, воспользуемся результатами работы [2]. Представим множество приборов P в виде объединения двух подмножеств: $P = P_1 \cup P_2$ с числом элементов, равным p_1 и p_2 соответственно. Величины p_1 и p_2 пропорциональны суммам длительностей заданий в Z_1 и Z_2, т.е.

$$p_1 = \left\lceil p \sum_{i \in Z_1} \tau_i \Big/ \sum_{i \in Z} \tau_i \right\rceil, \quad p_2 = p - p_1.$$

Задания Z_2 будут выполняться только приборами из P_2, а задания из Z_1 – всеми приборами из P. Пусть $P_1 = \{1, \ldots, p_1\}$, $P_2 = \{1, \ldots, p_2\}$.

Опишем процедуру $\Pi(F')$, которая определяет, существует ли допустимое расписание длины, не превосходящей заданной величины F', и строит его в случае положительного ответа. Процедура начинает работу с построения расписания выполнения непрерываемых заданий Z_2 на приборах P_2. Для этого используется алгоритм, описанный в разд. 4. Далее, с помощью алгоритма, описанного в [2], строится расписание выполнения прерываемых заданий Z_1 на приборах P_1, а также, если это возможно, на приборах P_2. Если при этом $F_j \le F'$ при всех $j = m_1 + 1, \ldots, m$ и работы Z_1 удалось "упаковать" в интервале $[0, F']$, то расписание длины, не превосходящей F', построено. В противном случае величину F' следует увеличить. В [2] указан метод построения такого отрезка $[L_1, L_2]$, что для некоторого $T' \in [L_1, L_2]$ существует допустимое расписание. Здесь

$$L_1 = \max\left(\frac{\sum\limits_{i \in Z} \tau_i}{p}, L_3, L_4 \right),$$

$$L_2 = \max\left(\frac{\sum\limits_{i \in Z_1} \tau_i + \min\limits_{j=p_1+1,\ldots,p} F_j}{p_1 + 1}, L_3, L_4 \right),$$

$$L_3 = \max_{j=p_1+1,\ldots,p} F_j, \quad L_4 = \max_{i \in Z_2} \tau_i.$$

Далее, применяя дихотомию к отрезку $[L_1, L_2]$, с помощью процедуры П можно определить величину $F_{\min}(S)$. Если $F_{\min}(S) \le F$, то допустимое расписание построено. В противном случае допустимое расписание не найдено. Применяя дихотомию к отрезку $[0, \sum_{i \in Z} s_i^0]$, с помощью процедуры $\Pi(F')$ вычисляется величина $S_{\min}(F)$.

Multiprocessor scheduling for a non-uniform set of jobs with not renewable resource

M.G. Furugyan

Abstract. The problem of scheduling in the multiprocessor computing system for a case when a part of jobs allows interruptions and switches from one processor to another, and a part does not allow is considered. Jobs are characterized by duration and a directive interval. Besides processors, in the system there is an additional not renewable resource, and duration of execution of each work linearly depends on quantity of this resource allocated to it. The minimum values of an additional resource and directive term at which there is an feasible schedule are defined.

Keywords: multiprocessing system, feasible schedule, directive term, available time, not renewable resource.

Литература

1. В.С. Танаев, В.С. Гордон, Я.М. Шафранский. Теория расписаний. Одностадийные системы. М., Наука, 1984.

2. Д.Р. Гончар, М.Г. Фуругян. Минимаксная задача планирования вычислений в многопроцессорной системе со смешанным набором работ. «Системы управления и информационные технологии», (2009), № 2 (36), 36 – 39.

3. Е.О. Косоруков, М.Г. Фуругян. Некоторые алгоритмы распределения ресурсов в многопроцессорных системах. «Вестн. МГУ. Сер. 15, Вычисл. математика и кибернетика», (2009), № 4, 34 – 37.

4. М.Г. Фуругян. Оптимальная коррекция директивных интервалов в задаче построения многопроцессорного расписания с дополнительным ресурсом. «Изв. РАН. ТиСУ». (2015), № 2, 107 – 116.

5. М.Г. Фуругян. Алгоритм решения задачи планирования вычислений в многопроцессорной системе с нефиксированными параметрами. «Труды НИИСИ РАН. Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты», (2018), т. 8, № 4, 86 – 89.

Эволюция и обучение в модели взаимодействия инвесторов и производителей

З. Б. Сохова[1], В. Г. Редько[2]
ФГУ ФНЦ НИИСИ РАН, Москва, Россия
E-mail's: [1]*zarema.sokhova@gmail.com,* [2]*vgredko@gmail.com*

Аннотация. В статье построена и исследована эволюционная модель взаимодействия инвесторов и производителей в экономической системе. Анализируются процессы *эволюции и обучения* в данном сообществе. Выполнен детальный анализ взаимодействия между обучением и эволюцией. Модель исследована с помощью компьютерного моделирования.

Ключевые слова: многоагентные системы, инвесторы, производители, эволюция, экономика.

Введение

В работах [1, 2] была предложена и исследована модель взаимодействия инвесторов и производителей в прозрачной экономической системе. В этой системе происходит открытый обмен информацией внутри экономического сообщества. При этом каждый инвестор, принимая решение о вкладах в производителей, знает свойства производителей и учитывает намечаемые вклады инвесторов.

В настоящей работе предлагается другой подход, когда нет открытого обмена информацией внутри сообщества инвесторов и производителей. В этом случае задача инвесторов существенно усложняется. Инвесторы должны найти разумное распределение капиталовложений в производителей, не зная заранее свойств производителей. Мы рассматриваем процесс эволюции сообщества и обучение инвесторов. Для простоты предполагаем, что производители могут быть эффективными или неэффективными. А инвесторы находят эффективных производителей, используя обучение и эволюцию аналогично работам [3-5].

1. Описание эволюционной модели

Пусть количество инвесторов и производителей равно N и M, соответственно. Предполагаем, что N и M велики: $N, M >> 1$, и что N и M не меняются в ходе эволюции. Рассматриваем эволюцию популяции инвесторов. Каждое поколение инвесторов живет в течение T периодов времени, T – время жизни одного поколения. Время дискретно: $t = 1, \ldots, T$. После $t = T$ происходит переход к следующему поколению эволюции.

Считаем, что половина производителей имеет эффективность $k_i = 1$ (эффективный производитель), а вторая половина – $k_i = 0$ (неэффективный производитель) (здесь и далее i – номер производителя, $i = 0, \ldots, M$). Эффективности производителей задаются в начале жизни первого поколения случайным образом и не меняются в течение всего эволюционного процесса. Формально считаем, что имеется вектор эффективностей производителей $\boldsymbol{G}_{pro}$, компоненты которого равны k_i.

В каждом периоде t инвесторы делают капиталовложения в производителей, а в конце периода получают прибыль. Считаем, что инвесторам выгодно делать капиталовложения в эффективных производителей.

Каждый инвестор имеет генотип и фенотип. Генотипы инвесторов $\boldsymbol{G}_{inv\,j}$ формируются из *степеней доверия* g_{ij}, где g_{ij} — это степень доверия j-го инвестора к i-му производителю (здесь и далее j – номер инвестора, $j = 0, \ldots, N$). Если инвестор доверяет производителю, то соответствующий символ в генотипе равен 1, иначе символ равен 0. Таким образом, каждый инвестор j имеет генотип, определяемый цепочкой $\boldsymbol{G}_{inv\,j} = \{g_{ij}\}$, g_{ij} равно 0 или 1. В начале эволюции генотипы инвесторов случайны. При переходе к новому поколению происходит отбор инвесторов и мутации их генотипов. То есть, генотипы потомков инвесторов отличаются от генотипов родителей небольшими мутациями. При мутации символ g_{ij} в генотипе инвестора с вероятностью p_m заменяется на 0 или 1.

Каждый инвестор характеризуется еще своим фенотипом. В качестве фенотипа

выступает *текущая* степень доверия $\boldsymbol{D}_j = \{d_{ij}\}$, которая в момент рождения агента инвестора совпадает с генотипом $\boldsymbol{G}_{inv\,j}$, т. е. $d_{ij} = g_{ij}$. Генотипы g_{ij} изменяются только в процессе эволюции, а текущие степени доверия (фенотипы) d_{ij} могут меняться от периода к периоду в *процессе обучения*. Инвесторам нужно найти цепочку степеней доверия с компонентами, равными 0 либо 1, которая наиболее близка к цепочке эффективностей производителей $\boldsymbol{G}_{pro}$. Оптимальную цепочку для фенотипов $\boldsymbol{D}_{max}$ каждый инвестор может найти в процессе обучения, а цепочка для генотипов $\boldsymbol{G}_{max}$ может быть найдена в процессе эволюции. Очевидно, что обе оптимальные цепочки совпадают с цепочкой вектора эффективностей производителей $\boldsymbol{G}_{pro}$, т. е. $\boldsymbol{G}_{max} = \boldsymbol{D}_{max} = \boldsymbol{G}_{pro}$.

В конце жизни каждого поколения происходит отбор инвесторов в следующее поколение в соответствии с конечными фенотипами. Если инвестор отбирается в следующее поколение, то он получает генотип $\boldsymbol{G}_{inv\,j}$ от родителя (с мутациями), а фенотипы инвесторов $\boldsymbol{D}_j$ в начале жизни каждого поколения равны генотипам $\boldsymbol{G}_{inv\,j}$. Фенотипы инвесторов $\boldsymbol{D}_j$ меняются в каждом периоде t путем обучения.

2. Процесс обучения

Обучение для каждого инвестора происходит методом проб и ошибок следующим образом. В цепочке фенотипа инвестора $\boldsymbol{D}_j$ каждый символ заменяется случайным образом на 0 или 1. После этого j - й инвестор оценивает новую возможную прибыль от i-го производителя P_{ij}. Почти во всех случаях считаем, что прибыль j-го инвестора от i-го производителя равна $P_{ij} = k_i\, d_{ij}$. Кроме того, считаем, что, если инвестор доверяет производителю ($d_{ij} = 1$), а прибыли нет ($k_i = 0$), это приводит к ненужному расходу инвестора, поэтому в этом случае прибыль равна –1: $P_{ij} = -1$ (Таблица 1).

Таблица 1. Прибыль инвестора P_{ij} при различных d_{ij} и k_i

d_{ij}	k_i	P_{ij}
0	0	0
0	1	0
1	0	-1
1	1	1

Затем сравниваются прибыли P_{ij} до замены и после замены символа в генотипе. Если прибыль после замены увеличилась, то принимается новое d_{ij}. Если прибыль после замены уменьшилась, то восстанавливается старое значение d_{ij}.

В конце каждого периода рассчитывается суммарная прибыль $sumP_j$ каждого инвестора:

$$SumP_j = \sum_{i=1}^{M} P_{ij}. \qquad (1)$$

Если инвестор правильно обучился, то $sumP_j = M/2$.

Таким образом, при обучении происходит приближение цепочек фенотипов $\boldsymbol{D}_j$ к оптимальной цепочке $\boldsymbol{G}_{max}$.

Анализ схемы обучения с учетом Таблицы 1 показывает, что расстояние по Хеммингу между цепочкой фенотипов $\boldsymbol{D}_j$ и оптимальной цепочкой $\boldsymbol{G}_{max} = \boldsymbol{G}_{pro}$ для j-го инвестора определяется выражением:

$$\rho(\boldsymbol{D}_j, \boldsymbol{G}_{max}) = M/2 - sumP_j\,. \qquad (2)$$

3. Отбор инвесторов

В конце поколения происходит отбор инвесторов в следующее поколение в соответствии с их приспособленностями. Считаем, что приспособленность каждого инвестора равна:

$$f_j = \exp(sumP_j - M/2) + \varepsilon\,, \qquad (3)$$

где малый положительный параметр ε ($0 < \varepsilon << 1$) учитывает случайность внешней среды. При $sumP_j = M/2$ приспособленность максимальна. Без обучения приспособленность мала.

Для отбора используется *метод пропорционального отбора* (fitness proportionate selection) [3, 6]. В новую популяцию отбирается ровно N инвесторов. Отбор инвесторов зависит от конечных фенотипов $\boldsymbol{D}_j$, а по наследству передаются генотипы $\boldsymbol{G}_{inv\,j}$ (с малыми мутациями), которые инвесторы получили от своих родителей.

Таким образом, рассматривается *дарвиновская эволюция*, так как по наследству передаются потомкам генотипы (с малыми мутациями), а не фенотипы.

Дополнительно можно учесть нагрузку на инвесторов при обучении. В этом случае будем считать, что приспособленность инвестора уменьшается под действием этой нагрузки:

$$fm_j = \exp(-\alpha r)\ \{\exp(sumP_j - M/2) + \varepsilon\}\ , \quad (4)$$

где $\alpha > 0$ параметр, который учитывает нагрузку, связанную с обучением, $r = |sumP0_j - sumP_j|$ – разница между суммарными прибылями до обучения ($sumP0_j$) и после обучения ($sumP_j$) для рассматриваемого инвестора. Таким образом, чем больше интенсивность обучения, тем большую нагрузку несет инвестор.

4. Компьютерный анализ модели

4.1. Общий подход и параметры моделирования

Предложенная выше модель исследована с помощью компьютерного моделирования. При численных расчетах определялась динамика средних по популяции расстояния по Хеммингу до оптимума $\langle\rho\rangle = \langle\rho(\boldsymbol{G}_{inv\,j}, \boldsymbol{G}_{max})\rangle$ и суммарной прибыли инвесторов $\langle SumP\rangle$.

Моделирование проводилось для двух режимов: 1) с обучением и 2) без обучения, в этом случае считается, что фенотипы все время равны генотипам $\boldsymbol{D}_j = \boldsymbol{G}_{inv\,j}$ и, тем самым, отбор согласно приспособленностям (3) происходит в соответствии с генотипами. Во втором случае имеем режим «чистой эволюции». Также проводилось моделирование в режиме 3) с дополнительной нагрузкой на обучение. В этом случае приспособленность инвестора рассчитывалась по формуле (4).

Параметры расчетов были выбраны следующим образом. Длина цепочки $\boldsymbol{G}_{pro} = \{k_i\}$, определяющей эффективности производителей k_i, была достаточно велика: $M = 100$. $k_i = 1/0$ для эффективного/неэффективного производителя. Количество инвесторов полагаем также равной 100, т. е. $N = 100$, интенсивность мутаций (вероятность замены каждого символа генотипа в одном поколении) $p_m = N^{-1} = 0.01$, число периодов времени в течение одного поколения полагаем равным $T = 2$. Параметр $\varepsilon = 10^{-5}$. Полученные результаты были усреднены по 2000 расчетам.

Ниже представлены результаты компьютерного моделирования.

4.2. Результаты моделирования

На рис. 1 представлена зависимость от номера поколения T_G среднего по популяции расстояния по Хеммингу $\langle\rho\rangle = \langle\rho(\boldsymbol{G}_{inv\,j}, \boldsymbol{G}_{max})\rangle$ между цепочками генотипов $\boldsymbol{G}_{inv\,j}$ и оптимальной цепочкой $\boldsymbol{G}_{max}$. Приспособленности особей определялись выражением (3). Данный эксперимент иллюстрирует, что при отсутствии обучения эволюционный процесс не приводит к нахождению оптимального генотипа $\boldsymbol{G}_{max}$. Если же обучение присутствует, эволюция приводит к нахождению оптимальной цепочки.

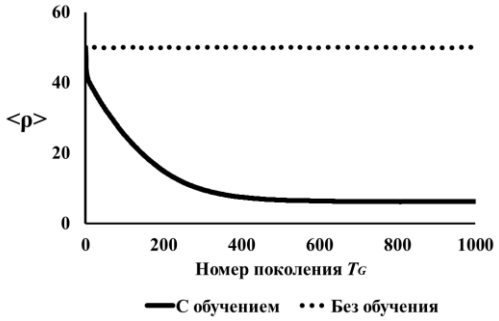


Рис. 1. Зависимость среднего по популяции расстояния между цепочками генотипов инвесторов $\boldsymbol{G}_{inv\,j}$ и оптимальной цепочкой $\boldsymbol{G}_{max}$ $\langle\rho\rangle = \langle\rho(\boldsymbol{G}_{inv\,j}, \boldsymbol{G}_{max})\rangle$ от номера поколения T_G

Как и в работе [3], в данной модели не происходит уменьшения ρ в процессе «чистой эволюции» за счет того, что все инвесторы в этом режиме имеют приближенно одинаковое значение приспособленности, равное ε. Отбор инвесторов в новое поколение осуществляется вероятностно, а так как приспособленности одинаковые, то не происходит селекции лучших инвесторов.

Динамика средней суммарной прибыли инвесторов *SumP* для этих режимов представлена на рис. 2.

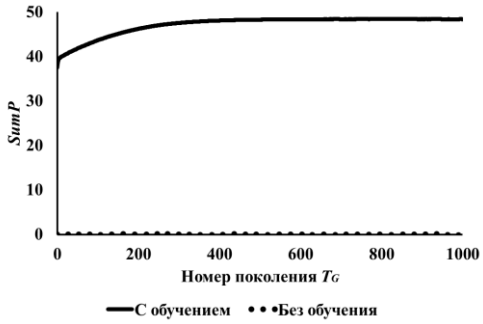


Рис. 2. Зависимость средней суммарной прибыли *SumP* инвесторов от номера поколения T_G

Видно, что в модели с обучением средняя суммарная прибыль *SumP* приближается к максимально возможной средней прибыли равной *M*/2. В модели без обучения средняя суммарная прибыль близка к 0.

На рис. 3 показано, как происходит эффект ускорения эволюционного процесса за счет обучения. Постепенное уменьшение расстояния между цепочками генотипов $\rho(\boldsymbol{G}_{inv\,j}, \boldsymbol{G}_{max})$ происходит следующим образом. При обучении расстояния $\rho = \rho(\boldsymbol{D}_j, \boldsymbol{G}_{max})$ становятся достаточно малыми и распределение инвесторов $n(\rho)$ по фенотипам $\boldsymbol{D}_j$ смещается в сторону меньших ρ при этом приспособленности инвесторов значительно различаются и в результате в новое поколение отбираются инвесторы с малыми значениями $\rho = \rho(\boldsymbol{D}_j, \boldsymbol{G}_{max})$.

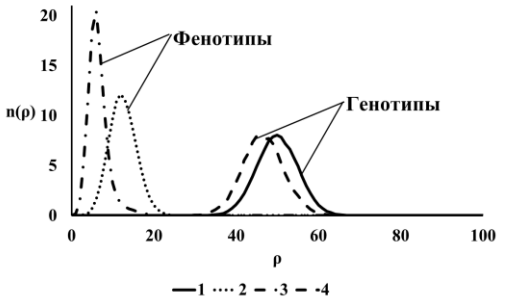


Рис. 3. Распределение инвесторов $n(\rho)$ по величинам ρ в первом поколении эволюции: 1 – распределение по $\rho = \rho(\boldsymbol{G}_{inv\,j}, \boldsymbol{G}_{max})$ для исходных генотипов до обучения, 2 – распределение по $\rho = \rho(\boldsymbol{D}_j, \boldsymbol{G}_{max})$ для фенотипов инвесторов после обучения, но еще до отбора, 3 – распределение по $\rho = \rho(\boldsymbol{D}_j, \boldsymbol{G}_{max})$ для отобранных инвесторов, 4 – распределение по $\rho = \rho(\boldsymbol{G}_{inv\,j}, \boldsymbol{G}_{max})$ для генотипов отобранных инвесторов в конце поколения

Таким образом, в результате отбора происходит селекция инвесторов, генотипы которых также приближаются к $\boldsymbol{G}_{max}$. В результате при переходе к новому поколению величина $\rho = \rho(\boldsymbol{G}_{inv\,j}, \boldsymbol{G}_{max})$ в популяции уменьшается, соответственно увеличивается суммарная прибыль $SumP_j$, и в ходе эволюции средняя суммарная прибыль для популяции инвесторов приближается к оптимальной прибыли, которую может получить инвестор.

Также был проведен анализ влияния нагрузки на обучении инвесторов на моделируемые процессы. При этом использовалось выражение (4) для приспособленности инвесторов. Показано, что учет нагрузки на обучение приводит к ускорению нахождения оптимальной цепочки $\boldsymbol{G}_{max}$ инвесторами и к росту скорости роста прибыли инвесторов.

Заключение

Таким образом, построена и исследована модель взаимодействия между обучением и эволюцией для специального случая взаимодействия популяции инвесторов и популяции производителей.

Показано, что обучение инвесторов может значительно ускорять эволюционный поиск максимальной эффективности вложений их капиталов в производителей. Нагрузка на обучение может проводить к дополнительному ускорению этого поиска.

Настоящая работа выполнена в рамках государственного задания по проведению фундаментальных научных исследований по теме "Исследование нейроморфных систем обработки больших данных и технологии их изготовления" (№ 0065-2019-0003).

Interaction of evolution and learning in the economic model of capital allocation

Z.B. Sokhova, V.G. Red'ko

Abstract. The article proposes an evolutionary model of interaction between investors and producers in the economic system. The processes of evolution and learning in this community are analyzed. The interaction between learning and evolution is analyzed in detail. The model is investigated by computer simulation. The mechanisms of interaction between learning and evolution are investigated.

Keywords: multi-agent systems, investors, producers, evolution, economy.

Литература

1. V.G. Red'ko, Z.B. Sokhova. Iterative method for distribution of capital in transparent economic system. «Optical memory and neural networks (information optics)», v. 26 (2017), No. 3, 182 − 191.
2. В.Г. Редько, З.Б. Сохова. Модель взаимодействия инвесторов и производителей в прозрачной экономической системе. «Экономика и математические методы». том 54 (2018), № 2, 50 – 61.
3. В.Г. Редько. Модель взаимодействия между обучением и эволюционной оптимизацией. «Математическая биология и биоинформатика (электронный журнал)». т. 7(2012). № 2. 676–691. URL: http://www.matbio.org/2012/Redko_7_676.pdf
4. G.E. Hinton, S.J. Nowlan. How learning can guide evolution. «Complex Systems». V. 1(1987). No. 3. 495–502.
5. G. Mayley. Guiding or hiding: Explorations into the effects of learning on the rate of evolution. «Proceedings of the Fourth European Conference on Artificial Life (ECAL 97) (Eds. Husbands P., Harvey I.)». Cambridge, Massachusetts: MIT Press,1997, 135–144.
6. В.Г. Редько. Моделирование когнитивной эволюции: На пути к теории эволюционного происхождения мышления. Изд. 2, испр. и доп. М., ЛЕНАНД/URSS, 2018.

Моделирование чувства причинности. Первые результаты

В. Г. Редько
ФГУ ФНЦ НИИСИ РАН, Москва, Россия
E-mail: vgredko@gmail.com

Аннотация. В статье построена модель формирования чувства причинности. Рассмотрена модель двух типов автономных агентов: 1) с чувством причинности и 2) без такого чувства. Агенты с чувством причинности запоминают связи между текущими ситуациями, своими действиями и результатами действий. На основе запомненной информации эти агенты строят базы знаний, характеризующие такие связи, а затем используют эти базы знаний при выборе действий в своем поведении. Агенты без чувства причинности выбирают действия случайным образом. Модель проанализирована путем компьютерного моделирования при целенаправленном поведении агентов. Показано, что агенты с чувством причинности находят цель значительно быстрее, чем агенты без чувства причинности.

Ключевые слова: автономные агенты, чувство причинности, целенаправленное поведение.

Введение

В «Исследовании о человеческом познании» (1748 г.) Давид Юм подверг сомнению понятие причинной связи [1]. А именно, он задался вопросом: почему, когда мы видим, что за одним явлением A постоянно следует другое B, то мы приходим к выводу, что A является причиной B? Например, когда мы наблюдаем, что Солнце освещает камень, и камень нагревается, то мы говорим, что солнечный свет есть причина нагревания камня.

