Федеральное государственное бюджетное учреждение науки
Научно-исследовательский институт системных исследований РАН

ТРУДЫ НИИСИ РАН

ТОМ 4 N 2

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ СЛОЖНЫХ СИСТЕМ:

ТЕОРЕТИЧЕСКИЕ И ПРИКЛАДНЫЕ АСПЕКТЫ

МОСКВА
2014

Тематика номера:
Математическое моделирование и визуализация, моделирование в микроэлектронике, информационные технологии, вопросы программирования и вычислительной математики

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

The topic of the issue:
Mathematical modeling and visualization, modeling in microelrctronics, information technologies, problems of programming and computational mathematics

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

Заведующий редакцией: Ю.Н. Штейников

Издатель: Российская академия наук, НИИСИ РАН,
117218, Москва, Нахимовский проспект 36, к. 1

СОДЕРЖАНИЕ

IV. ТЕОРЕТИЧЕСКИЕ ВОПРОСЫ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ

V. ИЗ ИСТОРИИ НАУКИ И ТЕХНИКИ

О возможности оптимизации некоторых функций библиотеки линейной алгебры с помощью векторного сопроцессора

А.А. Бурцев

кандидат физико-математических наук

Аннотация: В НИИСИ РАН в качестве расширения универсальных микропроцессоров семейства КОМДИВ разработан специализированный 128-разрядный сопроцессор, позволяющий ускорять вычисления над векторами комплексных и вещественных чисел одинарной и двойной точности. В статье представлены результаты применения этого сопроцессора, направленные на повышение скорости исполнения основных функций обработки векторов и матриц, обычно используемых для решения типовых задач линейной алгебры. Рассматриваются приёмы оптимизации кода с учётом особенностей сопроцессора, выявляются недостатки, препятствующие его эффективному применению, и предлагаются возможные пути их преодоления.

Ключевые слова: микропроцессоры семейства КОМДИВ, векторный сопроцессор, библиотека BLAS

Введение

Реалии современного мира диктуют острую потребность создания отечественных микропроцессоров, способных обеспечивать высокую производительность научно-технических и инженерных расчётов даже в экстремально жёстких условиях их эксплуатации.

Для удовлетворения этой потребности в НИИСИ РАН разрабатывается [1] семейство высокопроизводительных микропроцессоров КОМДИВ, в которых в качестве расширения базовой архитектуры предлагаются специализированные сопроцессоры, ориентированные (согласно принципу “встречной оптимизации”) на ускоренное исполнение заданного набора математических функций, наиболее часто употребляемых при решении задач определённой области применения.

Так, микропроцессор К64РИО (1890ВМ6) помимо обычного 64-разрядного сопроцессора (СР1) плавающей арифметики был дополнен 64-разрядным сопроцессором (СР2) комплексной арифметики, а микропроцессор К128РИО (1890ВМ7) был оснащён 128-разрядным сопроцессором (СР2) обработки сигналов.

В дальнейшем вычислительный блок сопроцессора комплексной арифметики был расширен до 128-разрядов и дополнен рядом полезных команд, позволяющих ускорить исполнение ряда типичных операций обработки векторов и матриц. В результате такого развития получился 128-разрядный сопроцессор (СР3), который можно охарактеризовать как векторный. Он, как предполагается, будет функционировать в составе новых высокопроизводительных микропроцессоров (1890ВМ8) семейства КОМДИВ.

Далее в статье предпринимается попытка оценить, какой выигрыш может дать применение этого сопроцессора для повышения скорости исполнения наиболее употребительных функций обработки векторов и матриц, используемых для решения типичных задач линейной алгебры. Данная оценка производится в отношении ряда характерных функций известной библиотеки линейной алгебры BLAS [2], которая часто используется в приложениях, служащих образцовыми примерами тестов компьютерной производительности.

1. Общая характеристика векторного сопроцессора

Векторный сопроцессор (CPV) содержит 64 128-битных регистра, в каждом из которых можно хранить:

- 1 комплексное число двойной точности (DC);
- 2 комплексных числа одинарной точности (SC);
- 2 вещественных числа двойной точности (DR);
- 4 вещественных числа одинарной точности (SR).

Для каждого из этих 4-х форматов значений, помещённых в его регистры, CPV обеспечивает соответствующие ему вычислительные команды. Например, каждая из команд умножения с накоплением (см. таблицу 1) выполняет свойственную ей операцию для одной тройки (z,y,x) величин типа DC, либо двух троек типа SC или DR, либо сразу для четырёх троек типа SR.

Таблица 1. Вычислительные команды группы умножения

1 DC (9)*	2 SC (7)*	2 DR (5)*	4 SR (4)*	cmd z,y,x
cmul.d	**cmul.s**	**vmul.d**	**vmul.s**	z=y·x
cmadd.d	**cmadd.s**	**vmadd.d**	**vmadd.s**	z=z+y·x
cmsub.d	**cmsub.s**	**vmsub.d**	**vmsub.s**	z=z-y·x
cmaddsub.d	**cmaddsub.s**	**vmaddsub.d**	**vmaddsub.s**	z=z+y·x
(t)* - *длительность в тактах (если команда уже в кэше)*				y=z-y·x

Длительность вычислительных команд зависит от типа обрабатываемых величин (см. в скобках: 4 такта для SR, 5 для DR, 7 для SR и 9 для DC). Но поскольку CPV разрешает на каждом такте начинать исполнение новой команды вычислительного потока, то в итоге можно добиться такой максимальной производительности CPV, при которой n его вычислительных команд смогут исполниться всего за n+8 тактов.

Правда, для этого необходимо параллельно с вычислениями осуществлять ещё и своевременную подкачку обрабатываемых данных из памяти в регистры и обратно. Но такую возможность CPV предоставляет.

Команды CPV для работы с памятью (см. таблицу 2) позволяют загрузить 128-битное значение целиком в регистр или заполнить его старшую и младшую половину 64-битными значениями по отдельности. Можно одной командой (vldq) загрузить два соседних регистра двумя 128-битными смежными словами из памяти. Аналогичные команды (vsd, vsdm, vsdq) предусмотрены и для сохранения значений регистров в памяти.

Таблица 2. Команды работы с памятью

	команды загрузки	команды сохранения
256 /32 L2(5)*	**vldq**, **vldqx**	**vsdq**, **vsdqx**
64 /8 L1(3)*	**vld**, **vldh**, **vldx**, **vldhx**, **vldlx**	**vsd**,**vsdh**, **vsdx**, **vsdhx**
128 /16 L1(3)* *если данные уже в L1(L2)-кэше*	**vldm**, **vldd**, **vlddh**, **vldmx**, **vlddx**, **vlddhx**, **vlddlx**	**vsdm**, **vsdd**, **vsddh**, **vsdmx**, **vsddx**, **vsddhx**

Для указания виртуального адреса в этих командах можно использовать базовую адресацию с относительным смещением (vld) или базовую индексную (vldx). Получаемый адрес должен быть выровнен на границу обрабатываемого слова, т.е. кратен 8, 16 или 32 (см./d).

Заметим, что описываемые здесь команды CPV нацелены на ускоренное исполнение в 128-разрядном режиме и не обеспечивают проверки на возникновение каких-либо исключительных ситуаций.

Архитектурой КОМДИВ предусмотрена возможность в каждом такте взять на исполнение две очередные команды, если эти команды разных потоков. Это позволяет совместить во времени вычислительные команды над одной группой данных с командами загрузки/сохранения в/из памяти другой группы данных.

В CPV самыми продуктивными командами над вещественными числами являются команды перемножения матрицы 2×2 на вектор из 2-х чисел, которые предназначены ускорить одну из важнейших операций линейной алгебры – перемножение матриц. Одна такая команда перемножения с накоплением (см. **mvmadd** в таблице 3) выполняет 8 операций (4 умножения и 4 сложения) над вещественными числами двойной точности или 16 арифметических операций одинарной точности.

Таблица 3. Команда перемножения матрицы на вектор

M_1=(a,b) M_2=(c,d) V=(x,y)	**mvmadd.d** Z,M,V (9)* M *задаёт два соседних регистра* (M_1,M_2)	Z= Z+M×V : Z= Z+(a·x+b·y,c·x+d·y)
M_1=(a,b,c,d) M_2=(e,f,g,h) V=(x,y,z,u)	**mvmadd.s** Z,M,V (7)* M *задаёт два соседних регистра* (M_1,M_2)	Z= Z+ (a·x+b·y, e·x+f·y, c·z+d·u, g·z+h·u)

А самая “мощная” команда (**cmaddsub**), предусмотренная в CPV над комплексными числами, осуществляет 10 арифметических операций (4 умножения и 6 сложений) над вещественными числами двойной точности или 20 арифметических операций над вещественными числами одинарной точности. Одной такой командой реализуется базовая операция цифровой обработки сигналов, называемая “бабочкой Фурье”.

Таким образом, векторный сопроцессор позволяет достичь (на тактовой частоте 1 ГГц) пиковой производительности 16 ГФлопс на задачах перемножения матриц и 20 ГФлопс на задачах преобразования Фурье.

2. Краткая характеристика библиотеки BLAS

BLAS (Basic Linear Algebra Subprograms) – это библиотека подпрограмм (функций), которая создавалась, чтобы обеспечить универсальным интерфейсом (API) разработчиков прикладных программ, нацеленных на решение разнообразных задач линейной алгебры (например, таких, как получение LU-разложения матрицы, вычисление определителя матрицы, решение систем линейных уравнений и др.).

BLAS первоначально создавалась на языке Фортран, но впоследствии стала доступной и для разработчиков Си-программ. В Интернете свободно доступны несколько разновидностей этой библиотеки. Среди них особый интерес представляет GotoBLAS [3], код которой оптимизирован под различные машинные архитектуры. Предпринимаются попытки повысить производительность BLAS-функций за счёт распараллеливания на многоядерных процессорах [4] или с помощью векторного расширения (например, Intel AVX [5]).

Функции BLAS принято подразделять на 3 уровня. Номер уровня функции определяется порядком величины количества операций с плавающей запятой (а также количеством перемещений данных), которые потребуется исполнить в данной функции (см. таблицу 4).

Таблица 4. Классификация BLAS-функций

	операций/перемещений данных	типичные функции
1	O(N) / O(N)	**x**SCAL, **x**AXPY, **x**DOT
2	$O(N^2)$ / $O(N^2)$	**x**GEMV, **x**TRSV
3	$O(N^3)$/ $O(N^2)$	**x**GEMM, **x**TRSM
N - размер вектора или порядок матрицы, **x**=Z,C,D,S		

Названия функций приводятся здесь с буквой-префиксом **x** (Z,C,D,S), означающей, для какого типа величин (DC, SC, DR, SR) предназначена эта функция.

На каждом уровне выделим самые характерные функции, которые чаще всего употребляются в приложениях, использующих эту библиотеку (см. таблицу 5).

Таблица 5. Характерные BLAS-функции

	имя (параметры)	схематичное описание функции
1	**x**SCAL (n,α,X,ix)	$X=\alpha\cdot X$: { $X_i=\alpha\cdot X_i$, i=1…n }
1	**x**AXPY (n,α,X,ix,Y,iy)	$Y=\alpha\cdot X+Y$: { $Y_i=\alpha\cdot X_i+Y_i$, i=1..n }
1	**x**DOT (n,X,ix,Y,iy)	$dot=X^T\times Y$: { $dot=\sum(X_i\cdot Y_i)_{i=1..n}$ }
2	**x**GEMV (tz,m,n,α, Z,lz,X,ix,β,Y,iy)	$Y=\alpha\cdot Z^*_{m\times n}\times X+\beta\cdot Y$, $Z^*=Z\|Z^T\|Z^H$: { $Y_i=\beta\cdot Y_i+\alpha\cdot\sum(Z^*_{ik}\cdot Y_k)_{k=1..n}$, i=1..m }
2	**x**TRSV (ul,tz,d,n, Z,lz,Y,iy)	$Z\times X=Y$? → $Y_{n\times1}=Z^{*-1}{}_{n\times n}\times Y_{n\times1}$ $Z^{*-1}=(Z)^{-1}\|(Z^T)^{-1}\|(Z^H)^{-1}$
3	**x**GEMM (tx,ty,m,n,k, α,X,lx,Y,ly,β,Z,lz)	$Z_{m\times n}=\alpha\cdot X^*_{m\times k}\times Y^*_{k\times n}+\beta\cdot Z_{m\times n}$: { $Z_{ij}=\beta\cdot Z_{ij}+\alpha\cdot\sum(X^*_{is}\cdot Y^*_{sj})_{s=1..n}$ }, i=1..m, $X^*=X\|X^T\|X^H$, $Y^*=Y\|Y^T\|Y^H$ j=1..n
3	**x**TRSM (s,ul,tz,d,m,n, α,Z,lz,Y,iy)	$Z\times X=\alpha\cdot Y$? → $Y_{m\times n}=\alpha\cdot Z^{*-1}{}_{m\times m}\times Y_{m\times n}$ $X\times Z=\alpha\cdot Y$? → $Y_{m\times n}=\alpha\cdot Y_{m\times n}\times Z^{*-1}{}_{n\times n}$ $Z^{*-1}=(Z)^{-1}\|(Z^T)^{-1}\|(Z^H)^{-1}$

Далее проанализируем, какой выигрыш в повышении производительности этих функций может обеспечить применение векторного сопроцессора CPV.

3. BLAS-функции 1-го уровня

Каким бы совершенным не был имеющийся аппаратный исполнитель, эффективность реализации требуемой функции, в конечном счёте, во многом определяется совершенством её программного кода. Поэтому анализ возможного выигрыша в производительности, который может обеспечить векторный сопроцессор (CPV), будем сопровождать обсуждением известных приёмов оптимизации кода, которые дают эффект при построении программы с учётом как общих принципов архитектуры КОМДИВ, так и отдельных особенностей сопроцессора CPV, в частности.

3.1. DSCAL

Начнём такой анализ с функций, обрабатывающих одиночный вектор. А в качестве типичного их представителя рассмотрим функцию умножения вектора на скаляр: **x**SCAL(n,α,X,ix). Под вектором здесь и далее понимается ряд значений, расположенных в памяти с некоторым шагом (ix), начиная с заданного адреса (X). Заметим, что элементы вектора будут следовать в памяти рядом друг за другом только при шаге ix=1. Далее такой вектор будем называть непрерывным, чтобы отличать его от произвольного, для которого ix≠1. Примерами непрерывного вектора могут служить строки матрицы в Си-программе, а примерами произвольного – её столбцы. (Для Фортран-программы будет всё наоборот, т.к. компилятор расположит её в памяти уже не по строкам, а по столбцам). Заметим, что адрес i-го элемента (i=0,..,n-1) произвольного вектора потребуется вычислять по формуле: X+i×ix×sT, где sT – размер элементов (в байтах), равный sT=4,8,8,16 для значений типа SR,DR,SC,DC соответственно.

Основной цикл функции DSCAL (n,α,X,ix), составленной на языке Си в расчёте на обработку (на CP1) непрерывного вектора (ix=1) с элементами типа DR:

```
#define vt double
void DSCAL(int n, vt a, vt *X, int ix) {
 int i; /* в предположении, что ix=1 */
 for (i=0; i<n; i++) X[i]=a*X[i];
}//DSCAL
```

будет переведён (если не применять оптимизацию) в ассемблерный код примерно следующего вида:

Loop: ## вариант 1	назначение регистров:
Ldc1 FX,0(Xi)	FS = значению параметра **a** ;
Mul.d FX,FX,FS	FX = значению эл-та X[i] ;
Sdc1 FX,0(Xi)	I – регистр-счётчик цикла;
addiu Xi,Xi,sT;	Xi – регистр адреса эл-та X[i];
addiu I,I,1	N – регистр параметра **n** ;
bne I,N,Loop	sT = 8 (размер элемента в байтах)

Выделенные в этом цикле основные команды:

Ldc1 ; **M**ul.d ; **S**dc1	L-M----S-	такты: 2+5+2=9

загрузки (**L**), сохранения (**S**) и вычислительного действия (**M**) должны исполняться строго последовательно и потому на обработку каждого элемента потребуется **9** тактов (на CP1), а на обработку всего вектора из **n** элементов – **9·n** тактов соответственно.

Ускорить обработку вектора можно за счёт совмещения во времени вычислительных операций с операциями загрузки/сохранения. Для этого используются следующие приёмы оптимизации кода.

Приём №1. Во-первых, можно просто изменить порядок следования этих команд в цикле так, чтобы на i-ом шаге каждого цикла вычислительная операция для i-го элемента сопровождалась опережающей загрузкой (i+1)-го элемента и сохранением результата для (i-1)-го элемента:

Sdc1 F2,-sT(Xi)	i-1: L-M----**S**-
Mul.d F2,F1,FS	i: L-...**M**----S-
Ldc1 F1,sT(Xi)	i+1: **L**-...M----S-

Тогда эта тройка команд займёт всего 6 тактов. (Правда, в этом случае для корректной обработки первого и последнего элемента вектора потребуются дополнительные действия до и после цикла).

Приём №2. Во-вторых, можно предусмотреть в теле цикла две фазы: в 1-ой фазе каждого шага цикла вычислительную обработку элементов одной группы (A) сопровождать загрузкой и сохранением элементов другой группы (B), а во 2-ой фазе каждого шага цикла – наоборот:

Mul.d FA,FA,FS	потактовая диаграмма:
Sdc1 FB,0(Bi)	B_{i-1}: **S**-
Ldc1 FB,sX(Bi)	A_i : L-**M**----.**S**-
Mul.d FB,FB,FS	B_i : **L**-**M**----.S-
Sdc1 FA,0(Ai)	A_{i+1}: **L**-M----.S-
Ldc1 FA,sX(Ai)	|на i-ом шаге |

В этом варианте 6 команд цикла для обработки 2-х элементов займут 10 тактов, а для обработки вектора длиной **n** (n=2·p) потребуется **5·n** (10·p) тактов. Но это без учёта особой обработки элементов первой A-группы и последней B-группы, а также команд обеспечения цикла: I:=I+1; Ai:=Ai+sX; Bi:=Bi+sX; (sX=2·sT).

Приём №3. Допустим, разрешается очередную команду вычислительного потока начинать исполнять уже на следующем такте, не дожидаясь завершения предыдущей команды этого же потока. (CP1 это позволяет в особом режиме работы). Тогда в целях дальнейшей оптимизации кода можно организовать кратную обработку вектора группами по **k** элементов.

На каждом шаге цикла будем применять тройку основных команд (L,M,S) попеременно ко всем элементам одной группы (j=1,..,k), а **k** возьмём таким (=5), чтобы следующая команда этой тройки могла применяться к j-ому элементу группы уже без задержки:

Ldc1 F1,1*sT(Xi);...	j= | 16 тактов |
Ldc1 Fk,k*sT(Xi);	1: **L**-...**M**----**S**-
Mul.d F1,F1,FS; ...	2: **L**-...**M**----**S**-
Mul.d Fk,Fk,FS;	3: **L**-...**M**----**S**-
Sdc1 F1,1*sT(Xi);...	4: **L**-...**M**----**S**-
Sdc1 Fk,k*sT(Xi);	5: **L**-...**M**----**S**-
ad**D**iu Xi,Xi,k*sT	**D**

Этот вариант в каждом цикле обрабатывает группу из 5-ти элементов за 16 тактов, а вектор длиной **n** (n=5·p) – уже примерно за **n·16/5** (16·p) тактов соответственно (это если не учитывать команды организации цикла).

Приём №4. Архитектура КОМДИВ позволяет в каждом такте приступить к исполнению сразу двух команд разных потоков. Это позволяет получить выигрыш в тактах (почти в 2 раза), если удачно расставить вычислительные команды в коде так, чтобы они попеременно чередовались с командами работы с памятью.

Объединим кратную обработку вектора (приём №3) с двухфазным построением цикла (приём №2) так, чтобы вычислительные операции над k элементами одной группы (Ai, i=1..k) совместить с параллельным

исполнением операций загрузок и сохранений для k элементов другой группы (Bi, i=1..k), а затем наоборот.

Проиллюстрируем это диаграммой (см. таблицу 6). На каждом шаге цикла на обработку 10 элементов затрачивается 20 тактов (по 10 тактов на 16 команд каждой фазы), значит, для обработки вектора из **n** элементов в этом варианте потребуется **n·2** (20·p) тактов.

Такой вариант обеспечивает предельную производительность, которую можно "выжать" для функции DSCAL при обработке непрерывного вектора на CP1.

Таблица 6. Потактовая диаграмма (n=10·p, k=5)

```
    | 1-ая фаза   | 2-ая фаза  |
A1: M---------.S-    L-
A2:  M---------.S-    L-
A3:   M---------.S-    L-
A4:    M---------.S-    L-
A5:     M---------.S-    L-
                   D
B1: S-    L-    M----
B2:  S-    L-    M----
B3:   S-    L-    M----
B4:    S-    L-    M----
B5:     S-    L-    M----
        D
```

При обработке же произвольного вектора команду приращения адреса (ad**D**iu) приходится выполнять уже после каждой команды загрузки или сохранения элемента. Поэтому на каждом шаге цикла придётся исполнить 50 команд (по 5L+5D+5M+5S+5D=25 в каждой фазе). В лучшем случае на это уйдёт 25 тактов. Значит, на обработку произвольного вектора из **n** элементов (n=10·p) потребуется **2,5·n** (25p=25·n/10) тактов.

Проанализируем теперь, как можно поднять производительность этой функции с помощью векторного сопроцессора (CPV).

Пусть подлежащий обработке непрерывный вектор выровнен по адресу, кратному 32. Тогда можно выгодно использовать команды **vLdq** и **vSdq** для загрузки и сохранения сразу 4-х его элементов в 2 соседних регистра. Необходимые вычисления для такой четвёрки можно осуществить двумя командами **vMul.d**. Применяя уже знакомые приёмы кратной групповой двухфазной обработки для k=16 и n=32·p, получаем такой ассемблерный код для тела основного цикла, обрабатывающего 2 группы A и B по 16 элементов в каждой:

```
   Loop: # для группы A:        # для группы B:
1  vMul.d VAj,VAj,VS ;    vSdq VBj,2j*sT(Bi);
ф  ... для j=0,1,2,3      ... для j=0,2,4,6
а                         adDiu Bi,Bi,2*k*sT
з  vMul.d VAj,VAj,VS ;    vLdq VBj,2j*sT(Bi);
а  ... для j=4,5,6,7      ... для j=0,2,4,6
2  vSdq VAj,2j*sT(Ai) ;   vMul.d VBj,VBj,VS ;
ф  ... для j=0,2,4,6      ... для j=0,1,2,3
а  adDiu Ai,Ai,2*k*sT
з  vLdq VAj,2j*sT(Ai) ;   vMul.d VBj,VBj,VS ;
а  ... для j=0,2,4,6      ... для j=4,5,6,75
   addiu I,I,1 ;          bne   I,P,Loop
```

В нём в каждой фазе 8 команд v**M**ul.d совмещаются с 4 командами v**S**dq и 4 командами v**L**dq, так что на все эти 32 команды будет затрачиваться 32/2=16 тактов. Значит, для обработки вектора из **n** элементов потребуется 16·p, т.е. примерно **n/2** тактов. (Это опять же без учёта команд организации цикла).

Таким образом, применение CPV позволяет ускорить почти **в 4 раза** (n/2 вместо 2n тактов) исполнение функции DSCAL для непрерывного вектора из DR-элементов. Но такой выигрыш, к сожалению, не получается получить при обработке произвольного вектора. Попробуем разобраться, почему.

Всё дело в том, что элементы произвольного вектора не располагаются в памяти друг за другом, поэтому при реализации основного цикла функции DSCAL:

```
for (i=0,I=0;i<n;i++,I+=ix) X[I]=a*X[I];
```

на загрузку 4-х элементов в два регистра (V0,V1) вместо одной команды (vldqx) теперь требуется 8 команд:

```
vLdhx V0,I(X); adDu I,I,sX   ## если бы ix=1
vLdlx V0,I(X); adDu I,I,sX    vLdqX V0,I(X)
vLdhx V1,I(X); adDu I,I,sX    addu I,I,4*sT
vLdlx V1,I(X); adDu I,I,sX   ## sX=ix·sT
```

Аналогичная серия команд (v**S**d**h**x, v**S**d**l**x) потребуется и вместо одной команды сохранения (vsdqx).

При кратной обработке вектора группами по 16 элементов на 8 вычислительных команд (v**M**ul.d) теперь приходится потратить 32 команды для загрузки и 32 для сохранения элементов. Соотношение этих команд при обработке группы непрерывного вектора было **8:9** (8M|4L+4S+1D), а для произвольного получается **8:64** (8M|16L+16S+32D). Даже если эти 72 команды удастся удачно распределить попарно (чтобы каждый такт захватывать на исполнение 2 команды), то на обработку одной группы уйдет 36 тактов, а на вектор длиной **n**=16·p потребуется **2,25·n** (36·n/16) тактов.

Таким образом, функцию DSCAL, обрабатывающую произвольный вектор из DR-элементов, удастся ускорить на CPV всего лишь **на 11%** (2,5n/2,25n=1,11).

Такой удручающий результат сначала может показаться странным. В самом деле, ведь команда v**M**ul.d CPV позволяет (за те же 5 тактов) исполнить сразу две вычислительные операции вместо такой же одной (умножение вещественных двойной точности), которую исполняет команда **M**ul.d CP1. Следовательно, можно было бы рассчитывать на то, что CPV обеспечит ускорение функции DSCAL хотя бы в 2 раза. Но такой расчёт оказывается ошибочен, ибо он не учитывает, что ускорение сам**о**й вычислительной команды, действующей над регистрами, не приведёт к значительному подъёму производительности всей функции, если оно не будет поддержано ускоренной работой команд, обеспечивающих доставку обрабатываемых значений из памяти в регистры и обратно.

3.2. SSCAL

Отсутствие команд, обеспечивающих быструю загрузку и требуемую упаковку в регистры сразу нескольких элементов произвольного вектора, а также обратную распаковку и сохранение их в памяти, ещё более значительно сказывается на замедлении производительности функции SSCAL при обработке значений типа SR. Используемая для её реализации на CPV основная вычислительная команда v**M**ul.s позволяет заменить 4 аналогичные команды **M**ul.s CP1. Казалось бы, можно рассчитывать, что функция SSCAL станет работать на CPV в 4 раза быстрее, чем на CP1. Но оказывается, что на такой выигрыш можно рассчитывать только лишь при обработке непрерывного вектора.

При обработке же произвольного вектора вариант реализации этой функции на CPV даже уступает в скорости исполнения варианту её реализации на CP1. Рассмотрим подробнее, почему это происходит.

Заметим, что CPV не предоставляет команд для загрузки в свои регистры (и сохранения из них) одиночных 32-битных значений. Поэтому SR-элементы произвольного вектора приходится переправлять из памяти в регистры CPV (и обратно) отдельными 64-битными частями через регистры CP1, используя особые команды для обмена с ними (см. таблицу 7).