Фактически Юм задался вопросом: что заставляет нас делать выводы о происходящих в природе явлениях? Что лежит в основе этих выводов? Юм попытался понять, откуда мы берем основание заключать, что A есть причина B. Он посмотрел на этот вопрос, как он пишет, со всех сторон и не нашел никакого другого основания, кроме некоторого внутреннего чувства привычки. То есть, имеется какое-то наше внутреннее свойство, которое заставляет нас утверждать, что если за A постоянно следует B, то A есть причина B. И это внутреннее чувство заставляет нас после того, когда мы сделали такое умозаключение, и снова видим событие A, ожидать, что за A вновь последует и событие B.

В настоящей работе предлагается формальная модель внутреннего чувства привычки у автономных агентов. Причем модель строится в максимально общем виде, чтобы характеризовать наиболее общие свойства понятия причинность. Модель проанализирована на примере целенаправленного поведения путем компьютерного моделирования.

1. Описание модели

Предполагаем, что имеется агент, взаимодействующий со внешней средой. Имеется N_S ситуаций, в которые может попадать агент. Одна из ситуаций является целевой S_G, считаем, что в этой ситуации агент получает некую награду. Время дискретно: $t = 1, 2,\ldots$ Имеется N_A возможных действий агента. Каждый такт времени агент выполняет одно действие. Связь между ситуациями, действиями агента и результатами действий четко определена. Эта связь между текущей ситуаций $S_{current}$ и текущим действием $A_{current}$ и следующей ситуацией S_{next} характеризуется соотношением:

$$\{S_{current}, A_{current}\} \rightarrow S_{next} \ . \qquad (1)$$

Выражение (1) характеризует причинные связи взаимодействия агента со внешней среды.

Имеется два типа агентов: 1) агент с внутренним чувством причинности, 2) агент без такого чувства.

Агент с чувством причинности предварительно самообучается. Перед обучением он находится в некоторой стартовой ситуации S_S. В процессе самообучения этот агент познает и использует причинные связи. Он выполняет случайно различные действия и в конце концов находит целевую ситуацию S_G. После этого агент находит те ситуации (пред-целевые ситуации), из которых можно

попасть в целевую. При этом он запоминает причинные связи вида (1) и действия, которые приводят к целевой ситуации. Далее агент продолжает анализ поиска действий, приводящих к пред-целевым ситуациям. При этом он запоминает новые причинные связи вида (1). Также происходит запоминание числа действий, которые приводят к целевой ситуации из рассматриваемой ситуации. Аналогичный анализ и запоминание связей происходят до тех пор, пока агент не дойдет до стартовой ситуации S_S. В результате этого обучения формируется база знаний агента. Пример такой базы знаний, полученной в результате обучения, приведен в Таблице 1.

Таблица 1. База знаний агента с чувством причинности

Текущая ситуация, $S_{current}$	Текущее действие, $A_{current}$	Следующая ситуация, S_{next}	$\rho(S_{current}, S_5)$	$\rho(S_{next}, S_5)$
S_3	A_5	S_2	3	2
S_2	A_2	S_4	2	1
S_4	A_1	S_5	1	0

В таблице приведен результат отдельного компьютерного расчета. Число ситуаций и действий для приведенного результата расчета было равно 5: $N_S = 5$, $N_A = 5$. Индексы ситуаций и действий в таблице обозначают их номера. Стартовая ситуация была S_3: $S_S = S_3$, целевая ситуация была S_5: $S_G = S_5$. В этой таблице $\rho(S_{current}, S_5)$ / $\rho(S_{next}, S_5)$ – расстояние между ситуацией $S_{current}$ / S_{next} и ситуацией цели S_5; это расстояние ρ равно количеству действий, необходимых для достижения цели S_5 из рассматриваемой ситуации ($S_{current}$ или S_{next}). Согласно таблице агент может мысленно представить последовательность ситуаций и действий, которые приводят к ситуации цели. База знаний характеризует цепочку действий ($A_5 \rightarrow A_2 \rightarrow A_1$), выполняя которые, агент может, начиная от стартовой ситуации S_3, перейти к целевой ситуации S_5.

Отметим, что рассмотренная схема обучения близка к схеме формирования базы знаний в моделях формирования планов поведения рыбами или воронами [2, 3]. Аналогичная база знаний строится модельными воронами и в работе [4]. Подчеркнем, что в настоящей модели агент с чувством причинности формирует базу знаний самостоятельно в результате самообучения. При этом агент явно использует чувство причинности, а именно он запоминает причинные связи между ситуациями, действиями и результатами действий в соответствии с выражением (1).

Далее агент с чувством причинности использует базу знаний, т. е. выполняет действия, приводящие к уменьшению расстояния между текущей ситуацией и целевой ситуацией. Для базы знаний, представленной в таблице 1, агенту, использующему эту таблицу, достаточно выполнить 3 действия (A_5, A_2, A_1). Он выполняет их за 3 такта времени.

Агент, не имеющий чувства причинности, также может прийти к целевой ситуации, выполняя различные действия, случайно выбирая их из числа возможных. При этом ситуации будут меняться согласно правилам, характеризуемым выражением (1). Хотя правил этих агент без чувства причинности не знает и не запоминает, он просто ведет случайный поиск.

2. Результаты компьютерного моделирования

Модель анализировалась путем компьютерного моделирования. Агенты с чувством причинности предварительно самообучались и формировали базы знаний (аналогичные приведенной в таблице 1), а затем использовали эти базы при формировании поведения, направленного на достижение целевой ситуации. Число ситуаций N_S и число возможных действий N_A варьировалось. Для определенности считалось, что $N_A = N_S$. Определялось время (число тактов), за которое агенты находили целевую ситуацию S_G. Для получения надежных данных результаты усреднялись по 10000 независимым расчетам. Ошибка в результатах расчетов была порядка 1%. Зависимость времени нахождения целевой ситуации от числа ситуаций N_S для обоих видов агентов приведена на рис. 1.

Видно, что агенты с чувством причинности, запоминающие и использующие свои знания, находят целевую ситуацию значительно быстрее, чем агенты без чувства причинности.

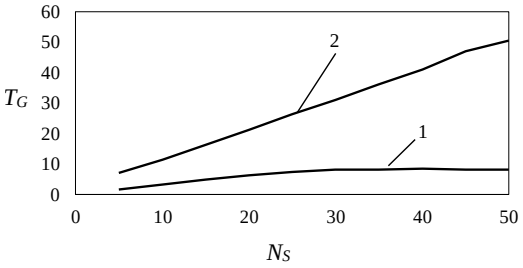


Рис. 1. Зависимость времени нахождения T_G целевой ситуации S_G от числа ситуаций N_S. 1 – агенты с чувством причинности, 2 – агенты без чувства причинности. Усреднено по 10000 независимым расчетам.

Выводы

Таким образом, построена и проанализирована модель чувства причинности автономных агентов. Агенты с чувством причинности в процессе самообучения запоминают причинные связи между ситуациями, своими действиями и результатами действий. А затем, используя полученные знания, находят путь к целевой ситуации. Агенты без чувства причинности находят целевую ситуацию случайным образом. Путем компьютерного моделирования показано, что агенты с чувством причинности находят целевую ситуацию значительно быстрее, чем агенты без чувства причинности. Модель пока исследована в максимально упрощенном и максимально общем случае. В дальнейшем возможно использование иерархической структуры причинных связей, использование связей между знаниями различных типов.

Подчеркнем, что использование свойства причинности, использование предсказаний – одно из ключевых свойств познания закономерностей внешнего мира. Изучение этого свойства важно при моделировании когнитивной эволюции, той эволюции, в результате которой произошло наше мышление, используемое в научном познании природы [5]. Отметим также, что рассмотренные в настоящей статье базы знаний могут рассматриваться как простые внутренние модели автономных агентов [6].

Настоящая работа выполнена при финансовой поддержке РФФИ, грант № 19-01-00331.

Modeling of feelings of causality. First results

V.G. Red'ko

Abstract. The article builds a model of the formation of a feeling of causality. The model considers two types of autonomous agents: 1) with a feeling of causality and 2) without such a feeling. Agents with the feeling of causality remember the causal relations and use their knowledge at their behavior. The model was analyzed by computer simulation. It is shown that agents with the feeling of causality find a goal much faster than other agents.

Keywords: the feeling of causality, autonomous agents, goal-directed behavior.

Литература

1. Д. Юм. Исследование о человеческом познании. Соч. в 2-х томах, т. 2. М.: Мысль, 1966, 5–169.
2. V.G.Red'ko, V.A.Nepomnyashchikh, E.A.Osipova. Models of fish exploratory behavior in mazes. "Biologically Inspired Cognitive Architectures", vol. 15 (2015), 9–16.
3. V.G.Red'ko, V.A.Nepomnyashchikh. Model of plan formation by New Caledonian crows. "Procedia Computer Science", vol. 71 (2015), 248–253. URL: http://www.sciencedirect.com/science/article/pii/S1877050915036820
4. V.G.Red'ko, M.S.Burtsev. Modeling of mechanism of plan formation by New Caledonian crows. "Procedia Computer Science", vol. 88 (2016), 403–408. URL: http://www.sciencedirect.com/science/article/pii/S1877050916317124
5. В.Г.Редько. «Моделирование когнитивной эволюции: На пути к теории эволюционного происхождения мышления». Изд. 2, испр. и доп. М.: URSS, 2018.
6. В.Г.Редько. Внутренние модели внешней среды, формируемые и используемые автономными агентами. «Труды НИИСИ РАН», т. 8 (2018), № 6, 59–64.

О проблемах применения векторного сопроцессора для ускорения обработки массивов данных, не выровненных в памяти

А.А. Бурцев

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail: burtsev@niisi.msk.ru

Аннотация: Векторные сопроцессоры, относящиеся к классу процессоров SIMD-архитектуры, позволяют значительно ускорить обработку больших массивов данных. Но для этого требуется, как правило, чтобы такие массивы были расположены в памяти по кратным адресам и обрабатывались большими порциями кратной длины. Эти жёсткие требования существенно ограничивают возможности применения таких SIMD-сопроцессоров. Чтобы снять имеющиеся ограничения и тем самым расширить область применения этих сопроцессоров, необходимо решить для них так называемую «проблему выравнивания», т.е. усовершенствовать применяемое программное (и/или аппаратное) обеспечение так, чтобы стало возможным обрабатывать с помощью векторного сопроцессора ещё и такие массивы данных, которые могут быть расположены в памяти по произвольному адресу, не обязательно выровненному по границе 64-, 128- или 256-разрядного машинного слова.

Способы решения этой проблемы и являются основным предметом обсуждения в данной статье. Предлагаемые возможные приёмы обработки не выровненных массивов данных сопровождаются демонстрацией примеров программ, разработанных автором для векторного сопроцессора CPV микропроцессора BM8 семейства КОМДИВ.

Ключевые слова: SIMD-архитектура, микропроцессоры семейства КОМДИВ, векторный сопроцессор, проблема не выровненного доступа к памяти.

Введение

В современных суперкомпьютерных системах ускорение высокопроизводительных вычислений (HPC) обеспечивается, как правило, за счёт того или иного вида параллелизма, т.е. за счёт исполнения программы, разработанной для решения вычислительной задачи, не одним, а сразу несколькими процессорами. При этом, естественно, предполагается, что алгоритм решения задачи поддаётся распараллеливанию, т.е. разбиению на такие участки, которые могут исполняться разными процессорами параллельно или одновременно (т.е. в одно и то же время). И ожидается (чисто теоретически), что в результате такой совместной параллельной работы многих (n) процессоров можно рассчитывать на общее ускорение вычислительной обработки в несколько (n) раз (если, конечно, каждый из n процессоров сможет выполнить свою часть работы одновременно с другими и независимо от них).

Применяются разные виды параллельной обработки (параллелизма) вычислительной программы: на разных компьютерах, связанных сетью (1); на разных компьютерах, связанных общей памятью (2); на разных процессорах (процессорных ядрах) одного компьютера (3); и наконец, возможен даже параллелизм на уровне исполнения одной команды (4).

Как особую разновидность упомянутого выше 4-го вида параллелизма, отметим использование специализированных сопроцессоров SIMD-архитектуры, ориентированных на единообразную параллельную обработку элементов больших массивов данных. Поскольку в таких сопроцессорах одна и та же вычислительная команда применяется одновременно сразу к нескольким (k) величинам, содержащимся в разных частях (секциях) используемых регистров, то одной командой можно выполнить одинаковую операцию сразу над несколькими (k) элементами массива, если загрузить в соответствующие регистры требуемое число (k) очередных элементов массива, подлежащих вычислительной обработке.

В настоящее время такие сопроцессоры можно встретить (в качестве расширения) почти в каждом семействе современных микропроцессоров. Это, например, MMX, SSE, SSE2, SSE3 (Intel,AMD); AltiVec/VMX (Apple, IBM, Motorola); MSA (MIPS SIMD Architecture) [4]. В семействе микропроцессоров

КОМДИВ примером такого SIMD-сопроцессора является векторный сопроцессор CPV микропроцессора BM8 [1]. Применение такого CPV позволило значительно ускорить типичные функции обработки векторов и матриц, характерные для задач линейной алгебры и цифровой обработки сигналов, что было реально подтверждено на практике путём разработки ряда экспериментальных тестовых программ [2,3].

Однако, весьма значительного ускорения (в несколько раз, см. таблицы 9 и 15-20 в [2]) удалось добиться лишь для непрерывных векторов кратной длины (=16·t), элементы которых располагаются в памяти друг за другом и начинаются с кратного (/16 или /32) адреса, выровненного по границе блока машинных слов, читаемых или записываемых одной коман-дой CPV в один 128-разрядный регистр CPV (для команд vldm/vsdm) или в два смежных (для команд vldq/vsdq) регистра CPV. Разработанные тогда для CPV процедуры обработки векторов не могли корректно обрабатывать массивы, не расположенные (не выровненные) по кратному адресу (а также массивы произвольной длины, не кратной 16). Почему? А пото-му, что для этого надо было каким-то образом преодолеть ограничения касательно кратности длины и адреса подвергаемого(-ых) обработке массива(-ов).

Подобного рода ограничения являются существенным недостатком почти всех SIMD-сопроцессоров, кроме тех, где такое ограничение уже преодолено аппаратным способом, как например, в MSA (см. команды LD.W SD.W в [4]). А если аппаратура не поддерживает необходимый доступ к памяти по не выровненным адресам, приходится преодолевать это ограничение программным способом. А для этого надо выработать приёмы решения так называемой проблемы обработки «не выровненных» (misaligned) данных. Другими словами, требуется усовершен-ствовать применяемое программное обеспечение так, чтобы стало возможным обрабатывать с помощью векторного сопроцессора (в частности, CPV) ещё и такие массивы данных, которые могут быть расположены в памяти по произвольному адресу, не обязательно выровненному по границе 128- или 256-разрядного машинного слова.

Способы решения этой проблемы и являются основным предметом обсуждения в данном докладе. Предлагаемые возможные приёмы обработки не выровненных массивов данных сопровождаются демонстрацией примеров программ, разработанных автором для векторного сопроцессора CPV микропроцессора BM8 семейства КОМДИВ. А также приводятся оценки, в какой степени снижается производительность и возрастает трудоёмкость (сложность) программ, усовершенствованных в таком направлении.

1. Специфика обработки массивов данных на векторном сопроцессоре CPV

Применение векторного сопроцессора CPV позволяет значительно (в разы!) ускорить такие процедуры обработки массивов данных, элементы которых могут обрабатываться одновременно (параллельно) и независимо друг от друга. Это было подтверждено практической разработкой для CPV микропроцессора BM8 семейства КОМДИВ ряда типичных функций обработки векторов и матриц, используемых при решении задач линейной алгебры [2] и цифровой обработки сигналов [3].

Однако, эффективная реализация таких функций на CPV первоначально была выполнена в расчёте на то, что обрабатываемые ими массивы данных удовлетворяют особым требованиям: это должны быть массивы кратной длины (1), и они обязательно должны быть расположены в памяти по так называемым выровненным адресам (2). Необходимость соблюдения этих требований была продиктована спецификой исполнения вычислительных команд векторного сопроцессора CPV, а также его команд доступа к памяти.

Каждая вычислительная команда CPV исполняет одну и ту же операцию сразу над несколькими секциями данных, загруженных в регистры CPV. Для определённости будем далее применять такие команды, которые обрабатывают 128-разрядные регистры CPV так, как будто в их 4-х секциях (в разрядах соответственно: 0-31, 32-63, 65-95, 96-127) расположены четыре (4) 32-разрядных числа в формате плавающей точки одинарной точности (величины в таком формате на языке Си объявляются типа float). Таким образом, одной командой CPV можно выполнить сразу 4 операции (по одной в каждой секции) над данными, загруженными в регистры CPV.

Чтобы обеспечить с той же скоростью подкачку данных в обрабатываемые регистры, в CPV предусмотрены команды (vLdm,vSdm), позволяющие загрузить в регистр CPV сразу 4 числа типа float, расположенных в памяти подряд друг за другом, а также выгрузить такие

4 числа из регистра CPV обратно, записав их в память в том же порядке в виде единого 16-байтного блока. Оптимальным сочетанием таких вычислительных команд и команд работы с памятью как раз и удаётся в итоге добиться на CPV 4-х кратного ускорения обработки массивов вещественных данных одинарной точности (с элементами типа float).

Чтобы пояснить, как достигается такой выигрыш, сравним различные реализации (на языке Си, на ассемблере для CP1 и для CPV) простейших типичных функций обработки векторов вещественных чисел одинарной точности, непрерывно (с шагом 1) расположенных в памяти (заметим, что выбранные нами функции являются характерными представителями 1-ого уровня функций библиотеки BLAS, см. таблицу 1):

Таблица 1. Типичные функции обработки векторов (библиотеки BLAS 1-го уровня)

имя (параметры)	схематичное описание функции
sSCAL1 (n,α,X)	X=α·X : { X_i=α·X_i, i=0…n-1 }
sCOPY1 (n,Y,X)	Y= X : { Y_i= X_i, i=0..n-1 }
sAXPY1 (n,Y,X,α)	Y=Y+α·X:{Y_i=Y_i+α·X_i,i=0..n-1}

Для полной характеристики этих функций сначала представим возможные варианты их реализаций на языке Си:

C_**sSCAL1**	void C_**sSCAL1** (int N,float *A,float *X) { int n; float a=*A; for (n=0; n<N; n++) X[n]= a*X[n]; }
C_**sCOPY1**	void C_**sCOPY1** (int N,float *Y,float *X) { int n; for (n=0; n<N; n++) Y[n]=X[n]; }
C_**sAXPY1**	void C_**sAXPY1**(int N, float *Y,float *X,float *A) { int n; float a=*A; for (n=0; n<N; n++) Y[n]=Y[n]+a*X[n]; }

Для сравнительной характеристики различных вариантов их реализаций, рассчитанных на применение обычного сопроцессора (CP1) вещественных чисел плавающей арифметики или векторного сопроцессора (CPV), представим схематично основные действия (команды), которые должны исполняться ими в теле цикла на очередном его шаге для обработки одного текущего элемента или сразу целой группы из нескольких (K) элементов массива:

	C_**sSCAL1** **K**=1	cp1_**sSCAL1** **K**=2
Load	lwc1 FX,0(X)	lwc1 FX,0(X); lwc1 FZ,4(X)
Oper	mul.s FX,FX,FA	mul.s FX,FX,FA;
Store	swc1 FX,0(X)	swc1 FX,0(X); swc1 FZ,4(X);

	cpv_**sSCAL1** **K**=16
Load	vldm X1,0(X); vldm X2,16(X); vldm X3,32(X); vldm X4,48(X);
Oper	vmul.s X1,X1,VA; vmul.s X2,X2,VA vmul.s X3,X3,VA; vmul.s X4,X4,VA
Store	vsdm X1,0(X); vsdm X2,16(X); vsdm X3,32(X); vsdm X4,48(X);

	C_**sCOPY1** **K**=1	cp1_**sCOPY1** **K**=2
Load	lwc1 FX,0(X)	lwc1 FX,0(X); lwc1 FZ,4(X)
Store	swc1 FX,0(X)	swc1 FX,0(Y); swc1 FZ,4(Y);

	cpv_**sCOPY1** **K**=16
Load	vldm X1,0(X); vldm X2,16(X); vldm X3,32(X); vldm X4,48(X);
Store	vsdm X1,0(Y); vsdm X2,16(Y); vsdm X3,32(Y); vsdm X4,48(Y);

	C_**sAXPY1** **K**=1	cp1_**sAXPY1** **K**=2
Load	lwc1 FX,0(X) lwc1 FY,0(Y)	lwc1 FX,0(X); lwc1 FY,0(Y); lwc1 FZ,4(X); lwc1 FV,4(Y)
Oper	madd.s FY,FY,FX,FA	madd.s FY,FY,FX,FA madd.s FV,FV,FZ,FA
Store	swc1 FY,0(Y)	swc1 FY,0(Y); swc1 FV,4(Y)

	cpv_**sAXPY1** **K**=16
Load	vldm X1,0(X); vldm Y1,0(Y); vldm X2,16(X); vldm Y2,16(Y); vldm X3,32(X); vldm Y3,32(Y); vldm X4,48(X); vldm Y4,48(Y)
Oper	vmadd.s Y1,X1,VA; vmadd.s Y2,X2,VA; vmadd.s Y3,X3,VA; vmadd.s Y4,X4,VA;
Store	vsdm Y1,0(Y); vsdm Y2,16(Y); vsdm Y3,32(Y); vsdm Y4,48(Y);

где: **K** – количество элементов, обрабатываемых на очередном шаге цикла;
Load – блок команд чтения (загрузки) значений элементов массива из памяти в регистр(ы);
Store – блок команд записи (выгрузки) значений элементов из регистра(ов) в память;
Oper – блок команд вычислительной обработки значений элементов массива.

Теперь сравним по производительности различные варианты реализации одной и той

же функции. В таблицах 2-4 представлены результаты измерений длительности (в тактах) исполнения каждого варианта функции на микропроцессоре BM8. Измерения производились особой тестовой программой, которая запускала каждый вариант функции 2 раза (для тех же векторов той же длины) на микропроцессоре BM8 с частотой ядра 400 МГц и частотой памяти 400 МГц. Первый запуск предназначался, чтобы гарантировать попадание подвергаемых обработке данных и испытываемого кода команд в кэш, и только на втором запуске проводились требуемые замеры длительностей исполнения функции.

Таблица 2. Сравнение вариантов реализации функции sSCAL1 по производительности

Длина	C	CP1	CPV	C/ CP1	CP1/ CPV	C/ CPV
32	321	204	84	1.574	2.429	3.821
64	603	364	120	1.657	3.033	5.025
|128	1179	678	191	1.739	3.550	6.173
256	2331	1318	319	1.769	4.132	7.307
512	4635	2598	591	1.784	4.396	7.843
1024	**9243**	**5158**	**1138**	**1.792**	**4.533**	**8.122**
2048	**18480**	**10300**	**2248**	**1.794**	**4.582**	**8.221**
4096	38421	22055	6453	1.742	3.418	5.954
8192	38421	44077	12853	1.743	3.429	5.978
16384	153633	88109	25653	1.744	3.435	5.989
32768	307233	176173	51253	1.744	3.437	5.994

Таблица 3. Сравнение вариантов реализации функции sCOPY1 по производительности

Длина	C	CP1	CPV	C/ CP1	CP1/ CPV	C/ CPV
32	194	147	63	1.320	2.333	3.079
64	348	259	89	1.344	2.910	3.910
|128	668	483	141	1.383	3.426	4.738
256	1308	931	250	1.405	3.724	5.232
512	2752	1827	591	1.506	3.989	6.009
1024	**5148**	**3619**	**869**	**1.422**	**4.165**	**5.924**
2048	10550	7751	2197	1.361	3.528	4.802
4096	22850	18129	8613	1.260	2.105	2.653
8192	46116	36909	18988	1.249	1.944	2.429
16384	92196	73773	37932	1.250	1.945	2.431
32768	184356	147501	75820	1.250	1.945	2.431

Таблица 4. Сравнение вариантов реализации функции sAXPY1 по производительности

Длина	C	CP1	CPV	C/ CP1	CP1/ CPV	C / CPV
32	420	299	92	1.405	3.250	4.565
64	804	547	130	1.470	4.208	6.185
|128	1565	1059	210	1.478	5.043	7.452
256	3101	2083	375	1.489	5.555	8.269
512	6173	4131	690	1.494	5.987	8.946
1024	**12317**	**8227**	**1330**	**1.497**	**6.186**	**9.261**
2048	24869	16955	3186	1.467	5.322	7.806
4096	51927	35887	8682	1.447	4.133	5.981
8192	104487	71727	17471	1.457	4.105	5.981
16384	208935	143407	34879	1.457	4.112	5.990
32768	417831	286767	69695	1.457	4.115	5.995

Анализируя результаты тестовых прогонов, представленные в этих таблицах, можно заметить, что применение CPV обеспечивает значительное ускорение обработки (от 2,5 до 6 раз) даже по отношению к оптимизированному ассемблерному варианту, применяющему команды CP1. А максимальный коэффициент ускорения достигается при обработке векторов длиной **N=1024 (4.165** для **sCOPY1, 6.186** для **sAXPY1**) и **N=2048** (**4.582** для **sSCAL1**). Это объясняется тем, что именно такой длины вектора целиком умещаются в кэш-память данных 1-го уровня (L1-кэш).