Таблица 7. Команды обмена между регистрами CPV и CP1

чтение из регистра CP1 в регистр CPV (3)*	**vmthfp** FR,VR	VR.Hi:=FR
	vmtlfp FR,VR	VR.Lo:=FR
запись в регистр CP1 из регистра CPV (3)*	**vmfhfp** FR,VR	FR:=VR.Hi
	vmflfp FR,VR	FR:=VR.Lo

Одной командой vldq можно загрузить в два регистра (V0,V1) сразу 8 SR-элементов непрерывного вектора. Но для загрузки тех же 8-ми SR-элементов произвольного вектора потребуется аж 24 команды, т.к. на загрузку 4-х SR-элементов в каждый регистр Vi (i=0,1) приходится тратить по 12 команд:

```
Lw   R,0(X); adDu X,X,sX ## R:=X[0]
Lwc1 F,0(X); adDu X,X,sX ## F.Lo:=X[1]
mthc1 R,F  ;             ## F.Hi:=X[0]
vmthfp F,Vi;          ## Vi.Hi:=X[0]|X[1]
Lw   R,0(X); adDu X,X,sX ## R:=X[2]
Lwc1 F,0(X); adDu X,X,sX ## F.Lo:=X[3]
mthc1 R,F  ;             ## F.Hi:=X[2]
vmtlfp F,Vi;          ## Vi.Lo:=X[2]|X[3]
```

Аналогичная серия из 24-х команд потребуется и вместо одной команды сохранения (vsdq).

Значит, при кратной обработке произвольного вектора группами по 32 элемента на 8 вычислительных команд v**M**ul.s приходится уже 96 команд для загрузки и 96 команд для сохранения элементов. В результате доля самих вычислительных команд по отношению к командам, обеспечивающим подкачку данных, стремительно падает. Их соотношение для произвольного вектора теперь **8:192** (хотя для непрерывного вектора оно сохраняется прежним **8:9**). Причём из этих 200 команд 128 одного и того же потока работы с памятью, и потому на их исполнение уйдёт не менее 128 тактов. Тогда на обработку SR-вектора длиной **n**=32·p потребуется как минимум **4·n** (128·n/32) тактов.

При реализации же функции SSCAL на CP1 обработать группу из 8 SR-элементов можно за 40 команд: (8×**M**ul.s+8×**L**wc1+8×**S**wc1+16×ad**D**u), которые в оптимальном варианте можно исполнить за 20 тактов. Так что обработка вектора длиной **n**=8·p элементов займёт всего **2,5·n** (20·n/8) тактов. Вот почему реализация функции SSCAL(n,α,X,ix) на CPV для произвольного вектора (ix≠1) оказывается невыгодной (4n>2,5n).

3.3. CSCAL и ZSCAL

Векторный сопроцессор CPV обеспечивает солидное преимущество (над CP1) при реализации функций обработки векторов комплексных величин, поскольку предоставляет обширный набор команд комплексной арифметики. Ведь каждая из них сразу исполняет операцию, которую на сопроцессоре CP1 приходится реализовывать серией из нескольких команд вещественной арифметики. Например:

CPV	эквивалентная реализация на CP1	
cmul.d	= 2×mul.d + madd.d + msub.d	4
cmul.s	= (2×mul.s + madd.s + msub.s) ×2	8
cmadd.d	= 3×madd.d + msub.d	4
cmadd.s	= (3×madd.s + msub.s) ×2	8

Чтобы получить более эффективную реализацию на CPV функций CSCAL и ZSCAL, можно применить все рассмотренные ранее приёмы оптимизации.

Полученные оценки производительности представим в виде итоговой таблицы (см. таблицу 8).

Таблица 8. Оценки производительности функции SCAL

ix≠1	Load Xi	Xi=a·Xi	Store Xi	M:L+S+D=T/k[1)]	O(**n**)[2)]
1×DC	v**L**dm+**D**	c**M**ul.d	v**S**dm+**D**	1:1+1+2=3	**3·n**
2×SC	2×**L**+2×**D**	c**M**ul.s	2×**S**+2×**D**	1:2+2+4=36/16	**2,25·n**
2×DR	2×**L**+2×**D**	v**M**ul.d	2×**S**+2×**D**	1:2+2+4=36/16	**2,25·n**
4×SR	8×**L**+4×**D**	v**M**ul.s	8×**S**+4×**D**	1:8+8+8=128/32	**4·n**
ix=1	v**L**dq	2×**M**	v**S**dq	8:4+4+1=9/8t[3)]	**≈ n/t**

[1)] M:L+S+D=T/k – форма записи, выражающая отношение числа вычислительных команд (M) к командам загрузки (L), сохранения (S) и приращения адреса (D), а также затраченное на них кол-во тактов (T) при условии обработки вектора группами по k эл-тов;
[2)] оценка общего кол-ва тактов для вектора из **n** элементов;
[3)] **t** – кол-во чисел в регистре: **t** =1,2,2,4 для DC,SC,DR,SR;

Благодаря отмеченному преимуществу CPV всегда будет обеспечивать весьма значительный выигрыш в производительности для любых функций обработки векторов и матриц комплексных чисел. Поэтому дальнейший анализ применимости CPV сосредоточим в большей степени на функциях обработки вещественных векторов.

3.4. AXPY

Рассмотрим теперь, как изменится оценка применимости CPV в отношении функции, которой требуется использовать не один, а два вектора. Возьмём для примера функцию сложения двух векторов AXPY. Основной цикл функции **x**AXPY (n,α,X,ix,Y,iy):

```
for (i=0,IY=0,IX=0; i<n; i++)
{ Y[IY]=a*Y[IY]+X[IX]; IY+=iy; IX+=ix; }
```

требует теперь в 2 раза больше команд загрузки:

```
Load Xi; | Load Yi; |  Yi=a·Yi+Xi;  | Store Yi
```

Вследствие чего изменяется баланс вычислительных команд (ВК) и команд работы с памятью (КП). А поскольку доля ВК становится меньше, то и выигрыш от использования CPV уменьшается (см. табл.9).

Таблица 9. Оценки производительности функции AXPY

ix≠1	Load Xi,Yi	Yi=a·Yi+Xi	Store Yi	M:L+S+D=T/k	O(**n**)
1×DC	(v**L**dm+**D**)×2	c**M**add.d	v**S**dm+**D**	1:2+1+3=4	**4·n**
2×SC	(2×**L**+2×**D**)×2	c**M**add.s	2×**S**+2×**D**	1:4+2+6=7	**3,5·n**
2×DR	(2×**L**+2×**D**)×2	v**M**add.d	2×**S**+2×**D**	1:4+2+6=7	**3,5·n**
4×SR	(8×**L**+4×**D**)×2	v**M**add.s	8×**S**+4×**D**	1:16+8+12=24	**6·n**
ix=1	v**L**dq×2	2×**M**	v**S**dq	8:8+4+1=12/8t	**3n/2t**

3.5. DOT

Применение CPV эффективно лишь при обработке векторов большими группами элементов. При этом количество (**k**) элементов в обрабатываемой группе следует выбирать таким, чтобы основную вычислительную команду можно было бы выполнять на каж-

дом такте без каких-либо задержек. Если такая команда длится **h** тактов, а **k** взять меньшим (**k**<**h**), то будут возникать задержки в начале каждого цикла обработки очередной группы (на h-k тактов). Поскольку **k** определяет кратность длины (**n**=**k**·p) вектора, подлежащего обработке, то для упрощения вычислений (индексов и адресов элементов) значение **k** обычно выбирают равным ближайшей степени двойки (**k**= 8, 16, 32).

Разбивка векторов на группы может быть произвольной, если элементы векторов можно обрабатывать в любом порядке независимо друг от друга, как это, например, допускалось функциями SCAL и AXPY. Но такое возможно не всегда.

Некоторые функции могут потребовать учитывать зависимости между элементами или даже навязывать строго определённый порядок обработки элементов. В таком случае групповая обработка вектора может оказаться затруднительной или вообще невозможной.

В ряде случаев, чтобы добиться эффективной реализации функции на CPV, имеющиеся зависимости между элементами надо лишь разумно учитывать для получения удачного разбиения вектора на группы. Продемонстрируем это на примере функции вычисления скалярного произведения двух векторов **x**DOT.

Если в основном цикле **x**DOT (n,X,ix,Y,iy) :

```
for (i=0,S=0,IY=0,IX=0; i<n; i++)
{ S=S+Y[IY]*X[IX]; IY+=iy; IX+=ix; }
```

использовать команду умножения с накоплением результата в одном и том же регистре (c**M**add.d S,Xi,Yi), то следующую такую же команду можно будет исполнять не сразу, а лишь после завершения предыдущей.

Чтобы избежать возможных задержек и добиться эффективной реализации **x**DOT на CPV, будем обрабатывать за каждый шаг цикла такую группу элементов, каждая пара (Xj,Yj) которых накапливает своё произведение в отдельном регистре Sj (c**M**add.d Sj,Xj,Yj). А итоговую сумму S посчитаем после завершения основного цикла, просуммировав накопленные в регистрах Sj (j=1,..,k) произведения.

Заметим, что при обработке элементов типа SC или DR в 2-х частях (старшей и младшей) каждого регистра Sj будет накапливаться отдельная сумма. А при обработке SR-элементов таких отдельно накапливаемых сумм будет уже 4. Впоследствии, конечно, потребуется слить все эти суммы в одну. Для этого в CPV предусмотрена особая команда (VSUM), позволяющая суммировать числа, содержащиеся в одном регистре.

Для функции DOT характерен такой же баланс ВК:КП, как и для функции SCAL. И поскольку при обработке векторов большого размера (т.е. с возрастанием **n**) доля итоговых команд суммирования сокращается, то производительность функции DOT будет приближаться к той, что характерна для функции SCAL.

4. BLAS-функции 2-го уровня

Функции этого уровня предполагают такой вид обработки, в котором наряду с векторами обязательно используется и одна матрица. Квадратичный порядок $O(n^2)$ обработки её элементов и определяет, главным образом, сложность реализации самой функции.

Анализ функций 1-го уровня уже показал, что применение CPV далеко не во всех случаях позволяет поднять их производительность. Чтобы понять, для каких функций 2-го уровня (и в каких случаях) можно улучшить реализацию, если применить CPV, а для каких это сделать не удастся и почему, рассмотрим два типичных примера.

4.1. TRSV

Начнём с функции **x**TRSV (ul,tz,d,n,Z,lz,Y,iY). И рассмотрим самый простейший случай её применения, когда требуется получить вектор X[n], являющийся результатом решения системы линейных уравнений:

$$
\begin{array}{lcl}
Z_{11}*X_1 & = & Y_1 \\
Z_{21}*X_1+Z_{22}*X_2 & = & Y_2 \\
Z_{31}*X_1+Z_{32}*X_2+\ Z_{33}*X_3 & = & Y_3 \\
\quad\quad\quad\quad\quad\quad\quad\ldots & & \\
Z_{n1}*X_1+Z_{n2}*X_2+\ Z_{n3}*X_3+\ \ldots\ Z_{nn}*X_n & = & Y_n
\end{array}
$$

При вызове функции предполагается, что коэффициенты Z_{ij} заданы нижней треугольной матрицей Z[n][n]. А результат (вектор X) требуется записать на место вектора Y[n], первоначально задающего правую часть.

Алгоритм получения этого результата основывается на так называемом обратном ходе метода Гаусса, который предполагает, что элементы вектора должны вычисляться строго последовательно друг за другом, так как перед вычислением очередного элемента X_i необходимо получить все предыдущие X_j (j=1,..,i-1):

$X_i = (Y_i - \sum(Z_{ij}\cdot X_j)_{j=1,..,i-1}) / Z_{ii}$ (для i=1,..,n)

Это обстоятельство не позволяет применить групповую обработку при вычислении элементов вектора X, а значит, применение CPV в этом случае вряд ли поможет обеспечить повышение производительности.

4.2. GEMV

Основная нагрузка при реализации функции GEMV(tz,m,n,α,Z,lz,X,ix,β,Y,iy) приходится на осуществление операции умножения матрицы $Z_{m\times n}$ на вектор X_n. Такое умножение, как известно, можно реализовать путём вычислений отдельных скалярных произведений каждой строки матрицы на вектор X, т.е. многократным применением функции DOT. Но при такой реализации выигрыш от применения CPV не превысит тот, что позволяет функция DOT.

Более того, в самом неблагоприятном случае, когда в операции умножения должна использоваться транспонированная матрица Z, придётся применять функцию DOT для произвольных векторов (столбцов матрицы и вектора X). И тогда вместо ожидаемого выигрыша в производительности можно даже будет получить и проигрыш (причина подобного проигрыша была подробно проанализирована в п.3.2).

Однако, CPV допускает и другой способ реализации функции GEMV, предполагающий использование специальных команд mvmadd (см. табл.3). Каждая команда mvmadd.d позволяет умножить матрицу 2×2 на вектор из 2-х элементов, позволяя заменить 2 команды vmadd.d, используемые для накопления скалярных произведений (в функции DDOT). Значит, используя её, можно рассчитывать на подъём производительности функции DGEMV.

Поясним схематично алгоритм эффективной реализации DGEMV с применением этой команды для матрицы Z[m][n] и векторов X[n], Y[m] кратных размеров m=16·p, n=2·q (см. таблицу 10).

Вектор Y вычисляем группами по 16 элементов. В начале обработки каждой группы загружаем очередные 16 элементов вектора Y в 8 регистров CPV VYj (j=0,1,..,7) и сразу умножаем их на скаляр β. Далее выполняем умножение матричного блока из следующих 16-ти строк матрицы Z на вектор X. Это умножение (с накоплением в регистрах VYj) выполняем поэтапно.

Таблица 10. Схема алгоритма DGEMV

```
for (r=0; r<m; r+=16) {
   for j=0..7 { load VYj ← ( β·Y[r+2j], β·Y[r+2j+1] ) }
   for (s=0; s<n; s+=2) {
      load VX ← { α·X[s], α·X[s+1] }
      for j=0..7 {
         load VAj ← { Z[r+2j][s], Z[r+2j][s+1] }
         load VBj ← { Z[r+2j+1][s], Z[r+2j+1][s+1] }
         mvmadd.d VYj,VAj,VX
      }
   }
   for j=0..7 { store VYj → ( Y[r+2j], Y[r+2j+1] ) }
}
```

Используются регистры CPV: **VY0**, …,**VY7, VX**
и соседние пары регистров: **(VA0,VB0)**,…,**(VA7,VB7)**

На каждом этапе сначала загружаем очередные два элемента вектора X в регистр VX и тут же домножаем их на скаляр α. Теперь из каждой j-ой пары строк матрицы очередные 2 элемента загружаем в соседние регистры ZAj, ZBj, образуя в них матрицу 2×2, которую и умножаем на регистр VX с добавлением к регистру VYj. Завершая обработку группы, сохраняем из регистров VYj (j=0,1,..,7) новые значения вычисленных 16-ти элементов вектора Y.

Во внутреннем цикле этого алгоритма на 16 команд загрузки теперь приходится 8 команд mvmadd.d вместо 16 команд vmadd.d, т.е. улучшился баланс ВК:КП= 8M:16L+16D (вместо 16M:16L+16D). Это может дать выигрыш в тактах (20:24) исполнения внутреннего цикла и обеспечить в итоге общий подъём производительности примерно на 20% (24/20).

Хотелось бы, конечно, за счёт применения команд mvmadd получить выигрыш в 2 раза. Возможно ли это? Помогут ли тут команды быстрой загрузки vldq, vldqx? Увы, нет. Всё дело в том, что эти команды позволяют загрузить 4 элемента одной строки матрицы в соседние регистры, а для применения команды mvmadd в этом алгоритме требуется, чтобы в соседние регистры попадали 2 элемента одной строки и 2 элемента следующей строки. Поэтому загрузку элементов матрицы приходится выполнять командами vldmx. Это-то и не позволяет получить такой желательный баланс ВК:КП, чтобы доля ВК в нём оказывалась определяющей.

5. BLAS-функции 3-го уровня

Для функций 3-го уровня характерно значительное превышение доли вычислительных операций над операциями обмена с памятью: $O(n^3)$ против $O(n^2)$. В силу этого они становятся своеобразным полигоном, т.е. той привлекательной сферой применения, где CPV может проявить свои лучшие качества.

5.1. GEMM

Центральное место в этой области применения, конечно же, занимает функция **x**GEMM, которая обеспечивает операцию универсального матричного перемножения с накоплением: $Z_{m\times n}=\alpha\cdot X_{m\times k}\times Y_{k\times n}+\beta\cdot Z_{m\times n}$.

В общем случае вызов этой функции сопровождается обширным списком параметров:

xGEMM (tx,ty, m,n,k, α,X,lx, Y,ly, β,Z,lz)

которые позволяют задавать каждую из перемножаемых матриц (X,Y) не только в обычном, но и в транспонированном виде (см. tx,ty) или сопряжённом (для комплексных), а также допускают, что в этой операции могут участвовать не только матрицы целиком, но и их отдельные прямоугольные части, называемые матричными блоками. В таком случае дополнительными параметрами (lx,ly,lz) уточняются размеры строк, с помощью которых были объявлены матрицы в программе: X[][lx], Y[][ly], Z[][lz] и которые определяют, с какими интервалами располагаются строки обрабатываемых матриц в памяти.

В случае обычных матриц операцию GEMM можно выразить таким широко известным классическим алгоритмом:

```
for (i=0; i<m; i++) {
  for (j=0; j<n; j++) { S=0;
    for (t=0; t<k; t++)
      S=S+X[i][t]*Y[t]*[j];
    Z[i][j]= β*Z[i][j]+ α*S;
}}
```

Однако, такая реализации операции GEMM оказывается неэффективной для обработки матриц больших размеров на современных машинах, обладающих быстродействующей кэш-памятью.

Поэтому на практике применяют другие варианты реализации GEMM [6], которые учитывают иерархическую структуру памяти и позволяют существенно сократить количество вынужденных операций обмена между медленной и быстрой памятью, обязательно возникающих при обработке больших матриц.

Для эффективной реализации GEMM на CPV был создан оригинальный алгоритм. В нём не только применялись известные приёмы поэтапного исполнения операции матричного перемножения с расщеплением обрабатываемых матриц на части: матричные панели (полосы) и блоки. Но и учитывались важнейшие особенности самого CPV.

В первую очередь, был сделан упор на максимальное использование уже опробованной нами высокопроизводительной команды mvmadd. Кроме того, для сокращения общего числа загрузок элементов матриц в регистры CPV было принято решение использовать часть регистров CPV в качестве своеобразной надстройки (0-го уровня) над имеющейся кэш-памятью, чтобы хранить в них наиболее часто используемые блоки перемножаемых матриц.

Продемонстрируем этот алгоритм на примере реализации функции DGEMV для обычных (не транспонированных) матриц Z[m][n], X[n][k], Y[k][m] кратных

размеров: m=4·g, n=8·f, k=8·h. Для этого разобьём заданные матрицы на блоки небольшого размера:

Z11	...	Z1j	...	Z1f
...	...	...	...	...
Zi1	...	**Zij**	...	Zif
...	...	...	...	...
Zg1	...	Zgj	...	Zgf

+=

X11	...	X1p	...	X1h
...	...	...	...	...
Xi1	...	**Xip**	...	Xih
...	...	...	...	...
Xg1	...	Xgp	...	Xgh

×

Y11	...	Y1j	...	Y1f
...	...	...	...	...
Yp1	...	**Ypj**	...	Yif
...	...	...	...	...
Yh1	...	Yhj	...	Yhf

$Z=\{Zij_{4\times8}\}$, $X=\{Xip_{4\times8}\}$, $Y=\{Ypj_{8\times8}\}$, i=1..g; j=1..f; p=1..h.

И выразим основной алгоритм DGEMM (см. таблицу 11) в виде 3-х вложенных циклов, многократно выполняющих операции матричного умножения с накоплением над малыми блоками ($Zij_{4\times8}$, $Xip_{4\times8}$, $Ypj_{8\times8}$), из которых складываются исходные матрицы. Размеры этих блоков подбираются такими, чтобы при исполнении внутреннего цикла (по i) все участвующие в нём блоки могли уместиться в кэш-памяти.

Таблица 11. Общая схема алгоритма DGEMM

Требуется получить: $Z_{m\times n}=\alpha\cdot X_{m\times k}\times Y_{k\times n}+\beta\cdot Z_{m\times n}$

$Z_{m\times n}$= β·$Z_{m\times n}$; {умножение всех элементов матрицы на **β** }
for p= 1..h {
  for j= 1..f {
  for i= 1..g { **$Zij_{4\times8}$= $Zij_{4\times8}$ + α·$Xip_{4\times8}$×$Ypj_{8\times8}$;** } /*!A!*/
}}

Заметим также, что для умножения матрицы Z на скаляр β (в самом начале DGEMM), можно применять оптимизированный на CPV аналог функции SCAL.

Эффективность исполнения внутреннего цикла (по i), в конечном счёте, является определяющим звеном общей производительности функции DGEMM. Поэтому данному участку алгоритма (выделенному с пометкой !A!) уделим самое пристальное внимание. Рассмотрим детально, как его можно оптимально реализовать на CPV (см. таблицу 12).

По сути этот цикл реализует перемножение целой полосы (p-ой панели) матрицы X (Xip,i=1..g) на один малый блок Ypj матрицы Y. Каждую r-ую строку этой полосы надо умножить на матричный блок Ypj, и домножив образовавшийся вектор на скаляр α, добавить к r-ой строке соответствующей j-ой полосы матрицы Z.

Таблица 12. Схема умножения полосы на матричный блок

Требуется: for i= 1..g { **$Zij_{4\times8}$= $Zij_{4\times8}$ + α·$Xip_{4\times8}$×$Ypj_{8\times8}$;** }

for s=0..7 for q=0,2,4,6
  { load **VYqs** ←(α·**Ypj**[q][s], α **Ypj**[q+1][s]) **}**
for i= 1..g {
  for r=0..3 {
  for s=0,2,4,6 { load **VZrs** ← (Zij[r][s],Zij[r][s+1]) **}**
  for q=0,2,4,6 { load **VXrq** ← (Xip[r][q],Xip[r][q+1]) **}**
  }
  for q=0,2,4,6 {
  for r=0..3 {
  for s=0,2,4,6 mvmadd.d VZrs,VYqs,VXrq
  }}
  for r=0..3 for s=0,2,4,6
  { store **VZrs** → (Zij[r][s],Zij[r][s+1]) **}**
}// for g

распределение регистров CPV по парам: **(VYq0,VYq1)**,…,**(VYq6,VYq7), q=0,2,4,6** (для mvmadd.d) **(VXr0,VXr2)**,**(VXr4,VXr6), r=0..3** (для vldqx) **(VZr0,VZr2)**, **(VZr4,VZr6), r=0..3** (для vldqx)

Поскольку в этом цикле повсеместно используется матричный блок Ypj, то целесообразно перед таким циклом загрузить весь блок Ypj в регистры CPV, сразу умножив все его элементы на скаляр α и разместив их так, чтобы в дальнейшем было удобно сразу применять команду mvmadd к прочитанной в регистр паре соседних элементов очередной строки матрицы X.

Разместим элементы блока Ypj в 32-х регистрах CPV VYqs (q=0,2,4,6; s=0..7) так, чтобы s-ый столбец блока оказался в регистрах (VY0s,VY2s,VY4s,VY6s), и назначим регистрам такие номера, чтобы пары соседних регистров (VYq0,VYq1),…, (VYq6,VYq7) для q=0,2,4,6 могли представлять матрицы 2×2 в команде mvmadd.

На каждом i-ом шаге цикла загружаем 4 строки блока Zij в 16 регистров CPV VZrs (r=0..3, s=0,2,4,6), а 4 строки блока Xip – в 16 регистров VXrq (r=0..3, q=0,2,4,6). Для загрузки 8 элементов каждой строки в 4 регистра потребуется 2 команды vLdqx и одна команда (adDu) для приращения индексного регистра. Итого 2×4×(2L+1D)=16L+8D команд.

Далее исполняем основной вычислительный блок, состоящий из 3-х вложенных циклов. В нём к каждому регистру VZrs (r=0..3,s=0,2,4,6) требуется добавить 2-х элементный вектор, являющийся результатом операции умножения r-ой строки блока Xip на s-ую пару столбцов блока Yjp. Для исполнения этой операции потребуется поэтапно над регистрами, представляющими отдельные части r-ой строки и s-ой пары столбцов, выполнить команду: mvMadd.d VZrs,VYqs,VXrq 4 раза для q=0,2,4,6. Но такая команда исполняется 9 тактов. Поэтому для предотвращения ненужных задержек будем исполнять один этап такой операции умножения сразу для всей группы строк (r=0..3) и всех 4-х пар столбцов (s=0,2,4,6). Это займёт 16M команд на этап и 64M на весь вычислительный блок каждого i-го шага внутреннего цикла.

Наконец, вычисленные таким способом новые значения регистров VZrs (r=0..3, s=0,2,4,6) в конце i-го шага цикла сохраняем в памяти как обновлённый блок Zij матрицы Z. На это потребуется 8 команд vSdqx и 4 команды adDu, т.е. 8S+4D команд.

Итак, на каждом шаге внутреннего цикла (по i) выполняется операция умножения (с накоплением) для матричных блоков: $Zij_{4\times8}$= $Zij_{4\times8}$ + α·$Xip_{4\times8}$×$Ypj_{8\times8}$. На это затрачивается 64M+16L+8S+12D=100 команд. Причём доля самих команд mvMadd почти в 2 раза превышает долю команд, обеспечивающих подкачку данных, т.е. получается хороший баланс ВК:КП= 64:36=16:9.

Можно перестроить внутренний цикл по i на две фазы (см. приём №2) и совместить вычислительные команды над одной группой строк (r=0,1) блоков Zij и Xip с командами загрузки и сохранения для другой группы строк (r=2,3) этих блоков. Тогда исполнение каждого шага внутреннего цикла можно будет ужать до 70 тактов. (После 8-ми команд mvMadd каждого из 4-х этапов потребуется 1 такт задержки, да ещё 2 такта для команд организации самого цикла).

В результате представленный алгоритм, исполняя функцию DGEMM для матриц большого размера (но помещающихся целиком в кэше), будет обеспечивать высокую производительность, которую можно оценить как 91% (=64/70) от пиковой. Вот почему именно на функции GEMM можно получить максимальный выигрыш от применения CPV.

Доля ВК в рассмотренном алгоритме DGEMM столь высока, что позволяет сохранить благоприятный баланс ВК:КП даже в случае обработки транспонированных матриц. Подобный алгоритм обеспечивает достаточно высокую эффективность также и для функции SGEMM, т.е. в случае его применения для обработки элементов типа SR.

5.2. TRSM

Ещё одной часто используемой BLAS-функцией 3-го уровня является TRSM(s,ul,tz,d,m,n,α,Z,lz,Y,ly). Она предназначена для решения левостороннего (Z×X=α·Y) или правостороннего (X×Z=α·Y) матричного уравнения для всевозможных вариантов представления треугольной матрицы Z. Установкой параметров можно задать не только, с какой стороны (s) в этом уравнении участвует матрица Z, но и уточнить, является она нижней или верхней треугольной (ul), транспонированная ли она (tz) или сопряжённая (для комплексных), единичная ли у неё диагональ (d) или нет. Функция может применяться не только к матрицам целиком, но и к отдельным матричным блокам, поэтому при её вызове задаются также параметры (lz,ly), уточняющие размеры строк используемых матриц.

Функцию TRSM можно считать расширенным вариантом функции TRSV, ибо она решает систему линейных уравнений (с одной и той же матрицей коэффициентов) уже не для одного, а сразу для нескольких векторов неизвестных, собранных в единую матрицу в виде столбцов (для Z×X=α·Y) или строк (для X×Z=α·Y). Значит, при реализации функции TRSM можно применять описанный ранее (в 5.1) метод последовательного получения элементов искомого вектора сразу ко всей группе неизвестных векторов, объединённых в матрицу. Именно применение групповой обработки открывает возможность повысить производительность функции TRSM с помощью CPV.

Проиллюстрируем сказанное в виде схемы алгоритма функции DTRSM (см. таблицу 13) для левостороннего варианта с нижней треугольной матрицей Z[m][m] для кратного **n** (n=32·f). В нём искомая матрица (X) получается на месте заданной матрицы Y[m][n], а выигрыш от применения CPV обеспечивается путём вызовов реализованных на CPV функций DSCAL и DAXPY для обработки непрерывных векторов кратного размера. (Напомним, что в таком случае применение CPV позволяет повысить производительность этих функций почти в 4 раза).

Таблица 13. Схема алгоритма функции DTRSM

```
Y[m×n]= α·Y[m×n] ; {умножение всех элементов матрицы на α }
for (k=0; k<m; k++) {
   DSCAL(n,1.0/Z[k][k],&Y[k][0],1); // Y[k] ←Y[k]/Z[kk]
   for (i=k+1; i<m; i++)
     if (Z[i][k] != 0.0) // Y[i] ← Y[i] – Y[k]·Z[ik]
       DAXPY(n,-Z[i][k],&Y[k][0],1,&Y[i][0],1);
}// for k
```

К сожалению, аналогичным алгоритмом для правостороннего варианта подобный выигрыш обеспечить не удаётся, т.к. в нём функции DSCAL и DAXPY придётся применять уже не к строкам, а к столбцам матрицы Y, т.е. к произвольным векторам. А в этом случае существенного выигрыша в производительности этих функций CPV, увы, не позволяет обеспечить.

5.3. TRSM на основе GEMM

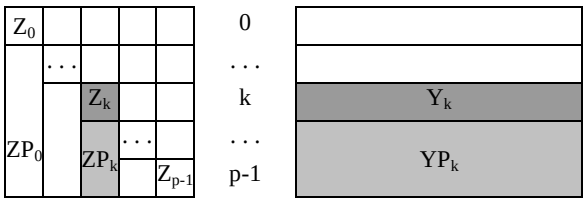
Функция GEMM является универсальной в том смысле, что многие другие функции обработки матриц можно выразить через неё. И если с помощью CPV удаётся добиться её высокой производительности, то тогда открывается возможность повысить и производительность этих других функций тоже.

В качестве примера рассмотрим, как можно обеспечить более эффективную реализацию функции DTRSM с помощью применения оптимизированной на CPV функции DGEMM. Продемонстрируем схематично алгоритм (см. таблицу 14) такой реализации для того же левостороннего варианта с нижней треугольной матрицей Z[m][m] при кратных **m** и **n** (m=w·p, n=32·f).

Таблица 14. Схема алгоритма DTRSM на основе DGEMM

```
Y[m×n]= α·Y[m×n] ; {умножение всех элементов матрицы на α }
for (k=0;k<p; k++) { Ik= k·w; Lk= m-(Ik+w);
  //решаем матричное уравнение: Z[k]×X[k]=Y[k] и X[k]→ Y[k]
   DTRSM(s,ul,tz,d,w,n,1.0,&Z[Ik][Ik],lZ,&Y[Ik][0],ly);
  // перемножаем с вычитанием YP[k]← YP[k] – ZP[k] × Y[k]
   if (Lk>0) DGEMM(tz,ty,Lk,n,w,
    -1.0,&Z[Ik+w][Ik],lz,&Y[Ik][0],ly,1.0,&Y[Ik+w][0],ly);
}// for k
```

Представим себе, что матрица Z состоит из p вертикальных панелей (полос) шириной по w элементов, а её диагональ выстроена как лестница из квадратных блоков размером по w элементов, каждый из которых представляет собой нижнюю треугольную матрицу:



Аналогично и матрицу Y представим состоящей из p горизонтальных полос размером по w элементов.

Тогда решение исходного матричного уравнения Z×X=α·Y можно получить за p этапов. На каждом k-ом (k=0,1,..,p-1) этапе сначала требуется решить матричное уравнение $Z_k \times X_k = Y_k$ для k-го малого диагонального блока Z_k и k-ой полосы Y_k. А затем следует подкорректировать значения элементов всех оставшихся ещё не обработанных полос матрицы Y (обозначенных как блок полос YP_k) путём вычитания из них результата матричного перемножения только что вычисленной в качестве решения полосы X_k на матричной блок коэффициентов ZP_k, лежащий в k-ой панели матрицы Z ниже диагонального блока Z_k : $YP_k \leftarrow YP_k - ZP_k \times X_k$.

Заметим, что в этом алгоритме применяется прежняя функция DTRSM (из п.5.2), чтобы получить решение матричного уравнения для блоков малых размеров. А бо́льшая часть вычислительной работы выпадает на высокопроизводительную функцию DGEMM, за счёт которой и удаётся поднять итоговую производительность функции DTRSM новой реализации.

6. Результаты тестов BLAS-функций

В представленных ниже таблицах (см. таблицы 15-17) содержатся показатели эффективности применения векторного сопроцессора (CPV) для некоторых функций обработки векторов и матриц библиотеки BLAS. Они получены экспериментальным путём в результате серии прогонов (на RTL-модели) тестов этих функций.

В полях этих таблиц содержится коэффициент выигрыша, который показывает, во сколько раз функция исполняется быстрее при её реализации на ассемблере с использованием команд CPV. Быстрее означает, что она выполняется за меньшее количество тактов, чем в случае реализации той же функции обычной (специально не оптимизированной) Си-программой, использующей сопроцессор вещественной арифметики (C_CP1).

Таблица 15. Выигрыш от применения CPV для AXPY

	n	DC	SC	DR	SR
AXPY ix=1	32	**4.03**	**10.15**	**3.17**	**4.18**
	256	**8.85**	**16.47**	**7.63**	**10.12**
	1024	**10.41**	**18.14**	**8.78**	**13.45**
AXPY ix≠1	32	**3.82**	**4.08**	**1.61**	**0.91**
	256	**2.83**	**3.05**	**1.51**	**0.93**
	1024	**2.87**	**2.48**	**1.45**	**0.97**

Таблица 16. Выигрыш от применения CPV для GEMV

	m×n	DC	SC	DR	SR
GEMV	32×32	**12.16**	**17.90**	**5.25**	**4.21**
	64×64	**13.03**	**21.99**	**3.87**	**3.82**
	256×256	**1.73**	**17.08**	**2.01**	**3.54**

Таблица 17. Выигрыш от применения CPV для GEMM

	m×n×k	DC	SC	DR	SR
GEMM	16×16×16	**9.28**	**13.69**	**6.42**	**8.70**
	32×32×32	**11.88**	**22.70**	**13.75**	**21.00**
	64×64×64	**14.67**	**31.77**	**34.06**	**30.14**
	128×128×128			**37.24**	**60.50**

В основном, эти показатели практически подтверждают приведённые ранее теоретические оценки.

Таблица 18. Сравнение DAXPY CPV с DAXPY GotoBLAS2

	n	C_CP1	Goto-BLAS2	CPV	C_CP1 / CPV	C_CP1 / Goto	Goto / CPV
ix=1	32	364	180	115	**3.17**	**2.02**	**1.57**
	256	2715	1234	356	**7.63**	**2.20**	**3.47**
	1024	11886	5942	1353	**8.78**	**2.00**	**4.39**
ix≠1	32	443	256	275	**1.61**	**1.73**	**0.93**
	256	3183	2124	2105	**1.51**	**1.50**	**1.01**
	1024	16503	9386	11404	**1.45**	**1.76**	**0.82**

Таблица 19. Сравнение DGEMV CPV с DGEMV GotoBLAS2

m×n	C_CP1	Goto-BLAS2	CPV	C_CP1 / CPV	C_CP1 / Goto	Goto / CPV
32×32	6913	5357	1318	**5.25**	**1.29**	**4.06**
64×64	29172	16322	7529	**3.87**	**1.79**	**2.17**
256×256	467921	336960	233040	**2.01**	**1.39**	**1.45**

Таблица 20. Сравнение DGEMM CPV с DGEMM GotoBLAS2

m×n×k	C_CP1	Goto-BLAS2	CPV	C_CP1 / CPV	C_CP1 / Goto	Goto /CPV
16×16×16	30367	11600	4732	**6.42**	**2.62**	**2.45**
32×32×32	220438	63800	16029	**13.75**	**3.46**	**3.98**
64×64×64	3282795	454000	96379	**34.06**	**7.23**	**4.71**
128×...×128	26023211	3576700	698745	**37.24**	**7.28**	**5.12**

Существенно, что представленные коэффициенты выигрыша от применения CPV, как правило, значительно превышают аналогичные коэффициенты выигрыша, которые обеспечивают те же функции библиотеки GotoBLAS2, содержащей ассемблерное ядро, оптимизированное для сопроцессора CP1 архитектуры КОМДИВ. (Исключением является функция DAXPY, производительность которой не удаётся поднять с помощью CPV в случае шага ix≠1). Это подтверждают приведённые здесь таблицы 18-20.

В них для каждой из функций над вещественными числами двойной точности (DR): DAXPY, DGEMV и DGEMM сравниваются показатели производительности трёх её различных реализаций: 1) обычной (неоптимизированной) Си-функции (C_CP1); 2) функции библиотеки GotoBLAS2 (версии 1.13) с оптимизированным ассемблерным ядром; 3) ассемблерной функции, применяющей команды CPV.

Слева в колонках таблиц приводятся полученные показатели числа тактов, затраченных на исполнение функции, а справа – вычисленные на их основе коэффициенты выигрыша (по отношению к C_CP1) для CPV и GotoBLAS2. Наконец, в самой правой колонке содержится показатель, характеризующий, во сколько раз реализованная с помощью CPV функция превышает по производительности аналогичную оптимизированную функцию библиотеки GotoBLAS.

7. Выводы и предложения

Проведённый анализ и полученный практический опыт разработки ряда BLAS-функций для имеющегося варианта векторного сопроцессора (CPV) позволяют сделать следующие выводы о возможности его применения для повышения производительности функций обработки векторов и матриц.

1. CPV позволяет значительно ускорить обработку непрерывных векторов и почти не даёт никакой выгоды для обработки одиночных произвольных векторов.

2. При обработке векторов комплексных величин CPV обеспечивает более существенный выигрыш, чем при обработке вещественных.

3. Выигрыш от применения CPV возможен только при обработке векторов и матриц кратных размеров.

4. Вектора и матрицы, обрабатываемые с применением CPV, должны располагаться в памяти выровненными по кратному адресу.

5. CPV позволяет получить более значительный выигрыш при обработке матриц (в Си-программе), если такую обработку удаётся совершать по строкам, а не по столбцам.

Таким образом, применение CPV позволяет значительно повысить производительность для многих BLAS-функций. Но, к сожалению, не для всех. И не во всех случаях их применения.

Самой болезненной нерешённой проблемой, сдерживающей эффективное применение CPV, остаётся проблема обеспечения своевременной подкачки данных из памяти в регистры CPV (и обратно) при обработке произвольных векторов. Особенно остро эта проблема встаёт для векторов из 32-битных элементов.

Она могла бы окончательно разрешиться, если бы в CPV были добавлены команды ускоренной групповой загрузки (и сохранения) элементов произвольного вектора (X) в парные регистры (V0,V1), аналогичные тем командам (vldq, vsdq), что CPV обеспечивает для подкачки элементов непрерывного вектора. Например, это могли бы быть команды вида:

`vLdqI V0,I(X)`	V0←{X[0], X[I],X[2·I],X[3·I]} V1←{X[4·I], X[5·I],X[6·I],X[7·I]}
`vSdqI V0,I(X)`	V0→{X[0], X[I],X[2·I],X[3·I]} V1→{X[4·I], X[5·I],X[6·I],X[7·I]}

Возможно, было бы полезно осуществить такое совмещение некоторых CPV-регистров с парами регистров сопроцессора CP1:

`V0=(F0,F1), V1=(F2,F3),... V15=(F30,F31).`

Это позволило бы для работы с регистрами CPV напрямую использовать команды CP1 и наоборот. Тогда, например, для загрузки 4-х SR-элементов произвольного вектора в один регистр CPV потребовалось бы уже не 12 команд, а только 10.

Число команд, затрачиваемых на загрузку (и сохранение) можно было бы сократить, если бы в командах CPV обеспечивалась автоинкрементная адресация (например, как это сделано в сопроцессоре CP2 микропроцессора К128РИО). Тогда связка из двух команд могла бы превратиться в одну усложнённую команду:

`Lwc1 F,0(X); adDu X,X,sX`	`Lwc1 F,0(X)+sX`

(и вместо тех же 12 команд стало бы 8 или даже 6).

Могут ли эти предложения быть реально реализованы? Очевидно, что на этот вопрос способны ответить только разработчики аппаратуры. Несомненно лишь то, что в любом случае решение обозначенной проблемы невозможно без какого-то существенного усовершенствования самого CPV и/или связанной с ним процессорной архитектуры.

Автор выражает благодарность В.В.Цветкову. за предоставленные данные, характеризующие показатели производительности исполнения функций DAXPY, DGEMV и DGEMM библиотеки GotoBLAS2 на процессоре CP1 КОМДИВ-архитектуры.

On a possibility to optimize some library functions of linear algebra by means of a vector coprocessor

A.A. Burtsev

Abstract: In Scientific Research Institute for System Analysis of the Russian Academy of Sciences (SRISA RAS) as expansion of universal microprocessors of the KOMDIV family the specialized 128-bit coprocessor allowing to accelerate calculations over vectors of complex and real numbers of unary and double accuracy has been developed. The article presents results of use of this coprocessor directed on increase the execution speeds of the main functions for vectors and matrixes processing which are usually applied for the solution of typical tasks of linear algebra. Methods of code optimization taking into account features of the coprocessor are considered, the shortcomings interfering its effective application come to light, and possible ways of their overcoming are offered.

Keywords: microprocessors of the KOMDIV family, vector coprocessor, library BLAS.

Литература

1. В.Б.Бетелин. Отечественные суперкомпьютерные технологии экзафлопсного класса – необходимое условие обеспечения технологической конкурентноспособности России в XXI веке. «Программные продукты и системы», 2013, № 4 (104), с. 4-9.

2. URL: http://ru.wikipedia.org/wiki/Basic_Linear_ Algebra_Subprograms (дата обращения: 14.05.2014)

3. URL: http://www.cs.utexas.edu/users/flame/goto/ signup_first.html (дата обращения: 14.05.2014).

4. Л.Гервич, Б.Я.Штейнберг, М.Юрушкин. Программирование экзафлопсных систем. «Открытые системы», 2013, № 8 (194), с.26-29.

5. Genry Gregory. Оптимизация функций линейной алгебры библиотеки Intel® MKL под Intel® AVX на примере DGEMM. URL: https://software.intel.com/ru-ru/articles/ (дата обращения: 14.05.2014)

6. Goto Kazushige, Robert A. van de Geijn. Anatomy of high-performance matrix multiplication. «ACM Trans. on Mathematical Software», vol. 34, No. 3, 2008, pp.1-25.

Средства функциональной верификации в SystemVerilog

Д.Е.Гурьев, А.В.Семашко, Н.Ю.Шубин[1]

1- кандидат физико- математических наук

Аннотация: Язык SystemVerilog имеет мощные средства функциональной верификации. Рассматривается согласованное применение этих средств на примере модельного проекта цифровой схемы.

Ключевые слова: функциональная верификация, электронное устройство.

Введение

С ростом сложности электронных устройств резко возрастает и сложность верификации их проектов, и сложность специальных средств для решения этих задач.

В языке проектирования и верификации SystemVerilog [1-5] интегрируются достижения ряда предшествующих языков и стандартов, что делает его самым мощным языком проектирования из существующих сегодня. Язык содержит средства для описания проектируемого устройства, тестирующего окружения, тестовых воздействий. В нем также имеются специальные средства, предназначенные для автоматизации решения задач верификации. К таким средствам относятся механизмы утверждений (предложения **assert**, **assume**), позволяющие проверять утверждения о проектируемом устройстве в процессе моделирования, и механизмы моделей функционального покрытия (**covergroup**), позволяющие оценивать полноту функционального теста.

Важной особенностью механизмов утверждений и моделей покрытия является возможность задавать условия на языке временных последовательностей, специфицируя и учитывая темпоральную логику поведения автоматов.

Механизмы утверждений и моделей покрытия по отдельности уже обсуждались в предшествующих работах [6-7]. В дополнение к этим работам, в данной работе мы продемонстрируем сквозное, согласованное применение этих механизмов на примере небольшого модельного проекта.

1. «Проектируемое устройство»

«Проектируемым устройством» в нашем примере будет упрощенный кодер протокола MIL-STD-1553B. Это протокол обмена данными по последовательной линии с использованием «манчестерского» кода, при котором 0 и 1 кодируются перепадом 0–>1 и 1–>0 соответственно, передаваемым посередине интервала передачи данного бита. Передача данных производится 16-битными словами со скоростью 1Мбит/с.

Передаваемое слово дополняется синхроимпульсом, занимающим 3 разряда, и битом контроля четности, и, таким образом, занимает 20 разрядов, или 20мкс времени передачи. В протоколе используется два вида синхроимпульса, чтобы отличать «командные слова» от «слов данных». Формат передаваемых слов приведен на рис. 1.

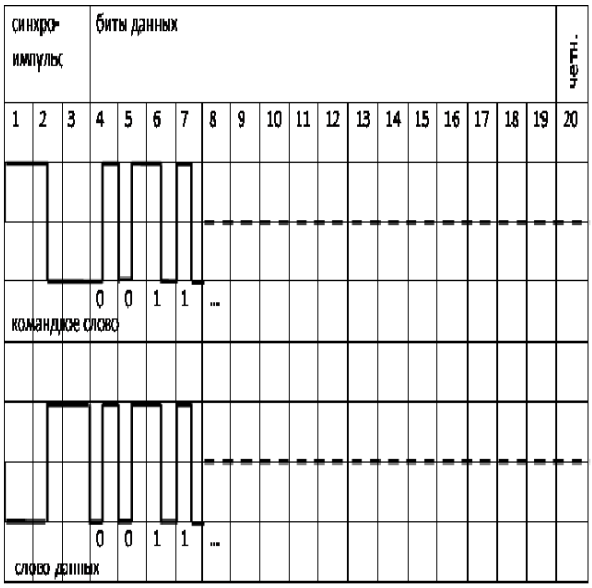


Рис. 1. Формат слов протокола

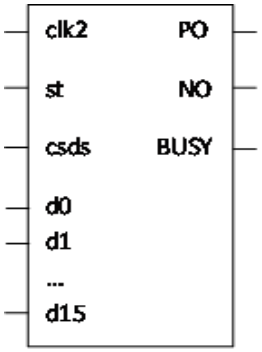


Рис. 2. Проектируемое устройство

Входы и выходы проектируемого кодера изображены на рис. 2. Их назначение следующее:

- clk2 – вход синхронизации 2 МГц,
- st – управляющий вход «начать передачу»,
- csds – вход выбора типа синхроимпульса,
- d0–d15 – входы данных,
- PO - выход, единичное значение которого командует аналоговому оборудованию канала сформировать положительную полуволну,
- NO - выход, единичное значение которого командует аналоговому оборудованию канала сформировать отрицательную полуволну,
- BUSY – выход, единичное значение которого сигнализирует, что кодер выполняет кодирование и передачу слова.

Пара сигналов PO, NO интерпретируется так: 00 – нет сигнала, 10 – положительный импульс, 01 – отрицательный импульс, комбинация 11 запрещена.
Будем считать, что наш кодер по сигналу st считывает данные со входов csds, d0-d15, и со следующего тактового сигнала clk2 начинает формировать слово протокола на выходах PO, NO, по половине разряда слова протокола на каждый такт синхронизации.

Потребуем также, чтобы окружающее оборудование сбрасывало сигнал st в следующем такте после установки кодером сигнала BUSY, и не устанавливало сигнал st до тех пор, пока сигнал BUSY не будет сброшен.

Описание проектируемого модуля в SystemVerilog выглядит так:

```
module encoder(clk2, st, csds, d, PO, NO, BUSY);
input clk2, st, csds, [15:0] d;
output PO, NO, BUSY;
endmodule;
```

(В этом описании заданы входы и выходы модуля, но не задано никакой реализации логики.)

2. Описание функциональности устройства при помощи утверждений.

SystemVerilog позволяет задать различные утверждения о работе устройства, которые в дальнейшем будут проверяться при моделировании (или учитываться средствами формальной верификации). Для простых проектов вроде нашего модельного примера при помощи этих средств даже возможно задать полную функциональную спецификацию устройства.

Для задания утверждений используются предложения **assert** и **assume**. В предложениях **assert** указываются условия, которым должно удовлетворять проектируемое устройство. В предложениях **assume** указываются условия, которые, как предполагается, будет выполнять окружающее оборудование. Эти два типа условий по-разному обрабатываются в средствах формальной верификации. В системах верификации, основанных на моделировании, нарушение любого из таких условий приводит к неуспешному результату моделирования, с той разницей, что нарушение условия, указанного в **assert**, свидетельствует об ошибке в проверяемом проекте, тогда как нарушение условия, указанного в **assume**, – об ошибке в проверяющем тесте или тестирующем окружении.

Прежде всего, запретим некорректную комбинацию на выходах PO и NO:

```
assert !(PO && NO);
```

Теперь потребуем, чтобы после установки входа st модуль выдавал правильную последовательность сигналов на выходы PO и NO. При этом мы будем учитывать, что одновременная установка этих выходов уже запрещена.

```
property encoding;
logic csds_, [15:0] d_, parity;
(@negedge clk2) !st ##1 st, d_ = d, csds_ = csds, parity = d[0] ⊕ d[1] ⊕ d[2] ⊕ d[3] ⊕ d[4] ⊕ d[5] ⊕
d[6] ⊕ d[7] ⊕ d[8] ⊕ d[9] ⊕ d[10] ⊕ d[11] ⊕ d[12] ⊕ d[13] ⊕ d[14] ⊕ d[15]
|=>
csds_? PO: NO ## 1 // 1-я половина синхроимпульса
csds_? PO: NO ## 1
csds_? PO: NO ## 1
csds_? NO: PO ## 1 // 2-я половина синхроимпульса
csds_? NO: PO ## 1
csds_? NO: PO ## 1
d_[0]? PO: NO ## 1 // 1-й бит данных – первая половина
d_[0]? NO: PO ## 1 // 1-й бит данных – вторая половина
d_[1]? PO: NO ## 1 // 2-й бит данных – первая половина
d_[1]? NO: PO ## 1 // 2-й бит данных – вторая половина
…
d_[15]? PO: NO ## 1 // 15-й бит данных – первая //половина
d_[15]? NO: PO ## 1 // 15-й бит данных – вторая //половина
parity? PO : NO ## 1 // бит четности – первая половина
parity? NO : PO;      // бит четности – вторая половина
endproperty;
assert property encoding;
```

Это требование более сложное, и его невозможно выразить в виде предложения **assert** с простым логическим условием. Здесь использовано предложение **property** (свойство), включающее локальные переменные (csds_, d_, parity) и определения последовательностей. Предложения SystemVerilog **property** служат для задания сложных условий, которые могут включать, в частности, временные последовательности. В данном предложении **property**:

- выражение (@**negedge** clk2) указывает, что вычисление свойства будет производиться при спадающем фронте сигнала синхронизации clk2,
- выражение «!st ## 1 st» – это выражение последовательности, означающее, что в предыдущем

такте st было установлено в 0, а в текущем – в 1, оно используется как условие запуска вычисления свойства,

- здесь «## 1» - указание задержки между элементами последовательности в тактах указанных часов (в нашем случае – в **negedge** clk2), SystemVerilog позволяет указывать любые положительные значения, а также их диапазоны и специальный символ $ - «до конца моделирования»,

- присваивания «d_ = d, csds_ = csds, parity =…» выполняются при выполнении условия запуска и «защелкивают» значения на входах d, csds в соответствующих локальных переменных, а также вычисляется бит четности parity,

- оператор «|=>» требует, чтобы со следующего такта после обнаружения последовательности «!st ##1 st» производилась проверка условия, указанного справа от него (аналогичный оператор SystemVerilog «|–>» требует проверки условия не со следующего, а с текущего такта),

- условие, указанное справа от «|=>», также является выражением последовательности SystemVerilog, оно считается выполненным, если в каждый из заданных моментов времени указанное для этого момента выражение является истинным; нетрудно видеть, что это условие задает предписанные значения для выходов PO, NO в последовательные моменты времени.

Заключительное предложение **assert** требует, чтобы выполнение указанного свойства проверялось.

Кроме этого, следует специфицировать поведение сигнала BUSY:

```
assert property (@negedge clk2) !st ##1 st => BUSY [*40];
assert property (@negedge clk2) !st && !BUSY => !BUSY;
```

Первое условие требует, чтобы после установки входа st в 1 сигнал BUSY тоже был установлен и оставался установленным еще 40 тактов («[*40]»). Второе условие требует, чтобы сигнал BUSY не устанавливался самопроизвольно при значении на входе st, сброшенном в 0. Дополним эти два условия еще условием «тишины» на выходах PO, NO при сброшенном BUSY:

```
assert BUSY || !PO && !NO;
```

В заключение списка утверждений сформулируем требования к окружению: сбрасывать сигнал st после установки BUSY и не устанавливать его, пока BUSY не будет сброшен. В данном случае правильнее использовать предложение **assume**, а не **assert**:

```
assume property (@negedge clk2) !BUSY ##1 BUSY |=> !st;
assume property (@negedge clk2) !st && BUSY |=> !st;
```

Наше модельное проектируемое устройство очень простое, и приведенные утверждения образуют его полную функциональную спецификацию.