Как видим, на CPV удаётся достичь даже более чем 4-х кратного ускорения обработки векторов. Такой дополнительный выигрыш в производительности обеспечивается за счёт возможности взятия на исполнение в каждом такте сразу двух команд разных потоков: например, одной вычислительной команды и одной команды доступа к памяти.

Но для достижения такой максимальной производительности вычислительные команды CPV (vmul.s,vmadd.s), а также команды чтения (vldm) и записи (vsdm) необходимо применять не поодиночке, а целой группой (минимум по 4), чтобы не возникало никаких лишних задержек при их исполнении. Такая необходимость обусловлена тем, что каждая из этих команд исполняется не за один, а за несколько тактов. Но в течение их исполнения процессор может выбирать на исполнение следующие команды. А задержка будет происходить, если последует команда, требующая результаты исполнения одной из уже взятых команд. Так функционирует конвейер команд.

Команды vmul.s и vmadd.s исполняются за 4 такта, а командам vldm и vsdm требуется как минимум 3 такта, если данные уже представлены в L1-кэше, а то и больше, если требуется их подкачка из оперативной памяти. Поэтому для эффективной реализации на CPV (чтобы минимизировать задержки) обработка элементов массива осуществляется группой по 16 элементов (4 командами vldm загружаются 16 элементов в 4 регистра CPV). Вследствие

чего для такого варианта реализации и требуется соответственно, чтобы длина обрабатываемого массива была кратна 16 (см. 1-ое особое требование).

Чтобы команды vldm и vsdm могли корректно читать элементы данных из массива и записывать их обратно, задаваемые в этих командах адреса должны быть кратны 16. Это значит, что начало массива (и начало каждой очередной четвёрки его элементов) должно располагаться в памяти по адресу, кратному 16 (в кодовом значении такого адреса младшие 4 бита должны быть нулевыми), иначе говоря, такие массивы должны быть выровнены в памяти по границе 16-байтного блока.

При аппаратном исполнении этих команд доступа к памяти 4 младших бита адреса игнорируются (они просто полагаются нулевыми). Вот почему корректная обработка на CPV массивов, не выровненных по кратному адресу, становится невозможной. Более того, попытка исполнить команду vsdm по не выровненному адресу может вообще привести к непредсказуемой порчи памяти. Этим и объясняется необходимость соблюдения второго особого требования, которое предъявляется к массивам, обрабатываемым командами CPV.

2. Возможные пути решения проблемы выравнивания

Как же всё-таки применить CPV для ускорения обработки любых массивов? Для этого есть два пути. Первый путь (весьма трудоёмкий): обеспечить такую модификацию программного (а возможно и аппаратного) обеспечения, чтобы обработке на CPV могли подвергаться любые массивы, а не только те, которые расположены в памяти по правильно выровненной кратной границе. Второй путь (более простой): гарантировано обеспечить расположение обрабатываемых на CPV массивов в памяти по выровненным адресам. Но можно ли этого всегда добиться в программе на языке Си? И если да, то как?

Будем полагать, что при простом объявлении массива величин типа float Си-компилятор отведёт для его размещения участок памяти, начиная с адреса, кратного 4. (При адресации с точностью до байта одна величина типа float занимает 4 байта). Но специальной директивой (см. aligned) можно дать указание компилятору начинать размещение в памяти массива, начиная с адреса, кратного указанному (см. aligned(16)):

```
float dataX [N]
__attribute__ (( aligned(16) ));
```

Сложнее обеспечить это при динамическом выделении памяти под массив. Для этого вместо стандартной функции malloc для выделения памяти надо воспользоваться особой функцией (см. **our_malloc**), которая выделяет блок памяти с запасом, а потом назначает в нём адрес размещения массива такой, какой соответствует требуемой кратности:

```
#include <stdint.h>
#define MIN_ALNMNT 16
/* кратность адреса, д.быть степенью двойки */
static void *our_malloc(size_t n){
void *p0, *p;
if (!(p0=malloc(n + MIN_ALNMNT)))
  return (void *) 0;
p=(void*)(((uintptr_t)p0+MIN_ALNMNT)
   &(~((uintptr_t)(MIN_ALNMNT-1))));
*((void **) p - 1) = p0;
return p;
}
```

Для освобождения же этой памяти надо тоже использовать свою особую процедуру:

```
static void our_free(void *p)
{ if(p) free(*((void **)p - 1)); }
```

Аналогичные процедуры придётся предусмотреть и для размещения локальных данных в стеке при выполнении программы (а это уже зона ответственности компилятора). Но даже если всё это предусмотреть, окончательно решить возникшую проблему выравнивания всё равно не удастся. Почему? А потому, что может потребоваться обрабатывать на CPV не весь массив, а какую-то его часть или обрабатывать массив, который оказался частью какой-то более сложной структуры. И при этом может опять-таки оказаться, что требуемый для обработки участок массива начинается с не выровненного правильным образом адреса. И что же тогда делать? Отказываться совсем от его высокопроизводительной обработки на CPV и переходить на его «медленную» обработку на CP1?

Значит, придётся всё же решать проблему выравнивания, продвигаясь по первому обозначенному пути. И прежде всего, попробуем проанализировать, можно ли решить эту проблему чисто программным способом, т.е. путём лишь усовершенствования программного обеспечения, применяемого для обработки массивов на CPV.

3. Обработка на CPV одиночного не выровненного массива

Попытаемся всё же найти возможность ускоренной обработки на CPV массивов даже для тех случаев, когда они располагаются в памяти, начиная с «неправильного», не выровненного должным образом адреса, т.е. с адреса, не отвечающего требуемой кратности (/16). Но поскольку нам приходится обрабатывать массивы вещественных чисел типа float, то будем полагать, что заданный адрес начала массива выровнен, по крайней мере, на границу 4-х байтовой величины. (Если это всё же не так, то все используемые далее приёмы надо будет применять уже не для обработки 4-х байтовых величин типа float, а для обработки величин размером в 2-х байтовое полуслово или даже в один байт).

Начнём с обработки одиночного массива. И в качестве модельного примера возьмём задачу умножения вектора на скаляр (см. функцию **sSCAL1**). Сначала разберёмся, как обрабатывать на CPV массивы произвольной длины, а не только требуемой кратности.

Допустим, длина заданного массива N не кратна 16 (как это требуется для прямого применения функции `cpv_sSCAL1`). В таком случае условно представим массив состоящим из двух частей. Первую часть массива длиной m (где m=16*p - максимальное целое, кратное 16-ти, не превышающее N) будем обрабатывать на CPV (вызывая функцию `cpv_sSCAL1`), а вторую его часть (так называемый «хвост») будем обрабатывать на CP1:

```
void Ccpv_sSCAL1
(int N, float *A, float *X ){
 int i,m; float a= *A;
 m= N&(~0xF);   /*  m= 16*p <= N  */
/* обработка 1-ой части массива на CPV*/
 cpv_sSCAL1(m,A,X);
/* обработка 2-й части массива (хвоста) на CP1 */
 for (i=m; i<N; i++) X[i]=a*X[i] ;
}//Ccpv_sSCAL1
```

Теперь представим, что массив может начинаться с не выровненного адреса. В этом случае выделим ту его начальную часть (длиной k), которая расположена от начала массива и до того элемента, который располагается на границе с кратным адресом (/16). И сначала обработаем эту часть элементов на CP1. А далее, начиная с элемента, расположенного по кратному адресу, проведём обработку оставшихся (N-k) элементов массива:

```
void Ccpv_sSCAL1
(int N, float *A, float *X ){
 int i,m,s,k; float a= *A;
/*  s=0..3  смещение от кратной границы  */
 s=((int)(X)>>2)&0x3;
 if (s>0) { k=4-s;
/* обработка начальной 0-й части массива на CP1*/
   for (i=0; i<k; i++) X[i]=a*X[i];
 } else k=0;
 m= (N-k)&(~0xF); /*  m= 16*p <= N  */
/* обработка 1-ой части массива на CPV*/
 cpv_sSCAL1(m,A,&X[k]);
/* обработка 2-й части массива (хвоста) на CP1 */
 for (i=m+k; i<N; i++) X[i]=a*X[i];
}//Ccpv_sSCAL1
```

После такой модификации функция (**Ccpv_sSCAL1**) будет обеспечивать обработку любого массива, даже если он не выровнен правильно по адресу, кратному 16 (но в предположении, что его адрес всё же кратен 4). При этом большая часть массива будет обрабатываться командами CPV, что обеспечит в итоге общую высокую производительность такой обработки. И проведённые тестирования модифицированной функции это в целом подтверждают (см. таблицу 5).

Вспомним, что прежний вариант функции (`cpv_sSCAL1`) для вектора длиной 2048 обеспечивал коэффициент ускорения (по сравнению с CP1) **4.582** (см. таблицу 2). А модифицированная нами функция (`Ccpv_sSCAL1`), которая умеет обрабатывать в том числе и не выровненные массивы, обеспечивает на длинах от 2048 до 2080 коэффициент ускорения в диапазоне от **4.215** до **4.519,** т.е. с незначительной потерей производительности (в худшем случае всего лишь на 9%).

Таблица 5. Сравнительная оценка производительности новой функции **sSCAL1**

Длина	s	C	CP1	**CPV**	C / CP1	**CP1/ CPV**	C / CPV
2048	**0**	**18478**	**10361**	**2293**	**1.783**	**4.519**	**8.058**
2049	**1**	**18485**	**10349**	**2455**	**1.786**	**4.215**	**7.530**
2050	2	18494	10365	2324	1.784	4.460	7.958
2051	3	18503	10359	2332	1.786	4.442	7.934
2052	0	18512	10370	2345	1.785	4.422	7.894
2053	1	18521	10369	2355	1.786	4.403	7.865
2054	2	18530	10374	2364	1.786	4.388	7.838
2055	3	18542	10382	2369	1.786	4.382	7.827
2056	0	18548	10384	2381	1.786	4.361	7.790
2057	1	18560	10392	2395	1.786	4.339	7.749
2058	2	18569	10394	2404	1.787	4.324	7.724
2059	3	18578	10402	2405	1.786	4.325	7.725
2060	0	18587	10404	2421	1.787	4.297	7.677

2061	1	18596	10412	2431	1.786	4.283	7.650
2062	2	18605	10414	2440	1.787	4.268	7.625
2063	3	18617	10425	2443	1.786	4.267	7.621
2064	0	18623	10424	2332	1.787	4.470	7.986
2065	**1**	**18635**	**10435**	**2473**	**1.786**	**4.220**	**7.535**
2066	2	18644	10437	2354	1.786	4.434	7.920
2067	3	18653	10445	2356	1.786	4.433	7.917
2068	0	18662	10447	2374	1.786	4.401	7.861
2069	1	18671	10455	2382	1.786	4.389	7.838
2070	2	18680	10457	2391	1.786	4.373	7.813
2071	3	18692	10468	2396	1.786	4.369	7.801
2072	0	18698	10467	2413	1.786	4.338	7.749
2073	1	18710	10478	2426	1.786	4.319	7.712
2074	2	18719	10480	2435	1.786	4.304	7.687
2075	3	18728	10488	2436	1.786	4.305	7.688
2076	0	18737	10490	2457	1.786	4.269	7.626
2077	1	18746	10498	2462	1.786	4.264	7.614
2078	2	18755	10500	2471	1.786	4.249	7.590
2079	3	18767	10511	2474	1.785	4.249	7.586
2080	0	**18773**	**10510**	**2365**	**1.786**	**4.444**	**7.938**

4. Приёмы обработки на CPV двух одинаково не выровненных массивов

Если в обработке участвуют не один, а два массива (**X** и **Y**), и каждый из них может быть не выровнен по кратному адресу, то придётся предусматривать различные варианты их обработки в зависимости от разных вариантов смещений (**sX**,**sY**) адресов этих массивов относительно кратной границы. Но в самом благоприятном случае, когда такие смещения у них одинаковы (**sX**=**sY**), обработку можно обеспечить одной функцией, используя такие же приёмы разделения массива(ов) на три части, которые были только что рассмотрены нами на примере обработки одиночного массива для функции **Ccpv_sSCAL1**.

Продемонстрируем для этих случаев аналогичные функции, обеспечивающие решение задач копирования векторов (Ccpv_**sCOPY1**) и добавления к вектору другого вектора, умноженного на скаляр (Ccpv_**sAXPY1**) в том числе и для тех векторов, которые могут оказаться не выровнены по кратному адресу (но с одинаковыми смещения от кратной границы):

```
void Ccpv_sCOPY1
(int N, float *Y, float *X ) {
 int i,m,sX,sY,k;
 sX=((int)(X)>>2)&0x3; /* sX= 0..3 */
 sY=((int)(Y)>>2)&0x3; /* sY= 0..3 */
/* предполагается, что sX=sY  !!! */
 if (sY>0) { k=4-sY;
/* обработка начальной 0-й части массивов на CP1*/
   for (i=0; i<k; i++) Y[i]=X[i];
 }else k=0;
   m= (N-k)&(~0xF);   /* m= 16*p <= N */
/* обработка 1-й части массивов на CPV*/
 cpv_sCOPY1(m,&Y[k],&X[k]);
/* обработка 2-й части массивов (хвостов) на CP1 */
 for (i=m+k; i<N; i++) Y[i]=X[i] ;
}//Ccpv_sCOPY1
```

```
void Ccpv_sAXPY1
(int N,float *Y,float *X,float *A) {
 int i,m,sX,sY,k; float a= *A;
 sX=((int)(X)>>2)&0x3; /* sX= 0..3 */
 sY=((int)(Y)>>2)&0x3; /* sY= 0..3 */
/* предполагается, что sX=sY  !!! */
 if (sY>0) { k=4-sY;
/*обработка начальной 0-й части массивов на CP1*/
   for (i=0; i<k; i++) Y[i]=Y+a*X[i];
 }else k=0;
 m= (N-k)&(~0xF);   /* m= 16*p <= N */
/*обработка 1-й части массивов на CPV*/
 cpv_sAXPY1(m,&Y[k],&X[k],A);
/*обработка 2-й части массивов (хвостов) на CP1*/
 for (i=m+k; i<N; i++) Y[i]=Y+a*X[i];
}//Ccpv_sAXPY1
```

5. Приёмы обработки на CPV двух по-разному не выровненных массивов

Наконец, рассмотрим самые сложные случаи обработки двух массивов, когда у каждого из них адрес смещён на ненулевое расстояние от кратной границы, причём так, что эти смещения у них разные (**sX≠sY**).

Если измерять эти смещения количеством элементов, на которые «подвинулся» массив относительно кратной границы, то каждое из этих смещений (**sX**,**sY**) может принимать 4 значения (0,1,2,3), а, значит, всего возможно 16 вариантов их комбинаций. Но эти 16 комбинаций можно разбить на 4 «родственные» группы в зависимости от того, как они соотносятся между собой, так чтобы в одну группу попадали те варианты, в которых разница между этими смещениями одинакова. Пусть величина этой разницы (pos) определяет, насколько «сдвинут» текущий обрабатываемый элемент вектора **X** относительно соответствующего ему элемента вектора **Y**: pos= sX-sY. А чтобы «родственные» разницы свести в одну группу, будем вычислять величину pos по модулю 4 так, чтобы она принимала только положительные значения от 0 до 3:

```
pos= sX-sY; if (pos<0) pos=pos+4;
```

Для каждой разрабатываемой функции (**sCOPY1** и **sAXPY1**) в качестве стартового примем за основу тот вариант алгоритма, который был рассмотрен нами в п.4 для случаев, когда оба вектора X и Y имеют одинаковые смещения адресов. Поскольку новые результирующие значения элементов будут записываться в массив Y (а для функции **sAXPY1** из массива Y будут ещё и читаться прежние значения), то будет разумно разделить всю обработку массивов на три части так, чтобы основная часть алгоритма (на CPV) могла исполняться в предположении, что задаваемый ей массив Y уже выровнен по кратному адресу (/16).

Тогда при исполнении вызванной в основной части функции (cpv_**sCOPY1** или cpv_**sAXPY1**) для чтения очередной группы значений 16-ти элементов массива Y в регистры CPV Y1,..,Y4 и записи их новых значений обратно в массив можно оставить прежний состав команд (см. столбцы Load и Store для этих функций в п.1). А вот для загрузки в регистры CPV X1,..,X4 очередной группы значений 16-ти элементов массива X следует предусмотреть различные последовательности команд в зависимости от того, по какому варианту из 4-х возможных (определяемому значением pos) предстоит проводить заказанную обработку массивов.

Это значит, что для исполнения основной части обработки массивов на CPV придётся подготовить 4 разных варианта каждой функции (cpv_**sCOPY1**_p и cpv_**sAXPY1**_p) для p=0,1,2,3. Первые варианты функций для значения pos=0 уже готовы: это те функции, которые были созданы (и рассмотрены нами в п.1) в предположении, что задаваемые им массивы X,Y располагаются по выровненным (кратным 16) адресам в памяти. (Будем просто считать, что cpv_**sCOPY1**_0=cpv_**sCOPY1** и cpv_**sAXPY1**_0=cpv_**sAXPY1**).

Выясним, какие действия следует предусмотреть для других вариантов функций в блоке загрузки Load тела основного цикла, чтобы значения очередных 16-ти элементов массива X могли быть корректно прочитаны из памяти и загружены в регистры X1,..,X4 векторного сопроцессора CPV. И наглядно продемонстрируем их на примерах разработки 3-х вариантов блоков Load для функций cpv_**sCOPY1**_p (p=1,2,3).

5.1 Вариант CPV-функции для pos=2

Чтобы при pos=2 корректно прочитывать из массива, задаваемым адресом X, очередную четвёрку его элементов (начиная с j-го), необходимо использовать две команды vldm, загружая такую четвёрку по частям. При этом в 1-ой команде vldm надо задавать адрес элемента с индексом j-2 (&X[j-2]), а во второй − адрес элемента с индексом j+2 (&X[j+2]). Именно такие адреса в этом случае будут правильно выровненным (см. рис.1).

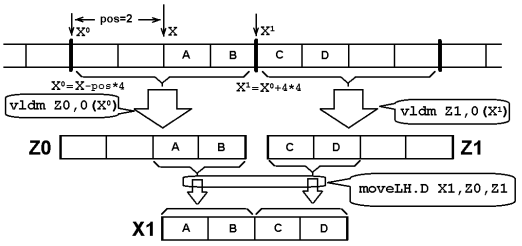


Рис. 1. Схема загрузки очередной четвёрки элементов массива в регистр CPV при pos=2.

После выполнения первой такой команды:

```
vldm Z0,(j-2)*4(X)
```

в регистр Z0 загружается четвёрка элементов:

(X[j-2],X[j-1],X[j],X[j+1]),

а после аналогичной команды:

```
vldm Z1,(j+2)*4(X)
```

в регистр Z1 будет загружена следующая за ними четвёрка элементов:

(X[j+2],X[j+3],X[j+4],X[j+5]).

Теперь с помощью команды CPV:

```
moveLH.D X1,Z0,Z1
```

можно осуществить слияние младшей половинки регистра Z0 и старшей половинки регистра Z1 так, чтобы сформировать нужную нам четвёрку элементов:

(X[j],X[j+1],X[j+2],X[j+3])

в заданном регистре X1.

Применим рассмотренный приём для загрузки очередных 16-ти элементов (4-х четверок) массива X в регистры CPV X1,..,X4 в блоке действий Load тела основного цикла при разработке CPV-функции cpv_**sCOPY1**_2 :

перед циклом	`X=X-2*4; vldm Z0,0(X);`
блок **Load** в теле цикла	`vldm Z1,16(X); vldm Z2,32(X);` `vldm Z3,48(X); vldm Z4,64(X);` **/*{Y!}*/** `moveLH.D X1,Z0,Z1;` `moveLH.D X2,Z1,Z2;` `moveLH.D X3,Z2,Z3;` `moveLH.D X4,Z3,Z4;` `VSWAP.S Z0,Z4,0 /* copy Z4=>Z0 */`
в конце тела цикла	`X=X+16*4; Y=Y+16*4;` `/* продвижение указателей` `на следующую группу элементов*/`

Заметим, что при переходе на следующий шаг цикла для обработки очередной группы из 16-ти элементов нужное в регистре Z0 значение можно просто скопировать из регистра Z4.

5.2 Вариант CPV-функции для pos=1

При варианте pos=1 для корректного чтения из массива, определяемым адресом X, очередной четвёрки его элементов (начиная с j-го), необходимо задавать в командах vldm адреса элементов с индексами j-1 (&X[j-1]) и j+3 (&X[j+3]) соответственно (см. рис.2).

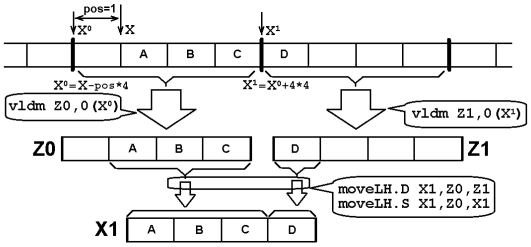


Рис. 2. Схема загрузки очередной четвёрки элементов массива в регистр CPV при pos=1**.**

Аналогичным приёмом (рассмотренным в п.5.1) загрузим сначала двумя командами:

`vldm Z0,(j-1)*4(X)`

и `vldm Z1,(j+3)*4(X)`

две стоящие рядом четвёрки элементов:

`(X[j-1],X[j],X[j+1],X[j+2])` и

`(X[j+3],X[j+4],X[j+5],X[j+6])`

в регистры `Z0` и `Z1` соответственно.

Теперь необходимо составить четвёрку значений в регистре `X1` из 3-х младших значений регистра `Z0` и одного старшего значения регистра `Z1`. Заметим, что одной командой CPV такое действие выполнить сразу же не удаётся. Однако, это можно осуществить, выполнив последовательно две команды `moveLH`. Сначала командой:

`moveLH.D X1,Z0,Z1`

можно записать (причём можно в тот же регистр `X1`) промежуточное значение:

`(X[j+1],X[j+2],X[j+3],X[j+4])`,

а затем другой командой:

`moveLH.S X1,Z0,X1`

сформировать в регистре `X1` требуемое окончательное значение:

`(X[j],X[j+1],X[j+2],X[j+3])`.