3. Оценка полноты функциональной верификации

Оценка полноты функциональной верификации SystemVerilog вычисляется как полнота покрытия пространства входов и состояний функциональными тестами. Для выполнения такой оценки в SystemVerilog имеются средства для описания моделей покрытия (**covergroup**). Модель представляет собой множество счетчиков покрытия значений сигналов или переменных и их комбинаций. Модель также включает способ вычисления интегральной оценки полноты верификации. Попробуем задать функциональное покрытие для нашего проекта.

Потребуем, чтобы проверялись все возможные передаваемые слова при всех возможных типах синхроимпульса.

```
covergroup all_data;
syn: coverpoint csds
{
options.weight = 0;
}
data: coverpoint d
{
options.weight = 0;
}
synXdata: cross syn, data iff (st);
endgroup;
```

Каждая модель **covergroup** включает в себя множество точек покрытия **coverpoint** (в нашем примере – syn и data). Для каждой такой точки система автоматически создает счетчики покрытия («**bins**») для каждого возможного значения сигнала или переменной. В нашем примере будет создано 2 счетчика для syn и 2^{16} счетчиков для data. Модель может содержать также «пересечения» (декартовы произведения) множества значений точек покрытия **cross** (в нашем примере – synXdata), для пересечения также автоматически создаются счетчики для каждого элемента множества значений (в нашем примере их будет 2^{17}). Именно покрытие этого произведения нас и интересует. Чтобы доля покрытия точек syn и data не влияла на интегральную оценку полноты, этим точкам заданы опции «options.weight = 0». Условие «**iff** (st)» указывает, что факты покрытия значений synXdata будут регистрироваться только при значении входа st, равного 1 (действительно, при сброшенном st кодер никак не должен реагировать на входы csds и d, и такие тесты ничего не проверяют).

Правильно было бы также проверить кодер на правильную реакцию на сигнал st и на отсутствие самопроизвольной передачи данных. Учтем, что кодеру требуется помнить предыдущие состояния не более чем

40 тактов, увеличим эту величину вдвое, и будем считать, что отсутствие самопроизвольной передачи в течение 80 тактов доказывает нам невозможность такой передачи вообще. Важно также проверить это условие после того, как хотя бы одно слово было передано, чтобы исключить самопроизвольную передачу как эхо на предыдущую передачу. В итоге. приходим к тестовой последовательности из 80 тактов, в течение которых сигнал st сброшен в 0, 2х тактов, в которых он устанавливается в 1, за которым следует еще 39 тактов, в течение которых выполняется передача (сигнал st сброшен в 0), и затем следует еще 80 тактов, в которых st сброшен в 0. Чтобы потребовать, чтобы в тесте обязательно была такая последовательность, добавим в **covergroup** all_data еще одну точку покрытия, описывающую именно эту последовательность:

```
seq: coverpoint st
{
bins seq = 0 [*80] => 1 => 1 => 0 [*39] => 0 [*80];
}
```

Здесь значение, для которого вводится счетчик покрытия, является последовательностью во времени. Для отсчета этой последовательности для **covergroup** придется задать синхронизирующее значение («(@**negedge** clk2)»). В результате наша модель примет вот такой вид:

```
covergroup all_data (@negedge clk2)
syn: coverpoint csds
{
options.weight = 0;
}
data: coverpoint d
{
options.weight = 0;
}
synXdata: cross syn, data iff (st);
seq: coverpoint st
{
bins seq = 0 [*80] => 1 => 1 => 0 [*39] => 0 [*80];
}
endgroup;
```

Предложение **bins** позволяет задать конкретное значение или набор конкретных значений, для которых создаются счетчики покрытия. Все созданные автоматически счетчики, пересекающиеся с явно заданными счетчиками, удаляются, но те, которые не пересекаются – остаются. Для явной отмены автоматически созданных счетчиков служат предложения **ignore_bins** и **illegal_bins** (второе предложение вызывает сообщение об ошибке при покрытии данного значения).

Наше проектируемое устройство очень маленькое, и для него создание и выполнения теста, перебирающего все комбинации входных значений, не является проблемой. В более реалистичных проектах это часто нецелесообразно, а порою и невозможно. Сейчас мы покажем, как при помощи предложений SystemVerilog можно снизить требования к тесту.

Ограничим данные, подаваемые на входы [15:0] d, некоторыми характерными значениями, плюс еще заданное количество разных тестовых данных не из этого списка:

```
covergroup some_data (@negedge clk2)
syn: coverpoint csds
{
options.weight = 0;
}
data: coverpoint d
{
bins zeroes = 16'h 0;
bins ones = 16'h FFFF;
bins r1 [16] = { 16'h 0001, 16'h 0002, 16'h 0004, 16'h 0008,
16'h 0010, 16'h 0020, 16'h 0040, 16'h 0080,
16'h 0100, 16'h 0200, 16'h 0400, 16'h 0800,
16'h 1000, 16'h 2000, 16'h 4000, 16'h 8000 };
bins r0 [16] = { 16'h FFFE, 16'h FFFD, 16'h FFFB, 16'h FFF7,
16'h FFEF, 16'h FFDF, 16'h FFBF, 16'h FF7F,
16'h FEFF, 16'h FDFF, 16'h FBFF, 16'h F7FF,
16'h EFFF, 16'h DFFF, 16'h BFFF, 16'h 7FFF };
bins aaaa = 16'h AAAA;
bins five = 16'h 5555;
options.auto_bin_max = 1000;
options.weight = 0;
}
synXdata: cross syn, data iff (st);
seq: coverpoint st
{
bins seq = 0 [*80] => 1 => 1 => 0 [*39] => 0 [*80];
}
endgroup;
```

В этой модели покрытия для точки data задано 36 счетчиков покрытия для особенных комбинаций значений и 1000 счетчиков покрытия, между которыми SystemVerilog автоматически распределяет оставшиеся значения, по возможности равномерно. Полное покрытие такой модели будет означать, что проверено не менее 1036 разных значений [15:0] d, в том числе – все значения, перечисленные явно.

Таким образом, еще не начиная проектирования, мы сформулировали требования к проектируемому устройству при помощи предложений **assert** и **assume** (см. п. 2), и требования к тестовому обеспечению при помощи предложения **covergroup**.

4. Функциональное тестирование

Как уже говорилось, средства SystemVerilog позволяют смоделировать окружение и организовать

выполнение тестов на модели. В нашем случае это можно было бы сделать так:

```
wire clk2, st, csds, [15:0] d;  // Цепи связи проверяемого
//устройства с окружением
wire PO, NO, BUSY;

encoder enc(clk2, st, csds, d, PO, NO, BUSY);
// Экземпляр модели проверяемого устройства

task write_to_device(input syn, input [15:0] data);
// Задача (подпрограмма) записи данных в устройство
csds = syn;
/обеспечивает соглашения о взаимодействии
// с устройством
d = data;
@(posedge clk2) st = 1;
@(posedge clk2 && BUSY) #1 st = 0;
@(negedge BUSY);   // Ожидание, пока BUSY не будет
//сброшен в 0
endtask;

program test;
initial
// Тест на молчание кодера в отсутствии управляющих
//сигналов
repeat (80) @(posedge clk2);
write_to_device(1, 16'h 0);
repeat (80) @(posedge clk2);
$display("Test 1 coverage = ", $get_coverage());

// Тест передачи особенных значений
shortint VALUES [36] = {
16'h 0, 16'h FFFF, 16'h AAAA, 16'h 5555.
16'h 0001, 16'h 0002, 16'h 0004, 16'h 0008,
16'h 0010, 16'h 0020, 16'h 0040, 16'h 0080,
16'h 0100, 16'h 0200, 16'h 0400, 16'h 0800,
16'h 1000, 16'h 2000, 16'h 4000, 16'h 8000,
16'h FFFE, 16'h FFFD, 16'h FFFB, 16'h FFF7,
16'h FFEF, 16'h FFDF, 16'h FFBF, 16'h FF7F,
16'h FEFF, 16'h FDFF, 16'h FBFF, 16'h F7FF,
16'h EFFF, 16'h DFFF, 16'h BFFF, 16'h 7FFF
};
for (int i = 0; i < 36; i++)
for (int j = 0; j < 2; j++)
{
write_to_device(j, VALUES[i]);
$display("Test 2 coverage = ", $get_coverage());
}

// Тест передачи случайных значений
for (int i = 0; i < 4000; i++)
for (int j = 0; j < 2; j++)
{
write_to_device(j, $urandom);
int cv = $get_coverage();
$display("Test 3 coverage = ", cv);
if (cv >= 100) break;
}
$finish();
endprogram;
```

Разумеется, к этому заданию должны быть добавлены все предложения **assert**, **assume**, из п. 2 и описание **covergroup** some_data из п.3.

В этом примере системная функция $urandom возвращает случайное значение, системная функция $get_coverage() возвращает интегральную оценку полноты покрытия (значения от 0 до 100), а системная функция $dispalay выводит значения в стандартный вывод. В результате работы модели в стандартный вывод будет выводиться:

```
Test 1 coverage = 0
Test 2 coverage = 1
Test 2 coverage = 2
…
Test 3 coverage = 91
Test 3 coverage = 92
…
```

Системная функция $finish() завершает сеанс моделирования.

По завершении работы модели будут выданы отчеты о нарушениях условия **assert**, **assume**, и непокрытых элементах покрытия.

Мы рассмотрели пример функциональной спецификации проекта, построение модели функционального покрытия для этого проекта и реализацию комплекта функциональных тестов. Было продемонстрировано применение конструкций языка SystemVerilog для этих целей. Как видим, решать эти задачи средствами SystemVerilog возможно до начала собственно проектирования устройства или независимо от проектирования.

5. Верификация больших проектов

Рассмотренный пример очень маленький – иначе изложение было бы безразмерным – и на нем незаметны проблемы, возникающие при верификации больших проектов.

Полная спецификация функциональности при помощи **assert** и **assume**, как правило, рассматривается как нецелесообразная. По крайней мере, объем этой спецификации сравним с объемом самого проекта устройства. Часто проверяют лишь условия, представляющиеся наиболее значимыми или связанные с наиболее критичными обстоятельствами поведения, такими как поведение вблизи граничных точек. С другой стороны, в отличие от нашего примера, большой проект редко верифицируется как черный ящик. Проверяемые условия могут формулироваться не только для входов и выходов устройства, но и его для внутренних точек, в частности – для интерфейсных цепей, связывающих его составные части. Последний случай тоже иллюстрируется нашим примером, если мы представим encoder частью некоторого объемлющего проекта.

(Предложения **assume** в таком случае содержат требованиями к остальным компонентам проекта, и их лучше заменить на **assert** с тем же утверждением)

Полное покрытие всего множества комбинаций сигналов в больших проектах, как правило, является невыполнимым требованием. Множество покрываемых комбинаций сокращают, оставляя лишь наиболее критичные. В нашем примере мы показали, как можно выполнять такое сокращение средствами SystemVerilog.

Проектируемые устройства, как правило, связаны с внешним миром достаточно сложными интерфейсами, которые могут представлять собой протоколы шин передачи данных. Для эффективного решения задачи верификации используются специальные библиотеки и связанные с этими библиотеками методологии, такие как VMM [8], UVM [9]. Обычно библиотека уже содержит заготовку модели тестирующего окружения (testbench), которую надо только настроить под конкретный проект. Однако, архитектура этих моделей достаточно сложна сама по себе, что часто служит препятствием их внедрения в процесс проектирования. Для облегчения перехода к их использованию предложена методика поэтапного внедрения [10].

Заключение

На конкретном примере рассмотрено применение средств функциональной верификации языка SystemVerilog – утверждений и моделей покрытия. Пример, включающий сквозное, согласованное применение механизмов утверждений, моделей покрытия и организации тестирования на модели, представлен впервые. Авторы надеются, что работа будет иметь важное методическое значение для внедрения средств функциональной верификации SystemVerilog в процессы проектирования электронного оборудования на регулярной основе.

Functional Verification Tools in SystemVerilog

D.E.Guriev, A.V.Semashko, N.Ju.Shubin

Abstract: SystemVerilog language incorporates powerful tools for functional verification. A coordinated use of these tools in application to a digital circuit design is considered in this paper.

Keywords: functional verification, electronic device

Литература

1. Описание языка System Verilog – Unified Hardware Design, Specification, and Verification Language, Часть I. Сборник научных трудов под ред. В.Б. Бетелина, П.П. Кольцова, А.С. Яицкова, НИИСИ РАН, Москва, 2009.

2. Описание языка System Verilog – Unified Hardware Design, Specification, and Verification Language, Часть II. Сборник научных трудов под ред. В.Б. Бетелина, П.П. Кольцова, А.С. Яицкова, НИИСИ РАН, Москва, 2010.

3. Описание языка System Verilog – Unified Hardware Design, Specification, and Verification Language, Часть III. Сборник научных трудов под ред. В.Б. Бетелина, П.П. Кольцова, А.С. Яицкова, НИИСИ РАН, Москва, 2010.

4. Описание языка System Verilog – Unified Hardware Design, Specification, and Verification Language, Часть IV. Сборник научных трудов под ред. В.Б. Бетелина, П.П. Кольцова, А.С. Яицкова, НИИСИ РАН, Москва, 2011.

5. Описание языка System Verilog – Unified Hardware Design, Specification, and Verification Language, Часть V. Сборник научных трудов под ред. В.Б. Бетелина, П.П. Кольцова, Д.Е.Гурьева, НИИСИ РАН, Москва, 2013.

6. Г.А. Яицкова. Функциональное покрытие в SystemVerilog. Труды IT+S&E'2012, Ялта-Гурзуф, 2012.

7. Г.А. Яицкова. SystemVerilog Утверждения в SystemVerilog-2009. «Образовательные ресурсы и технологии», № 2 (5), Москва, 2014.

8. Introduction to Design Verification with VMM – a Quickstart Guide. Synopsys, Inc., 2011. URL: https:// www.vmmcentral.org/pdfs/intro_to_dv_with_vmm.pdf

9. UVM Cookbook. URL: https://verificationacademy.com/cookbook/uvm

10. И.В.Селиванов, Н.Ю.Шубин. UVM EXPRESS – упрощенная методика внедрения UVM. «Образовательные ресурсы и технологии», № 2 (5), Москва, 2014.

Программное обеспечение для приема и передачи данных по высокоскоростному каналу в мультипроцессорных комплексах реального времени

Т.К. Грингауз, А.Н. Онин

Аннотация: В мультипроцессорных системах обработки цифровой информации на базе процессоров КОМДИВ64-РИО, КОМДИВ128-РИО, функционирующих в среде RapidIO, данные от аналого-цифровых преобразователей (АЦП) вводятся по высокоскоростному каналу (ВСК). Преобразование данных из формата ВСК в формат PRIO ([1]) осуществляется с помощью специальных аппаратных устройств («устройства ВСК») на RapidIO, требующих конфигурирования. Для поддержки конфигурирования устройств ВСК, приема и передачи данных в формате ВСК в системах цифровой обработки информации под управлением ОС РВ Багет 3.3 разработано программное изделие «Пакет поддержки устройств ВСК в среде RapidIO для операционной системы реального времени» (ПП ВСК-РИО). В статье описаны архитектура, функциональность и принцип функционирования ПП ВСК-РИО.

Ключевые слова: программное обеспечение, архитектура, функциональность

1. Определение основных понятий и постановка задачи

В НИИСИ РАН разрабатываются аппаратные средства и средства общего программного обеспечения, предназначенные для использования в составе мультипроцессорных систем обработки цифровой информации в режиме реального времени. Аппаратные средства создаются на базе универсальных и специализированных процессоров, функционирующих в коммутируемой коммуникационной среде RapidIO[1]: КОМДИВ64-РИО (микросхема 1890ВМ6Я, [2]), КОМДИВ128-РИО (микросхема 1890ВМ7Я, [3]). В качестве программной платформы используется операционная система ОС РВ Багет 3.3, принадлежащая семейству операционных систем реального времени ОС РВ Багет [5] (далее - ОС РВ). Совокупность аппаратных средств, функционирующих под управлением ОС РВ, далее будем называть целевой платформой. Программы, предназначенные для исполнения на целевой платформе, будем называть целевыми программами.

В мультипроцессорных системах обработки цифровой информации на целевой платформе передача внешних данных от аналого-цифровых преобразователей (АЦП) к обрабатывающим процессорам производится с использованием аппаратного интерфейса, называемого высокоскоростным каналом (ВСК). ВСК – это двунаправленный последовательный интерфейс «точка-точка», передающий данные в специфическом формате («формат ВСК»). Интерфейс реализован в контроллере ВСК, входящем в состав разработанной в НИИСИ РАН микросхемы 1890ВГ18Я ([4]). Микросхема представляет собой гибридное устройство [1], не содержащее процессорных элементов, взаимодействующее со средой RapidIO по интерфейсу RapidIO 8 LP-LVDS (далее- PRIO) [1]. Контроллер ВСК обеспечивает прием и передачу данных между устройствами на RapidIO и внешними устройствами. При приеме данных по последовательному каналу производится их десериализация, разбиение на сегменты и распределение сегментов по процессорам через RapidIO. При передаче данных контроллер ВСК сериализует данные, приходящие по PRIO, и направляет их в выходной последовательный канал.

Ниже будут рассматриваться системы, в которых внешние потоки данных в формате ВСК направляются в процессоры КОМДИВ128-РИО, а передача данных по RapidIO в контроллер ВСК может осуществляться как с КОМДИВ128-РИО, так и с КОМДИВ64-РИО. Такой подход оправдан, поскольку КОМДИВ128-РИО обладает специализированным 128-разрядным сопроцессором, ориентированным на цифровую обработку сигналов, и именно этот тип процессора целесообразно использовать для первичной обработки данных ВСК.

Под устройствами ВСК понимаются:

-контроллер ВСК микросхемы 1890ВГ18Я [4] (далее - «контроллер ВСК»),

-канал контроллера DMA для приема данных ВСК через интерфейс RapidIO процессора КОМДИВ128-РИО [3] (далее - «КВСК»).

Перед тем, как начать прием или передачу данных по ВСК, необходимо проинициализировать [1] среду RapidIO и сконфигурировать устройства ВСК. Под конфигурированием устройств понимается программирование их регистров с целью обеспечения требуемой функциональности. Конфигурирование устройств ВСК должны осуществлять целевые программы.

При преобразовании входных потоков из формата ВСК в формат PRIO контроллер ВСК использует информацию о распределении потоков по процессорам. Информация заранее вносится в память контроллера на этапе его конфигурирования.

Конфигурирование контроллеров ВСК осуществляется путем отправки в них конфигурационных данных по RapidIO с удаленных процессоров. Так же осуществляются запуск и останов контроллера. Конфигурационные данные отправляются пакетами MAINTENANCE[1].

Контроллер ВСК передает данные в среду RapidIO пакетами NWRITE, NWRITE_R [1]. В заголовки формируемых контроллером пакетов вносится информация о том, что данные получены от ВСК. Эта информация используется на приемной стороне контроллером RapidIO при передаче пакета в накристальную память (SRAM) процессора, а также контроллером DMA при последующей передаче в динамическую память (DPRAM). Пересылка в DPRAM осуществляется посредством КВСК. Канал КВСК должен быть предварительно сконфигурирован на основе априорной информации о поступающих в него потоках данных. Конфигурирование, запуск и останов КВСК должна производить программа, функционирующая на локальном процессоре.

Для передачи данных с процессора на RapidIO в выходной последовательный канал контроллера ВСК могут использоваться пакеты MAINTENANCE или NWRITE.

Функциональное программное обеспечение (ФПО) систем с обменом данными по ВСК должно обладать следующими свойствами.

Программы для процессоров, обеспечивающих конфигурирование контроллеров ВСК, должны иметь доступ к априорной информации о потоках данных в конфигурируемых контроллерах, интерпретировать ее, формировать на ее основе пакеты MAINTENANCE [1] с конфигурационными данными и отправлять их по RapidIO в конфигурируемые контроллеры ВСК.

Программы для процессоров, обеспечивающих прием данных от контроллеров ВСК, должны иметь доступ к априорной информации о приходящих к ним потоках данных, интерпретировать ее, конфигурировать на ее основе локальные КВСК, вычитывать и обрабатывать данные, приходящие в DPRAM от КВСК.

Программы для процессоров, обеспечивающих передачу данных от контроллеров ВСК, должны формировать данные в формате десериализованных данных ВСК и передавать их по RapidIO в заранее сконфигурированные на передачу контроллеры ВСК.

При разработке программного обеспечения с перечисленными свойствами возникают проблемы технического и технологического характера:

- отсутствие в ОС РВ пользовательского интерфейса для приема данных от КВСК и для отправки пакетов NWITE,
- необходимость разработки специальных драйверов, функционирующих в системном процессе KERNEL, для осуществления доступа к регистрам физических устройств (в том числе, к регистрам КВСК) из защищенного POSIX-процесса ОС РВ,
- трудоемкость составления конфигурационных данных для устройств ВСК на основании информации о потоках данных ВСК в системе,
- трудоемкость программирования регистров физических устройств,
- дублирование информации при описании конфигурации источника и приемника потока данных,
- сложность унификации программного кода для различных процессоров мультипроцессорной системы при многообразии конфигураций их КВСК.

С целью преодоления перечисленных проблем и упрощения прикладного программирования разработано программное изделие, выполняющее роль промежуточного слоя между ОС РВ и ФПО - «Пакет поддержки устройств ВСК в среде RapidIO для операционной системы реального времени» (ПП ВСК-РИО). Программное изделие предназначено для поддержки конфигурирования устройств ВСК, приема и передачи данных в формате ВСК в мультипроцессорных системах цифровой обработки информации на основе процессоров КОМДИВ64-РИО, КОМДИВ128-РИО.

2. Общая характеристика ПП ВСК-РИО

2.1 Архитектура и принцип функционирования ПП ВСК-РИО

Перечислим основные принципы, на которых базировалась разработка ПП ВСК-РИО:

- технологическое отделение этапа составления конфигурационных данных устройств ВСК от этапа разработки кода целевых программ;
- выполнение этапа составления конфигурационных данных устройств ВСК на инструментальной ЭВМ (далее - «ИЭВМ») в среде ОС Linux;
- автоматизация преобразования информации о потоках ВСК-данных в конфигурационные данные ВСК-устройств,
- представление совокупности конфигурационных данных ВСК-устройств системы в формате, который позволял бы включить эти данные в тексты целевых программ (далее – «описание конфигурации»),
- разработка программного интерфейса целевых программ для интерпретации описания конфигурации, осуществления конфигурирования устройств ВСК, приема и передачи данных в формате ВСК на целевой платформе,
- обеспечение возможности разработки унифицированного кода целевых программ в мультипроцессорной среде за счет специфического структурирования описания конфигурации,
- разработка в составе программного изделия драйвера ОС РВ, функционирующего в процессе KERNEL и обеспечивающего выполнение функций конфигурирования устройств ВСК, приема и передачи данных в формате ВСК в защищенных процессах POSIX.

Использование специфического формата описания конфигурации, автоматизация подготовки описания конфигурации на ИЭВМ и последующая расшифровка функциями программного интерфейса на целевой

платформе позволяет обеспечить прозрачность программирования регистров устройств ВСК для разработчиков ФПО.

В соответствии с описанным выше подходом программное изделие ПП ВСК-РИО реализовано в виде двух программ: «Конфигуратор ВСК» и «Библиотека ВСК».

Конфигуратор ВСК – это инструментальное средство, предназначенное для составления конфигурационных данных устройств ВСК на основании информации о потоках ВСК в системе. Функционирует в среде Linux на «ИЭВМ» под управлением Java-машины. Позволяет с помощью графического интерфейса задать распределение потоков входных данных по принимающим процессорным элементам на RapidIO. Создает соответствующие этому распределению конфигурационные данные устройств ВСК в формате, пригодном для включения в целевые программы.

Библиотека ВСК предназначена для настройки устройств ВСК, приема и передачи данных в формате ВСК на целевой платформе. Библиотека ВСК включает функции для инициализации устройств ВСК в соответствии с конфигурационными данными, а также функции для приема и передачи данных по каналам ВСК. На процессоре КОМДИВ128-РИО могут исполняться все функции библиотеки. На процессорах КОМДИВ64-РИО могут исполняться только те функции библиотеки, которые предназначены для удаленного конфигурирования, синхронизации устройств ВСК или для отправки данных в контроллер ВСК.

Функции библиотеки могут исполняться в системном процессе ОС РВ (KERNEL), а также в защищенном пользовательском процессе (POSIX). Выполнение функций библиотеки в процессе POSIX обеспечивается специальным драйвером («драйвер ВСК»), запускаемым при старте ОС РВ. Функция инициализации драйвера ВСК входит в состав библиотеки ВСК.

ПП ВСК-РИО обеспечивает следующую технологию разработки СПО.

На ИЭВМ с помощью графического интерфейса, предоставляемого конфигуратором ВСК, пользователь формирует файл в текстовом формате с описанием параметров конфигурации устройств ВСК, участвующих в обмене данными в формате ВСК (далее – «файл конфигурации ВСК»). Посредством конфигуратора ВСК файл конфигурации ВСК преобразовывается: в него добавляется секция, содержащая представление параметров конфигурации в виде целочисленного массива в формате языка Си (далее – «массив конфигурационных данных»). Файл конфигурации ВСК с массивом конфигурационных данных включается директивой #include в тексты прикладных программ на языке Си, предназначенных для исполнения на целевой платформе.

В прикладной программе конфигурационные данные должны быть определены как целочисленный массив и инициализированы с помощью файла конфигурации ВСК, например:

```
 static int configVSK [] = // массив конфигурационных данных
 {
#include "01-1-red.vsk"     // файл конфигурации ВСК
 };
```

Целевые программы осуществляют конфигурирование устройств ВСК и обмен данными в формате ВСК с помощью функций библиотеки ВСК в соответствии с описанием конфигурации. Функции библиотеки ВСК используют указатель на массив конфигурационных данных в качестве входного параметра.

2.2 Унификация кода целевых программ в мультипроцессорной среде

Для мультипроцессорной среды исполнения характерен подход, при котором на всех процессорах функционирует одна и та же программа. На разных процессорах исполняются разные ветки этой программы в зависимости от уникального идентификатора процессора. Если программа использует входные данные, задаваемые в тексте, и эти данные различны для разных процессоров, то необходимо реализовать механизм вычленения данных для каждого процессора из общего набора во время исполнения программы.

ПП ВСК-РИО обеспечивает возможность разработки унифицированных целевых программ для мультипроцессорной среды. Массив конфигурационных данных, включаемый в текст программы, содержит совокупность конфигурационных данных для всех ВСК-устройств комплекса. Программа, исполняемая на процессоре, должна идентифицировать локальные или удаленные устройства ВСК, подлежащие конфигурированию с данного процессора, и вычленить конфигурационные данные для каждого из этих устройств. Такая возможность обеспечивается форматом массива конфигурационных данных. Идентификация устройств и вычленение конфигурационных данных производится функциями библиотеки ВСК. В качестве ключа, по которому производится идентификация процессора и относящихся к нему конфигурационных данных, используется идентификатор RapidIO [1] (далее - «РИО-идентификатор»).

При использовании ПП ВСК-РИО предполагается, что RapidIO инициализируется статически [1] до начала использования обменов по ВСК (средства инициализации RapidIO не входят в ПП ВСК-РИО). Статическая инициализация устанавливает соответствие РИО-идентификаторов конкретным устройствам априори, и это соответствие не изменяется при запуске программ на целевой платформе. Целевая программа, исполняющаяся на процессоре, может узнать его РИО-идентификатор средствами ОС РВ.

РИО-идентификаторы всех устройств ВСК, участвующих в конфигурировании или в приеме данных, должны быть известны заранее и передаваться

конфигуратору ВСК в составе входных данных. Конфигуратор предоставляет опциональную возможность употреблять символические имена РИО-идентификаторов. Символические имена РИО-идентификаторов должны быть определены с помощью макросов #define в специальном заголовочном файле (далее - «файл списка идентификаторов»). В случае использования конфигуратором файла списка идентификаторов этот файл должен включаться в прикладную программу.

Массив конфигурационных данных, формируемый конфигуратором ВСК, содержит данные двух типов: 1) данные для локального конфигурирования КВСК, 2) данные для удаленного конфигурирования микросхем 1890ВГ18Я. Данные для удаленного конфигурирования интерпретируются функциями библиотеки ВСК, отправляющими пакеты MAINTENANCE с удаленного процессора в микросхему. Таким образом, целевой программе должна быть передана информация о том, какой контроллер ВСК с какого процессора должен конфигурироваться.

Идеология ПП ВСК-РИО предполагает, что конфигурирование всех контроллеров ВСК осуществляется с единого процессора, называемого «главным». Главный процессор предназначен для управления обменом данных в формате ВСК между устройствами ВСК. С главного процессора можно осуществлять конфигурирование, запуск и останов контроллеров ВСК, синхронизацию контроллеров ВСК с принимающими процессорами 1890ВМ7Я, синхронизацию начала приема данных на разных принимающих процессорах 1890ВМ7Я. Главный процессор должен иметь доступ по сети RapidIO ко всем конфигурируемым микросхемам 1890ВГ18Я и ко всем принимающим процессорам.

Информация о том, какой из процессоров назначен главным, вносится в массив конфигурационных данных на этапе использования конфигуратора ВСК. Пользователь может явно указать РИО-идентификатор главного процессора или назначить главный процессор неявно. В последнем случае в качестве главного будет автоматически выбран первый из списка процессоров, участвующих в приёме данных ВСК. Целевая программа интерпретирует массив конфигурационных данных с помощью функций библиотеки ВСК и выполняет удаленное конфигурирование только в том случае, если выполняется на главном процессоре.

3. Краткая справка по архитектуре и функциональности устройств ВСК

Ниже приводятся сведения об устройствах ВСК [3,4].в объеме, необходимом для описания функциональности ПП ВСК-РИО.

3.1 Описание входного потока данных ВСК

Входной поток данных ВСК представляет собой последовательность пачек, разделенных служебным словом «начало пачки». Пачки состоят из 36-разрядных слов. 4 старших разряда интерпретируются ка «тэг», остальные 32 разряда трактуются как содержательные данные. Поле тэга предназначено для кодирования цвета и типа данных. Контроллер ВСК воспринимает два цвета данных: «красный» и «зеленый». Кодируются следующие типы данных: «данные», «начало пачки», «конфигурационные данные» и «пустые данные». Данные разных цветов во входном потоке могут быть перемешаны.

3.2 Преобразование потоков данных контроллером ВСК

3.2.1 Прием входного потока данных ВСК

Внешний разъем микросхемы 1890ВГ18Я позволяет принимать данные в формате SRIO (один канал 4x или два канала 1x) или в формате ВСК. Режим использования микросхемы (ВСК или SRIO) устанавливается путем отправки конфигурационных данных по RapidIO с удаленного процессора.

Данные ВСК поступают в контроллер ВСК по двум каналам (канал №1, канал №2). Каждый из каналов подразделяется на «канал А», «канал В». Каждый из каналов 1А,1В, 2А, 2В может принимать входной поток данных ВСК или передавать поток данных в последовательный канал. Таким образом, контроллер может принимать до 4 входных потоков данных ВСК. Поскольку с точки зрения аппаратного устройства каналы №1 и №2 идентичны, далее будем говорить просто о канале ВСК, разделенном на каналы А и В.

В состав канала ВСК входит генератор тестовых данных, который может выдавать поочередно пачки красного и зеленого цвета. Данные от генератора предназначены для использования в отладочных целях.

Потоки данных в контроллере ВСК коммутируются посредством «блока управления каналами». В зависимости от настройки контроллера производится установка направления потоков данных красного и зеленого цвета в следующих направлениях:

- из канала А в канал В или в канал приемного FIFO (выход на RapidIO);
- из канала В в канал А или в канал приемного FIFO (выход на RapidIO);
- из встроенного генератора тестовых данных на вход канала А, в канал В или в канал приемного FIFO (выход на RapidIO);
- из порта передачи данных на ВСК (вход с RapidIO) в канал А, в канал В или в канал приемного FIFO (выход на RapidIO).

В контроллере ВСК реализован механизм изменения цвета данных в каналах А и В. Перекраска данных необходима в ситуации, когда по обоим каналам А и В поступают данные одного цвета, и оба эти потока направляются в приемное FIFO. По выходе из FIFO данные уже не привязаны к каналам (А или В), а различаются только цветом. Если не перекрасить

данные в одном из каналов, то разделить потоки от каналов A и B невозможно.

Реализован режим объединения нескольких пачек в одну в приемном FIFO. Такой режим может использоваться для повышения эффективности передачи данных по RapidIO.

3.2.2 Преобразование входных потоков данных ВСК в пакеты RapidIO

Данные из приемного FIFO блока управления каналами попадают в контроллер, осуществляющий упаковку данных в пакеты RapidIO. Контроллер убирает у слов признаки цветности и упаковывает их в 64-разрядные слова. Предусмотрен режим контроллера, при котором последовательность байтов в 64-разрядных словах может быть изменена.

На этапе формирования пакетов RapidIO происходит разбиение пачек данных на сегменты и распределение сегментов по обрабатывающим процессорам с использованием «механизма дескрипторов» [4]. В памяти контроллера на этапе его конфигурирования создаются четыре связных списка, называемых «цепочками локальных дескрипторов». Каждая цепочка привязана к определенному потоку данных («канал №1, красный», «канал №1, зеленый», «канал №2, красный», «канал №2, зеленый»). Адреса базовых дескрипторов фиксированы, списки закольцованы: последний дескриптор ссылается на первый. Логически цепочка локальных дескрипторов представляет собой таблицу, устанавливающую соответствие размера очередного сегмента РИО-идентификатору вычислительного узла, которому этот сегмент должен быть отправлен. Кроме того, локальный дескриптор содержит поля для указания признака мультикастингового режима передачи (MC), признаков начала пачки (PS) и виртуального начала пачки (VPS).

Под мультикастинговым режимом понимается режим передачи данных, при котором одни и те же данные пачки отправляются двум вычислительным узлам. Для указания дополнительного вычислительного узла используется дополнительный дескриптор, а в основном дескрипторе устанавливается специальный бит, сообщающий контроллеру о дополнительном дескрипторе.

Признак PS устанавливается для дескрипторов первого сегмента пачки. Признак VPS устанавливается опционально в случае распределения сегментов одной пачки на несколько вычислительных узлов. Признак VPS призван информировать о новой пачке вычислительные узлы, на которые не приходит признак PS.

Вычитывая последовательно данные из приемного FIFO, контроллер по номеру канала и цвету данных определяет соответствующую цепочку и на основании содержимого ее текущего дескриптора формирует сегмент для отправки на RapidIO. По достижении указанного в дескрипторе размера происходит «закрытие дескриптора» и переход к обработке следующего локального дескриптора.

Сегменты передаются на RapidIO пакетами NWRITE по 256 байт. Передача сегмента завершается отправкой пакета NWRITE_R, который «закрывает» текущий дескриптор. В заголовки пакетов записываются РИО-идентификаторы источника (SrcID), приемника (DestID) и адрес блока памяти приемника, по которому следует записать данные. В качестве SrcID записывается РИО-идентификатор микросхемы 189ВГ18Я. В качестве DestID записывается РИО-идентификатор удаленного вычислительного узла, указанный в текущем локальном дескрипторе. Механизм определения адреса удаленного блока памяти зависит от режима использования контроллера ВСК. ПП ВСК-РИО ориентирован на использование режима «упрощенного приема пачки» [4], реализованного специально для отправки данных в процессор КОМДИВ128-РИО. В накристальной памяти процессора КОМДИВ128-РИО для приема пакетов от контроллеров ВСК выделены четыре фиксированных адреса, соответствующих двум каналам (№1 и №2) и двум цветам данных («канал №1, красный», «канал №1, зеленый», «канал №2, красный», «канал №2, зеленый»). Старшие два бита упомянутых адресов, записываемые в поле xam заголовка пакета, специфичны и отличают пакеты, приходящие от контроллера ВСК, от всех остальных пакетов, приходящих в контроллер RapidIO. При использовании режима упрощенного приема пачки в качестве адреса пакета RapidIO контроллер ВСК записывает предопределенный адрес, соответствующий номеру канала и цвету данных. КВСК принимающего процессора должен быть предварительно сконфигурирован на прием потока данных от микросхемы 1890ВГ18Я с РИО-идентификатором SrcID из соответствующего канала соответствующего цвета. До начала приема данных от КВСК программа, функционирующая на процессоре, должна выделить в динамической памяти приемные буфера для каждого принимаемого потока данных и записать их адреса в регистры КВСК. Пакеты, поступившие в накристальную память КОМДИВ128-РИО по адресу, указанному в заголовке пакета, передаются посредством КВСК в динамическую память процессора в соответствии с конфигурацией КВСК.

В составе пакета, закрывающего текущий дескриптор, передается информация об ошибках приема данных ВСК, зафиксированных контроллером. Ошибка ErrorDataLost означает, что во входящей по ВСК пачке обнаружены лишние данные, и эти данные были утеряны. Ошибка ErrorNoData означает, что во входящей по ВСК пачке недостает данных. Переданная информация об ошибках фиксируются в специальных регистрах КВСК принимающего процессора.

Контроллер ВСК поддерживает два режима передачи данных по RapidIO: 1) с ожиданием ответа от принимающего процессора после отправки сегмента данных, 2) без ожидания ответа. Ожидание ответа снижает скорость передачи данных ВСК по RapidIO, но повышает надежность передачи.

3.3 Прием данных от контроллера ВСК процессором КОМДИВ128-РИО

Управление приемом данных посредством КВСК осуществляется ядром процессора с использованием регистров, программируемых через интерфейс PIO.

Процессор КОМДИВ128-РИО может принимать данные ВСК от восьми микросхем 1890ВГ18Я. От одного контроллера ВСК могут приходить следующие потоки данных:

- из 1-го канала контроллера ВСК, данные красного цвета;
- из 1-го канала контроллера ВСК, данные зелёного цвета;
- из 2-го канала контроллера ВСК, данные красного цвета;
- из 2-го канала контроллера ВСК, данные зелёного цвета.

Таким образом, на одном процессоре можно обрабатывать потоки данных от 32 источников. Источник идентифицируется совокупностью трех признаков: РИО-идентификатор микросхемы 1890ВГ18Я, номер канала (№1 или №2) и цвет данных.

Приходящие в КОМДИВ128-РИО пакеты с данными ВСК размещаются в накристальной памяти по четырем фиксированным адресам в соответствии с номером канала и цветом данных. Для пересылки данных с каждого из четырех адресов в динамическую память выделен свой блок КВСК. В КВСК проводится фильтрация пакетов по РИО-идентификаторам источника пакета. При конфигурировании КВСК РИО-идентификаторы источников должны быть занесены в регистры SID0..7, предназначенные для хранения информации об избранных РИО-идентификаторах источников. С каждым избранным РИО-идентификатором связаны четыре специальных регистровых файла (соответственно двум каналам и двум цветам), предназначенные для определения адреса размещения пакета в динамической памяти. Каждому принимаемому потоку соответствует свой регистровый файл. Адрес регистрового файла определяется на основе информации о РИО-идентификаторе источника, о номере канала и цвете данных, получаемой из заголовка пакета RapidIO.

КВСК осуществляет прием потока данных от одного источника в два буфера. Данные принимаются сегментами. Размер буфера должен быть кратен размеру сегмента. Память для приемных буферов должна быть физически непрерывной.

Для информации о приемных буферах в регистровом файле предусмотрены следующие поля. В полях ADR0, ADR1 содержится информация об адресах приемных буферов 0 и 1. В поле OFFS содержится информация о размере сегментов в приемных буферах 0 и 1. В полях CNT0, CNT1 содержатся счетчики свободных сегментов в приемных буферах 0 и 1. Поле SEL указывает, какой из буферов (0 или 1) должен использоваться для приема данных в текущий момент. Перечисленные поля должны быть инициализированы программным путем через интерфейс PIO. В процессе работы КВСК модифицирует их значения.

В регистровом файле предусмотрено поле STAT, предназначенное для записи дополнительной информации о переданном сегменте данных. Поле STAT содержит биты для фиксации ошибок ErrorDataLost, ErrorNoData и признаков PS,VPS,MC, передаваемых в пакете NWRITE-R.

При поступлении сегмента данных от избранного источника в SRAM КВСК вычисляет адрес регистрового файла, считывает его регистры и в зависимости от поля SEL выбирает один из двух приемных буферов. Вычисляются новые значения регистров для выбранного буфера: ADRx = ADRx + OFFS, CNTx = CNTx – 1.Если CNTx равен 0, то поле SEL инвертируется. Заполняется поле STAT. Далее на основании анализа поля CNTx производятся следующие действия:

а) если значение равно 0, то производится переключение SEL и прием пакета в другой буфер;

б) если оба счетчика CNTx равны 0, то прекращается обработка пакета и осуществляется переход к приему следующего пакета с потерей текущих данных и формированием информации об ошибке,

в) если значение CNTx не равно 0, то вычисляется адрес в DPRAM на основании поля ADRx и осуществляется DMA-пересылка данных в память.

Помимо регистровых файлов, обеспечивающих прием отдельных потоков данных, в КВСК реализованы регистры, позволяющие контролировать устройство в целом. В частности, к ним относятся регистр управления, регистр маскирования прерывания, регистр прерываний. Записью полей START и STOP регистра управления можно стартовать или остановить работу КВСК. Регистры, обеспечивающие работу с прерываниями, позволяют установить возбуждение и обработку прерываний в ошибочных ситуациях, среди которых можно выделить следующие практически важные случаи:

- обнуление счетчика CNT0,
- обнуление счетчика CNT1,
- обнуление обоих счетчиков CNT0, CNT1 (невозможность сохранить принятый пакет),
- наличие в пакете NWRITE-R ошибки ErrorDataLost ,
- наличие в пакете NWRITE-R ошибки ErrorNoData.

4 Программная модель потоков данных

Для приема сегментов потока данных целевая программа должна выделить приёмные буфера. Количество приёмных буферов и количество сегментов данных, которые необходимо помещать в каждый из приёмных буферов, определяет пользователь в соответствии со своей задачей. Должно быть выделено не менее двух буферов для каждого принимаемого потока данных. Размеры сегментов данных во всех приёмных буферах должны быть равны между собой. Приёмные буфера могут отличаться по количеству сегментов. Привязка адреса текущего буфера к полям ADR0, ADR1 регистрового файла

КВСК осуществляется функциями библиотеки ВСК прозрачно для пользователя.

Для идентификации потока данных, обеспечения приема данных в буфера и контроля целостности библиотека ВСК использует совокупность структур, определяемую как «приемный канал». Приемные каналы инициализируются функциями библиотеки ВСК. При инициализации приемного канала ему автоматически присваивается номер (идентификатор). Идентификатор связывает между собой структуры приемного канала и используется функциями библиотеки.

Приемный канал описывается структурой VskSource, в которой указываются следующие атрибуты:

- идентификатор приёмного канала,
- РИО-идентификатор источника данных (микросхемы 1890ВГ18Я),
- номер канала (1 или 2) контроллера ВСК,
- цвет данных (красный или зелёный),
- размер сегмента данных, в словах,
- количество сегментов в приёмном буфере,
- количество приёмных буферов,
- предполагаемая частота поступления данных в приёмный канал (число сегментов за секунду, используется только в технологических целях при отладке прикладных программ).

С приемным каналом связана структура VskChannelState, поля которой отражают текущее состояние регистрового файла и другую служебную информацию, позволяющую контролировать состояние канала. В частности, поле flags содержит информацию об ошибках приема данных потока.

Приемные каналы инициализируются функциями библиотеки ВСК. Поддерживаются два способа инициализации: 1) с передачей атрибутов канала и массива адресов приемных буферов в параметрах функции инициализации, 2) с передачей атрибутов каналов в составе массива конфигурационных данных и автоматическим выделением памяти под приемные буфера каждого канала. В последнем случае на этапе использования конфигуратора ВСК для каждого потока данных помимо обязательных параметров, определяющих настройку аппаратуры, должны дополнительно указываться параметры «программной модели» (количество сегментов в приёмном буфере, количество приёмных буферов).

При инициализации приёмного канала открывается приём данных в первые два буфера из массива приемных буферов.

В библиотеке ВСК реализованы функции запуска и останова приема данных по приемному каналу, а также функции чтения данных из приемного канала (с ожиданием данных или без ожидания). Функция чтения возвращает указатель на новые принятые данные (на «новый» сегмент), или 0 — в случае, если новые данные отсутствуют. После того, как получен последний сегмент текущего приёмного буфера, функция осуществляет переключение приёма сегментов данных в следующий приёмный буфер. По исчерпании массива приемных буферов в качестве очередного выбирается первый буфер массива.

5 Функциональность конфигуратора ВСК

Конфигуратор ВСК функционирует на ИЭВМ под управлением ОС Linux. С помощью графического интерфейса конфигуратора на основании информации, вводимой пользователем, на ИЭВМ создается описание конфигурации устройств ВСК мультипроцессорной системы, которое сохраняется в файле конфигурации ВСК.

Конфигуратор обеспечивает ввод следующей информации о РИО-идентификаторах устройств ВСК:

- РИО-идентификаторы микросхем 1890ВГ18Я, используемых для приема/передачи данных в формате ВСК;
- РИО-идентификаторы принимающих процессоров;
- РИО-идентификатор главного процессора.

РИО-идентификаторы могут вводиться как в числовом, так и в символьном виде. В последнем случае файл списка идентификаторов (заголовочный файл в формате языка Си) должен подаваться на вход конфигуратора ВСК.

Для каждой микросхемы 1890ВГ18Я конфигуратор позволяет ввести следующие данные:

-признаки активности встроенного генератора данных для каналов №1 и №2 контроллера ВСК;

- направление потоков данных красного и зеленого цвета в каналах №1 и №2 контроллера ВСК;
- изменение цвета данных в каналах А и В каналов №1 и №2 контроллера ВСК;
- параметры объединения пачек данных в контроллере ВСК;
- распределение сегментов данных пачки между принимающими процессорами 1890ВМ7Я;
- РИО-идентификаторы дополнительных процессоров 1890ВМ7Я для приема данных в режиме Multicasting;
- перегруппировка 64-разрядных слов данных в каналах А и В каналов №1 и №2;
- признак ожидания контроллером ВСК ответа от принимающего процессора после отправки сегмента данных.

Опционально для каждого выходного потока контроллера ВСК могут быть сохранены параметры программной модели приемного канала: РИО-идентификатор контроллера ВСК, номер канала (1 или 2), цвет данных, размер сегмента данных, количество сегментов в приёмном буфере, количество приёмных буферов, частота поступления сегментов данных.

На рисунках 1 и 2 приведен внешний вид окон конфигуратора ВСК.

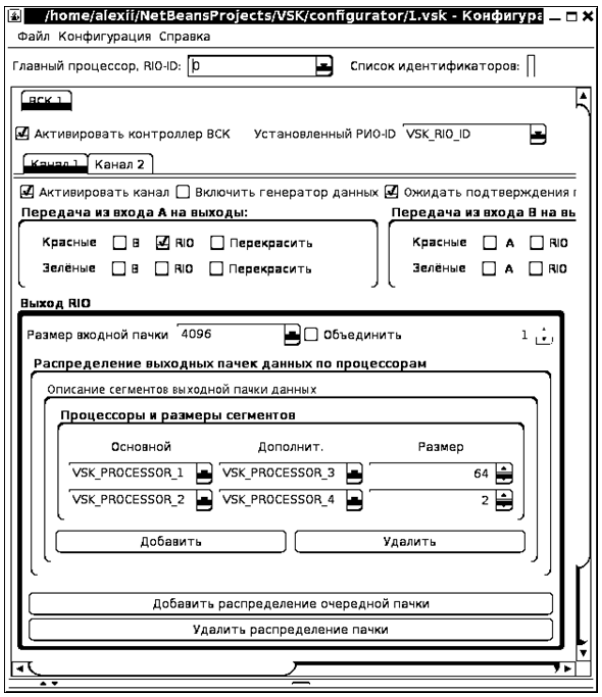


Рис. 1 Распределение пачки данных красного цвета по четырём процессорам с использованием режима Multicasting

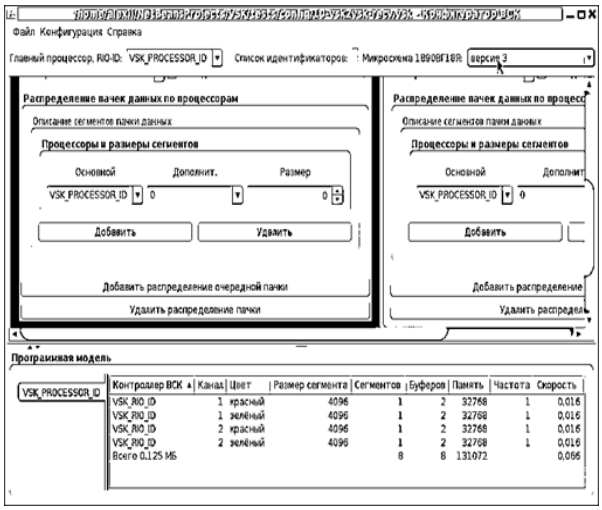


Рис. 2. Настройка программной модели приемного канала

Конфигуратор сохраняет данные в файле конфигурации ВСК. Имя файла имеет расширение «.vsk».

Файл конфигурации ВСК состоит из двух секций:

- секция формального описания конфигурации контроллеров ВСК,
- секция массива конфигурационных данных.

Секция формального описания конфигурации начинается со строки «#if 0», заканчивается строкой «#endif» и содержит введенные пользователем данные в виде комментариев в формате языка Си. Секция массива конфигурационных данных представляет собой массив 32-разрядных слов типа int, представленных в шестнадцатеричном формате. Каждый элемент массива в соответствии с внутренним протоколом ПП ВСК-РИО интерпретируется одним из следующих способов:

- как РИО-идентификатор конфигурируемого контроллера ВСК (микросхемы 1890ВГ18Я),
- как РИО-идентификатор принимающего процессора КОМДИВ128-РИО,
- как адрес и слово данных получателя конфигурируемого контроллера ВСК (для отправки по сети RapidIO),
- как адрес и слово данных принимающего процессора КОМДИВ128-РИО (для записи в регистры КВСК процессора),
- как служебное слово.

Интерпретация массива осуществляется функциями библиотеки ВСК.

Файл конфигурации ВСК в части секции формального описания может быть изменен с помощью текстового редактора или с помощью графического интерфейса конфигуратора. Специальная опция конфигуратора позволяет привести в соответствие секцию массива конфигурационных данных с обновленным содержанием секции формального описания конфигурации.

6 Функциональность библиотеки ВСК

Функции библиотеки ВСК исполняются на процессорах КОМДИВ64-РИО, КОМДИВ128-РИО под управлением ОС РВ в KERNEL-процессе и в защищенном POSIX-процессе.

Библиотека ВСК использует в качестве входных данных файлы конфигурации ВСК, созданные с помощью конфигуратора ВСК.

Библиотека ВСК обеспечивает выполнение следующих действий:

- удаленное конфигурирование контроллеров ВСК с процессоров КОМДИВ64-РИО, КОМДИВ128-РИО и локальное конфигурирование КВСК процессоров КОМДИВ128-РИО в соответствии с описанием конфигурации,
- запуск и останов устройств ВСК,
- прием потоков данных ВСК на процессорах КОМДИВ128-РИО,
- формирование на процессорах КОМДИВ64-РИО, КОМДИВ128-РИО пачек данных в формате ВСК и передачу их в контроллер ВСК по сети RapidIO,
- работа с прерываниями КВСК процессора КОМДИВ128-РИО (что позволяет осуществлять контроль потери данных).

Запуск и останов КВСК осуществляется с локального процессора КОМДИВ128-РИО. Запуск и останов контроллеров ВСК могут осуществляться с процессоров КОМДИВ64-РИО, КОМДИВ128-РИО.

Функции библиотеки ВСК в соответствии с их назначением можно условно разделить на следующие группы:

- функция инициализации драйвера ВСК,
- функции создания структур приемных каналов,
- функции конфигурирования устройств ВСК,
- функции инициализации устройств ВСК и приемных каналов,

- функции старта приема данных,
- функции останова приема данных,
- функции приема данных на процессорах КОМДИВ128-РИО,
- функции передачи данных в контроллер ВСК по RapidIO,
- функции работы с прерываниями КВСК процессора КОМДИВ128-РИО,
- функции доступа к флагам и структуре состояния приемного канала,
- отладочные и служебные функции.

Инициализация драйвера ВСК осуществляется функцией int vskDeviceInit (). Функция исполняется только в ядре операционной системы (процесс KERNEL) при старте ОС РВ, для ее выполнения необходимо специальным образом сконфигурировать образ ОС РВ.

Функции создания структур приемных каналов:

int vskChannels() - получить список приёмных каналов, заданных в файле конфигурации ВСК,

int vskSources() - заполнить массив структур источников данных приёмного канала.

Функции конфигурирования устройств ВСК:

int vskConfig() - конфигурирование устройств ВСК из заданного списка РИО-идентификаторов,

int vskConfigInitStart() - конфигурирование устройств ВСК, инициализация приёмных каналов, активация КВСК, синхронный запуск контроллеров ВСК, выделение памяти для приёмных буферов в соответствии с массивом конфигурационных данных,

int vskSendConfig - отправить пачку конфигурационных данных в выбранный канал контроллера ВСК.