Применив рассмотренный приём для чтения массива X и формирования в регистрах CPV X1,..,X4 очередных 16-ти элементов (4-х четверок) в блоке действий Load тела основного цикла, получим вариант CPV-функции `cpv_sCOPY1_1`:

	`cpv_sCOPY1_1 (K=16, pos=1)`
перед циклом	`X=X-1*4; vldm Z0,0(X);`
блок **Load** в теле цикла	`vldm Z1,16(X); vldm Z2,32(X);` `vldm Z3,48(X); vldm Z4,64(X);` `/*{Y!}*/` `moveLH.D X1,Z0,Z1;` `moveLH.D X2,Z1,Z2;` `moveLH.D X3,Z2,Z3;` `moveLH.D X4,Z3,Z4;` **`moveLH.S X1,Z0,X1;`** **`moveLH.S X2,Z1,X2;`** **`moveLH.S X3,Z2,X3;`** **`moveLH.S X4,Z3,X4;`** `VSWAP.S Z0,Z4,0 /* copy Z4=>Z0 */`
в конце тела цикла	`X=X+16*4; Y=Y+16*4;` `/* продвижение указателей на следующую группу элементов */`

5.3 Вариант CPV-функции для pos=3

При разработке CPV-функций для варианта pos=3 для чтения очередной четвёрки элементов (начиная с j-го), задаём в командах vldm адреса элементов с индексами j-3 и j+1 (&X[j-3] и &X[j+1]) соответственно (см. рис.3).

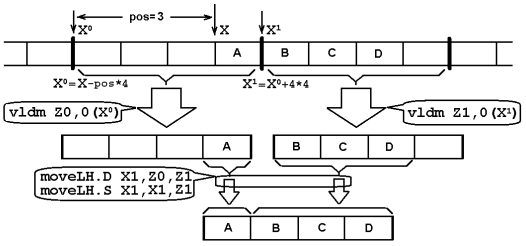


Рис. 3. Схема загрузки очередной четвёрки элементов массива в регистр CPV при pos=3**.**

И сначала тем же способом (рассмотренным в п.5.1) загружаем сначала командами:

`vldm Z0,(j-3)*4(X)` и

`vldm Z1,(j+1)*4(X)`

в регистры `Z0` и `Z1` четвёрки:

`(X[j-3],X[j-2],X[j-1],X[j])` и

`(X[j+1],X[j+2],X[j+3],X[j+4])`.

А далее выполняем последовательно тоже две команды `moveLH`. Сначала командой:

`moveLH.D X1,Z0,Z1`

записываем значение:

`(X[j-1],X[j],X[j+1],X[j+2])`

в регистр `X1`, а затем другой командой:

`moveLH.S X1,X1,Z1`

формируем в регистре X1 окончательное требуемое значение:

(X[j],X[j+1],X[j+2],X[j+3]).

Применяя такой приём для загрузки всех 4-х регистров CPV X1,..,X4 очередными 16-тью элементами, получаем соответственно вот такой вариант блока действий Load тела основного цикла для CPV-функции cpv_**sCOPY1**_3 :

	cpv_**sCOPY1**_3 (**K=16, pos=3**)
перед циклом	X=X-**3***4; vldm Z0,0(X);
блок **Load** в теле цикла	vldm Z1,16(X); vldm Z2,32(X); vldm Z3,48(X); vldm Z4,64(X); **/*{Y!}*/** moveLH.D X1,Z0,Z1; moveLH.D X2,Z1,Z2; moveLH.D X3,Z2,Z3; moveLH.D X4,Z3,Z4; **moveLH.S X1,X1,Z1;** **moveLH.S X2,X2,Z2;** **moveLH.S X3,X3,Z3;** **moveLH.S X4,X4,Z4;** VSWAP.S Z0,Z4,0 /*copy Z4=>Z0 */
в конце тела цикла	X=X+16*4; Y=Y+16*4; /* продвижение указателей на следующую группу элементов*/

Поскольку команды CPV moveLH относятся к вычислительному потоку, их можно «переплести» (т.е. разместить с чередованием) с командами чтения vldm следующей группы элементов в регистры Z1,..,Z4 так, чтобы процессор мог на каждом такте запускать исполнение сразу двух команд разных потоков. Выполнив такую оптимизацию, получим более производительные варианты реализации функций cpv_**sCOPY1**_p для p=1,2,3.

5.4 Итоговая версия основной функции копирования векторов

Конечно, из всех разнообразных вариантов cpv_**sCOPY1**_p (p=0,1,2,3) функции копирования векторов, приготовленных для исполнения на CPV, из основной части функции (Ccpv_**sCOPY1**) должен вызываться только тот, которому соответствует значение разницы (pos), образовавшейся между смещениями (**sX**,**sY**) адресов заданных массивов X,Y. Чтобы вызывать нужную функцию cpv_**sCOPY1**_p по значению p=pos, в Си-программе совсем не обязательно устраивать множественные ветвления или применять оператор switch. Гораздо эффективней применить в таком случае вызов требуемой функции из таблицы:

```
typedef int (*sCOPY1_func)
        (int N, float *Y, float *X );
/* Table of CPV-functions: */
static sCOPY1_func cpv_sCOPY1_fT[] =
{ cpv_sCOPY1_0, cpv_sCOPY1_1,
  cpv_sCOPY1_2, cpv_sCOPY1_3 };
```

В результате получится вот такая итоговая версия реализации основной функции Ccpv_**sCOPY1**:

```
/* One of CPV-functions, called in Ccpv_sCOPY1*/
static sCOPY1_func cpv_sCOPY1_fv;
void Ccpv_sCOPY1
(int N,float *Y,float *X){
 int i,m,sX,sY,k,pos;
 sX=((int)(X)>>2)&0x3; /* sX= 0..3 */
 sY=((int)(Y)>>2)&0x3; /* sY= 0..3 */
 pos= sX-sY; if (pos<0) pos=pos+4;
 cpv_sCOPY1_fv= cpv_sCOPY1_fT[pos];
 if(sY>0) {  k=4-sY;
  for(i=0;i<k;i++) Y[i]=X[i];
 }else k=0;
 m=(N-k)&(~0xF); /*  m= 16*p <= N  */
/*обработка основной части массивов на CPV*/
 cpv_ sCOPY1_fv(m,&Y[k],&X[k]);
/*обработка оставшейся части массивов на CP1*/
 for (i=m+k; i<N; i++) Y[i]=X[i];
}//Ccpv_sCOPY1
```

которая способна выполнить задачу копирования векторов даже в том случае, если один из них или оба расположены в памяти по не выровненному адресу.

5.5 Особенности разработки CPV-вариантов и основной функции sAXPY1

При разработке CPV-вариантов функций cpv_**sAXPY1**_p для p=1,2,3 следует модифицировать их блоки загрузки Load в теле основного цикла, применяя для формирования в регистрах CPV X1,.., X4 нужных значений элементов массива X те же приёмы, что были продемонстрированы для соответствующих вариантов функции cpv_**sCOPY1**_p для pos=2,1,3. При этом для блоков Load функций cpv_**sAXPY1**_p можно позаимствовать уже приготовленные блоки Load функций cpv_**sCOPY1**_p, лишь дополнив их командами загрузки регистров CPV Y1,..,Y4 прежними значениями соответствующих элементов массива Y (в точках, отмеченных комментарием **/*{Y!}*/** после загрузки регистров Z1,..,Z4):

```
/*{Y!}*/ vldm Y1,0(Y);  vldm Y2,16(Y);
         vldm Y3,32(Y); vldm Y4,48(Y);
```

После такой модификации в блоках Load окажется 2 группы команд vLdm (по 4 команды в каждой) вместе с одной (для pos=2) или дву-

мя (для pos=1,3) группами из 4-х команд вида moveLH. А всего в теле основного цикла функции cpv_**sAXPY1**_p (вместе с блоком Operation и блоком Store) будет исполняться 13 команд CPV потока работы с памятью (8 команд v**L**dm, одна команда VSWAP и 4 команды v**S**dm), 8 (или 12) команд CPV вычислительного потока (8 или 4 команды mo**V**eLH и 4 команды v**M**add.s), а также 4 команды целочисленного потока (3 команды для подсчёта количества шагов цикла и приращений значений указателей для массивов X,Y и одна команда условного перехода). Команды разных потоков можно «переплести» так, чтобы по две команды могли захватываться на исполнение в одном такте цикла. Проведя такую оптимизацию, получим достаточно эффективные версии CPV-функций cpv_**sAXPY1**_p (для p=1,2,3).

Для вызова же из основной функции Ccpv_**sAXPY1** нужной CPV-функции cpv_**sAXPY1**_p в зависимости от значения pos применим уже рассмотренный выше приём вызова функции по таблице:

```
typedef int (*sAXPY1_func)
(int N,float *Y,float *X,float *A);
/* Table of CPV-functions: */
static sAXPY1_func cpv_sAXPY1_fT[]={
 cpv_sAXPY1_0, cpv_sAXPY1_1,
 cpv_sAXPY1_2, cpv_sAXPY1_3 };
/* One of CPV-functions, called in Ccpv_sAXPY1*/
static sAXPY1_func cpv_sAXPY1_fv;
void Ccpv_sAXPY1
(int N,float *Y,float *X,float *A){
 int i,m,sX,sY,k,pos; float a= *A;
 sX=((int)(X)>>2)&0x3;
 sY=((int)(Y)>>2)&0x3; /* sX,sY= 0..3 */
 pos= sX-sY; if (pos<0) pos=pos+4;
 cpv_sAXPY1_fv= cpv_sAXPY1_fT[pos];
 if (sY>0) { k=4-sY;
  for(i=0;i<k;i++) Y[i]=Y+a*X[i];
 }else k=0;
  m=(N-k)&(~0xF); /* m= 16*p <= N */
/*обработка основной части массивов на CPV*/
 cpv_ sAXPY1_fv(m,&Y[k],&X[k],A);
/*обработка оставшейся части массивов на CP1*/
 for (i=m+k;i<N;i++) Y[i]=Y+a*X[i];
}//Ccpv_sAXPY1
```

5.6 Оценка производительности новых функций sCOPY1 и sAXPY1

Для сравнительной оценки производительностей новых вариантов функций **sCOPY1** и **sAXPY1**, полученных после их модификации для обеспечения обработки ими в том числе и не выровненных правильно массивов, приведём результаты их тестирования, выполненные как и ранее на микропроцессоре BM8 с частотой ядра 400 Мгц и частотой памяти 400 МГц.

Таблица 6. Сравнительная оценка производительности новой функции **sCOPY1**

Длина	sX	sY	C	CP1	**CPV**	C/ CP1	**CP1/ CPV**	C/ CPV
1024	**0**	**0**	**5154**	**3662**	**928**	**1.407**	**3.946**	**5.554**
1025	1	1	5154	3669	1013	1.405	3.622	5.088
1026	2	2	5159	3658	943	1.410	3.879	5.471
1027	3	3	5166	3673	953	1.406	3.854	5.421
1028	1	0	5335	3681	1428	1.449	2.578	3.736
1029	2	1	5174	3676	1420	1.408	2.589	3.644
1030	3	2	5336	3688	1425	1.447	2.588	3.745
1031	**0**	**3**	**5342**	**3683**	**1430**	**1.450**	**2.576**	**3.736**
1032	2	0	5453	3682	1068	1.481	3.448	5.106
1033	3	1	5194	3697	1057	1.405	3.498	4.914
1034	0	2	5348	3686	1062	1.451	3.471	5.036
1035	**1**	**3**	**5246**	**3698**	**1085**	**1.419**	**3.408**	**4.835**
1036	3	0	5251	3706	1453	1.417	2.551	3.614
1037	0	1	5245	3704	1466	1.416	2.527	3.578
1038	1	2	5412	3710	1465	1.459	2.532	3.694
1039	**2**	**3**	**5536**	**3711**	**1476**	**1.492**	**2.514**	**3.751**
1040	0	0	5277	3710	944	1.422	3.930	5.590
1041	1	1	5236	3719	1023	1.408	3.635	5.118
1042	2	2	5599	3714	959	1.508	3.873	5.838
1043	3	3	5365	3726	966	1.440	3.857	5.554
1044	1	0	5594	3734	1439	1.498	2.595	3.887
1045	2	1	5586	3732	1440	1.497	2.592	3.879
1046	3	2	5563	3738	1445	1.488	2.587	3.850
1047	0	3	5878	3739	1450	1.572	2.579	4.054
1048	2	0	5485	3738	1070	1.467	3.493	5.126
1049	3	1	5563	3747	1071	1.485	3.499	5.194
1050	0	2	5420	3742	1076	1.448	3.478	5.037
1051	1	3	5667	3754	1081	1.510	3.473	5.242
1052	3	0	5742	3762	1473	1.526	2.554	3.898
1053	0	1	5552	3760	1480	1.477	2.541	3.751
1054	1	2	6654	3766	1485	1.767	2.536	4.481
1055	2	3	5801	3767	1490	1.540	2.528	3.893
1056	0	0	**5567**	**3766**	**957**	**1.478**	**3.935**	**5.817**

Прежний вариант функции cpv_**sCOPY1** на векторах длиной 1024 обеспечивал коэффициент ускорения (по сравнению с CP1) **4.165** (см. таблицу 3). А новая функция Ccpv_**sCOPY1**, которая после проведённой модификации умеет обрабатывать в том числе и не выровненные массивы, обеспечивает теперь (см. таблицу 6) на векторах длиной от 1024 до 1056 коэффициент ускорения в диапазоне от **2.514** до **3.946.** При этом для случаев, характеризующихся значением pos=2 (3.408), производительность падает всего лишь на 18%, в то время как для случаев, когда pos=1 (2.576)

или =3 (2.514), потеря производительности оказывается более ощутимой: она падает примерно на 38-40% от максимальной (4.165).

Таблица 7. Сравнительная оценка производительности новой функции **sAXPY1**

Длина	sX	sY	C	CP1	**CPV**	C/ CP1	**CP1/ CPV**	C/ CPV
1024	**0**	**0**	**12323**	**8278**	**1398**	**1.489**	**5.921**	**8.815**
1025	1	1	12330	8286	1588	1.488	5.218	7.764
1026	2	2	12342	8283	1426	1.490	5.809	8.655
1027	3	3	12354	8299	1435	1.489	5.783	8.609
1028	1	0	12366	8314	1455	1.487	5.714	8.499
1029	2	1	12378	8314	1474	1.489	5.640	8.398
1030	3	2	12390	8327	1486	1.488	5.604	8.338
1031	**0**	**3**	**12402**	**8330**	**1497**	**1.489**	**5.564**	**8.285**
1032	2	0	12414	8331	1457	1.490	5.718	8.520
1033	3	1	12426	8347	1459	1.489	5.721	8.517
1034	0	2	12438	8344	1465	1.491	5.696	8.490
1035	**1**	**3**	**12450**	**8363**	**1477**	**1.489**	**5.662**	**8.429**
1036	3	0	12462	8378	1551	1.487	5.402	8.035
1037	0	1	12474	8378	1576	1.489	5.316	7.915
1038	1	2	12486	8391	1576	1.488	5.324	7.923
1039	**2**	**3**	**12606**	**8394**	**1594**	**1.502**	**5.266**	**7.908**
1040	0	0	12646	8395	1418	1.506	5.920	8.918
1041	1	1	12566	8411	1600	1.494	5.257	7.854
1042	2	2	12588	8408	1438	1.497	5.847	8.754
1043	3	3	12546	8427	1452	1.489	5.804	8.640
1044	1	0	12663	8442	1475	1.500	5.723	8.585
1045	2	1	12570	8442	1488	1.489	5.673	8.448
1046	3	2	12582	8455	1500	1.488	5.637	8.388
1047	0	3	12594	8458	1512	1.489	5.594	8.329
1048	2	0	12606	8459	1459	1.490	5.798	8.640
1049	3	1	12618	8475	1472	1.489	5.757	8.572
1050	0	2	12630	8472	1484	1.491	5.709	8.511
1051	1	3	12642	8491	1496	1.489	5.676	8.451
1052	3	0	12654	8506	1571	1.488	5.414	8.055
1053	0	1	12880	8506	1584	1.514	5.370	8.131
1054	1	2	12813	8519	1596	1.504	5.338	8.028
1055	2	3	12900	8522	1614	1.514	5.280	7.993
1056	0	0	**12826**	**8523**	**1438**	**1.505**	**5.927**	**8.919**

Значительно меньшую потерю производительности можно наблюдать у новой версии функции `Ccpv_`**`sAXPY1`**. Прежний вариант функции `cpv_`**`sAXPY1`** на векторах длиной 1024 обеспечивал коэффициент ускорения (по сравнению с CP1) **6.186** (см. таблицу 4). А новая функция `Ccpv_`**`sAXPY1`**, способная после проведённой модификации обрабатывать любые массивы вещественных чисел одинарной точности, в том числе и не выровненные по кратной границе (/16), обеспечивает теперь (см. таблицу 7) на векторах длиной от 1024 до 1056 коэффициент ускорения в диапазоне от **5.266** до **5.927.** Причём для случаев, характеризующихся значением pos=2 (5.662), pos =1 (5.564) и pos =3 (5.266), производительность новой функции **`sAXPY1`** падает всего лишь на 8, 10 и 15% соответственно.

Чем объяснить, что потеря производительности у новой функции **`sAXPY1`** оказывается не такой большой, как у новой функции **`sCOPY1`**? Видимо, всё дело в том, что при модификации CPV-функций `cpv_`**`sAXPY1`**`_p` удаётся в лучшей степени совмещать (в одном такте) вычислительные команды с командами доступа к памяти.

6. Выводы и предложения

Можно было бы, конечно, проанализировать ещё и другие часто используемые функции обработки массивов. Но уже и так становится понятной общая тенденция. С увеличением числа массивов, участвующих в векторной операции, объём кода, который требуется разработать для исполнения такой операции на CPV, будет только резко возрастать. И если для функции **sAXPY**, в которой участвуют 2 массива, пришлось вместо одной реализовать для CPV 4 разные процедуры (на ассемблере), то для решения задачи, в которой будут задаваться 3 массива (даже такой простой как получение третьего вектора суммой двух других Z=X+Y), таких процедур на CPV придётся заготовить уже 16. А если в параметрах операции потребуется 4 массива, то потребуется заготовить аж 64 ассемблерные процедуры, чтобы предусмотреть всевозможные разнообразные соотношения смещений у адресов всех 4-х массивов, участвующих в такой операции.

Поэтому вывод напрашивается такой: да, программным способом можно, конечно, решить проблему выравнивания. И возможная потеря производительности не является такой уж большой (до 40%). Подобные оценки потерь можно встретить и в других аналитических статьях, даже в тех случаях, когда для решения проблемы выравнивания предлагается та и или иная аппаратная поддержка [5].

Однако, такое «чисто программное» решение обозначенной проблемы приводит к совершенно неоправданному разрастанию объёма программного кода, который предстоит для этого создать. И дело не только в том, что при таком подходе к решению проблемы существенно вырастут затраты программистского труда и усложнится

разрабатываемое ими ПО. Ведь в разы вырастет и объём скомпонованного в итоге программного кода, который потребуется хранить в той же оперативной памяти. А при большом объёме используемого кода начнёт «пробуксовывать» кэш команд, и как следствие, падать общая производительность.

Поэтому есть сомнение, разумно ли решать такую проблему выравнивания «чисто программным» путём? Возможно, гораздо эффективнее можно справиться с этой проблемой с помощью соответствующей аппаратной поддержки?

Пожалуй, здесь уместно будет вспомнить, что для решения проблемы выравнивания при обработке 8-байтовых (64-разрядных) величин в системе команд микропроцессора ВМ8 семейства КОМДИВ (который является ветвью MIPS-архитектуры) уже предусмотрены команды для чтения из памяти (LDL, LDR) в целочисленные регистры основного процессора и записи из них в память (SDL, SDR) 8-байтовых блоков данных по произвольным адресам, которые не обязательно должны начинаться на 8-байтовой границе и быть кратными 8. Да, конечно, применение этих команд снижает производительность обработки массивов, но не намного. Тесты показали, что вариант реализации функции копирования sCOPY1 с их применением уступает всего лишь в 2 раза варианту той же функции, применяющей команды чтения/записи LD/SD 2-х величин типа float из памяти в 64-разрядные регистры и обратно.

Так что возможно и для векторного сопроцессора подобные команды загрузки из памяти в регистры CPV и обратно величин типа float могли бы оказаться очень полезными и позволили бы достойным образом решить проблему обработки на CPV не выровненных в памяти массивов. Если не сейчас, то хотя бы в ближайшей перспективе.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований по теме «Разработка архитектуры, системных решений и методов для создания микропроцессорных ядер и коммуникационных средств семейства систем на кристалле двойного назначения» (№0065-2019-0004).

About problems of application vector the coprocessor for processing acceleration the data arrays which are not aligned in memory

A.A.Burtsev

Abstract. The vector coprocessors belonging to the class of processors of SIMD architecture allow to accelerate processing of big data arrays considerably. But for this purpose it is required, as a rule, that such arrays were located in memory at the multiple addresses and were processed by large portions of multiple length. These strict requirements significantly limit possibilities of use of such SIMD coprocessors. To lift the available limits and by that to expand a scope of these coprocessors, it is necessary to solve for them a so-called "problem of alignment", i.e. to improve applied software (and/or hardware) providing so that became possible to process by means of the vector coprocessor also such data arrays which can be located in memory at any address which is not necessarily aligned on border of a 64-, 128-or 256-digit machine word.

Ways of the solution of this problem are also the main subject of discussion in this article. The offered possible methods of processing of not aligned data arrays are followed by demonstration of examples of the programs developed by the author for the vector coprocessor CPV of the VM8 microprocessor of the KOMDIV family.

Keywords: SIMD-architecture, microprocessors of the KOMDIV family, the vector coprocessor, problem of not aligned access to memory.

Литература

1. Микросхема 1890ВМ8Я. URL: http:// https://www.niisi.ru/1890ВМ8Я.pdf (дата обращения: 26.07.2017).

2. 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.

3. 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.

4. MIPS Architecture for Programmers. Volume IV-j: The MIPS64 SIMD Architecture Module. URL: https://s3-eu-west-1.amazonaws.com/downloads-mips/documents/MD00868-1D-MSA64-AFP-01.12.pdf (дата обращения: 14.11.2018).

5. Mauricio Alvarez, Esther Salamí, Alex Ramírez, Mateo Valero. Performance Impact of Misaligned Accesses in SIMD extensions. 2007 IEEE International Symposium on Performance Analysis of Systems & Software. URL: http://www.aes.tu-berlin.de/fileadmin/fg196/publication/old-juurlink/performance_impact_of_misaligned_accesses_in_simd_extensions.pdf. (дата обращения: 14.11.2018).

Метод импорта пространственных данных формата «шейп-файл» в систему БГИСРВ

П.В. Егоров

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: egorov@niisi.ras.ru

Аннотация. В статье описывается метод импорта пространственных данных формата «шейп-файл» в систему БГИСРВ.

Ключевые слова: пространственные данные, пространственный объект, импорт данных, цифровая карта, паспорт цифровой карты, метрика, классификатор, шейп-файл.

Введение

В статье рассматривается метод решения задачи импорта пространственных данных формата «шейп-файл» в геоинформационную систему на основе библиотеки БГИСРВ (далее система БГИСРВ).

ГОСТ Р 52438-2005 дает следующие определения понятиям «импорт данных» и «пространственные данные» [1]. Импорт данных - это прием данных из внешней среды путем их конвертирования для использования в данной геоинформационной системе в ее собственном формате. Пространственные данные – данные о пространственных объектах и их наборах. Пространственный объект - цифровая модель материального или абстрактного объекта реального или виртуального мира с указанием его идентификатора, координатных и атрибутивных данных.

Далее под библиотекой (для языка программирования Си) будет пониматься программа, реализующая набор функций (именованный блок программного кода, который выполняет определенную задачу), предназначенных для использования другими программами. Спецификация набора функций библиотеки будет называться ее прикладным программным интерфейсом.

Библиотека географической информационной системы для операционной системы реального времени (БГИСРВ) была разработана в НИИСИ РАН в 2001 году. Библиотека БГИСРВ предназначена для написания на языке Си прикладных программ, функционирующих под управлением ОС РВ Багет, и выполняющих сбор, хранение, анализ и графическую визуализацию пространственных данных [2]. Библиотека обеспечивает выполнение следующих групп функций: создание и редактирование электронных карт; управление отображением электронных карт; поиск объектов по различным условиям; загрузка векторной карты формата SXF; загрузка растровой карты формата PCX; загрузка морской карты формата S57, выполнение расчетов по карте.