Функции конфигурирования работают следующим образом. На процессоре КОМДИВ128-РИО конфигурируется локальный КВСК. Если процессор (КОМДИВ64-РИО или КОМДИВ128-РИО) опознает себя в качестве «главного», то он осуществляет удаленное конфигурирование контроллеров ВСК с заданными (явно или в массиве конфигурационных данных) РИО-идентификаторами. Перед конфигурированием выполняется останов всех указанных в конфигурации контроллеров ВСК (блокируется передача данных).

Функции инициализации устройств ВСК и приемных каналов:

int vskChannelInit() - инициализация приёмного канала,

void vskReceiverSetActive() - активация КВСК для приёма данных.

Функции старта приема данных:

void vskChannelStart() - стартовать передачу данных от контроллера ВСК для выбранного приёмного канала (по идентификатору канала),

void vskStart() - стартовать передачу данных от избранного источника (по РИО-идентификатору контроллера ВСК, номеру канала и цвету),

int vskStartAll() - стартовать передачу данных на всех контроллерах ВСК из заданного списка или из массива конфигурационных данных,

int vskStartAllSynchronized() - синхронизировать приёмники данных и запустить передачу данных на всех контроллерах ВСК.

Функции останова приема данных:

void vskChannelStop() - остановить передачу данных от контроллера ВСК для выбранного приёмного канала,

void vskStop() - остановить передачу данных от данных от избранного источника (по РИО-идентификатору контроллера ВСК, номеру канала и цвету),

int vskStopAll() - остановить передачу данных на всех контроллерах ВСК.

Функции приема данных на процессорах КОМДИВ128-РИО:

void * vskChannelGet() - получить указатель на очередной сегмент данных ВСК,

VskChannelFlags vskChannelGetSimple() - упрощённый приём данных ВСК. Функция предназначена для конфигурирования контроллеров ВСК и приёма данных ВСК от одного источника (текущий процессор должен принимать данные только от одного источника)

void * vskChannelWait() - ожидать очередной сегмент данных,

void * vskChannelWaitBuffer() - ожидать заполнения данными всего буфера - получить все сегменты данных одного буфера.

Функция передачи данных в контроллер ВСК по RapidIO:

int vskSend() - отправить пачку данных указанного цвета в выбранный канал контроллера ВСК.

Функции для работы с прерываниями КВСК:

void vskInterruptSet() - установить обработчик и маску прерываний от КВСК, сбросить установленные прерывания,

VskFlags vskReceiverFlags() - получить значения флагов прерываний КВСК, установить маску прерываний,

void vskReceiverGetCounters() - получить счётчики прерываний КВСК.

Функции доступа к флагам и структуре состояния приемного канала

VskChannelFlags vskChannelFlags() - получить значения флагов приёмного канала,

int vskChannelState() - получить копию структуры состояния приёмного канала.

В числе **служебных и отладочных функций** выделим **функции синхронизации процессоров:**

void vskEventSend() - отправить указанному процессору событие,

VskEventData vskEventWait() - ожидать событие,

void vskSynchronized() - синхронизировать локальный процессор с удалённым,

void vskSynchronizedQuickAll() - быстрая синхронизация всех процессоров.

Механизм синхронизации с использованием «событий» был разработан в силу необходимости синхронизировать конфигурирующие, принимающие процессоры и контроллеры ВСК перед началом обмена данными во избежание потери данных.

При разработке прикладных программ в начале приема рекомендуется соблюдать следующую хронологию:

- настройка всех контроллеров ВСК, участвующих в приеме данных,
- настройка приемных каналов на всех процессорах КОМДИВ128-РИО, участвующих в приеме данных,
- синхронный запуск приема данных на всех контроллерах ВСК с управляющего процессора.

7 Пример программы приёма данных ВСК от внутреннего генератора контроллера ВСК микросхемы 1890ВГ18Я

Ниже приведен исходный текст программы на языке программирования Си:

```
/*
* Генерация пачек данных по 4096 слов.
*  Микросхема генерации: 1890ВГ18Я
*  с РИО-ID = *VSK_RIO_ID.
* Выходной канал - 1. Цвет данных - красный.
*  Приём данных на процессоре
*  КОМДИВ128-РИО с РИО-идентификатором
*  VSK_PROCESSOR_ID.
* Для приёма пачек данных используются два
* буфера. Размер каждого буфера
*(SEGMENTS_COUNT * 4096) *слов.
* Память для буферов выделяется с помощью
*функции cp2c_malloc().
* Каждый буфер принимает SEGMENTS_COUNT
*сегментов данных.
* Размер сегмента - 4096 слов.
* Проверяются первые TEST_SIZE и последние
*TEST_SIZE слов каждой принятой пачки
* данных
*/
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
// Подключить заголовочный файл библиотеки
//ВСК
#include <vsk.h>
// Подключить файл определений РИО-
//идентификаторов
#include "list-RIO-ID.h"
/// Количество принимаемых пакетов
#define PACKAGE_COUNT 1000
/// Количество сегментов в буфере
#define SEGMENTS_COUNT 16
/// Размер пачки данных, в словах.
#define PACKAGE_SIZE 4096
/// Количество проверяемых слов пачки данных от
/// начала и от конца.
#define TEST_SIZE 32
int main ()
{
// Номер канала
int channel = 1;
// Цвет данных
VskColor color = VSK_RED;
// Количество приёмных буферов
int countBuffers = 2;

// Приём данных красного цвета из канала 1
static int configVSK [] =
{
#include "01-1-red.vsk"
};
// Конфигурировать контроллеры ВСК и
процессоры
vskConfig (configVSK, 0);
// Создать массив буферов для приёма пачек
//данных
int* buffers [countBuffers];
// Создать массив сегментов
int segments [countBuffers];
int i;
// Выделить память для приёмных буферов
for (i = 0; i < countBuffers; i++)
{
// Размер буфера
int size=PACKAGE_SIZE*4* SEGMENTS_COUNT;
buffers [i] = cp2c_malloc (size, 32);
// Указать количество сегментов в буфере
segments [i] = SEGMENTS_COUNT;
}
// Идентификатор ВСК
int vskID = VSK_RIO_ID;
// Инициализировать КВСК на приём данных от
//контроллера ВСК
int channelID = vskChannelInit (vskID, channel,
color, (void*) buffers,
segments, countBuffers);
// Активировать приёмник
vskReceiverSetActive (VSK_ACTIVE);
// Запустить передачу данных от ВСК
vskStartAll (configVSK, 0);
// Счётчик принятых пачек
int count = 0;
// Номер последнего ошибочного пакета
int lastCount = 0;
// Принять PACKAGE_COUNT пачек
while (count++ < PACKAGE_COUNT)
{
VskChannelFlags flags;
int* segment;
while (1)
{
// Получить сегмент данных
segment = vskChannelGet (channelID);
// Получить флаги (признаки ошибок)
flags = vskChannelFlags (channelID);
if (segment)
{break;}
}}
// Остановить передачу данных из контроллера
// ВСК
vskChannelStop (channelID);
// Освободить память буферов
for (i = 0; i < countBuffers; i++)
{ cp2c_free (buffers [i]); }
return 0; }
```

Software for data transmission and reception via high-speed serial channel in multiprocessor real-time systems

T.K. Gringauz, A.N.Onin

Abstract. Data input in multiprocessor real-time digital signal processing systems based on KOMDIV64-RIO, KOMDIV128-RIO microprocessors using RapidIO interconnect technology is fulfilled via high-speed serial channel. Conversion of data flow from serial format to PRIO format is performed by specific RapidIO devices ("high-speed channel devices"). Said devices are to be configured. Software development tool kit was created to support high-speed channel devices programming, data transmission and reception via high-speed serial channel in RapidIO real-time systems. This paper presents software architecture, functionality, operation principles.

Keywords: software, architecture, functionality

Литература

1. RapidIO Interconnect Specification (Revision 1.3) Available from: http://www.rapidio.org/specs/current.

2. С.Г.Бобков, С.И.Аряшев, М.Е.Барских, П.С.Зубковский, Е.В.Ивасюк. Высокопроизводительные расширения архитектуры универсальных микропроцессоров для ускорения инженерных расчетов. «Информационные технологии: ISSN: 1684-6400», 2014. №6. с. 27-37.

3. Научно-технический отчёт по ОКР «Разработка базовых узлов для создания субмикронных СБИС для электронных модулей ЭВМ с повышенной устойчивостью к воздействию нейтронов и тепловых режимов эксплуатации бортовой аппаратуры авиационной техники», М., НИИСИ РАН, 2011

4. Научно-технический отчёт по ОКР «Разработка системных контроллеров для высокопроизводительных микропроцессоров, М., НИИСИ РАН, 2011

5. А.Н. Годунов [и др.], Операционная система реального времени Багет 3.0. «Программные продукты и системы», Тверь, Научно-исследовательский институт «Центрпрограммсистем», 2010, № 4, с. 15 - 19

Методы оптимизации программ для процессора КОМДИВ128-РИО

М.С. Аристов

Аннотация: В работе рассмотрены методы оптимизации программ на языке ассемблера для сопроцессора цифровой обработки сигналов, входящего в состав процессора КОМДИВ128-РИО. Предложенные методы используют особенности архитектур процессора и сопроцессора таким образом, чтобы добиться высокой общей производительности. В работе приведены примеры программ на языке ассемблера с применением различных методов оптимизации. Таким образом, используя приведенные в статье методы оптимизации можно достичь пиковой производительности сопроцессора 8 Гфлопс.

Ключевые слова: метод оптимизации, программы

Введение

Задача обработки больших массивов данных в режиме реального времени является одной из актуальных проблем во многих областях науки и техники, в том числе при обработке сигналов, трехмерном моделировании, создании реалистичных изображений.

Для решения подобной задачи в НИИСИ РАН разработан и используется процессор КОМДИВ128-РИО [1].

Процессор КОМДИВ128 состоит из ядра – основного универсального процессора архитектуры КОМДИВ64 [2], DMA-контроллера и сопроцессора СР2, предназначенного для цифровой обработки сигналов. 64-разрядная архитектура сопроцессора СР2 позволяет достичь максимальной производительности – 8 Гфлопс. Особенность архитектуры сопроцессора СР2 состоит в том, что сопроцессор СР2 последовательно выполняет вычислительные операции над несколькими потоками данных (SIMD). Подобная архитектура накладывает определенные ограничения на возможности сопроцессора СР2, однако подобное решение позволяет достичь высокой производительности в случае цифровой обработки сигналов.

Сопроцессор СР2 имеет собственную память, поэтому перед запуском и после завершения работы СР2 необходимо выполнять пересылку данных между ОЗУ основного процессора и ОЗУ сопроцессора СР2. Задачу передачи данных между ОЗУ основного процессора и ОЗУ сопроцессора СР2 решает DMA-контроллер. Сопроцессор СР2, основной процессор и DMA-контроллер работают параллельно и независимо друг от друга, что можно использовать при обработке сигнальных данных.

В состав сопроцессора СР2 входят одна управляющая и четыре вычислительных секции. Каждая команда сопроцессора СР2 исполняется параллельно на всех вычислительных секциях. В одной команде сопроцессора СР2 сочетаются одна арифметическая команда и одна управляющая. Арифметическая команда выполняет арифметические действия над данными, хранящимися в регистрах вычислительных секций. Управляющая команда осуществляет чтение данных из регистров или запись данных в регистры. В одной команде сопроцессора СР2 одновременно задаются арифметическая и управляющая команды. Исполнение арифметической и управляющей команд, размещенных в одной команде сопроцессора СР2, происходит параллельно.

Используя особенности параллельной работы DMA-контроллера и сопроцессора СР2 можно, например, запустить чтение/запись данных, используя DMA-контроллер в первых половинах памяти вычислительных секций СР2, работу программы параллельно над данными – во вторых половинах памяти СР2, а по окончанию пересылки данных и работы СР2 запустить СР2 над первыми половинами памяти СР2, а чтение/запись над вторыми половинами памяти СР2. Подобная техника параллельной работы независимых устройств позволяет увеличить общую производительность в сравнении с последовательным запуском каждой из стадий работы программы.

Существуют различные методы оптимизации программ сопроцессора СР2, некоторые методы можно использовать в комбинации с другими.

Регистры сопроцессора СР2 64-разрядные, а архитектура команд устроена таким образом, что позволяет размещать в одном 64-разрядном регистре два 32-разрядных числа. Вычислительные операции так же оперируют 64-разрядными данными, что позволяет в одной вычислительной секции выполнять операции над двумя 32-разрядными числами за один такт.

В сопроцессоре СР2 выполнение всех команд конвейеризовано, то есть команда может быть передана на исполнение до того, как будет готов результат предыдущей команды. Количество тактов, за которое исполняется команда, зависит от команды. Для арифметических операций конвейер имеет длину восемь тактов, для управляющих – три такта. Таким образом, после запуска, например, команды сложения вещественных чисел, результат будет записан только на восьмом такте, а параллельная управляющая команда чтения из памяти секции – на третьем такте. Различные команды сопроцессора СР2 отрабатывают за различное количество тактов.

Обозначенная архитектурная особенность сопроцессора СР2 дает два возможных метода оптимизации вычислений. Первый метод

подразумевает развертку цикла, используя свободные регистры и объединяя схожие команды в блоки по восемь или шестнадцать команд.

Второй метод – это конвейеризация. Конвейеризацию необходимо использовать, если для развертки недостает регистров. Однако, так как результат работы команды записывается в регистр не сразу, а например на восьмом такте, то до записи результата в регистр его можно использовать в других вычислительных командах.

Настоящая статья посвящена методам оптимизации программ цифровой обработки сигналов с целью эффективного использования сопроцессора CP2.

1. Алгоритм работы CP2

Алгоритм работы сопроцессора CP2 состоит в следующем. Код, выполняемый на универсальном процессоре под управлением операционной системы, загружает в память инструкций сопроцессора CP2 программу (написанную на языке ассемблера для CP2 [3], откомпилированную и обработанную редактором связей), а в память данных сопроцессора CP2 – данные, используемые программой CP2, после чего инициирует выполнение загруженной программы на CP2. По окончании выполнения программы на CP2 устанавливается флаг, доступный для проверки коду, выполняемому на универсальном процессоре. Проверив данный флаг, можно выгрузить результат работы из памяти данных CP2 для дальнейшей обработки в среде операционной системы.

2. Оптимизация программы для CP2

При написании программ на языке ассемблера для CP2 нужно учитывать возможные оптимизации, что позволяет существенно сократить время вычислений на CP2. Ниже будут рассмотрены следующие методы оптимизации:

- «ps-параллелизм»;
- развертка цикла и конвейеризация;
- параллельное исполнение вычислительных и управляющих команд;
- совмещение вычислений и DMA-пересылок.

2.1. Метод «ps-параллелизм»

Метод «ps-параллелизм» состоит в том, чтобы использовать команды, работающие параллельно над парами действительных чисел везде, где это возможно.

Например, комплексное число хранится в регистре CP2 целиком, в старшей половине регистра действительная часть числа, а в младшей – мнимая. Действия над действительной и мнимой частями комплексного числа выполняются параллельно. Система команд сопроцессора CP2 позволяет при работе с действительными числами загрузить в такой регистр пару 32-разрядных чисел одинакового типа (**`float`** или **`int`**), лежащих в одном 64-разрядном машинном слове, и некоторые действия выполнять над этой парой чисел параллельно.

Однако не все команды CP2 позволяют использовать «ps-параллелизм». Например, команда **`sqrt`** [4] работает только с половинкой регистра, следовательно, для нее этот оптимизирующий прием недоступен.

2.2. Развертка цикла и конвейеризация

В сопроцессоре CP2 выполнение всех команд конвейеризовано, то есть запускаемая команда начинает работать до того, как будет готов результат предыдущей команды.

Различные команды сопроцессора CP2 отрабатывают за различное количество тактов. Например, результат работы команды **`sw`** [4], которая считывает данные из регистра CP2 и записывает их в память вычислительной секции, готов на третьем такте, а результат работы команды **`psadd`** [4], которая складывает пары вещественных чисел, готов на восьмом такте.

В приведенной ниже программе, осуществляющей сложение двух вещественных векторов, используются пустые команды **`nop`** [4]. Использование пустых команд необходимо для того, чтобы результат работы предыдущих команд был записан в регистр до исполнения следующей команды, которая подразумевает использование этого результата.

```
lw f1, (r1) + 1  ## Чтение данных по адресу
           ## из регистра r1 в регистр f1
           ## с постинкрементом регистра a1
lw f2, (r2) + 1  ## Чтение данных по адресу
           ## из регистра r2 в регистр f2
           ## с постинкрементом регистра a2
nop;;nop         ## Ожидание завершения
           ## выполнения команд на конвейере.
           ## Время готовности
           ## команд чтения - 3 такта
psadd f1, f1, f2 ## В регистр f1 помещаются
           ## два элемента вектора, равных
           ## сумме соответствующих элементов
           ## входных векторов
nop;;nop;;nop;;  ## Ожидание завершения
nop;;nop;;nop;;  ## выполнения команд
nop;;      ## на конвейере. Время готовности
           ## вычислительных команд - 7 тактов
sw f1, (r3) + 1  ## Запись по адресу,
           ## содержащемуся в регистре r3,
           ## данных из регистра f1
           ## с постинкрементом регистра a3
```

Данная программа расходует такты следующим образом:

- на считывание двух элементов каждого входного вектора – два такта;
- ожидание завершения команд обмена данными на конвейере – два такта;
- сложение пар прочитанных элементов – один такт;
- ожидание завершения арифметической операции на конвейере – семь тактов;
- запись результата – один такт.

Итого на считывание, сложение и запись двух элементов векторов затрачивается 13 тактов, значит время, затраченное на чтение, сложение и запись одного элемента каждого вектора в этой программе составляет 6.5 тактов.

Для оптимизации вышеприведенной программы, используя особенность архитектуры сопроцессора СР2, можно использовать два метода:

– развертка цикла в несколько раз (обычно в 4, 8 или 16 раз);

– конвейеризация цикла.

Эти методы допускают совместное использование.

Метод развертки цикла – это повторение команд с использование доступных регистров таким образом, чтобы пустые команды оказались не нужны. Развертка используется тогда, когда итерации цикла не зависят друг от друга по данным, или от этой зависимости можно избавиться, например, заменив одно накапливаемое значение на несколько. Аргументы команд внутри развернутого цикла готовы перед их вызовом, и нет необходимости вставлять пустую команду **nop** [4].

Если развертка возможна, то выполнить ее легко, и она дает большой прирост производительности, до восьми раз.

В приведенном выше примере развертку можно выполнить, используя свободные регистры, что позволит заменить пустые команды на вычислительные. В регистры **f0** – **f7** можно поместить 16 элементов первого вектора, а в регистры **f10** – **f17** - 16 элементов второго вектора. Результат сложения соответствующих элементов векторов можно поместить в регистры **f20** – **f27**. В завершение работы цикла выполняется запись 16 элементов из регистров **f20** – **f27** в соответствующие элементы вектора-суммы.

Ниже приведен пример программы, осуществляющей сложение двух вещественных векторов, оптимизированный с использованием метода развертки цикла:

```
ld f0, (r1) + 2  ## Чтение по адресу
        ## из регистра r1 в регистры f0 и f1
        ## с постинкрементом регистра a1
        ## Можно считать, что инкрементация
        ## регистра происходит «мгновенно»,
        ## таким образом, на момент выполнения
        ## следующей загрузки a1 содержит
        ## адрес 3-го элемента вектора
ld f2, (r1) + 2  ## Чтение по адресу
        ## из регистра r1 в регистры f2 и f3
        ## с постинкрементом регистра a1
ld f4, (r1) + 2  ## Чтение по адресу
        ## из регистра r1 в регистры f4 и f5
        ## с постинкрементом регистра a1
ld f6, (r1) + 2  ## Чтение по адресу
        ## из регистра r1 в регистры f6 и f7
        ## с постинкрементом регистра a1
ld f10, (r2) + 2 ## Чтение по адресу
        ## из регистра r2 в регистры f10 и f11
        ## с постинкрементом регистра a2
ld f12, (r2) + 2 ## Чтение по адресу
        ## из регистра r2 в регистры f12 и f13
        ## с постинкрементом регистра a2
ld f14, (r2) + 2 ## Чтение по адресу
        ## из регистра r2 в регистры f14 и f15
        ## с постинкрементом регистра a2
ld f16, (r2) + 2 ## Чтение по адресу
        ## из регистра r2 в регистры f16 и f17
        ## с постинкрементом регистра a2
do g0, Loop  ## Цикл с количеством итераций
             ## из регистра g0 и меткой Loop
  psadd f20, f0, f10  ## В регистр f20
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f21, f1, f11  ## В регистр f21
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f22, f2, f12  ## В регистр f22
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f23, f3, f13  ## В регистр f23
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f24, f4, f14  ## В регистр f24
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f25, f5, f15  ## В регистр f25
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f26, f6, f16  ## В регистр f26
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f27, f7, f17  ## В регистр f27
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  ld f0, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f0 и f1
        ## с постинкрементом регистра a1
  ld f2, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f2 и f3
        ## с постинкрементом регистра a1
  ld f4, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f4 и f5
        ## с постинкрементом регистра a1
  ld f6, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f6 и f7
        ## с постинкрементом регистра a1
  ld f10, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f10 и f11
        ## с постинкрементом регистра a2
  ld f12, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f12 и f13
        ## с постинкрементом регистра a2
  ld f14, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f14 и f15
        ## с постинкрементом регистра a2
  ld f16, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f16 и f17
        ## с постинкрементом регистра a2
  sd f20, (r3) + 2    ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f20 и f21
        ## с постинкрементом регистра a3
  sd f22, (r3) + 2    ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f22 и f23
        ## с постинкрементом регистра a3
```

```
  sd f24, (r3) + 2    ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f24 и f25
        ## с постинкрементом регистра a3
Loop: sd f26, (r3) + 2  ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f26 и f27
        ## с постинкрементом регистра a3
```

Данная программа внутри цикла, обозначенного меткой **Loop**, расходует такты следующим образом:

– на считывание по четыре элемента каждого входного вектора – восемь тактов;

– сложение прочитанных элементов – восемь тактов;

– запись четырех элементов выходного вектора – четыре такта.

Итого на считывание, сложение и запись 16 элементов векторов затрачивается 20 тактов. В данной программе на чтение, суммирование и запись одного элемента выходного вектора тратится 1.25 такта. Таким образом, используя данный метод, можно достичь увлечение производительности 5.2 раза. Однако следует отметить, что при работе с векторами размерности, не кратной 16, часть тактов будет работать впустую.

Но с методом развертки цикла могут быть связаны такие проблемы, как:

– дефицит регистров;

– нехватка кодов условий.

Дефицит регистров может возникнуть из-за того, что развертка цикла подразумевает использование дополнительных регистров, однако количество регистров сопроцессора СР2 ограничено, что может привести к невозможности использования этого метода. Аналогичная проблема касается использования кодов условий, так как в сопроцессоре СР2 всего семь кодов условий.

Для решения приведенных выше проблем можно использовать более универсальный метод оптимизации – конвейеризацию цикла. Основан этот метод на учете времени готовности результатов и использовании «старых» данных из регистра, пока вычисляются «новые». У этого метода есть существенный недостаток: вручную его выполнять трудоемко, а полученную программу сложно модифицировать.

Приведенный выше пример программы, осуществляющей сложение двух вещественных векторов, модифицированный с условием использования меньшего числа регистров, выглядит следующим образом:

```
ld f0, (r1) + 2    ## Чтение по адресу
        ## из регистра r1 в регистры f0 и f1
        ## с постинкрементом регистра a1
ld f10, (r2) + 2   ## Чтение по адресу
        ## из регистра r2 в регистры f10 и f11
        ## с постинкрементом регистра a2
ld f2, (r1) + 2    ## Чтение по адресу
        ## из регистра r1 в регистры f2 и f3
        ## с постинкрементом регистра a1
ld f12, (r2) + 2   ## Чтение по адресу
        ## из регистра r2 в регистры f12 и f13
        ## с постинкрементом регистра a2
do g0, Loop   ## Цикл с количеством итераций
              ## из регистра g0 и меткой Loop
  psadd f20, f0, f10  ## В регистр f20
        ## помещаются два элемента  вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f21, f1, f11  ## В регистр f21
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f22, f2, f12  ## В регистр f22
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  ld f0, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f0 и f1
        ## с постинкрементом регистра a1
  ld f10, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f10 и f11
        ## с постинкрементом регистра a2
  psadd f23, f3, f13  ## В регистр f23
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  ld f2, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f2 и f3
        ## с постинкрементом регистра a1
  ld f12, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f12 и f13
        ## с постинкрементом регистра a2
  psadd f20, f0, f10  ## В регистр f20
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f21, f1, f11  ## В регистр f21
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f22, f2, f12  ## В регистр f22
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  psadd f23, f3, f13  ## В регистр f23
        ## помещаются два элемента вектора,
        ## равных сумме соответствующих
        ## элементов входных векторов
  sd f20, (r3) + 2    ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f20 и f21
        ## с постинкрементом регистра a3
  sd f22, (r3) + 2    ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f22 и f23
        ## с постинкрементом регистра a3
  ld f0, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f0 и f1
        ## с постинкрементом регистра a1
  ld f10, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f10 и f11
        ## с постинкрементом регистра a2
  ld f2, (r1) + 2     ## Чтение по адресу
        ## из регистра r1 в регистры f2 и f3
        ## с постинкрементом регистра a1
  ld f12, (r2) + 2    ## Чтение по адресу
        ## из регистра r2 в регистры f12 и f13
        ## с постинкрементом регистра a2
  sd f20, (r3) + 2    ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f20 и f21
        ## с постинкрементом регистра a3
```

```
Loop: sd f22, (r3) + 2 ## Запись по адресу,
        ## содержащемуся в регистре r3,
        ## из регистров f22 и f23
        ## с постинкрементом регистра a3
```

Как видно из этого примера, в одном цикле по-прежнему обрабатываются 16 элементов, но для выполнения задачи использовано не 24 регистра (**f0** – **f7**, **f10** – **f17**, **f20** – **f27**), а только 12 (**f0** – **f3**, **f10** – **f13**, **f20** – **f23**). Этот метод оптимизации – единственный способ увеличить производительность при работе с битами условий, так как в СР2 этих битов всего семь. Время готовности результатов сравнения составляет семь тактов, что делает невозможным использование простой развертки программ, в которых есть команды сравнения.

Как и в предыдущем варианте программы, последние четыре команды загрузки при проходе последнего цикла работают впустую.

2.3. Параллельное исполнение вычислительных и управляющих команд

Полная команда сопроцессора СР2 компонуется из двух 32-разрядных элементарных команд. Эти команды входят в состав условных групп, допускающих параллельный запуск в течение одного такта: группы арифметических команд и группы операций пересылки и управления. Выполняются они параллельно на разных конвейерах. Следовательно, в зависимости от готовности аргументов, за один такт рабочей частоты можно выполнить вычислительную и управляющую операции. Подобное совмещение вычислительных и управляющих команд может увеличить производительность до двух раз.