Формат «шейп-файл» (Shapefile) – это формат пространственных данных (далее шейп-данных), разработанный американской компанией ESRI для геоинформационной системы ArcView GIS версии 2. На сегодняшний день существует большое количество программ, в том числе свободно распространяемых, для работы с данными в этом формате. Из-за своей распространенности формат стал де-факто стандартом для обмена данными между геоинформационными системами [3].

Расширение функционала библиотеки БГИСРВ за счет добавления в нее функций импорта шейп-данных позволило пользователям библиотеки получить доступ к пространственным данным не только отечественных, но и зарубежных геоинформационных систем.

Анализ рынка программного обеспечения на предмет наличия в нем сервиса импорта шейп-данных, реализующего, в частности, преобразование данных формата «шейп-файл» в формат данных БГИСРВ, показал, что на данный момент только программа ГИС «Панорама» [4] имеет подобный функционал. Если на основе имеющейся документации сравнить технологии импорта, задействованные в указанной программе и рассматриваемые в данной статье, то можно отметить следующее. В соответствии с документацией

ГИС «Панорама» суть ее технологии импорта заключается в применении прикладной библиотеки, с помощью которой реализуется программа, которая в свою очередь выполняет импорт шейп-данных. В данной статье технология импорта рассматривается шире, помимо описания прикладного программного интерфейса библиотеки, которая в модели обозначена с помощью пакета «Конвертор», и структуры прикладной программы, реализующей импорт данных, особое внимание уделено вопросу проектирования самой библиотеки, а именно дается описание статической семантики пакета «Конвертор» в виде диаграммы классов и динамической - в виде диаграмм деятельности операций пакета.

Особенностью предлагаемого в статье подхода является применение в операциях классов пакета «Конвертор» функций свободно распространяемой библиотеки «Shapefile C Library». Библиотека «Shapefile C Library» предоставляет возможность писать программы на языке программирования C для чтения, записи и обновления ESRI шейп-файлов и связанного с ними файла атрибутов [5]. Такой подход с точки зрения модели данных позволил заменить модель данных спецификации «шейп-файл» на более простую модель данных библиотеки «Shapefile C Library», что в свою очередь привело к уменьшению сложности исходного кода программы и соответственно повышению ее надежности. С точки зрения программиста такой подход к решению задачи импорта данных позволил уменьшить трудоемкость программирования и отладки.

Чтобы сделать описание технологии формализованным, в статье используются словарь и правила языка моделирования UML [6].

При разработке технологии импорта шейп-данных использовались методы прямого и обратного проектирования [6], а также метод сравнительного анализа. Методом обратного проектирования были получены концептуальные модели цифровых карт библиотек «Shapefile C Library» и БГИСРВ. Методом прямого проектирования были получены модели прикладной программы и пакета «Конвертор». Методом сравнительного анализа были сделаны выводы о различиях в моделях цифровых карт библиотек «Shapefile C Library» и БГИСРВ и затем на основе полученных данных предложен метод приведения модели цифровой карты библиотеки «Shapefile C Library» к модели цифровой карты библиотеки БГИСРВ.

2. Модель цифровой карты библиотеки БГИСРВ

Цифровая карта – это цифровая картографическая модель, содержание которой соответствует содержанию карты определенного вида и масштаба [7].

Модель классов цифровой карты библиотеки БГИСРВ показана на рисунке 1. Эта модель отражает только те аспекты цифровой карты, которые имеют значение при решении задачи импорта шейп-данных.

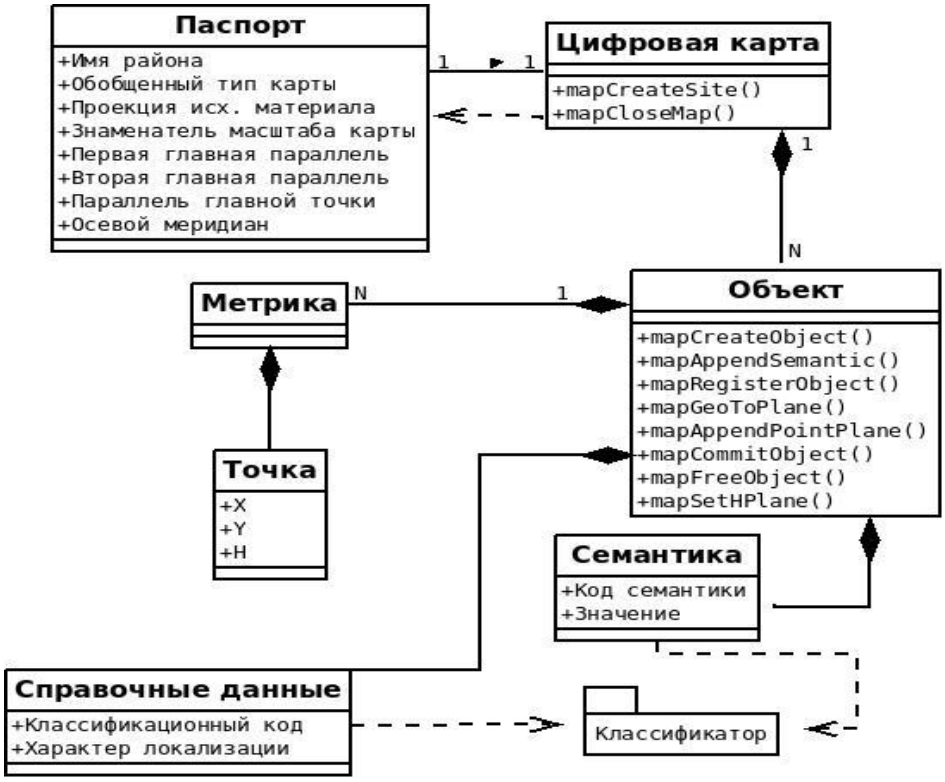


Рис. 1. Диаграмма классов цифровой карты БГИСРВ

На диаграмме цифровая карта представлена как контейнер объектов, на что указывает ассоциация композитного агрегирования между классами «Цифровая карта» и «Объект». В дальнейшем понятие контейнер будет применятся к тем сущностям диаграммы, которые включают в себя другие сущности, и для которых определена ассоциация композитного агрегирования, то есть такая ассоциация, в которой «объект-часть может принадлежать только единственному целому, и, кроме того, как правило, жизненный цикл частей совпадает с циклом целого: они живут и умирают вместе с ним» [8]. Этот термин будет использоваться как для классов, так и для их объектов. Далее в тексте и на диаграммах для обозначения объекта, в частности выполняющего роль контейнера, перед именем класса объекта будет ставиться двоеточие, а само название выделяться подчеркиванием. В случае необходимости именовать объект перед двоеточием будет указываться имя объекта.

У класса «Цифровая карта» имеются две операции – «mapCreateSite» и «mapCloseMap». Операция «mapCreateSite» создает контейнер «:Цифровая карта». Операция «mapCloseMap» сохраняет данные контейнера «:Цифровая карта» в долговременной памяти и освобождает связанные с ним ресурсы.

Класс «Паспорт» отображает сущность «Паспорт цифровой карты», которая является структурной единицей цифровой карты и содержит справочно-технологическую информацию в установленных формате и кодах. [2]. Структурная связь паспорта с цифровой картой на диаграмме показана с помощью ассоциации с множественностью один к одному между классами «Паспорт» и «Цифровая карта». Класс «Паспорт» содержит следующие атрибуты: «Имя района», «Обобщенный тип карты», «Проекция исх. материала», «Знаменатель масштаба карты», «Первая главная параллель», «Вторая главная параллель», «Параллель главной точки» и «Осевой меридиан». Класс «Паспорт» участвует в операции «mapCreateSite» класса «Цифровая карта», что на диаграмме обозначено с помощью отношения зависимости между указанными классами.

Класс «Объект» обозначает пространственные объекты цифровой карты.

Данные пространственного объекта делятся на следующие виды: справочные

данные, метрика и семантика, которые на диаграмме представлены одноименными классами. Класс «Объект» является контейнером для классов «Метрика», «Семантика» и «Справочные данные», что на диаграмме показано с помощью ассоциаций композитного агрегирования между классом «Объект» и перечисленными выше классами.

Справочные данные состоят из классификационного кода объекта (например, 62310000 - мосты) и кода характера локализации. В модели эти элементы данных представлены атрибутами «Код вида объекта» и «Характер локализации» класса «Справочные данные». Перечисленные элементы данных могут принимать строго определенные значения, которые указаны в классификаторе. На диаграмме это условие выражено с помощью отношения зависимости между классом «Справочные данные» и пакетом «Классификатор». На диаграмме для упрощения модели данные классификатора «свернуты» в пакет, структура данных классификатора будет рассмотрена ниже.

Характер локализации объекта (цифровой карты) - это вид геометрического представления объекта цифровой карты [7]. В соответствии с характером локализации объект может быть линейный, площадной и точечный. Виды объекта векторный и подпись здесь не рассматриваются, поскольку не используются в решении задачи импорта шейп-данных.

Площадной объект – это объект электронной карты, метрическое описание которого представлено последовательностью координат точек его замкнутого контура. Линейный объект – это объект электронной карты, метрическое описание которого представлено последовательностью координат его точек. Точечный объект – это объект электронной карты, метрическое описание которого представлено координатами одной точки [7].

Пространственный объект может состоять из нескольких составных частей, каждая из которых имеет свою метрику. Такой объект называется сложным или составным. На диаграмме тот факт, что объект может состоять из нескольких частей и соответственно иметь несколько метрик, отображен с помощью множественности 1 к N ассоциации композитного агрегирования между классами «Объект» и «Метрика». Далее по умолчанию будем предполагать, что объект состоит из одной части и имеет одну метрику.

Метрика объекта цифровой топографической карты – часть информации в составе объекта цифровой топографической карты, описывающая местоположение и плановые очертания объекта топографической карты [9]. Данные метрики объекта состоят из координат его точек. На диаграмме точки метрики отображены с помощью класса «Точка». Класс «Метрика» является контейнером для объектов класса «Точка», что показано с помощью ассоциации композитного агрегирования между ними. Каждая точка метрики объекта может иметь три измерения: X, Y и H - высота. Эти элементы данных представлены с помощью атрибутов «X», «Y» и «H» класса «Точка».

Данные семантики состоят из кода семантики и ее значения. На диаграмме эти элементы данных показаны с помощью атрибутов «Код семантики» и «Значение» класса «Семантика». Как и в случае справочных данных все допустимые коды семантик объекта определены в классификаторе, что на диаграмме показано отношением зависимости между классом «Семантика» и пакетом «Классификатор».

Операции класса «Объект» имеют следующее назначение:

mapCreateObject() – операция «конструктор», которая создает экземпляр класса «Объект» в контейнере «:Цифровая карта». Особенностью данной операции является то, что при создании экземпляра класса «Объект» также создается неинициализированный объект класса «Метрика». Надо отметить, что в случае сложного объекта, когда необходимо добавить метрики составных частей в контейнер «:Объект», для создания объекта класса «Метрика» дополнительно к этой операции используется операция «mapCreateSubject».

mapCreateSubject() – операция создает экземпляр класса «Метрика» для сегмента сложного объекта в контейнере «:Объект».

mapAppendSemantic() – операция создает экземпляр класса «Семантика» в контейнере «:Объект».

mapRegisterObject() – операция создает экземпляр класса «Справочные данные» в контейнере «:Объект».

mapAppendPointPlane() – операция создает экземпляр класса «Точка» в контейнере «:Метрика».

mapSetHPlane() – операция инициализирует атрибут «H» класса «Точка».

mapCommitObject() – операция сохраняет состояние объекта в долговременной памяти.

mapFreeObject() – операция «деструктор», которая освобождает ресурсы, связанные с объектом.

mapGeoToPlane() – операция преобразования координат из географической системы в прямоугольную.

Далее для удобства изложения рассмотренная модель цифровой карты будет представлена в виде пакета с названием «БГИСРВ».

В рассматриваемой технологии импорта данных наличие цифрового классификатора в модели цифровой карты БГИСРВ является обязательным условием. Классификатор содержит систематизированный перечень наименований и кодов объектов цифровых карт и их характеристик [7]. Диаграмма классов классификатора показана на рисунке 2.

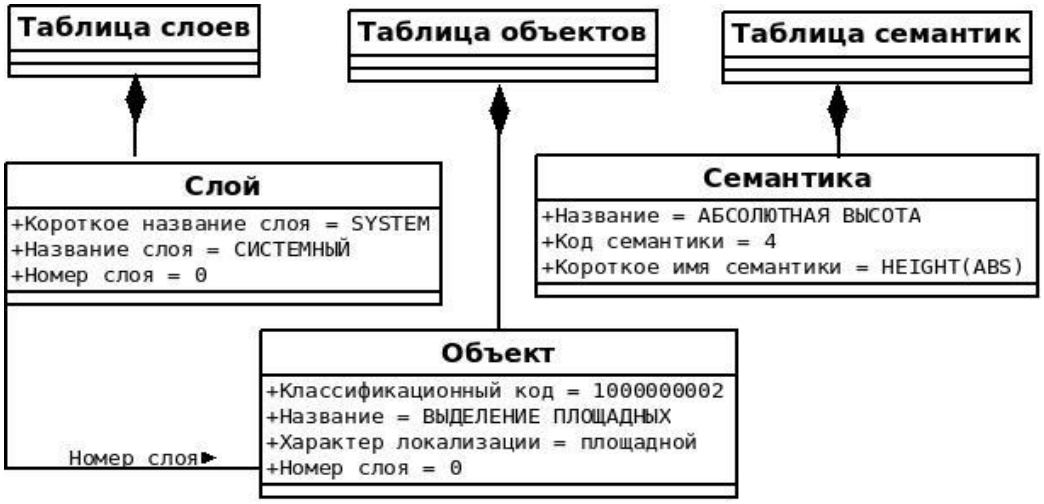


Рис. 2. Диаграмма классов классификатора

Структура данных классификатора организована в виде набора таблиц данных, которые в модели обозначены классами с именами, совпадающими с именами таблиц в классификаторе. Это следующие классы: «Таблица семантик», «Таблица слоев» и «Таблица объектов». Указанные классы являются соответственно контейнерами для объектов классов «Семантика», «Слой» и «Объект», что отображено на диаграмме с помощью ассоциаций композитного агрегирования между соответствующими классами. В модели отображены только те сущности классификатора, которые участвуют в решении задачи импорта шейп-данных. Подробное описание классификатора приведено в работе [10].

В таблице семантик хранится информация о характеристиках объектов цифровой карты. Характеристики объекта на диаграмме представлены классом «Семантика», который имеет следующие элементы данных: «Название», «Короткое имя семантики», а также «Код семантики». Атрибут «Название» содержит имя характеристики, атрибут «Короткое имя семантики» содержит кодовое слово для подписи полей в базах данных, атрибут «Код семантики» содержит уникальный номер семантики в таблице. В описании класса «Семантика», для примера, в качестве значений по умолчанию указаны значения атрибутов одной из записей таблицы семантик классификатора «OPENSTREETMAP» [11].

Таблица слоев содержит данные о слоях карты, которые на диаграмме изображены классом «Слой». Слой (пространственных данных) – подмножество пространственных объектов предметной области, обладающих тематической общностью и единой для всех слоев системой координат [1]. На диаграмме распределение объектов по слоям показано с

помощью классов «Слой» и «Объект» и ассоциации между ними, основанной на атрибуте «Номер слоя» перечисленных выше классов. Класс «Слой» характеризуется следующими элементами данных: «Название слоя», «Короткое название слоя» и «Номер слоя». Атрибут «Номер слоя» используется для связи объектов «:Слой» контейнера «:Таблица слоев» с сущностями «:Объект» контейнера «:Таблица объектов». Эта связь показана с помощью ассоциации с именем «Номер слоя» между классами «Слой» и «Объект». Атрибут «Название слоя» содержит название слоя. Атрибут «Короткое название слоя» служит для связи с названием полей в базах данных. Для примера, в описании класса «Слой» в качестве значений по умолчанию указаны значения атрибутов одной из записей таблицы слоев классификатора «OPENSTREETMAP».

Таблица объектов содержит справочные данные объектов, которые на диаграмме отображены с помощью класса «Объект». Этот класс имеет следующие атрибуты: «Название», «Характер локализации», «Номер слоя» и «Классификационный код». Атрибут «Название» содержит название объекта, атрибут «Номер слоя» содержит номер слоя и служит для связи элементов «:Объект» контейнера «:Таблица объектов» с объектами «:Слой» контейнера «:Таблица слоев», атрибут «Характер локализации» указывает вид геометрического представления объекта, атрибут «Классификационный код» содержит код вида объекта. В описании класса «Объект», для примера, в качестве значений по умолчанию указаны значения атрибутов одной из записей таблицы объектов классификатора «OPENSTREETMAP».

3. Описание шейп-типов пространственных объектов формата «шейп-файл»

Прежде, чем рассматривать модель цифровой карты библиотеки «Shapefile C Library», необходимо сделать обзор шейп-типов (shape type) пространственных объектов формата данных «шейп-файл». Подробное описание шейп-типов пространственных объектов формата «шейп-файл» приведено в работе [12].

Шейп-тип, как и характер локализации в модели цифровой карты библиотеки БГИСРВ, определяет вид геометрического представления пространственного объекта.

В спецификации формата «шейп-файл» список шейп-типов пространственных объектов делится на три группы.

1) Шейп-типы объектов в пространстве X и Y.

Координаты точек метрики объектов данной группы шейп-типов имеют два измерения: X и Y. В этой группе имеются следующие шейп-типы пространственных объектов.

Point (Точка) – тип объекта, который состоит из одной точки.

MultiPoint (Мультиточка) – тип объекта, представленный набором точек.

PolyLine (Полилиния) – тип объекта, который состоит из одного или нескольких сегментов, представляющих собой связанную последовательность из двух или более точек. Сегменты могут быть связаны и не связаны друг с другом, пересекаться и не пересекаться.

Polygon (Полигон) – тип объекта, который состоит из одного или нескольких сегментов. Сегмент представляет собой связную последовательность из четырех или более точек, которые образуют замкнутую, не самопересекающуюся «петлю».

2) Шейп-типы объектов в пространстве X, Y с измеряемым значением M.

В качестве примеров измеряемого значения M можно привести такие параметры объектов как глубина или расстояние на километровом столбе.

Объекты этой группы шейп-типов имеют те же геометрические параметры, что и объекты в пространстве X и Y, но к координатным данным точек их метрик X и Y добавлено измеряемое значение M. Они соответственно имеют названия PointM (Точка M), MultiPointM (Мультиточка M), PolyLineM (Полилиния M) и PolygonM (Полигон M).

3) Шейп-типы объектов в пространстве X, Y и Z.

У объектов этой группы шейп-типов координатные данные точек метрик имеют четыре измерения: X, Y, Z и M. Измерение Z предназначено для хранения значений высоты точки объекта. В этой группе имеются типы объектов подобные типам объектов в пространстве X, Y с измеряемым значением M, а именно PointZ (Точка Z), MultiPointZ (Мультиточка Z), PolyLineZ (Полилиния Z) и PolygonZ (Полигон Z), с той разницей, что к координатным данным точек

объектов этой группы типов добавлено измерение Z. Кроме того, в этой группе типов имеется особый шейп-тип MultiPatch (Мультипатч) – это тип 3D объекта, который состоит из поверхностей, отображающих его границы. Поверхность может быть в виде треугольников, вееров треугольников, полос треугольников или колец [13].

Кроме описанных выше шейп-типов в спецификации «шейп-файл» имеется особый вид типа объекта «NULL shape» особенностью объектов этого типа является то, что они не имеют метрики.

4. Модель цифровой карты библиотеки «Shapefile C Library»

Диаграмма классов цифровой карты библиотеки «Shapefile C Library» показана на рисунке 3.

В модели отображены только те элементы, которые участвуют в решении задачи импорта данных. Полное описание прикладного программного интерфейса библиотеки «Shapefile C Library» приведено в работе [5].

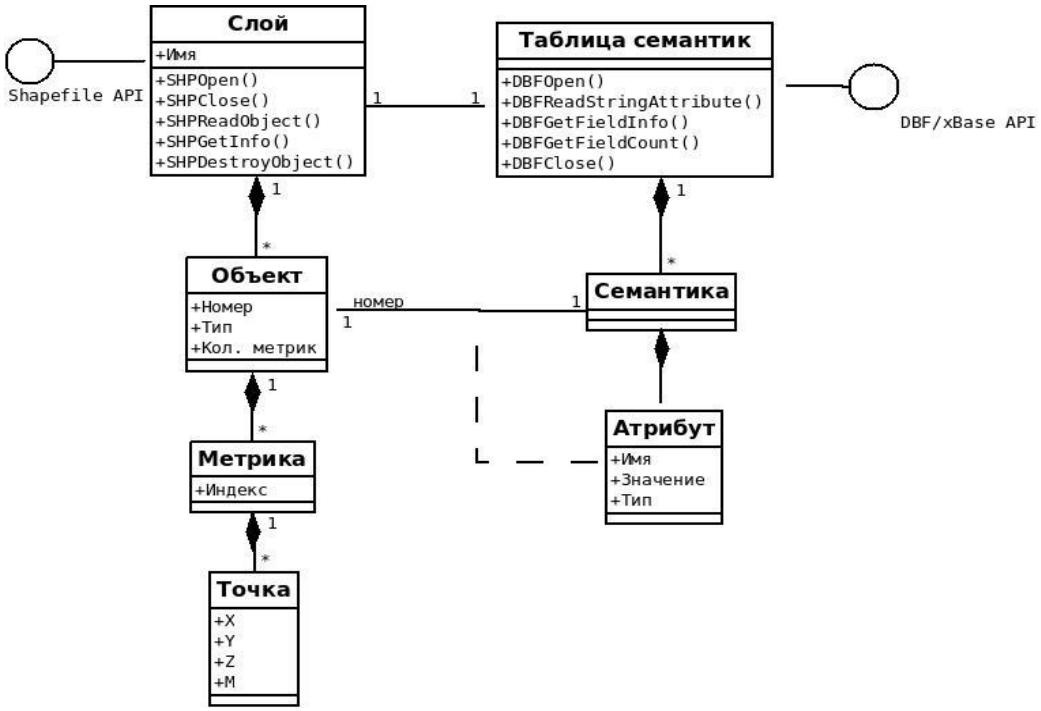


Рис. 3. Диаграмма классов цифровой карты библиотеки «Shapefile C Library»

Класс «Слой» обозначает слой пространственных объектов. В модели цифровой карты библиотеки «Shapefile C Library», в отличие от модели цифровой карты БГИСРВ, класс «Слой» является контейнером для класса «Объект». На диаграмме это отображено с помощью отношения композитного агрегирования между соответствующими классами. Класс «Слой» имеет один атрибут «Имя», который служит для идентификации объектов «:Слой» в операциях интерфейса «Shapefile API».

Класс «Слой» реализует интерфейс «Shapefile API», который включает в себя следующие операции:

SHPOpen() – операция «конструктор», которая используется для создания объекта класса «Слой» и восстановления его состояния из долговременной памяти.

SHPClose() – операция «деструктор», которая сохраняет состояние экземпляра класса «Слой» в долговременной памяти и удаляет связанные с ним ресурсы.

SHPGetInfo() – операция позволяет узнать количество объектов в слое.

SHPReadObject() – операция извлекает экземпляр класса «Объект» из контейнера «:Слой».

SHPDestroyObject() – операция «деструктор», которая освобождает связанные с объектом «:Объект» ресурсы.

Класс «Объект» обозначает пространственные объекты слоя. Данные пространственного объекта по виду делятся на координатные и атрибутивные, на диаграмме эти данные обозначаются соответственно классами «Метрика» и «Семантика». Класс «Объект» является контейнером для класса «Метрика», на диаграмме этот факт отображен с помощью ассоциации композитного агрегирования между указанными классами. У класса «Объект» имеются три атрибута: «Номер», «Тип» и «Кол. метрик». Атрибут «Номер» участвует в создании ассоциации между классами «Объект» и «Семантика», на диаграмме данная ассоциация имеет название «номер». Атрибут «Тип» определяет шейп-тип экземпляров класса «Объект». В контейнере «:Слой» объекты «:Объект», у которых есть метрика «:Метрика», имеют одинаковый шейп-тип. Атрибут «Кол. метрик» задает количество объектов «:Метрика» в контейнере «:Объект».

Как и в модели цифровой карты БГИСРВ, объект может быть сложным, то есть состоять из нескольких частей, каждая из которых описывается отдельной метрикой. На диаграмме это свойство пространственного объекта показано с помощью множественности один ко многим ассоциации между классами «Объект» и «Метрика».

Класс «Метрика» имеет атрибут «Индекс», с помощью которого при выполнении операций обеспечивается доступ к экземплярам класса «Метрика» в контейнере «:Объект».

Данные метрики состоят из координат точек объекта, которые на диаграмме отображены с помощью класса «Точка». Класс «Метрика» является контейнером для объектов класса «Точка», что на диаграмме показано с помощью ассоциации композитного агрегирования.

Координаты точки пространственного объекта в зависимости от его шейп-типа могут иметь от двух до четырех измерений, а именно X, Y, Z и M, в спецификации класса «Точка» указанные элементы данных представлены одноименными атрибутами.

Атрибутивные данные объекта на диаграмме представлены классом «Семантика», они структурно организованы в виде таблицы, которая на диаграмме представлена классом «Таблица семантик», который является контейнером для данных класса «Семантика», что показано с помощью ассоциации композитного агрегирования между указанными классами.

Между классом «Объект» и «Семантика» существует отношение один к одному, то есть для каждого объекта класса «Объект» всегда имеется один объект класса «Семантика», на диаграмме это свойство отображено с помощью ассоциации с множественностью один к одному между классами «Объект» и «Семантика».

Семантика объекта определяется с помощью набора атрибутов, которые представлены на диаграмме классом «Атрибут». Класс «Семантика» является контейнером для данных класса «Атрибут», что на диаграмме показано с помощью ассоциации композитного агрегирования между соответствующими классами. У класса «Атрибут» имеются следующие свойства: «Имя» - задает имя атрибута семантики; «Значение» - содержит значение атрибута; «Тип» - определяет тип данных атрибута. Свойство «Тип» может принимать следующие значения: «строка», «целое», «вещественное», «логическое» и «неизвестное».

У класса «Атрибут» имеется еще одно свойство – он выполняет роль класса ассоциаций между классами «Объект» и «Семантика». Класс ассоциаций позволяет определять для ассоциаций атрибуты, операции и другие свойства [8]. На диаграмме это свойство класса показано с помощью пунктирной линии, соединяющей класс «Атрибут» с ассоциацией между классами «Объект» и «Семантика». На уровне реализации это означает, что в контейнере «:Семантика» всегда имеется один объект «Номер:Атрибут» с именем «Номер», у которого свойство «Значение» содержит значение атрибута «Номер» контейнера «:Объект», с которым связан данный контейнер «:Семантика».

Класс «Таблица семантик» реализует интерфейс «DBF/xBase API», который включает следующие операции манипулирования данными таблицы:

DBFOpen() – операция «конструктор», которая используется для создания объекта «:Таблица семантик» и

восстановления его состояния из долговременной памяти.

DBFClose() – операция «деструктор», которая сохраняет состояние объекта «:Таблица семантик» в долговременной памяти и удаляет связанные с ним ресурсы.

DBFGetFieldInfo() – операция позволяет узнать значение свойств «Тип» и «Имя» объекта «:Атрибут».

DBFReadStringAttribute() – операция возвращает значение свойства «Значение» объекта «:Атрибут».

DBFGetFieldCount() – операция позволяет узнать количество объектов «:Атрибут» в контейнере «:Семантика».

Далее для удобства изложения рассмотренная модель цифровой карты библиотеки «Shapefile C Library» будет представлена в виде пакета с названием «ShapeLib».

5. Описание технологии импорта шейп-данных

Одной из особенностей рассматриваемой здесь технологии импорта шейп-данных является применение библиотеки «Shapefile C Library» для манипулирования данными в формате «шейп-файл». Такое решение позволяет заменить модель данных спецификации «шейп-файл» моделью данных цифровой карты библиотеки «Shapefile C Library», которая представлена пакетом «ShapeLib». В результате этого модель данных становится проще за счет инкапсуляции операциями классов пакета «ShapeLib» данных, отвечающих за сохранение и восстановление состояния объектов в памяти, таким образом в модели данных остаются только сущности, описывающие непосредственно пространственные данные.

В предлагаемом подходе задача импорта шейп-данных сводится к задаче приведения модели цифровой карты пакета «ShapeLib» к модели цифровой карты пакета «БГИСРВ».

При сравнении моделей цифровых карт пакетов «ShapeLib» и «БГИСРВ» можно сделать следующие выводы: в пакете «ShapeLib» отсутствуют такие сущности, как «Паспорт» и «Справочные данные»; в пакетах «ShapeLib» и «БГИСРВ» используются разные системы типизации геометрических видов пространственных объектов. Кроме того, между семантически эквивалентными сущностями пакетов, например, классами «Семантика», отсутствуют связи, с помощью которых можно было бы выполнить преобразование данных объектов сущностей пакета «ShapeLib» в данные объектов сущностей пакета «БГИСРВ».

В соответствии с вышеописанным, задачу приведения модели данных пакета «ShapeLib» к модели данных пакета «БГИСРВ» можно разбить на следующие подзадачи: а) создание объектов классов «Паспорт» и «Справочные данные»; б) преобразование типа пространственного объекта, которое позволяет определить значение атрибута «Характер локализации» экземпляра класса «:БГИСРВ::Справочные данные» на основе данных атрибута «Тип» пространственного объекта «:ShapeLib::Объект»; г) определение отношений между семантически эквивалентными сущностями моделей; в) преобразование данных контейнера «:Объект» пакета «ShapeLib» в данные контейнера «:Объект» пакета «БГИСРВ».

В предлагаемом подходе проблема отсутствия связей между семантически эквивалентными сущностями моделей решается за счет расширения модели данных цифровой карты пакета «БГИСРВ» с помощью нового пакета классов ассоциаций, которому было присвоено имя «Конвертор» (см. Рисунок 4).

Рис. 4. Пакет «Конвертор»

На диаграмме с помощью ассоциации с именем «расширяет» между пакетами «БГИСРВ» и «Конвертор» показано, что модель данных цифровой карты пакета «БГИСРВ» расширена с помощью элементов пакета «Конвертор». Кроме того, с помощью отношения зависимости между пакетом «Конвертор» и пакетами «ShapeLib», «БГИСРВ» и «Классификатор» показано, что в операциях пакета «Конвертор» участвуют классы

пакетов «ShapeLib», «БГИСРВ» и «Классификатор».

На рисунке 5 изображена диаграмма классов пакета «Конвертор», для наглядности в нее включены классы из пакетов «ShapeLib», «БГИСРВ» и «Классификатор». Чтобы различать классы из разных пакетов, на диаграмме перед именем класса через разделитель «::» будет указываться имя его пакета.

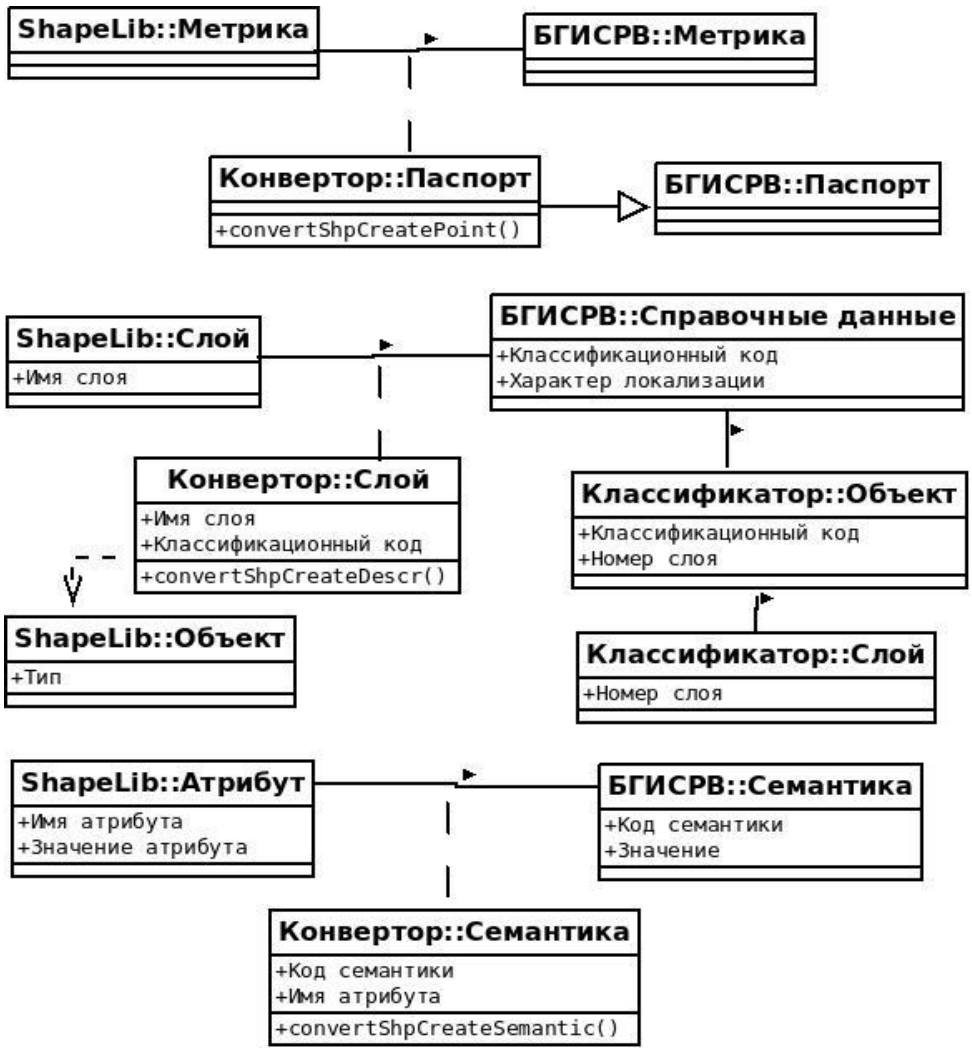


Рис. 5. Диаграмма классов ассоциаций пакета «Конвертор»

Класс ассоциаций «Конвертор::Паспорт» определяет связь между классами «Метрика» пакетов «БГИСРВ» и «ShapeLib». Он наследует атрибуты класса «БГИСРВ::Паспорт». На диаграмме указанное свойство обозначено с помощью отношения обобщения между классами «Паспорт» пакетов «Конвертор» и «БГИСРВ». Класс ассоциаций «Конвертор::Паспорт» имеет операцию «convertShpCreatePoint», которая выполняет инициализацию объекта «:БГИСРВ::Метрика» на основе данных объекта «:ShapeLib::Метрика». Алгоритм выполнения этой операции следующий. Из контейнера «:ShapeLib::Метрика» извлекается объект «:ShapeLib::Точка», данные которого в случае необходимости преобразуются из географической в прямоугольную систему координат операцией «mapGeoToPlane», затем с помощью операций «mapAppendPointPlane» и «mapSetHPlane» в контейнере «:БГИСРВ::Метрика» создается объект «:БГИСРВ::Точка», который инициализируется данными объекта

«:ShapeLib::Точка». Указанная последовательность действий выполняется для всех объектов «:ShapeLib::Точка» в контейнере «:ShapeLib::Метрика».

Класс ассоциаций «Конвертор::Слой» определяет связь между классами «ShapeLib::Слой» и «БГИСРВ::Справочные данные». Кроме того, с помощью класса ассоциаций «Конвертор::Слой» создается косвенная связь между классами «Слой» пакетов «ShapeLib» и «Классификатор». На диаграмме указанная косвенная связь образуется набором классов «ShapeLib::Слой», «Конвертор::Слой», «БГИСРВ::Справочные данные», «Классификатор::Объект» и «Классификатор::Слой». Класс ассоциаций «Конвертор::Слой» имеет атрибуты «Имя слоя» и «Классификационный код», которые участвуют в операции, выполняющей отображение данных атрибута «Имя слоя» объекта «:ShapeLib::Слой» в данные атрибута «Классификационный код» объекта «:БГИСРВ::Справочные данные». У класса ассоциаций «Конвертор::Слой» имеется операция «convertShpCreateDescr», которая имеет следующий алгоритм выполнения. С помощью операции «mapRegisterObject» в контейнере «:БГИСРВ::Объект» создается и инициализируется объект «:БГИСРВ::Справочные данные». В процессе инициализации выполняется преобразование данных атрибута «Имя слоя» объекта «:ShapeLib::Слой» в данные атрибута «Классификационный код» объекта «:БГИСРВ::Справочные данные» с помощью данных одноименных атрибутов объекта «:Конвертор::Слой», а также определяется характер локализации объекта «:БГИСРВ::Справочные данные». В указанной операции характер локализации объекта «:БГИСРВ::Справочные данные» определяется на основе данных атрибута «Тип» пространственного объекта «:ShapeLib::Объект» с помощью следующих правил [4] (см. Таблица 1):

Таблица 1. Правила преобразования типов

Номер правила	Тип объекта формата «шейп-файл»	Характер локализации БГИСРВ
1	Point (Точка)	Точечный
2	PointZ (Точка Z)	Точечный (с Н координатой)
3	PointM (Точка М)	Точечный (без параметра М)
4	MultiPoint (Мультиточка)	Точечный
5	MultiPointZ (Мультиточка Z)	Точечный (с Н координатой)
6	MultiPointM (Мультиточка М)	Точечный (без параметра М)
7	PolyLine (Полилиния)	Линейный (сложный)
8	PolyLineZ (Полилиния Z)	Линейный (сложный с Н координатой)
9	PolyLineM (Полилиния М)	Линейный (сложный без параметра М)
10	Polygon (Полигон)	Площадной
11	PolygonZ (Полигон Z)	Площадной (с Н координатой)
12	PolygonM (Полигон М)	Площадной (без параметра М)
13	MultiPatch (Мультипатч)	Не поддерживается
14	NULL shape	Не поддерживается

Применение данных атрибута «Тип» класса «ShapeLib::Объект» в операции «convertShpCreateDescr» класса ассоциаций «Конвертор::Слой», на диаграмме отображено с помощью отношения зависимости между соответствующими классами.

Класс ассоциаций «Конвертор::Семантика» определяет отношение между классами «ShapeLib::Атрибут» и «БГИСРВ::Семантика». Он имеет атрибуты «Код семантики» и «Имя атрибута», с помощью которых данные атрибута «Имя» объекта «:ShapeLib::Атрибут» преобразуются в данные атрибута «Код семантики» объекта «:БГИСРВ::Семантика», а также операцию с именем «convertShpCreateSemantic», которая инициализирует объект «:БГИСРВ::Семантика» на основе данных объекта «:ShapeLib::Атрибут». Алгоритм выполнения данной операции следующий. Для каждого объекта «:ShapeLib::Атрибут», за исключением объекта «Номер:ShapeLib::Атрибут», который содержит данные атрибута «Номер» объекта «:ShapeLib::Объект» и выполняет роль ассоциации между объектами классов «ShapeLib::Объект» и «ShapeLib::Семантика», с помощью операций «DBFGetFieldInfo» и «DBFReadStringAttribute» объекта «:ShapeLib::Таблица семантик» извлекаются данные атрибутов «Имя» и «Значение» объекта «:ShapeLib::Атрибут», а затем с помощью операции «mapAppendSemantic» контейнера «:БГИСРВ::Объект» в данном контейнере создается объект «:БГИСРВ::Семантика» и инициализируется полученными данными. В процессе инициализации данные атрибута «Имя» объекта «:ShapeLib::Атрибут» преобразуются в данные атрибута «Код семантики» объекта «:БГИСРВ::Семантика». При этом преобразовании используются значения атрибутов «Код семантики» и «Имя атрибута» объекта «:Конвертор::Семантика». Количество объектов «:ShapeLib::Атрибут» в контейнере «:ShapeLib::Семантика» определяется с помощью операции «DBFGetFieldCount» объекта «:ShapeLib::Таблица семантик».

В предлагаемом подходе для создания экземпляров перечисленных выше классов ассоциаций пакета «Конвертор» необходимо написать прикладную программу, диаграмма деятельности которой имеет следующий вид (см. Рисунок 6).

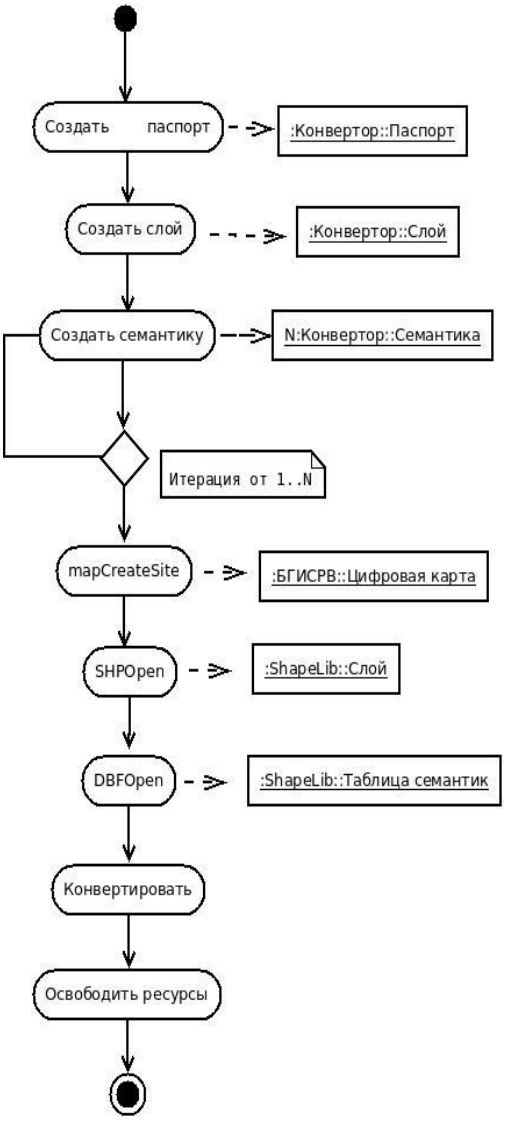


Рис. 6. Диаграмма деятельности прикладной программы

Далее, при описании диаграмм деятельности вместо термина «операция» диаграммы классов будут использоваться термины «деятельность» и «действие». Деятельность – это неатомарный набор действий внутри машины состояний. Действие – это исполняемое атомарное вычисление [6].

Действия «Создать паспорт», «Создать слой» и «Создать семантику» создают соответственно объекты: «:Конвертор::Паспорт», «:Конвертор::Слой», «N:Конвертор::Семантика». Следует обратить внимание на серию объектов «N:Конвертор::Семантика», в данном случае создается N объектов, где количество объектов определяется по формуле:

$$N = A - 1$$

где N – количество экземпляров класса «Конвертор::Семантика», A – количество экземпляров класса «ShapeLib::Атрибут» в контейнере «:ShapeLib::Семантика». Из формулы следует, что количество экземпляров класса «Конвертор::Семантика» на единицу меньше количества экземпляров класса «ShapeLib::Атрибут» в контейнере «:ShapeLib::Семантика», это связано с тем, что объект «:Конвертор::Семантика» не создается для объекта «Номер:ShapeLib::Атрибут», содержащего значение атрибута «Номер» экземпляра класса «ShapeLib::Объект», поскольку, как было сказано выше, объект «Номер:ShapeLib::Атрибут» является «служебным» и выполняет роль ассоциации между объектами классов «ShapeLib::Объект» и «ShapeLib::Семантика».

Реализация описанных выше действий может быть произвольной, в качестве примера приведем листинг фрагмента программы контрольной задачи на языке Си, реализующей действие «Создать паспорт».

```
CREATESITE sitedesc;
strcpy(sitedesc.MapName, "Example");
sitedesc.MapType = UTMWGS84;
sitedesc.Scale = 500000;
```

В данном примере программы в первой строке определяется переменная sitedesc, которая соответствует объекту «:Конвертор::Паспорт», с типом «CREATESITE», во второй строке определяется имя района работ, в третьей - обобщенный тип карты и в четвертой - знаменатель масштаба карты.

Действия «mapCreateSite», «SHPOpen» и «DBFOpen» создают соответственно объекты «:БГИСРВ::Цифровая карта» , «:ShapeLib::Слой» и «:ShapeLib::Таблица семантик».

Деятельность «Освободить ресурсы» сводится к выполнению действий «DBFClose», «SHPClose» и «mapCloseMap», с помощью которых сохраняются данные объектов в долговременной памяти и освобождаются связанные с ними ресурсы.

Деятельность «Конвертировать» выполняет отображение множества объектов «:ShapeLib::Объект» контейнера «:ShapeLib:Слой» в множество объектов «:БГИСРВ::Объект» контейнера «:БГИСРВ::Цифровая карта». Диаграмма деятельности «Конвертировать» показана на рисунке (см. Рисунок 7).

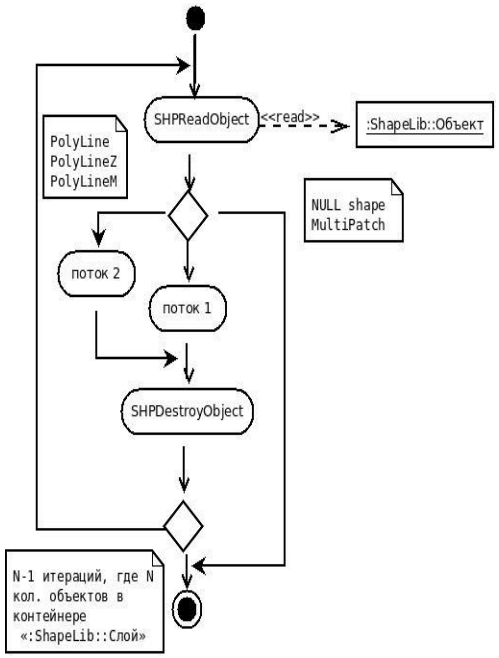


Рис. 7. Диаграмма деятельности «Конвертировать»

Алгоритм выполнения деятельности «Конвертировать» следующий. Сначала выполняется действие «SHPReadObject», которое извлекает из контейнера «:ShapeLib::Слой» объект «:ShapeLib::Объект». Затем анализируются данные атрибута «Тип» объекта «:ShapeLib::Объект», если атрибут имеет значения «MultiPatch» или «NULL shape», то деятельность завершается, если значения атрибута «PolyLine» или «PolyLineZ» или «PolyLineM», то происходит переход к деятельности с именем «поток 2», иначе активируется деятельность «поток 1». После этого выполняется действие «SHPDestroyObject», которое освобождает ресурсы, связанные с объектом «:ShapeLib::Объект». Описанная последовательность действий повторяется для всех объектов «:ShapeLib::Объект» контейнера «:ShapeLib::Слой».

Деятельность «поток 1» предназначена для создания объектов «:БГИСРВ::Объект» и инициализации их данными объектов «:ShapeLib::Объект» следующих шейп-типов: «Point», «PointZ», «PointM», «MultiPoint», «MultiPointZ», «MultiPointM», «Polygon», «PolygonZ» и «PolygonM». Диаграмма деятельности «поток 1» показана на рисунке 8.