Приведенный выше пример программы, измененный с использованием возможности совмещения вычислительных и управляющей операций, представлен ниже:

```
ld f0, (r1) + 2
ld f10, (r2) + 2
ld f2, (r1) + 2
ld f12, (r2) + 2
do g0, Loop
      psadd f20, f0, f10 ld f0, (r1) + 2
      psadd f21, f1, f11 ld f10, (r2) + 2
      psadd f22, f2, f12 ld f2, (r1) + 2
      psadd f23, f3, f13 ld f12, (r2) + 2
      psadd f24, f0, f10 ld f0, (r1) + 2
      psadd f25, f1, f11 ld f10, (r2) + 2
      psadd f26, f2, f12 ld f2, (r1) + 2
      psadd f27, f3, f13 ld f21, (r2) + 2
      psadd f20, f0, f10 ld f0, (r1) + 2
      psadd f21, f1, f11 ld f10, (r2) + 2
      psadd f22, f2, f12 ld f2, (r1) + 2
      psadd f23, f3, f13 ld f12, (r2) + 2
      psadd f24, f0, f10 sd f20, (r3) + 2
      psadd f25, f1, f11 sd f22, (r3) + 2
      psadd f26, f2, f12 sd f24, (r3) + 2
      psadd f27, f3, f13 sd f26, (r3) + 2
      ld f0, (r1) + 2
      ld f10, (r2) + 2
      ld f2, (r1) + 2
      ld f12, (r2) + 2
      sd f20, (r3) + 2
      sd f22, (r3) + 2
      sd f24, (r3) + 2
Loop: sd f26, (r3) + 2
```

Данная программа внутри цикла, обозначенного меткой **Loop**, расходует такты следующим образом:

- на считывание по четыре элемента каждого входного вектора – четыре такта;
- сложение прочитанных элементов + загрузка/запись – 16 тактов;
- запись результата – четыре такта.

Итого на считывание, сложение и запись 32 элементов векторов затрачивается 24 такта. Таким образом, в данной программе на чтение, суммирование и запись одного элемента каждого вектора тратится 0.75 такта. Однако следует отметить, что при работе с векторами размерности, не кратной 32, часть тактов будет работать впустую.

2.4. Совмещение вычислений и DMA-пересылок

DMA-контроллер и сопроцессор СР2 могут работать параллельно, поэтому можно запланировать выполнение вычислений параллельно с выгрузкой готовых и загрузкой новых данных.

Память каждой вычислительной секции СР2 делится на две части: младшую и старшую половины. При организации параллельной работы DMA-контроллера и сопроцессора СР2 необходимо следить, чтобы пересылки и вычисления работали с разными половинами памяти. В противном случае результат работы программы может оказаться неверным.

Для оптимального распределения очереди пересылок и запуска программы на СР2 следует использовать временные диаграммы.

Пример такой временной диаграммы показан на рис. 1.

Рис. 1. Временная диаграмма параллельной работы DMA-контроллера и сопроцессора СР2

Использование этого метода может дать прирост производительности до двух раз.

Заключение

Используя приведенные выше оптимизационные методы:

- «ps-параллелизм»;
- конвейерное исполнение команд;
- параллельное исполнение вычислительных и управляющих команд;

– совмещение вычислений и DMA-пересылок; можно достичь увеличения производительности в 34.5 раза по сравнению с исходным примером, приведенным в п.2.

Program optimization techniques for KOMDIV128-RIO processor

M.S. Aristov

Abstract: This paper presents assembly language program optimization techniques for digital signal coprocessor, the part of the processor KOMDIV128-RIO. The proposed techniques use coprocessor architecture features to achieve a high overall performance. Some examples of the assembly language programs using various optimization techniques are presented. Peak coprocessor performance (8 Gflops) could be achieved by use of the optimization techniques.

Keywords: method of optimization, programs

Литература

1. Микросхема интегральная 1890ВМ7Я (КОМДИВ128-РИО). Указания по применению. М.: НИИСИ РАН, 2009

2. Архитектура КОМДИВ для программистов. Том 1. Введение в архитектуру КОМДИВ. М.: НИИСИ РАН, 2010

3. Программное изделие «Ассемблер для специализированного сопроцессора СР2 в составе микропроцессора КОМДИВ128-РИО (АССК128)». Руководство программиста. М.: НИИСИ РАН, 2014

4. Микросхема интегральная 1890ВМ7Я (КОМДИВ128-РИО). Система команд. М.: НИИСИ РАН, 2009

Способ инициализации сети RapidIO с оконечными устройствами, имеющими разную разрядность идентификатора

Г.А. Лавринов

Аннотация: в статье рассматривается способ инициализации коммуникационной среды RapidIO с оконечными устройствами, имеющими идентификаторы 8 и 16 бит. Приведен алгоритм инициализации системы с динамическим обходом устройств RapidIO. Описан способ разбиения на домены коммуникационной среды для последующей инициализации и приведен пример такого разбиения.

Ключевые слова: RapidIO, алгоритм инициализации, маршрутизация, домен.

Коммуникационный интерфейс RapidIO допускает 8-ми и 16-ти битную адресацию оконечных устройств (ОУ). 8-ми битный адрес (идентификатор ОУ), позволяющий адресовать до 256 ОУ, существенно ограничивает производительность вычислительной системы. На сегодняшний день уже существует отечественная элементная база ОУ и коммутаторов с разной разрядностью идентификаторов. При этом 8-ми битными идентификаторами (ID) обладают микропроцессоры, ориентированные на обработку сигналов. Коммутаторы и универсальные микропроцессоры могут работать и с 8-ми битными, и с 16-ти битными ID. Для прикладного программного комплекса, реализуемого на этой элементной базе, в общем случае требуется больше чем 256 ОУ. Для ОУ различаются и записываются в разные поля настройки таблицы маршрутизации коммутаторов используется два метода: для 8-разрядных ID – стандартный, используется только локальная таблица (см. рис. 1), а для 16 разрядных ID – иерархический, используется либо локальная таблица для младшего байта ID, либо глобальная для старшего байта, в зависимости от значения поля BASE в регистре коммутатора ROUTE_BASE (см. рис. 2) [1]. Причем в стандартном методе маршрутизации может поддерживаться 256 ОУ с 16-разрядным ID с номерами в диапазоне от 256 до 511. Выбор метода маршрутизации осуществляется битом LUT_512 регистра порта коммутатора PORT_CONFIG. ОУ с поддержкой адресации 16-битных ID могут работать и с 8-битными ID, причем 8- и 16-битные ID у одного регистра BASE_DEVICE_ID [2]. Далее рассматри-

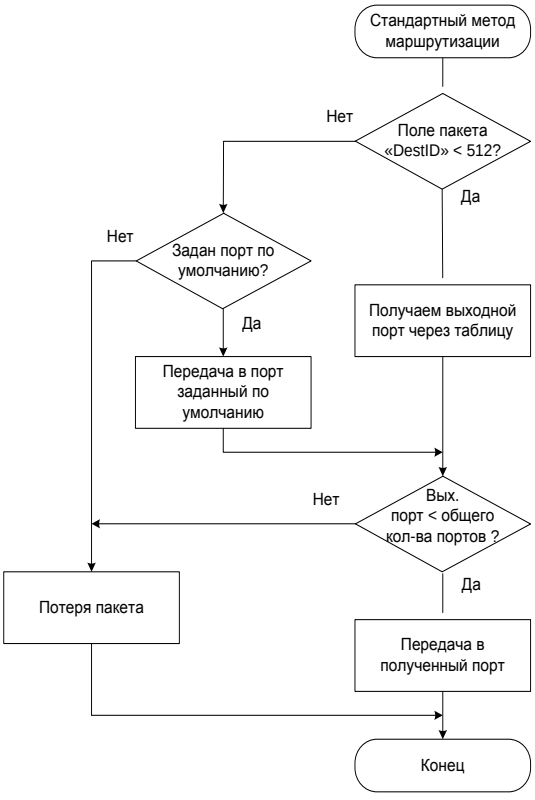


Рис. 1. Алгоритм стандартного метода маршрутизации

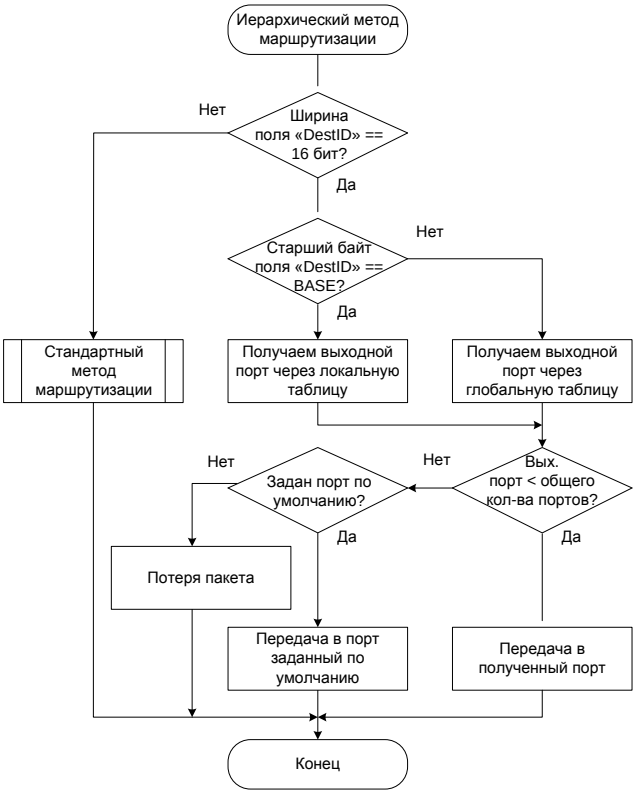


Рис. 2. Алгоритм иерархического метода маршрутизации

вается система из ОУ с разной разрядностью ID, при этом коммутаторы должны иметь возможность работать как с 8-и, так и 16-и разрядными полями пакета ID отправителя и приемника.

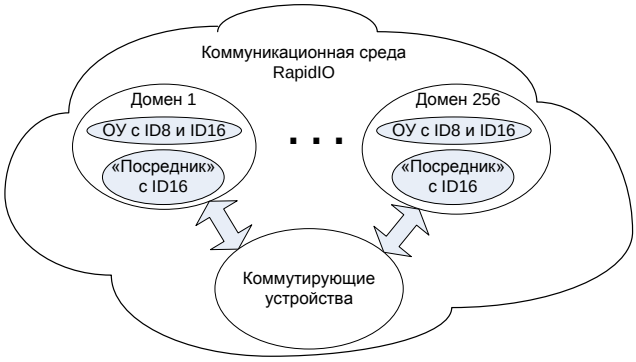


Рис. 3. Разбиение ОУ RapidIO на домены

Принцип подхода следующий: все пространство RapidIO делится на домены, в каждом домене выделяется «посредник» - это ОУ с 16-разрядным ID (см. рис. 3). С его помощью ОУ с 8-разрядным ID будет передавать данные ОУ из другого домена. Каждый домен имеет максимально возможное количество устройств 256. Так как, в случае иерархического механизма маршрутизации, для ОУ с 8-разрядным ID выделяется локальная таблица маршрутизации в коммутаторе, а домен при этом характеризуется числом в регистре ROUTE_BASE, при не совпадении старшего байта 16-разрядного ID идет выборка маршрутизации по глобальной таблице. Поэтому при инициализации ОУ с 8-разрядным ID необходимо настроить локальную таблицу маршрутизации в рамках домена. Каждое ОУ с 8-разрядным ID в домене должно знать 8-разрядное ID своего «посредника» в домене для передачи данных в другой домен. Сведения о «посреднике» могут быть записаны в переменные окружения программного обеспечения (ПО), либо может быть использован регистр RapidIO ОУ «Component и Tag» [2]. При таком подходе, каждое ОУ должно иметь программную поддержку, помимо приема/передачи сообщений по RapidIO, прием/передачу команды на исполнение другому ОУ [3].

Для инициализации среды RapidIO используются 2 способа: динамический и статический. На рисунке 4 приведен алгоритм обхода устройств RapidIO при динамической инициализации. Для начала необходимо знать функциональный состав вычислительного комплекса, чтобы заранее определить: возможен ли обмен между всеми ОУ. Известно, что количество доменов может быть 256, ОУ в домене тоже 256, «посредник» в домене 1. Тогда если после первого обхода выясняется, что количество «посредников» недостаточно, то полноценная передача от одного ОУ ко всем не возможна. Если же достаточно, то производится анализ на возможность работы в каждом домене ОУ, так как локальная таблица маршрутизации для каждого порта коммутатора должна быть настроена для определенного домена и не должна пересекаться с настройкой для других доменов. Только после этого ОУ программно разбиваются на домены, и применяется механизм иерархической маршрутизации с присвоением младшего байта 16-разрядного ID ОУ с 8-разрядным ID. После инициализации, у инициатора формируется массив структур вида:

{номер домена, ID ОУ, разрядность ОУ};

На основе этой структуры можно передавать данные ОУ, либо напрямую минуя «посредника», либо через него. Недостатком динамического алгоритма инициализации является отсутствие настройки маршрутов между всеми ОУ системы. Поэтому при таком подходе, возможен обмен между устройством, осуществляющем инициализацию, со всеми остальными и наоборот. А значит, после динамического обхода системы, необходимо проинициализировать маршруты между остальными устройствами. С целью ускорения передачи, в динамической инициализации потребуется передать массив структур посредством команд загружаемого ПО.

Разбиение на домены коммуникационной среды RapidIO применимо для разнородной системы (устройства с разной адресацией ID). Предлагается рассмотреть один из способов разбиения на домены, в случае динамического обхода среды. После начального обхода без присвоения ID подсчитывается количество ОУ в системе, количество коммутаторов и сохраняются маршруты к этим устройствам. По этим маршрутам определяется количество ОУ для каждого

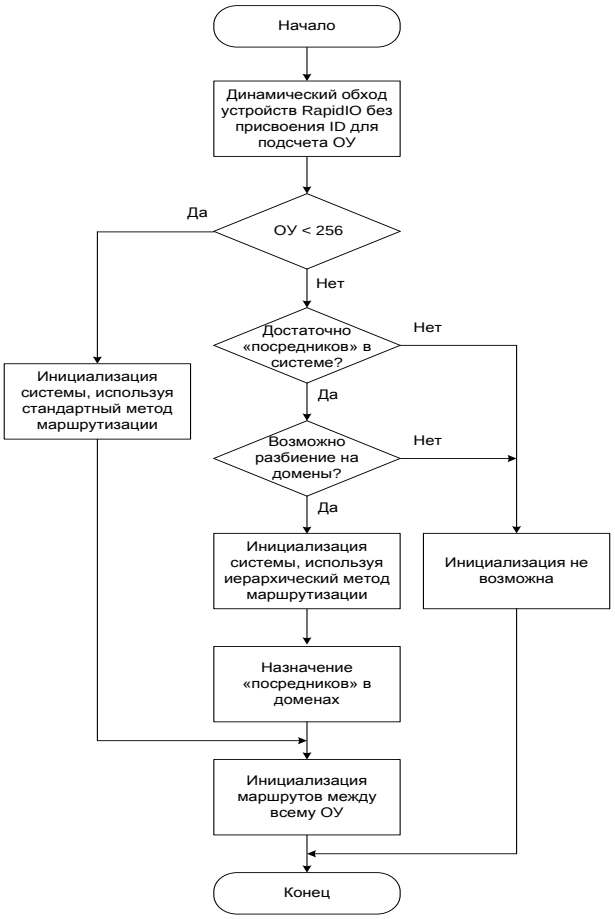


Рис. 4. Алгоритм инициализации системы с динамическим обходом ОУ с 8- и 16-разрядными ID

коммутатора. Другими словами, это те ОУ, в которых маршруты включают в себя маршрут коммутатора, для которого это количество подсчитывается. После этого, для каждого домена определен состав коммутаторов и ОУ для дальнейшей инициализации.

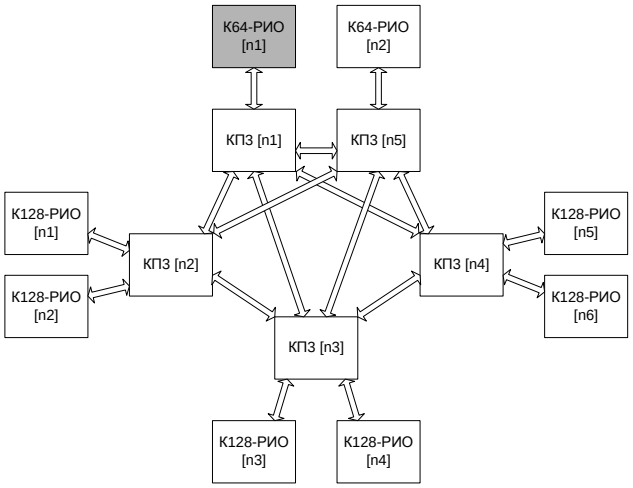


Рис. 5. Схема соединений RapidIO устройств

Для наглядного примера, рассматривается разбиение на домены небольшой системы, на основе коммутаторов КП3 (микросхема 1890КП3Я, с поддержкой 8- и 16-разрядной адресацией пакетов) и ОУ – процессоров К64-РИО (микросхема 1890ВМ6Я, 8- и 16-разрядная адресация на RapidIO) и К128-РИО (микросхема 1890ВМ7Я, 8-разрядная адресация на RapidIO). С максимально возможным количеством ОУ в домене равным пяти. Схема соединений RapidIO устройств показана на рисунке 5. Инициатор среды на рисунке подкрашен. После динамического обхода определятся количество ОУ (2 К64-РИО и 6 К128-РИО), коммутаторов (5 КП3) и их маршрутов. Диаграмма маршрутов оконечных устройств RapidIO приведена на рисунке 6. На диаграмме видно, что маршрут устройства КП3[n3] («n» сокр. от «number») входит в состав маршрутов 2-ух коммутаторов КП3[n4], [n5] и 5 ОУ К64-РИО[n2], К128-РИО[n3], [n4], [n5], [n6]. Тогда эти устройства войдут в домен 2, а оставшиеся в домен 1. После этого назначаются “посредники” в доменах: в домене 1 – это К64-RIO[n1], а в домене 2 – это К64-РИО[n2].

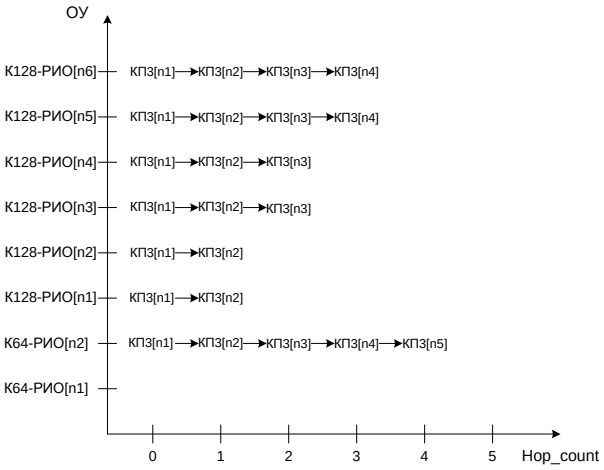


Рис. 6. Диаграмма маршрутов ОУ

Достоинством данного подхода является возможность обмена между устройствами RapidIO с 8- и 16-разрядными ID. Недостатком является увеличение времени передачи данных из-за участия в передачах устройств-«посредников». Кроме того, перед разбиением ОУ на домены требуется анализ связей между ними, либо изначально должен быть определен состав вычислительной машины, исходя из применения данного подхода.

Method to initialize RapidIO network with endpoints having different bits of the identifier

G.A. Lavrinov

Abstract. The article considers the approach to initialize RapidIO system with endpoints having identifiers 8 and 16 bits. Adduces an algorithm for system initialization with dynamic bypass of RapidIO devices. It discloses a method of partitioning into domains of the system for subsequent initialization and gives an example of such a partition.

Keywords: RapidIO, algorithm of initialization, routing, domain.

Литература

1. http://www.idt.com/products/interface-connectivity/serial-rapidio-solutions/rapidio-switches (страница, посвященная коммутаторам RapidIO компании IDT).
2. RapidIO Interconnect Specification. Part 3: Common Transport Specification. Revision 3.0. RapidIO Trade Association, 2013.
3. RapidIO Interconnect Specification. Annex 2: Session Management Protocol Specification. Revision 3.0. RapidIO Trade Association, 2013.

Способ интеграции электронного документооборота подразделения и делопроизводства учреждения

Г.Л.Левченкова

Аннотация: Настоящая статья является развитием темы удобства использования электронной информационной среды подразделения (документооборота [1]), доработанной с целью соблюдения требований ведения делопроизводства в учреждении.

Ключевые слова: документооборот, система автоматизации документооборота, управление записями, учет.

Введение

В большинстве организаций (и предприятий) в настоящее время внедряется электронное делопроизводство, и утверждаются "Инструкции по делопроизводству", устанавливающие, кроме всего прочего, порядок регистрации и хранения документов. По большому счету, организацию делопроизводства можно условно считать решающей две основных задачи: 1-я - обеспечение своевременного и грамотного создания документов и 2-я - организация работы с документами (получение-передача, обработка, учёт, регистрация, контроль, хранение, подготовка в архив, уничтожение). В силу индивидуальности решаемых проблем, сложившихся подходов, наличия технологических решений и исходя из технических возможностей организации создают свое информационное пространство. Это оправдано, поскольку у каждого предприятия своя специфика. Но с определенного момента, а именно с момента принятия стандарта ГОСТ Р 53898-2013 «Системы электронного документооборота. Взаимодействие систем управления документами. Технические требования к электронному сообщению», который устанавливает формат, состав и содержание электронного сообщения, обеспечивающего информационное взаимодействие систем управления документами, предприятия вынуждены переходить на более унифицированные, стандартизованные системы, предъявляющие определенные требования к ведению делопроизводства. Поддержка функционирования обновленных корпоративных информационных сред эксплуатации и модификации информационных систем в условиях быстро изменяющегося информационного общества отражается на процессах документопотоков, которые, в конечном итоге попадая в подразделения организации, должны быть соответствующим образом учтены. Часто изменяющиеся требования к поддержке информации о поступающих и исходящих, а также о создаваемых в подразделениях документах налагает практически непосильные обязательства на рядовых сотрудников учреждений, которым просто-напросто некогда заниматься вплотную вопросами учета и регистрации документов, проходящих через их подразделения.

В настоящей статье приводятся содержательные идеи, алгоритмы и методы интеграции "требований делопроизводства" и "документооборота, поддерживаемого в подразделениях" организации.

1. Общие положения

Делопроизводство призвано обеспечивать создание официальных документов (регистрацию), организацию их учета, движения и хранения. Понятие делопроизводства базируется на понятии документирования. Документирование – запись информации на различных носителях по установленным правилам.

Под "Документооборотом" понимается совокупность взаимосвязанных процедур, обеспечивающих движение документов с момента их создания, исполнения или отправки.

Документопоток – совокупность документов, выполняющих определенное целевое назначение в процессе документооборота.

Под "Документом" понимаются данные, зафиксированные на материальном носителе информации с реквизитами, позволяющими их идентифицировать (различить). Документы - носители первичной информации. Именно в документах информация фиксируется впервые.

Принято выделять централизованный, децентрализованный документооборот и документооборот уровня структурного подразделения.

В "централизованный" документооборот входит вся документация, подлежащая централизованной регистрации. За обеспечение централизованного документооборота отвечает, как правило, специальное подразделение, а все структурные единицы работают по единым утвержденным правилам.

При "децентрализованном" документообороте правила и методики, сформулированные в центральном аппарате, в самостоятельных структурных единицах могут быть изменены в

соответствии с характером деятельности или сложившимися традициями.

Документы, учитываемые только в структурных подразделениях, составляют "документооборот уровня структурного подразделения", и ответственность за организацию работы с такими документами несет секретарь структурного подразделения.

Именно о "документообороте уровня структурного подразделения" идет речь в данной статье ("Система ДОКУМЕНТООБОРОТ. [1]).

Обозначим, с какими именно документами имеет дело предприятие (организация), и с какими – рядовые сотрудники. Логично предположить, что с точки зрения "подлинности" все документы делятся на две группы: 1-я - оригиналы (происходит от латинского слова "originalis", что означает - первоначальный, самобытный; Как синоним слова оригинал употребляется "подлинник", что в переводе с латинского означает достоверный). 2-я – копии (повторное точное воспроизведение подлинников, включая все его внешние признаки; в настоящее время широко распространены ксерокопии и отсканированные электронные документы; обратим внимание на то, что копия не имеет юридической силы). Очевидно, что рядовые сотрудники в основном имеют дело с копиями документов. В подразделениях могут храниться оригиналы, но в работе в основном используются копии. Копии (или оригиналы), поступившие в подразделение, имеют реквизиты, присвоенные в делопроизводстве предприятия. В подразделении эти же документы (или их копии) учитываются как поступившие, и им вполне вероятно присваиваются еще какие-то идентификационные номера.

Итак, "делопроизводство предприятия" зафиксировало оригинал документа и присвоило ему регистрационный номер, а затем копия этого документа (а, может быть, и оригинал) попадает в подразделение и ей присваивается "номер по документообороту подразделения". Копии зарегистрированного документа могут передаваться в несколько подразделений, соответственно и количество учетных номеров множится. Единственное, что можно отметить, при занесении информации о поступившем документе (копии) в "документооборот подразделения", для того, чтобы однозначно можно было идентифицировать документ в дальнейшем, - это регистрационный номер, присвоенный в "делопроизводстве предприятия".

2. Актуальность

Рассмотрим разновидности задач, решаемых отдельно взятым подразделением организации, с точки зрения документопотока. Очевидно, что ряд задач связан с рассмотрением поступающих сведений, часть документов создается в процессе работы и, разумеется, оформляются ответы на запросы или формируются вопросы, представляющие собой исходящие документы.

По-видимому, основное требование, предъявляемое к "документообороту подразделения" – нужно уметь быстро найти нужный документ. Занося же в учетную систему подразделения только регистрационные номера, присвоенные "делопроизводством предприятия" и другие данные, извлеченные из него (делопроизводства), подразделение может потерять ориентацию в поисках нужного документа, потому что "в одну папку" такие документы не сложишь, а сложишь в разные – не найдешь. Да и если в одну папку сложить, то как их потом найти, если их сотни, а порядковых номеров нет? Поэтому приходится ставить свои, подразделенческие порядковые номера, но при этом заносить еще уйму всякой дополнительной информации для ускорения и упрощения дальнейшего поиска нужного документа. На самом деле это не такое уж трудоемкое занятие, но требует определенной доли аккуратности. Поскольку требования к информационному учету изменяются быстрее, чем внедряются соответствующие программные продукты, рекомендуем использовать всем хорошо известную распространенную программу "Microsoft Excel" для создания системы документооборота в подразделении [1]

В основе системы "документооборот"[1] используется самый простой принцип интеграции – "Информационно-ориентированный". Он основан на использовании одной и той же информации двумя и более системами, а для обеспечения работы со своей информацией у каждой системы имеется набор своих процедур.