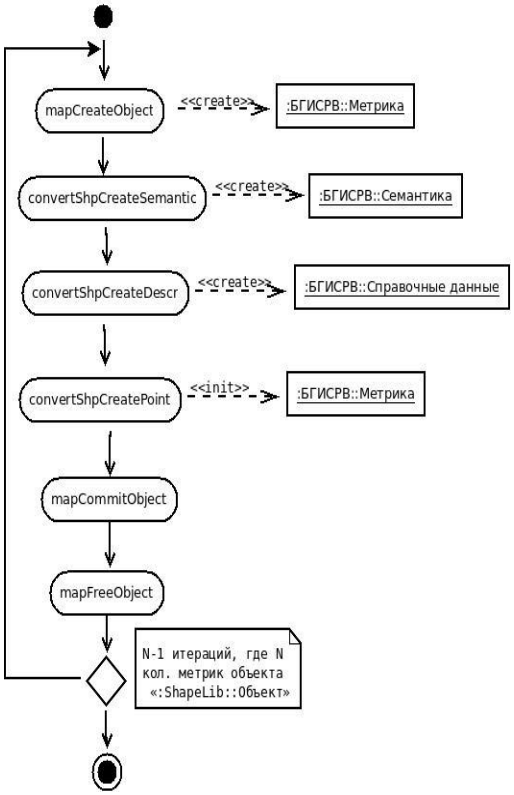


Рис. 8. Диаграмма деятельности «поток 1»

Особенность деятельности «поток 1» состоит в том, что в ней в случае сложных объектов «:ShapeLib::Объект» перечисленных выше шейп-типов в соответствии с правилами Таблицы 1 не создаются сложные объекты «:БГИСРВ::Объект», вместо этого для каждой метрики «:ShapeLib::Метрика» контейнера «:ShapeLib::Объект» создается контейнер «:БГИСРВ::Объект» с единственной метрикой «:БГИСРВ::Метрика». Алгоритм выполнения деятельности «поток 1» следующий. Сначала выполняется действие «mapCreateObject», которое создает контейнер «:БГИСРВ::Объект» и объект «:БГИСРВ::Метрика». Затем выполняются действия «convertShpCreateSemantic» и «convertShpCreateDescr», которые в контейнере «:БГИСРВ::Объект» создают и инициализируют соответственно объекты «:БГИСРВ::Семантика» и «:БГИСРВ::Справочные данные». После этого выполняется действие «convertShpCreatePoint», которое инициализирует объект «:БГИСРВ::Метрика» в контейнере «:БГИСРВ::Объект». Затем с помощью действия «mapCommitObject» состояние объекта «:БГИСРВ::Объект» сохраняется в долговременной памяти и с помощью действия «mapFreeObject» освобождаются ресурсы занятые этим объектом. Описанная последовательность действий повторяется для всех объектов «:ShapeLib::Метрика» контейнера «:ShapeLib::Объект».

Деятельность «поток 2» предназначена для создания объектов «:БГИСРВ::Объект» и инициализации их данными объектов «:ShapeLib::Объект» следующих шейп-типов: «PolyLine», «PolyLineZ» и «PolyLineM». В деятельности «поток 2» в случае сложных объектов «:ShapeLib::Объект» перечисленных выше шейп-типов в соответствии с Таблицей 1 правил создаются сложные объекты «:БГИСРВ::Объект», а именно для каждого контейнера «:ShapeLib::Объект» создается контейнер «:БГИСРВ::Объект» и для каждой метрики «:ShapeLib::Метрика» контейнера «:ShapeLib::Объект» создается метрика «:БГИСРВ::Метрика» в контейнере «:БГИСРВ::Объект». Диаграмма деятельности «поток 2» показана на рисунке 9.

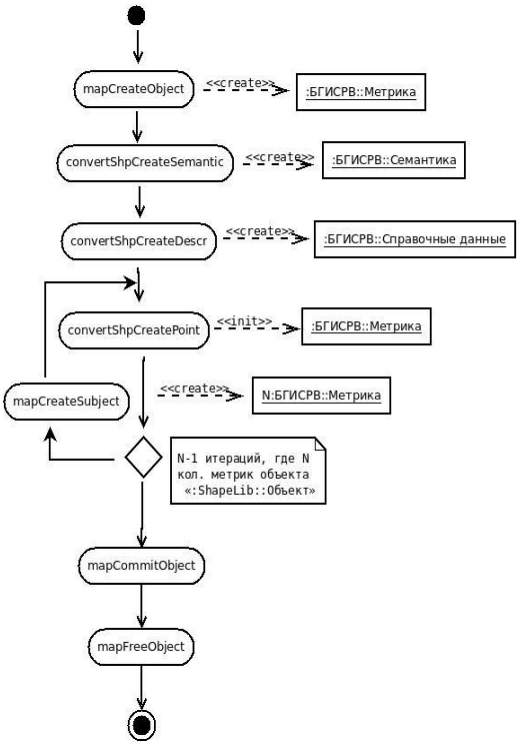


Рис. 9. Диаграмма деятельности «поток 2»

Алгоритм выполнения указанной деятельности следующий. С помощью действия «mapCreateObject» создаются контейнер «:БГИСРВ::Объект» и объект

«:БГИСРВ::Метрика». Затем выполняются действия «convertShpCreateSemantic» и «convertShpCreateDescr», которые в контейнере «:БГИСРВ::Объект» создают и инициализируют соответственно объекты «:БГИСРВ::Семантика» и «:БГИСРВ::Справочные данные». После этого выполняется действие «convertShpCreatePoint», которое инициализирует объект «:БГИСРВ::Метрика» в контейнере «:БГИСРВ::Объект». Далее, в случае сложного объекта, образуется цикл в котором для каждой метрики «:ShapeLib::Метрика» со значением атрибута «Индекс» N > 1 контейнера «:ShapeLib::Объект» с помощью действия «mapCreateSubject» пакета «БГИСРВ» в контейнере «:БГИСРВ::Объект» создается объект «N:БГИСРВ::Метрика» и инициализируется с помощью действия «convertShpCreatePoint». В конце деятельности «поток 2» выполняется действие «mapCommitObject», которое сохраняет состояние объекта «:БГИСРВ::Объект» в долговременной памяти, после чего выполняется действие «mapFreeObject», которое освобождает занятые этим объектом ресурсы.

6. Заключение

В статье с помощью средств моделирования языка UML приведено формальное описание метода импорта данных формата «шейп-файл» в систему БГИСРВ. При этом были получены следующие результаты: созданы модели цифровых карт библиотек «Shapefile C Library» и БГИСРВ, модель прикладной программы, а также модели пакета «Конвертор» в виде диаграммы классов, отображающей статическую семантику сущностей предметной области, и диаграмм деятельности, отображающих динамическую семантику. С помощью разработанных моделей был определен словарь изучаемой предметной области.

Программная реализация конвертора и ее тестирование с помощью контрольной задачи подтвердили корректность рассмотренного в данной статье метода импорта данных.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований по теме (проекту) «Исследование и реализация программной платформы для перспективных многоядерных процессоров» (№ 0065-2019-0002).

The method of importing spatial data of the format "shapefile" into the BGISPB system

P.V. Egorov

Abstract. The article describes the method of importing spatial data of the format "shapefile" into the BGISPB system

Keywords: spatial data, spatial object, data import, digital map, digital map passport, metric, classifier, shapefile.

Литература

1. ГОСТ Р 52438-2005 Географические информационные системы. Термины и определения.

2. Программное изделие Библиотека географической информационной системы для операционной системы реального времени, издание 5. Описание применения. ЮКСУ.90748-02 31 01.

3. Shapefile. https://ru.wikipedia.org/wiki/Shapefile (Дата обращения 22.02.2019).

4. Программное изделие геоинформационная система «ПАНОРАМА X64» (ГИС Панорама) Прикладные задачи. Импорт и экспорт данных формата Shapefile. http://gistoolkit.ru/download/doc/shapefile.pdf (Дата обращения 22.02.2019).

5. Shapefile C Library. http://shapelib.maptools.org (Дата обращения 22.02.2019)

6. Г. Буч, Д. Рамбо, А. Джекобсон. Язык UML. Руководство пользователя. М., ДМК, 2000.

7. ГОСТ 28441-99. КАРТОГРАФИЯ ЦИФРОВАЯ. Термины и определения.

8. М. Фаулер, К. Скотт. UML в кратком изложении. М., Мир, 1999.

9. ОСТ 68-3.1-98. Стандарт отрасли. Карты цифровые топографические. Общие требования.

10. Требования к навигационным картам. Формат цифрового классификатора rsc и библиотеки условных знаков. http://gistoolkit.ru/download/classifiers/formatrsc.pdf (Дата обращения 22.02.2019).

11. Классификатор слоев, семантических характеристик и объектов для отображения сведений на картах из открытых источников (OPENSTREETMAP). http://gistoolkit.ru/download/classifiers/osm_doc.pdf (Дата обращения 22.02.2019).

12. ESRI Shapefile Technical Description. https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf (Дата обращения 22.02.2019).

13. Мультипатчи. http://desktop.arcgis.com/ru/arcmap/latest/extensions/3d-analyst/multipatches.htm (Дата обращения 22.02.2019).

Лучевые числа и алгебры

А.В. Коганов

ФНЦ НИИСИ РАН, Москва, Россия, E-mail: akoganov@yandex.ru

Аннотация. В статье рассмотрен новый подход к теории гиперкомплексных чисел, при котором отрицательные числа рассматриваются как комплексное расширение неотрицательных чисел. Такие алгебры, названные лучевыми, обладают неожиданными свойствами, и классические комплексные и гиперкомплексные алгебры получаются из них путем факторизации. В частности, будет показано, что в лучевых алгебрах имеется несколько различных операций сопряжения. Некоторые из этих операций могут быть использованы в математической и теоретической физике. Подробно рассмотрен случай модификации алгебры комплексных чисел.

Ключевые слова: гиперкомплексные числа, комплексная единица, дистрибутивные алгебры, световой конус, стрела времени.

1. Введение

В некоторых направлениях современной теоретической и математической физики возникают затруднения из-за того, что естественные симметрии многомерных линейных пространств оказываются избыточными по отношению к симметриям моделируемых объектов. Наиболее ярким примером такой избыточности является так называемая стрела времени. Моделирование времени (или мировой линии) в форме числовой прямой требует признания равноправия обоих направлений времени. Но это явно противоречит как физическим экспериментам, так и бытовой практике. Поиск новых алгебраических конструкций характерен для современной математической физики, например, [2,9-11]. В основном, эти работы направлены на построение новых моделей пространства-времени или квантовой механики. В этой статье автор не ставит перед собой задачу разрешения всех подобных проблем в теории. Но будет построена алгебра чисел, у которых имеется ограничение степеней свободы, аналогичное нарушению симметрии времени. В конце статьи будет показано, как этим свойством можно воспользоваться для усиления одной из известных моделей нарушения симметрии пространства-времени. Близкие идеи автор изложил в заметках [14-17], где рассмотрена проблема генерации дискретного пространства-времени в теории квантовой генерации, совместимой с принципами релятивистской физики.

Основное содержание статьи посвящено изучению таких алгебр, которые предлагается называть лучевыми, а соответствующие им числа — лучевыми числами. Основная идея этой работы состоит в том, чтобы считать отрицательные числа комплексным расширением алгебры неотрицательных чисел. Соответственно, (-1) становится новой комплексной единицей, и каждая комплексная единица гиперкомплексных алгебр [1,2] оснащается дуальной отрицательной комплексной единицей. Базовой алгеброй становится положительная полуось с операциями сложения и умножения. Эта конструкция геометрически является лучом. Каждая комплексная единица тоже становится лучом в новом измерении. При этом, её отрицательный двойник порождает свой луч в новом измерении.

2. Луч как алгебра

Носитель $\mathbb{R}_{0+} = \{x \mid x \geq 0\}$. Операции $+ \quad \cdot$ Обе операции коммутативные, ассоциативные, сложение двусторонне дистрибутивно с умножением. Операции определяются как в действительной арифметике на положительных числах.

Свойства операций.

По сложению луч полугруппа. По умножению это группа на строго положительных числах и поглощающая точка 0 является отдельным от группы идеалом.

3. Действительные лучевые числа

Комплексные единицы $0,1,\nu$ (К-единицы).

Умножение единиц.

$$0\cdot 1=1\cdot 0=0;\quad 0\cdot\nu=\nu\cdot 0=0;\quad 0\cdot 0=0;$$
$$1\cdot 1=1;\quad 1\cdot\nu=\nu\cdot 1=\nu;\quad \nu\cdot\nu=1; \qquad (3.1)$$

Пояснение: К-единица ν соответствует (-1) в действительной алгебре.

Умножение двусторонне дистрибутивное по сложению. Сложение коммутативно.

Общий вид действительного лучевого числа:

$$x1+y\nu;\ x\ge 0,\ y\ge 0;\ \text{или}\ 0\,. \quad (3.2)$$

Сложение:

$$(x1+y\nu)+(a1+b\nu)=(x+a)1+(y+b)\nu; \quad (3.3)$$

$$0+(x1+y\nu)=(x1+y\nu)+0=x1+y\nu;$$

Умножение:

$$(x1+y\nu)(a1+b\nu)=(xa+yb)1+(xb+ya)\nu; \quad (3.4)$$

В этих формулах все коэффициенты перед К-единицами неотрицательные числа.

Геометрически это двумерный квадрант.

Алгебраически это полугруппа по умножению и сложению с нулевым поглощающим идеалом по умножению.

Теорема 3.1. Для лучевых двухкомпонентных чисел не возможно деление, как обратная к умножению операция. Доказательство.

$$(x_1\mathbf{1}+x_2\nu)(y_1\mathbf{1}+y_2\nu)=(x_1y_1+x_2y_2)1+(x_1y_2+x_2y_1)\nu$$

$$y1=x1/(x_1^2-x_2^2);$$

$$y2=-x2/(x_1^2-x_2^2);$$

Поскольку все компоненты x неотрицательны, то либо знаменатель равен нулю, либо компоненты y имеют разные знаки. Это невозможно для лучевого числа. □

Матричное представление лучевого действительного числа:

$$x\mapsto\begin{pmatrix}x_1 & x_2\\ x_2 & x_1\end{pmatrix}=[x]$$

Тогда

$$xy\mapsto\begin{pmatrix}x_1 & x_2\\ x_2 & x_1\end{pmatrix}\begin{pmatrix}y_1 & y_2\\ y_2 & y_1\end{pmatrix}=[xy]=[x][y].$$

$$x+y\mapsto\begin{pmatrix}x_1 & x_2\\ x_2 & x_1\end{pmatrix}+\begin{pmatrix}y_1 & y_2\\ y_2 & y_1\end{pmatrix}=[x+y]=[x]+[y]$$

4. Лучевые комплексные числа

К-единицы: $0, \mathbf{1}, \nu, \mathbf{i}, \mathbf{j}$

$$0\cdot\mathbf{1}=\mathbf{1}\cdot 0=0;\quad 0\cdot\nu=\nu\cdot 0=0;\quad 0\cdot 0=0;$$

$$0\cdot\mathbf{i}=\mathbf{i}\cdot 0=0;\quad 0\cdot\mathbf{j}=\mathbf{j}\cdot 0=0;$$

$$\mathbf{1}\cdot\mathbf{1}=\mathbf{1};\quad \mathbf{1}\cdot\nu=\nu\cdot\mathbf{1}=\nu;\quad \nu\cdot\nu=\mathbf{1};$$

$$1\cdot\mathbf{i}=\mathbf{i}\cdot 1=\mathbf{i};\quad 1\cdot\mathbf{j}=\mathbf{j}\cdot 1=\mathbf{j};$$

$$\mathbf{i}\cdot\mathbf{i}=\nu;\ \mathbf{i}\cdot\mathbf{j}=\mathbf{j}\cdot\mathbf{i}=1;\ \mathbf{j}\cdot\mathbf{j}=1;$$

$$\nu\cdot\mathbf{i}=\mathbf{i}\cdot\nu=\mathbf{j};\ \nu\cdot\mathbf{j}=\mathbf{j}\cdot\nu=\mathbf{i}; \quad (4.1)$$

Пояснение: К-единицы i и j соответствуют мнимой единице (i) и ($-i$) в комплексной алгебре.

Геометрически, комплексная лучевая алгебра соответствует 4-мерному квадранту.

Алгебраически это полугруппа по умножению и сложению с нулевым поглощающим идеалом по умножению. Умножению дистрибутивно по сложению. Обе операции коммутативны.

Общий вид числа

$$x\mathbf{1}+y\nu+a\mathbf{i}+b\mathbf{j} \quad (4.3)$$

Операции.

$$(x\mathbf{1}+y\nu+a\mathbf{i}+b\mathbf{j})+(x'\mathbf{1}+y'\nu+a'\mathbf{i}+b'\mathbf{j})=$$
$$=(x+x')\mathbf{1}+(y+y')\nu+(a+a')\mathbf{i}+(b+b')\mathbf{j} \quad (4.4)$$

$$(x\mathbf{1}+y\nu+a\mathbf{i}+b\mathbf{j})(x'\mathbf{1}+y'\nu+a'\mathbf{i}+b'\mathbf{j})=$$
$$=(xx'+yy'+ba'+ab')\mathbf{1}+(yx'+xy'+bb'+aa')\nu+$$
$$+(xa'+ax'+yb'+by')\mathbf{i}+(x'b+bx'+ya'+ay')\mathbf{j} \quad (4.5)$$

5. Общие теоремы лучевых алгебр

Теорема 5.1. Поскольку все числа в скобках неотрицательные, в лучевой алгебре любой размерности нет делителей нуля. Это свойство не зависит от алгебры комплексных единиц. И в обёртывающей алгебре любой лучевой алгебры нет делителей нуля.

Но из утверждения 4.1. не следует, что операция умножения обратима. Это связано с отсутствием отрицательных коэффициентов.

Теорема 5.2. Любая дистрибутивная лучевая алгебра является мультипликативно модулярной в метрике $L1$.

Доказательство. Обозначим К-единицы алгебры $1_1, 1_2, \ldots, 1_n$. Результат перемножения единиц обозначим $1_i\cdot 1_j=1_{ij}$, где двойной индекс соответствует одному из основных индексов в соответствие с таблицей умножения: $1_{ij}\in\{1_1;\ldots;1_n\}$. Тогда элемент алгебры имеет вид $x=\sum_{i=1}^{n}x_i1_i$. Тогда произведение имеет вид

$$xy=\sum_{i,j=1}^{n}x_iy_i1_{ij} \quad (5.1)$$

$$\|xy\|_{L1} = \sum_{i,j=1}^{n} x_i y_i; \qquad (5.2)$$

$$\|x\|_{L1} = \sum_{i=1}^{n} x_i; \quad \|y\|_{L1} = \sum_{i=1}^{n} y_i;$$

$$\|x\|_{L1}\|y\|_{L1} = \sum_{i,j=1}^{n} x_i y_i = \|xy\|_{L1}; \qquad (5.3)$$

□

Теорема 5.3. Деление в произвольной дистрибутивной лучевой алгебре сводится к решению некоторой системы линейных уравнений относительно координат чисел, рассмотренных как векторы. При этом, если полученное векторное решение имеет неотрицательные координаты, то это частное от деления. А если решения нет или оно имеет отрицательные координаты, значит деление данных лучевых чисел невозможно.

Доказательство. Используем обозначения из (5.1).Обозначим

$$W(k) = \left\{ij \mid 1_{ij} = 1_k\right\}; \qquad (5.4)$$

Тогда из 5.1 следует

$$xy = \sum_{i,j=1}^{n} x_i y_i 1_{ij} = \sum_{k=1}^{n}\left(\sum\nolimits_{ij\in W(k)} x_i y_i\right)1_k \qquad (5.5)$$

Если $x = z/y$, то $z = xy$. Получаем систему уравнений относительно неизвестного x:

$$z_k = \sum\nolimits_{ij\in W(k)} x_i y_i; \quad k = \overline{1,n}. \qquad (5.6)$$

Решение надо получить в неотрицательных числах.

Это доказывает теорему. □

Теорема 5.4. Любая лучевая алгебра аддитивно модулярна.

$$\|x+y\|_{L1} = \|x\|_{L1} + \|y\|_{L1} \qquad (5.7)$$

Это равенство имеет место в силу неотрицательности компонент лучевых чисел

$$\|x\|_{L1} = \sum_{i=1}^{n} x_i; \quad \|y\|_{L1} = \sum_{i=1}^{n} y_i;$$

$$\|x+y\|_{L1} = \sum_{i=1}^{n}|x_i + y_i| = \sum_{i=1}^{n}(x_i + y_i) =$$

$$= \sum_{i=1}^{n}(x_i) + \sum_{i=1}^{n}(y_i) = \sum_{i=1}^{n}|x_i| + \sum_{i=1}^{n}|y_i| = \|x\|_{L1} + \|y\|_{L1}$$

□

6. Деление в лучевой комплексной алгебре

Запишем систему уравнений (5.6) для комплексной лучевой алгебры (п. 4). В силу коммутативности этой алгебры систему можно записать для уравнения $z = yx$ в обычной форме с коэффициентами перед неизвестными.

$$\begin{cases} z1 = y1x1 + y2x2 + y4x3 + y3x4 \\ z2 = y2x_1 + y1x_2 + y3x_3 + y4x_4 \\ z3 = y3x_1 + y4x_2 + y1x_3 + y2x_4 \\ z4 = y4x_1 + y3x_2 + y2x_3 + y1x_4 \end{cases} \qquad (6.1)$$

Матрица $[y]$ этой системы даёт уравнение $[y]\vec{x} = \vec{z}$:

$$\begin{pmatrix} y_1 & y_2 & y_4 & y_3 \\ y_2 & y_1 & y_3 & y_4 \\ y_3 & y_4 & y_1 & y_2 \\ y_4 & y_3 & y_2 & y_1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} \qquad (6.2)$$

Если $\det[y] = 0$ система не имеет решения. деление не возможно. Это зависит только от делителя y. Если же $\det[y] \neq 0$, то решение есть. Но в лучевой алгебре требуется выполнение условия $\vec{x} \geq 0$. Если некоторая компонента числа x оказывается отрицательной, деление тоже не осуществимо. Рассмотрим эту систему подробнее. Введём обозначения.

$$W = \begin{pmatrix} y_4 & y_3 \\ y_3 & y_4 \end{pmatrix}; U = \begin{pmatrix} y_1 & y_2 \\ y_2 & y_1 \end{pmatrix};$$

$$V = \begin{pmatrix} y_3 & y_4 \\ y_4 & y_3 \end{pmatrix}; \qquad (6.3)$$

$$[y] = \begin{pmatrix} U & W \\ V & U \end{pmatrix} \qquad (6.4)$$

$$\det W = -\det V \qquad (6.5)$$

$$\det[y] = \det\begin{pmatrix} U & W \\ V & U \end{pmatrix} = \det^2 U + \det^2 V \qquad (6.6)$$

Из (6.6) следует, что $\det[y] \geq 0$.

При этом, если $\det[y] = 0$, то $\det U = \det V = 0$. Тогда (числа не отрицательные):

$$y_3 = y_4; \quad y_1 = y_2; \qquad (6.7)$$

Можно перейти от лучевого числа к обычному комплексному числу. Тогда в силу интерпретации лучевых К-единиц получим

$$y = y_1\mathbf{1} + y_2\nu + y_3\mathbf{i} + y_4\mathbf{j} \mapsto (y_1 - y_2) + i(y_3 - y_4) = C(y); \tag{6.8}$$

Это операция комплексификации.

$$\mathrm{Re}\,C(y) = (y_1 - y_2); \quad \mathrm{Im}\,C(y) = (y_3 - y_4);$$

Теорема 6.1. Если лучевое число y не может быть делителем никакого лучевого числа z, то $C(y) = 0$.

Доказательство. Пусть $\det[y] \neq 0$. Тогда уравнение $z = yx$ заведомо имеет решение $x = 1$ при $z = y$. А при $\det[y] = 0$ система не имеет решения. В силу (6.7), (6.8) это доказывает теорему. □

$$[y] = \begin{pmatrix} y_1 & y_2 & y_4 & y_3 \\ y_2 & y_1 & y_3 & y_4 \\ y_3 & y_4 & y_1 & y_2 \\ y_4 & y_3 & y_2 & y_1 \end{pmatrix} = \begin{pmatrix} U & W \\ V & U \end{pmatrix} \tag{6.9}$$

Если $\det[y] \neq 0$ то

$$[y][y]^{-1} = E = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.10}$$

Теорема 6.2. Отображение $y \mapsto [y]$ является матричным представлением лучевого комплексного числа. Доказательство. Непосредственно проверяется, что

$$[y + x] = [y] + [x]. \tag{6.11}$$

Покажем, что

$$[y][x] = [yx] \tag{6.12}$$

используя для числа x штрихованные обозначения структурных матриц, запишем

$$\begin{pmatrix} U & W \\ V & U \end{pmatrix}\begin{pmatrix} U' & W' \\ V' & U' \end{pmatrix} = \begin{pmatrix} UU'+WV' & UW'+WU' \\ VU'+UV' & UU'+WV' \end{pmatrix} = \begin{pmatrix} U'' & W'' \\ V'' & U'' \end{pmatrix}$$

Вычисляем.

$$WV' = \begin{pmatrix} y_4x_3 + y_3x_4 & y_4x_4 + y_3x_3 \\ y_4x_4 + y_3x_3 & y_4x_3 + y_3x_4 \end{pmatrix}$$

$$VW' = \begin{pmatrix} y_3x_4 + y_4x_3 & y_3x_3 + y_4x_4 \\ y_3x_3 + y_4x_4 & y_3x_3 + y_4x_4 \end{pmatrix}$$

$$VU' = \begin{pmatrix} y_3x_1 + y_4x_2 & y_3x_2 + y_4x_1 \\ y_4x_1 + y_3x_2 & y_4x_3 + y_3x_1 \end{pmatrix}$$

$$UW' = \begin{pmatrix} y_1x_4 + y_2x_3 & y_1x_3 + y_2x_4 \\ y_2x_4 + y_1x_3 & y_2x_3 + y_1x_4 \end{pmatrix}$$

$$WU' = \begin{pmatrix} y_4x_1 + y_3x_2 & y_4x_2 + y_3x_1 \\ y_3x_1 + y_4x_2 & y_3x_2 + y_4x_1 \end{pmatrix}$$

Видно, что все матрицы вида, соответствующего матрице представления числа, и это число равно произведению исходных чисел.

□

Следствие 1. Если $\det[y] \neq 0$, то существует матрица $[y]^{-1}$. Если $[y]^{-1} \geq 0$, то существует x, такой, что $yx = \mathbf{1}$, и $[x] = [y]^{-1}$. В этом случае возможно деление на лучевое число y:

$$z = xy \Rightarrow [x] = [z][y]^{-1}. \tag{6.13}$$

Если матрица $[y]^{-1}$ содержит отрицательные элементы, или этой матрицы не существует (т. е. $\det[y] = 0$), то деление на число y невозможно.

Следствие 2. Уравнение (6.10) позволяет непосредственно вычислить представление обратного числа, если оно существует. Пусть $\det[y] \neq 0$, $x = 1/y$. Тогда

$$x = \begin{pmatrix} y & y & y & y \\ y & y & y & y \\ y & y & y & y \\ y & y & y & y \end{pmatrix} \det{}^{-1}[y]$$

$$x_1 = \begin{pmatrix} 1 & y_2 & y_4 & y_3 \\ 0 & y_1 & y_3 & y_4 \\ 0 & y_4 & y_1 & y_2 \\ 0 & y_3 & y_2 & y_1 \end{pmatrix} \det{}^{-1}[y]$$

$$x_2 = \begin{pmatrix} y_1 & 1 & y_4 & y_3 \\ y_2 & 0 & y_3 & y_4 \\ y_3 & 0 & y_1 & y_2 \\ y_4 & 0 & y_2 & y_1 \end{pmatrix} \det{}^{-1}[y]$$

$$x_3 = \begin{pmatrix} y_1 & y_2 & 1 & y_3 \\ y_2 & y_1 & 0 & y_4 \\ y_3 & y_4 & 0 & y_2 \\ y_4 & y_3 & 0 & y_1 \end{pmatrix} \det{}^{-1}[y]$$

$$x_4 = \begin{pmatrix} y_1 & y_2 & y_4 & 1 \\ y_2 & y_1 & y_3 & 0 \\ y_3 & y_4 & y_1 & 0 \\ y_4 & y_3 & y_2 & 0 \end{pmatrix} \det{}^{-1}[y]$$

Обратное число найдено, если все полученные компоненты не отрицательные. В противном случае это число не существует. □

Заметим, что первый столбец представления задает представляемое лучевое комплексное число.

7. Нормализация лучевого комплексного числа

Операция нормализации лучевого комплексного числа соответствует восстановлению того комплексного числа, которому соответствует лучевое число, но с учетом разнесения положительных и отрицательных значений коэффициентов по разным лучам.

$$\left(x_1\mathbf{1} + x_2\nu + x_3\mathbf{i} + x_4\mathbf{j}\right)^{\odot} =$$
$$= \begin{cases} (x_1 - x_2)\mathbf{1} + 0\nu + (x_3 - x_4)\mathbf{i} + 0\mathbf{j} \Leftarrow x_1 \geq x_2, x_3 \geq x_4; \\ 0\mathbf{1} + (x_2 - x_1)\nu + (x_3 - x_4)\mathbf{i} + 0\mathbf{j} \Leftarrow x_1 < x_2, x_3 \geq x_4; \\ (x_1 - x_2)\mathbf{1} + 0\nu + 0\mathbf{i} + (x_4 - x_3)\mathbf{j} \Leftarrow x_1 \geq x_2, x_3 < x_4; \\ 0\mathbf{1} + (x_2 - x_1)\nu + 0\mathbf{i} + (x_4 - x_3)\mathbf{j} \Leftarrow x_1 < x_2, x_3 < x_4; \end{cases} \quad (7.1)$$

Опуская нулевые компоненты можно записать

$$\left(x_1\mathbf{1} + x_2\nu + x_3\mathbf{i} + x_4\mathbf{j}\right)^{\odot} =$$
$$= \begin{cases} (x_1 - x_2)\mathbf{1} + (x_3 - x_4)\mathbf{i} \Leftarrow x_1 \geq x_2, x_3 \geq x_4; \\ (x_2 - x_1)\nu + (x_3 - x_4)\mathbf{i} \Leftarrow x_1 < x_2, x_3 \geq x_4; \\ (x_1 - x_2)\mathbf{1} + (x_4 - x_3)\mathbf{j} \Leftarrow x_1 \geq x_2, x_3 < x_4; \\ (x_2 - x_1)\nu + (x_4 - x_3)\mathbf{j} \Leftarrow x_1 < x_2, x_3 < x_4; \end{cases}$$

Лемма 7.1. Значение $C(x)$ однозначно определяет значение $x^{\odot}$. Если $C(x) = A + iB$, то

$$x^{\odot} = \begin{cases} A\mathbf{1} + B\mathbf{i} \Leftarrow A \geq 0, B \geq 0; \\ A\nu + B\mathbf{i} \Leftarrow A < 0, B \geq 0; \\ A1 + B\mathbf{j} \Leftarrow A \geq 0, B < 0; \\ A\nu + B\mathbf{j} \Leftarrow A < 0, B < 0; \end{cases} \quad (7.2)$$

это непосредственно следует из определения комплексификации (6.8).

Теорема 7.1. Для любого лучевого комплексного числа x выполняются отношения:

сохранение комплексного образа нормализации

$$C(x) = C(x^{\odot}); \quad (7.3a)$$

$$\left\|C(x)\right\|_{L1} = \left\|x^{\odot}\right\|_{L1} \leq \left\|x\right\|_{L1} \quad (7.3b)$$

идемпотентность нормализации

$$(x^{\odot})^{\odot} = x^{\odot}; \quad (7.4)$$

алгебраическая транзитивность нормализации: если $F(z_1,\ldots,z_n) = y$ некоторое выражение в лучевой комплексной алгебре, то

$$(F(z_1,\ldots,z_n))^{\odot} = (F(z_1^{\odot},\ldots,z_n^{\odot}))^{\odot} = y^{\odot}. \quad (7.5)$$

Это значит, что нормализация является интроморфизмом (изоморфизмом на собственное подмножество) алгебры лучевых комплексных чисел.

Доказательство. Непосредственно из (6.8) и (7.2) следует (7.3). Из (7.2) сразу следует (7.4). Свойство (7.5) достаточно проверить для бинарных операций суммы и умножения

$$(x + y)^{\odot} = x^{\odot} + y^{\odot}; \quad (7.5a)$$

$$(x \cdot y)^{\odot} = x^{\odot} \cdot y^{\odot}. \quad (7.5b)$$

По лемме 7.1 Значение $C(x)$ однозначно определяет значение $x^{\odot}$.

Из (7.2)(7.3)

$$C((x + y)^{\odot}) = C(x^{\odot}) + C(y^{\odot}) = C(x + y),$$

что влечёт (7.5a).

$$C((xy)^{\odot}) = C(x^{\odot})C(y^{\odot}) = C(xy),$$

что влечёт (7.5b). □

Теорема 7.2. Пусть $F(z_1,\ldots,z_n) = y$ некоторое выражение в алгебре $(+\cdot)$ комплексных чисел. Пусть $x_1,\ldots,x_n$ набор лучевых комплексных чисел, и $C(x_1) = z_1,\ldots,C(x_n) = z_n$. Тогда выполняется свойство комплексной согласованности

$$C(F(x_1,\ldots,x_n)) = C(F(C(x_1),\ldots,C(x_n))) = y. \quad (7.6)$$

Доказательство. В силу того, что свойства ассоциативности, коммутативности и дистрибутивности имеются у обеих алгебр, достаточно проверить теорему для операций над базисными элементами (комплексными единицами). С учетом отношений

$$C(0) = 0; C(\mathbf{1}) = 1; C(\nu) = -1; C(\mathbf{i}) = i; C(\mathbf{j}) = -i; \quad (7.4)$$

утверждение теоремы очевидно.

Другой способ доказательства следует из (7.5a), (7.b), (7.3).

□

Теорема 7.3. Пусть уравнение обращения $yx=\mathbf{1}$ лучевого числа y (6.2) имеет решение x с действительными коэффициентами любых знаков. В общем случае это не лучевое число. Но операция нормализации к нему формально применима. Тогда

$$(y\cdot x^{\odot})^{\odot}=\mathbf{1} \qquad (7.5)$$

Доказательство. $C(y)\cdot C(x^{\odot})=1$. По теореме 7.2 $C((y\cdot x^{\odot})^{\odot})=1$. Но имеется единственное нормализованное лучевое число, комплексификация которого равна 1, это $\mathbf{1}$.

□

Замечание 7.1. Уравнение (7.2) определяет перевод комплексного числа $A+iB$ в лучевое $ray(A+iB)=x^{\odot}$. Это обозначение соответствующей операции.

Тогда по доказанному выше

$$ray(X+Y)=(ray(X)+ray(Y))^{\odot}; \qquad (7.6)$$

$$ray(X\cdot Y)=(ray(X)\cdot ray(Y))^{\odot};$$

$$C(x+y)=C(x)+C(y);$$

$$C(x\cdot y)=C(x)\cdot C(y);$$

Здесь заглавные буквы обозначают комплексные числа, а строчные, соответственно, лучевые числа.

□

8. Нормированные лучевые числа

Пусть x лучевое число с компонентами $(x_1,\ldots,x_n)$ в дистрибутивной и коммутативной алгебре. Тогда определим его нормировку по норме L_1 (см. теорему 5.2).

$$\mathrm{nrm}(x)=x/(\sum_{i=1}^{n}x_i). \qquad (8.1)$$

Если $y=\mathrm{nrm}(x)$, то

$$y_1+\ldots+y_n=1, \qquad (8.2)$$

$$y_1,\ldots,y_n>0. \qquad (8.3)$$

Это позволяет использовать такие лучевые числа как дискретное распределение вероятности на заданном числе элементарных событий. У этих чисел уже нет комплексной интерпретации, но их базовые единицы имеют смысл элементарных событий. Алгебра базовых единиц задает закон композиции этих событий: последовательная реализация пары элементарных событий приводит к одному из элементарных событий того же множества. Тогда, если имеются две нормированные модели x и y на одном множестве событий, то произведение xy задает распределение исходов независимых парных испытаний на этом множестве событий, а для нескольких нормированных чисел, произведение $x(1)\cdot x(2)\cdot\ldots\cdot x(T)$ задает распределение вероятности исходов в последовательных T-кратных испытаниях при том же законе парной композиции.

9. Сопряжения лучевых чисел

Сопряжения лучевого комплексного числа соответствуют перестановкам компонент или их линейным комбинациям. Исходное лучевое число

$$x=x_1\mathbf{1}+x_2\nu+x_3\mathbf{i}+x_4\mathbf{j}.$$

Различные сопряжения перестановок компонент этого числа.

$$x^{(1,2)}=x_2\mathbf{1}+x_1\nu+x_3\mathbf{i}+x_4\mathbf{j}; \qquad (9.1)$$

$$x^{(3,4)}=x_1\mathbf{1}+x_2\nu+x_4\mathbf{i}+x_3\mathbf{j}; \qquad (9.2)$$

$$x^{(1,2|3,4)}=x_1\mathbf{1}+x_2\nu+x_3\mathbf{i}+x_4\mathbf{j}; \qquad (9.3)$$

$$x^{(1,3|2,4)}=x_3\mathbf{1}+x_4\nu+x_1\mathbf{i}+x_2\mathbf{j}; \qquad (9.4)$$

$$x^{(1,4|2,3)}=x_4\mathbf{1}+x_3\nu+x_2\mathbf{i}+x_1\mathbf{j}; \qquad (9.5)$$

Сопряжение $(3,4)$ соответствует комплексному сопряжению:

$$C(x^{(3,4)})=C^{*}(x);$$

$$\mathrm{Re}\,C(x^{(1,2)})=\mathrm{Re}\,C(x),\quad \mathrm{Im}\,C(x^{(1,2)})=-\mathrm{Im}\,C(x). \qquad (9.6)$$

Сопряжение $(1,2)$ соответствует смене знака действительной части:

$$\mathrm{Re}\,C(x^{(1,2)})=-\mathrm{Re}\,C(x),\quad \mathrm{Im}\,C(x^{(1,2)})=\mathrm{Im}\,C(x). \qquad (9,7)$$

Сопряжение $(1,2\,|\,3,4)$ соответствует комплексной инверсии:

$$C(x^{(1,3|2,4)})=-C(x);$$

$$\mathrm{Re}\,C(x^{(1,2)})=-\mathrm{Re}\,C(x),\quad \mathrm{Im}\,C(x^{(1,2)})=-\mathrm{Im}\,C(x). \qquad (9,8)$$

Сопряжение $(1,3\,|\,2,4)$ соответствует отражению комплексного числа от биссектрисы:

$$\mathrm{Re}\,C(x^{(1,3|2,4)})=\mathrm{Im}\,C(x),\quad \mathrm{Im}\,C(x^{(1,3|2,4)})=\mathrm{Re}\,C(x). \qquad (9,9)$$

Сопряжение $(1,4\,|\,2,3)$ соответствует отражению комплексного числа от биссектрисы с инверсией:

$$\mathrm{Re}\,C(x^{(1,3|2,4)}) = -\mathrm{Im}\,C(x), \quad \mathrm{Im}\,C(x^{(1,3|2,4)}) = -\mathrm{Re}\,C(x). \tag{9,10}$$

Модульное сопряжение лучевого числа:

$$x^{(M1,2)} = (\max\{x_1; x_2\} - \min\{x_1; x_2\})\mathbf{1} + 0\nu + x_3\mathbf{i} + x_4\mathbf{j} \tag{9.11}$$

.

Сопряжение $(M1,2)$ соответствует сопряжению по отрицательной единице ν:

$$C(x^{(M1,2)}) = \left|\mathrm{Re}\,C(x)\right| + i\,\mathrm{Im}\,C(x). \tag{9.12}$$

Модульное комплексное сопряжение

$$x^{(M1,2|3,4)} = (\max\{x_1; x_2\} - \min\{x_1; x_2\})\mathbf{1} + 0\nu + x_4\mathbf{i} + x_3\mathbf{j} \tag{9.13}$$

$$C(x^{(M1,2|3,4)}) = \left|\mathrm{Re}\,C(x)\right| - i\,\mathrm{Im}\,C(x). \tag{9.14}$$

Приложение 1

Усиление постулата Уиллера-Фейнмана.

Постулат У-Ф:

При передаче действия от одного события к другому в пространстве Минковского происходит обратная передача действия от второго события к первому. При такой передача действия интеграл по времени заменяется интегралом по мировой линии кванта действия. А при обратном действии дифференциал меняется на комплексно сопряженное значение. Общее действие равно сумме прямого и обратного действия [9].

Тогда при передаче действия внутри светового конуса дифференциал действительный, и оба интеграла имеют одинаковые дифференциалы. Поэтому действие удваивается. А при передачи действия вне светового конуса дифференциал чисто мнимый. Тогда обратное действие имеет дифференциал, инвертированный по отношению к прямому действию. И сумма интегралов равна нулю. Поэтому событие может эффективно действовать только внутри светового конуса.

Остаётся не ясным вопрос, почему действие передается только из прошлого в будущее внутри светового конуса.

Модификация постулата У-Ф.

При передаче действия от одного события к другому в пространстве Минковского происходит обратная передача действия от второго события к первому. При такой передача действия интеграл по времени заменяется интегралом по мировой линии кванта действия. А при обратном действии дифференциал меняется на модульно комплексно сопряженное значение. Общее действие равно сумме прямого и обратного действия.

Тогда вне светового конуса всё сохраняется как раньше, и действие обнуляется. Но внутри светового конуса при действии назад по времени дифференциал мировой линии отрицательный. Но при обратном действии дифференциал становится положительным. Тогда суммарное действие обнуляется. В такой модели эффективное действие возможно только в положительном направлении времени.

Математическое изложение этого эффекта.

Обозначим $E(t,x)$ энергию объекта, переносящего действие в четырёх измерениях $\langle t, x\rangle, t \in \mathbb{R}, x \in \mathbb{R}^3$; в метрике ${}^1\mathbb{R}^3$ пространства Минковского. Пусть действие производится по некоторой прямой линии L в этом пространстве, соединяющей точки A и B как события. Тогда действие $W(A,B)$ измеряется интегралом по этой линии с дифференциалом

$$ds = \left(dt^2 - dx_1^2 - dx_2^2 - dx_3^2\right)^{1/2} \tag{П1.1}$$

$$W(A,B) = \int_{(t,x)\in L} E(t,x)ds \tag{П1.2}$$

Если учитывать противодействие, то по постулату У-Ф

$$W(A,B,A) = \int_{(t,x)\in L} E(t,x)ds + \int_{(t,x)\in L} E(t,x)(ds)^* \tag{П1,3}$$

где звёздочка $*$ означает комплексное сопряжение числа.

Если использовать модифицированный постулат У-Ф

$$W(A,B,A) = \int_{(t,x)\in L} E(t,x)ds + \int_{(t,x)\in L} E(t,x)(ds)^{\otimes} \tag{П1.4}$$

где знак $\otimes$ означает модульное комплексное сопряжение.

Если точка B лежит вне светового конуса точки A, то $ds = \mathrm{Im}\,ds$, и поэтому

$$ds^* = ds^{\otimes} = -ds. \tag{П1.5}$$

Тогда в обеих моделях

$$W(A,B,A) = 0. \tag{П1.6}$$

Если точка B лежит в световом конусе точки A, и $t(A) < t(B)$, то $ds > 0$, и поэтому

$$ds^* = ds^{\otimes} = ds. \tag{П1.7}$$

Тогда в обеих моделях

$$W(A,B,A) = 2W(A,B). \tag{П1.8}$$

Это действие в общем случае не равно нулю, если только энергия $E(t,x)$ не меняет свой знак в данном процессе.

Различие между моделями наблюдается в случае, когда точка B лежит в световом ко-

нусе точки A, и $t(B)<t(A)$. Тогда $ds<0$ на пути от A к, B и поэтому

$$ds^{*}=ds;$$
$$ds^{\otimes}=-ds; \qquad (П1,9)$$

В модели У-Ф

$$W(A,B,A)=2W(A,B) \qquad (П1.10)$$

В модификации модели У-Ф

$$W(A,B,A)=0 \qquad (П1.11)$$

Таким образом, модель У-Ф объясняет невозможность эффективного макроскопического действия вне светового конуса события, которое является причиной действия (источником энергии). Это не позволяет обратить время, используя относительность направления времени у тахионов. Но внутри светового конуса время остаётся двунаправленным в этой модели.

В модифицированной модели У-Ф сохраняется невозможность эффективного действия за пределами светового конуса. Но в этой модели невозможно также эффективное действие внутри светового конуса, но назад по времени. Остаётся только возможность передавать энергию внутри конуса вперёд по времени. Таким образом, эта модель объясняет как ограничения на скорость света, так и стрелу времени.

Ray numbers and algebras

A.V. Koganov

Abstract. The paper considers a new approach to the theory of Hypercomplex numbers, in which negative numbers are considered as a complex extension of non-negative numbers. Such algebras, called ray algebras, have unexpected properties, and classical complex and Hypercomplex algebras are obtained from them by factorization. In particular, it will be shown that there are several different conjugation operations in ray algebras. Some of these operations can be used in mathematical and theoretical physics. The case of modification of the algebra of complex numbers is considered in detail.

Keywords: Hypercomplex numbers, complex unit, distributive algebras, light cone, time arrow.

Литература

1. И. Л. Кантор, Ф. С. Солодовников. Гиперкомплексные числа. М. «Наука», 1973

2. Д. Г. Павлов. Гиперкомплексные числа и связанные с ними пространства. textarchive.ru

3. Г. Глейзер Комплексные числа. (ч. 1,2) // Математика, 2001, №10.

4. Н. И. Яцкин. Алгебра. Теоремы и алгоритмы. Иваново. Изд. «Ивановский государственный университет», 2006.

6. А.И. Кострикин Введение в алгебру. Ч. 1. Основы алгебры. М., Физматгиз, 2000.

7. А.Г. Курош Курс высшей алгебры. М., Наука, 1973.

8. Э.Б. Винберг Курс алгебры. М. Факториал Пресс, 2001.

9. Г. Г. Михайличенко. Математический аппарат теории физических структур. Горно-Алтайский государственный Университет. 1997г., г. Горно-Алтайск, 144с.

10. Ю. И. Кулаков, Ю. С. Владимиров, А. В. Карнаухов. Введение в теорию физических структур и бинарную геометрофизику. М. «Архимед», 1992г., 184с.

11. В.Х. Лев Трёхмерные геометрии в теории физических структур. // Вычислительные системы, Новосибирск, ИМ СОАН СССР, 1988г., вып. 125, с. 90-103.

12. Р. Фейнман. Характер физических законов. АСТ, 2016, 256с.

13. Р. Фейнман. КЭД – странная теория света и вещества. АСТ, 2018, 208с.

14. А.В. Коганов Факторизация свободной группы с сопряжёнными операциями, изоморфная плотному подмножеству линейного пространства. // 25-я Международная конференция «Математика. Компьютер. Образование», Дубна, 29.01-03.02.2018, тезисы, , R&C Dynamics, Москва, Ижевск, 2018, с. 145.

15. А.В. Коганов Принцип контравариантной генерации событий в физике. // Метафизика. 2018, №1(27), РУДН, Москва, с. 129-134. (ISSN 2224-7580) (РИНЦ, Российская электронная библиотека.)

16. А.В. Коганов. Контравариантная генерация мультиграфа квантовой гравитации и относительность близкодействия. // 17-я Всероссийская гравитационная конференция. Международная конференция по гравитации, космологии и астрофизике (RUSGRAV-17), 24-30 июня 2017г. Материалы конференции. Калининград, 2017; с.80.

17. А.В. Коганов Контравариантная модель квантовой гравитации и относительность близкодействия. // 13-я международная конференция Финслеровы обобщения теории относительности. Материалы конференции. 7-12.09.2017, Муром, научный городок «Перемиловы горы», Россия, с. 66-68.