Информационные системы зачастую обладают различными функционалом, логикой, архитектурой и форматом хранения данных. Причём большая часть таких систем создавалась разными разработчиками для решения определённых задач, а, следовательно, системы содержат лишь самые простые (на уровне передачи информации файлами) механизмы интеграции с другими системами. Но для занесения и использования данных в "Microsoft Excel" этого вполне достаточно.

В общем случае, в делопроизводстве каждый документ, отнесенный к числу регистрируемых, получает свой регистрационный номер, состоящий из порядкового номера в пределах регистрируемого массива документов, который, исходя из задач поиска, дополняется индексами по номенклатуре дел, классификаторам корреспондентов, исполнителей и др. Но, отметим, что для работы сотрудникам нужно иметь доступ также и к документам (справкам, например, или сводкам), которые не подлежат регистрации в "делопроизводстве предприятия". Кроме того, по номеру, присвоенному делопроизводством не всегда можно определить к какой категории относится данный документ. Например документ с номером № 01-22/2167 может оказаться служебной запиской

или письмом, а чтобы это узнать, нужно или анализировать адресата либо смотреть на копию документа, что не всегда удобно, да и время тратится неконструктивно. Информационная среда должна быть обозримой, понятной и удобной в использовании. Но у "делопроизводства" свои задачи, а у "документооборота подразделения" другие цели. Поэтому будем "затачивать" систему "документооборота подразделения" (систему ДОКУМЕНТООБОРОТ) так, чтобы удовлетворить нужды и запросы сотрудников, но в то же время соответствовать требованиям "делопроизводства".

3. Принципы занесения информации

Задачи, которые ставились при создании системы ДОКУМЕНТООБОРОТ для использования в подразделении либо на местах, следующие:

- структура учёта должна быть проста для понимания, наглядна, легка в применении;
- трудозатраты на поиск и выдачу справочных, статистических и других данных должны быть минимальны;
- доступ к информации об имеющихся материалах должен быть обеспечен всем заинтересованным лицам.

Система ДОКУМЕНТООБОРОТ разработана на базе "Microsoft Excel", при этом одна (каждая) учетная запись занимает одну строку (информация о документе (копии) располагается в одной строке), что позволяет, во-первых, выбирать (отбирать) нужные строки по фильтрам, и, во-вторых, по своему усмотрению добавлять столбцы (и менять их при необходимости местами), в которые можно заносить информацию согласно нуждам подразделения или требованиям делопроизводства. Кроме того, использование возможности создания гиперссылок в файлах-оглавлениях позволяет быстро получать доступ к документу в электронном виде.

Использование "Microsoft Excel" позволяет минимизировать системные требования к персональному компьютеру пользователя, снизить расходы на установку дополнительного программного обеспечения и обучение пользователей работе с ним.

Для соблюдения корректности структуры информационного поля системы ДОКУМЕНТООБОРОТ при занесении информации рекомендуется следовать основным принципам:

Прозрачность

Вопросы "где взять", "где посмотреть", "где лежит" возникают довольно часто. Поэтому одним из основных принципов хранения должна быть прозрачность. Любому должно быть понятно, где получить необходимую информацию.

Место

Для обеспечения прозрачности поместим всю информацию об учёте и хранении в одно место. Причём "местом" можно определить и каталог, и файл, и папку, и полку.

Каждому - своё

Третий принцип – "каждому – свое". Для каждой папки должна быть доступна своя опись. В каждом каталоге лежат "свои" файлы. У каждой темы – свое оглавление. И сотрудник, в ведении которого находится занесение учётных данных в систему, должен быть, в идеале, один. но может быть и у каждой части системы свой "ведущий". Документ, который имеет отношение к нескольким темам, несомненно, должен быть виден в каждой из них.

Доступность

Четвёртый принцип – доступность. Следует организовать учёт и хранение таким образом, чтобы к поиску информации был доступ у каждого заинтересованного лица. Применение этого принципа организации учёта позволяет не нагружать сверх меры ответственного за ведение документооборота поиском необходимых данных. Всё можно найти самому.

Аккуратность и чёткость

Пятый принцип – аккуратность и чёткость занесения данных. От того, насколько качественно (без опечаток) будут занесены данные и ссылки на взаимосвязанные материалы в систему учёта, зависит результативность последующих операций с документами. На один документ рекомендуется заводить по одной строке (за исключением особо выделенных случаев, описанных в последующих разделах).

Унификация

Принцип шестой – унификация терминологии, используемой при занесении информации, что существенно упрощает поиск объектов по ключевым словам.

Ключевые слова

Принцип седьмой – вносить как можно больше ключевых слов. Как в название организации-отправителя-получателя (для внешних документов, так и в "тему документа" ("О чем")). Каждое такое слово – дополнительный критерий отбора при поиске. В делопроизводстве заносится лишь самая общая информация о документах, зачастую вносится лишь "тема", указанная вверху документа, например "Протокол совещания № 21" или "О направлении счета", или "О производстве микросхем". И как, скажите на милость, из такого "оглавления" понять, о чем идет речь, не вытаскивая текст документа и не просматривая его глазами? Поэтому в дополнение (а то и вместо) к словам "Протокол совещания № 21" (который, кстати, уместно было бы видеть не в графе "О чем", а в колонке "Номер документа") следует написать "Вопросы применения микросхемы 1990КП8А в Системе "Бобер-23" (ОКР "Практика-3)". Вместо "О направлении счета" – "Прислали счет на оплату соединителей СНП" или "Мы выставили счет на оплату услуг по сборке изделия "Варан-5"". Вместо "О производстве микросхем" - "Запрашивают

информацию о количестве микросхем 1456ВУ45, предполагаемых к производству в 2015 году" или "Изменение технологии производства микросхем серии 3336".

Отметим, что для сотрудников подразделения обычно привычными являются понятия:

- входящий служебный документ (сл.записка, справка, сводка, приказ, распоряжение, …) – то, что создается в процессе работы и не выходит за пределы организации;

- входящий внешний документ (письмо, справка, сводка, книга, ГОСТ, …) – документ, созданный в сторонней организации;

- совместный документ - утвержденный представителями нескольких организаций (может быть как входящим, так и исходящим. Обычно он входящий, поскольку в подразделение передается (входит) копия. Но если этот документ создан сотрудником подразделения, то он может также быть зарегистрирован как исходящий (со взаимными ссылками - на входящую подписанную копию у электронного проекта и на электронный вид документа у полученной подписанной копии);

- исходящий служебный документ;

- исходящий "во вне" документ – направляемый в стороннюю организацию;

- хранящийся документ - если в ведении подразделения находятся документы, подлежащие хранению. Хранящиеся документы обычно подразделяются на "темы", то есть объединяются по тематике.

Исходя из этих предпосылок логично предположить, что чтобы эффективно искать документ (копию) следует создать всего около пяти-семи электронных описей (файлов-оглавлений list.xls), в которые заносить учетные данные документопотоков, проходящих через подразделение. В нашем примере (система ДОКУМЕНТООБОРОТ [1]) предложена следующая структура информации;

- ВХОДЯЩИЕ ДОКУМЕНТЫ (включая распорядительные, внешние и служебные);
- ИСХОДЯЩИЕ (подразделяются на СЛУЖЕБНЫЕ и "ДЛЯ СТОРОННИХ ОРГАНИЗАЦИЙ");
- ТЕМАТИЧЕСКИЕ (могут подразделяться на ТЕМЫ, перечни, списки адресов, …).

В файлах-оглавлениях обязательно нужно предусмотреть столбец, где будет заноситься регистрационный номер документа, полученный (проставленный) в "делопроизводстве организации". Это единственный номер, который однозначно идентифицирует документ в рамках предприятия (см. рисунок 1).

Обязательны к заполнению столбцы "в ответ на" (в связи с чем получен или направлен документ) и "ответная реакция на документ" (отчеты-результаты). Такие столбцы могут содержать помимо ссылок на электронные версии документа порядковый учетный номер по подразделенческому ДОКУМЕНТООБОРОТУ (чтобы можно было посмотреть на связанный документ воочию).

4. Дополнительные данные

Различают следующие виды документов:

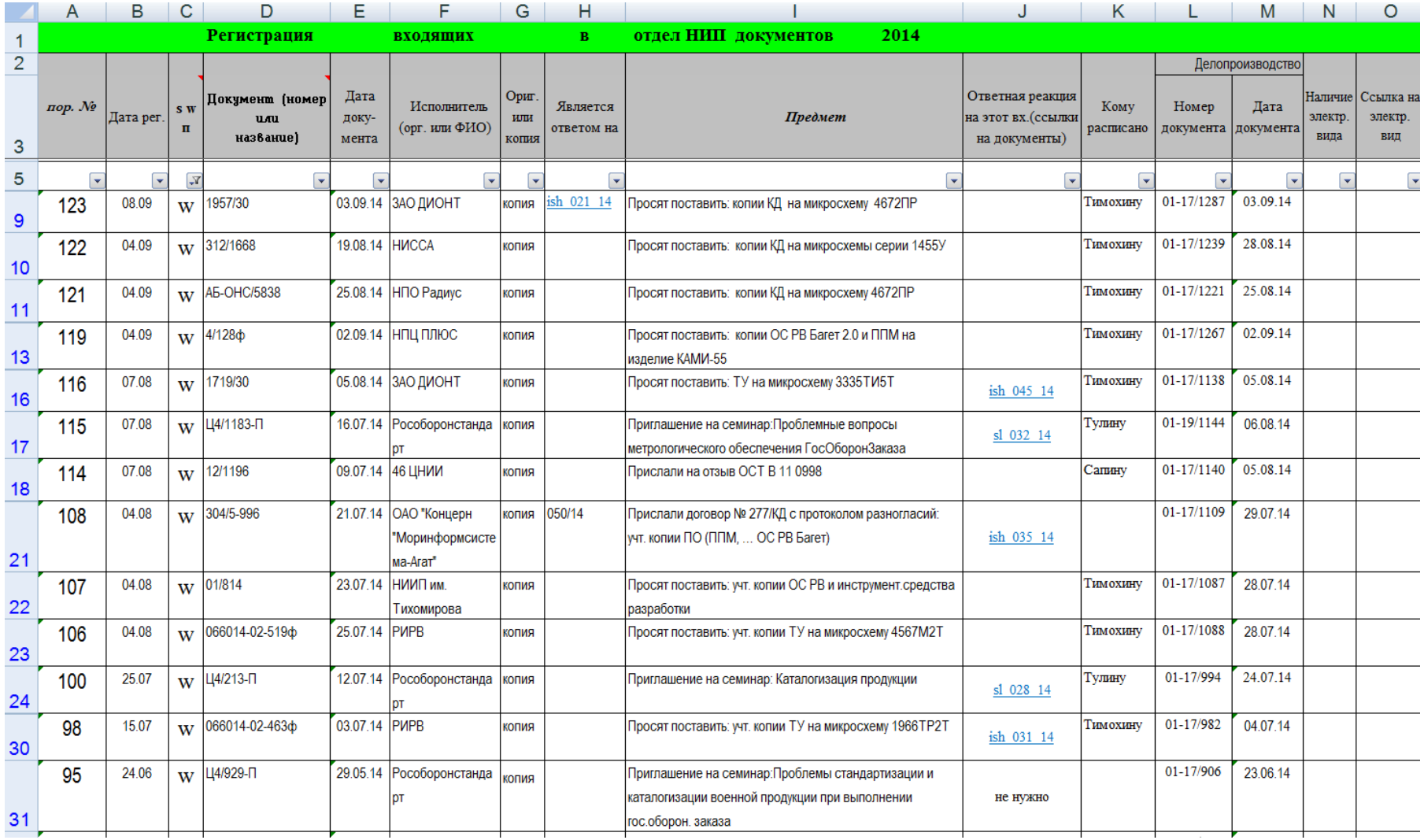
- по назначению (средства фиксации фактов (протоколы, отчеты) и средства передачи документов (письмо, телеграмма);
- по происхождению (служебные и личные документы);
- по форме (индивидуальные, типовые, трафаретные);
- по срокам исполнения и хранения (постоянные, долговременные, временные);
- по месту составления (внутренние и внешние (из сторонних организаций), совместные (решения, акты передач);
- по наименованию (проказы, указы, отчеты);
- по стадиям создания (выписки, отпуск, оригиналы);
- копии или оригиналы.

Вне зависимости от того, как именно идентифицируют некий "документ" в делопроизводстве, в системе ДОКУМЕНТООБОРОТ должен быть свой критерий поиска. Поэтому для занесения сведений об этих "классификациях" следует завести дополнительные колонки (столбцы).

Так, например, для входящих документов можно рекомендовать занесение символов w (внешний документ), s (служебный), п (распорядительный документ (приказ, распоряжение), …), в определенную колонку как признак папки (см. рисунок 1). Именно так, как того требует делопроизводство – такие ПАПКИ должны стоять в подразделении на полке (и иметь идентификатор "дело"). Если сотрудник хочет посмотреть какой-то входящий документ, то зайдя в список-оглавление входящих документов (list_whod.xls) и выбрав там по описанию нужный документ, он, ориентируясь на пометки столбца, в котором обозначено "где лежит", откроет соответствующую папку и обнаружит в ней документ под учетным (в системе ДОКУМЕНТООБОРОТ) порядковым номером. В данном случае порядковые учетные номера в папках будут с пропусками, потому что "подряд" они зарегистрированы во всех ВХОДЯЩИХ, а после распределения по папкам в списках-оглавлениях папок будут появляться "дыры". Оглавление папки легко получить отобрав в файле list_whod.xls строки по фильтру. Указав "w" в колонке фильтра с "принадлежностью к папке" мы получим оглавление папки ВХОДЯЩИЕ ВНЕШНИЕ ДОКУМЕНТЫ. (см. рисунок 2).

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Регистрация					входящих		в	отдел НИП документов 2014						
2												Делопроизводство			
3	пор. №	Дата рег.	s w п	Документ (номер или название)	Дата доку-мента	Исполнитель (орг. или ФИО)	Ориг. или копия	Является ответом на	Предмет	Ответная реакция на этот вх.(ссылки на документы)	Кому расписано	Номер документа	Дата документа	Наличие электр. вида	Ссылка на электр. вид
7															
8	124	24.09	S	сл.зап. 01-23/1567	23.09.14	Барин	ориг.	087/14	Просят организовать доступ к базе КД и ПД (электронному архиву)	отработано 24.09.14.	Рямову	01-23/1567	23.09.14		
9	123	08.09	W	1957/30	03.09.14	ЗАО ДИОНТ	копия	ish_021_14	Просят поставить: копии КД на микросхему		Тимохину	01-17/1287	03.09.14		
10	122	04.09	W	312/1668	19.08.14	НИССА	копия		Просят поставить: копии КД на микросхемы		Тимохину	01-17/1239	28.08.14		
11	121	04.09	W	АБ-ОНС/5838	25.08.14	НПО Радиус	копия		Просят поставить: копии КД на микросхему		Тимохину	01-17/1221	25.08.14		
12	120	04.09	П	ПРИКАЗ № П-107/А	11.08.14	Белин	копия		А.А.Шилов - ИО директора 25-26.08.2014	к сведению		№ П-107/А	11.08.14		
13	119	04.09	W	4/128ф	02.09.14	НПЦ ПЛЮС	копия		Просят поставить: копии ОС РВ и ПТМ		Тимохину	01-17/1267	02.09.14		
14	118	04.09	S	сл.зап. 01-22/2303	03.09.14	Караев	ориг.		Просят передать в ОРВС копии КД на 1907КХ018	сл.зап. 01-22/2331 05.09.14	Митько	01-22/2303	03.09.14		
15	117	31.07	П	ПРИКАЗ № П-106/А	08.08.14	Белин	копия		Е.А.Смирнов - ИО директора 12.08.2014	к сведению		№ П-106/А	08.08.14		
16	116	07.08	W	1719/30	05.08.14	ЗАО ДИОНТ	копия		Просят поставить: ТУ на микросхему 3335ТИ5Т	ish_045_14	Тимохину	01-17/1138	05.08.14		
17	115	07.08	W	Ц4/1183-П	16.07.14	Рособоронстанда рт	копия		Приглашение на семинар:Проблемные вопросы метрологического обеспечения ГосОборонЗаказа	sl_032_14	Тулину	01-19/1144	06.08.14		
18	114	07.08	W	12/1196	09.07.14	46 ЦНИИ	копия		Прислали на отзыв ОСТ В 11 0998		Сапину	01-17/1140	05.08.14		
19	113	29.07	S	сл.зап. 01-22/1836	11.07.14	Тировин	ориг.		Просят откорректировать стандарт САНП.7653.98	sl_013_14		01-22/1836	11.07.14		
20	112	05.08	S	Протокол № 156	24.07.14	Капов НИПА	копия		Нам передали: копию подписанного протокола совместного совещания по микросхемам 1919МИ (ОКР Вагон)	копия - в ТЕМУ ОКР Вагон	в_дело			есть	sl_812_14

Рис. 1. Вид файла-оглавления каталога "ВХОДЯЩИЕ"



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Регистрация входящих в отдел НИП документов 2014														
2												Делопроизводство			
3	пор. №	Дата рег.	s w п	Документ (номер или название)	Дата документа	Исполнитель (орг. или ФИО)	Ориг. или копия	Является ответом на	Предмет	Ответная реакция на этот вх.(ссылки на документы)	Кому расписано	Номер документа	Дата документа	Наличие электр. вида	Ссылка на электр. вид
5															
9	123	08.09	w	1957/30	03.09.14	ЗАО ДИОНТ	копия	ish_021_14	Просят поставить: копии КД на микросхему 4672ПР		Тимохину	01-17/1287	03.09.14		
10	122	04.09	w	312/1668	19.08.14	НИССА	копия		Просят поставить: копии КД на микросхемы серии 1455У		Тимохину	01-17/1239	28.08.14		
11	121	04.09	w	АБ-ОНС/5838	25.08.14	НПО Радиус	копия		Просят поставить: копии КД на микросхему 4672ПР		Тимохину	01-17/1221	25.08.14		
13	119	04.09	w	4/128ф	02.09.14	НПЦ ПЛЮС	копия		Просят поставить: копии ОС РВ Багет 2.0 и ППМ на изделие КАМИ-55		Тимохину	01-17/1267	02.09.14		
16	116	07.08	w	1719/30	05.08.14	ЗАО ДИОНТ	копия		Просят поставить: ТУ на микросхему 3335ТИ5Т	ish_045_14	Тимохину	01-17/1138	05.08.14		
17	115	07.08	w	Ц4/1183-П	16.07.14	Рособоронстандарт	копия		Приглашение на семинар:Проблемные вопросы метрологического обеспечения ГосОборонЗаказа	sl_032_14	Тулину	01-19/1144	06.08.14		
18	114	07.08	w	12/1196	09.07.14	46 ЦНИИ	копия		Прислали на отзыв ОСТ В 11 0998		Сапину	01-17/1140	05.08.14		
21	108	04.08	w	304/5-996	21.07.14	ОАО "Концерн "Моринформсистема-Агат"	копия	050/14	Прислали договор № 277/КД с протоколом разногласий: учт. копии ПО (ППМ, ... ОС РВ Багет)	ish_035_14		01-17/1109	29.07.14		
22	107	04.08	w	01/814	23.07.14	НИИП им. Тихомирова	копия		Просят поставить: учт. копии ОС РВ и инструмент.средства разработки		Тимохину	01-17/1087	28.07.14		
23	106	04.08	w	066014-02-519ф	25.07.14	РИРВ	копия		Просят поставить: учт. копии ТУ на микросхему 4567М2Т		Тимохину	01-17/1088	28.07.14		
24	100	25.07	w	Ц4/213-П	12.07.14	Рособоронстандарт	копия		Приглашение на семинар: Каталогизация продукции	sl_028_14	Тулину	01-17/994	24.07.14		
30	98	15.07	w	066014-02-463ф	03.07.14	РИРВ	копия		Просят поставить: учт. копии ТУ на микросхему 1966ТР2Т	ish_031_14	Тимохину	01-17/982	04.07.14		
31	95	24.06	w	Ц4/929-П	29.05.14	Рособоронстандарт	копия		Приглашение на семинар:Проблемы стандартизации и каталогизации военной продукции при выполнении гос.оборон. заказа	не нужно		01-17/906	23.06.14		

Рис. 2. Оглавление папки входящие внешние документы

Важным может оказаться выделение отдельных колонок для занесения следующей информации:

1) "Оригинал" или "копия"

2) Ответственные за выполнение поручений (либо ФИО исполнителя проекта документа – для ИСХОДЯЩИХ).

3) Где находится оригинал

Рассмотрим случай, когда в подразделение поступил оригинал документа "Протокол внутреннего совещания по порядку разработки микросхемы 1122ТИ в ОКР "Прибой-4"", изначально созданный (набран текст и распечатан документ) сотрудником этого подразделения. Запись о наличии этого протокола может фигурировать в нескольких файлах-оглавлениях, но оригинал следует тоже куда-то положить и должным образом учесть. Данный протокол, во-первых, учтен в оглавлении "исходящих служебных документов" (там же находится электронный вид документа-проекта), во-вторых, оригинал документа, видимо, может попасть в папку (дело), где хранятся оригиналы внутренних протоколов (если нет, то оригинал останется в папке "Входящих"), в третьих, копия протокола, по идее, должна попасть в тематическую папку – либо к микросхемам 1122ТИ либо к ОКР "Прибой-4", а, может быть, и туда и сюда, в четвертых, протокол должен быть зарегистрирован как входящий служебный документ с отметкой о дате поступления и ссылками, "где лежит оригинал" (учетный номер документа в соответствующей папке), ссылкой на номер электронного вида (проекта) в папке исходящих, ссылкой на тематическую папку, где также размещена копия этого документа.

4) Признак наличия документа в электронном виде – очень полезная вещь. Всегда нужно знать, доступен ли документ для редактирования или извлечения информации с целью её использования в создании других документов. Рекомендуется ставить ссылку на соответствующий регистрационный номер по ДОКУМЕНТООБОРОТУ в целях ускорения поиска электронного вида документа. Или, в идеале, устанавливать в поле со ссылкой на документ гиперссылку На рисунке 1 sl_812_14 означает, что электронный вид документа находится в каталоге исходящих служебных документов подразделения, зарегистрирован в файле-оглавлении 2014 года под номером 812, а подчеркивание символов (sl_812_14) означает, что в данном месте установлена гиперссылка на электронный вид документа. Кстати, поскольку рекомендуется именовать файлы согласно порядковому рег. номеру в файле-оглавлении, то и имя файла у документа скорее всего имеет вид sl_812_14.расширение.

5) Признак окончательной отработки документа. Несмотря на то, что какие-то действия по работе с документом уже привели к определенным результатам (написана ответная служебная записка или письмо, или составлен протокол, или проект приказа…), и в графе "Ответная реакция на документ" уже появились

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	Регистрация					входящих		в	отдел НИП документов	2014					
	пор. №	Дата рег.	s w п	Документ (номер или название)	Дата доку-мента	Исполнитель (орг. или ФИО)	Ориг. или копия	Является ответом на	Предмет	Ответная реакция на этот вх.(ссылки на документы)	Кому расписано	Делопроизводство: Номер документа	Делопроизводство: Дата документа	Наличие электр. вида	Ссылка на электр. вид
2	49	02.04	W	Положение о закуп.	17.03.14	Фирма КОНЦЕПТ	копия		Положение о закупке товаров, работ, услуг	к сведению				есть	049/14
3	49	02.04	S	Информ.письмо (эл.почта)	27.03.14	Варина	копия		Об организации закупок в рамках принятого Положения о закупке товаров, работ, услуг	к сведению				есть	049/14
4	49	02.04	П	ПРИКАЗ № П-45	27.03.14	Белин	копия		Об организации закупок в рамках принятого Положения о закупке товаров, работ, услуг	к сведению		№ П-45	27.03.14		

Рис. 3. "Заливка" для документов, отсутствующих в папке

соответствующие отметки и ссылки, но не всегда это означает, что работа по документу закончена полностью. Нужна колонка с признаком для отбора документов, по которым еще ведется какая-то деятельность.

При создании в подразделении электронного проекта документа (ИСХОДЯЩЕГО), он подлежит регистрации с соответствующим порядковым номером в исходящих служебных, или исходящих письмах, или, может быть, в какой-то ТЕМЕ. Если документ потом регистрируется в делопроизводстве предприятия, то присвоенный регистрационный номер указывается в отведенной для этого колонке. Бывают случаи, когда в подразделении в "папке на полке" не требуется подкалывать копию такого документа, например по причине, что он потом будет заведен как входящий (проект протокола был набран на компьютере сотрудником подразделения, а подписанный протокол или его копия позже поступили в подразделение как входящий). Либо документ является какой-то справкой, которую имеет смысл хранить только в электронном виде, а в печатном текст может оказаться нечитаемым. Или эта "справка" есть не что иное, как сохраненное электронное письмо… В таких случаях рекомендуется как-то отметить в файле-оглавлении ИСХОДЯЩИХ, что в "папке на полке" этого документа нет. Например, можно закрасить такую строку "заливкой серого цвета". В Exel можно такие строки (с заливкой или без выделения заливкой) выбирать по фильтру – если вдруг понадобится анализировать количество электронных и "бумажных" копий в папке. (см. рисунок 3). Аналогичная ситуация может возникнуть и в других файлах-оглавлениях. Либо ВХОДЯЩИМ регистрируется e-mail или проект, подлежащий доработке, или, например, поступил электронный документ "Методические указания", представляющий собой текст двух инструкций на более чем 100 листах каждая, да еще плюс бланки документов, которые бесполезно хранить на полке – используя их как шаблоны, из них нужно создавать электронные документы.

Отметим, что может произойти следующая ситуация: в подразделение поступила служебная записка, приложениями к которой являются копия письма из сторонней организации и оригинал акта приема-передачи по какой-то теме. В этом случае следует зарегистрировать все три документа одним регистрационным входящим номером, но проставить разные признаки "где лежит" и остальные атрибуты этих документов. И, соответственно, разместить документы в разных папках (по принадлежности: входящие служебные, входящие письма (переписка), ТЕМА, …). В примере, проиллюстрированном на рисунке 3, под одним номером зарегистрировано три документа, которые "относятся" к разным "папкам", но поступили одновременно и связанными между собой. А именно: в Приказе было указание на рассылку "Положения" по электронной почте.

Важная рекомендация: в целях облегчения «самоконтроля» при вводе информации в колонках (столбцах) ДОКУМЕНТООБОРОТА, по которым в дальнейшем предполагается вести отбор строк по фильтру, следует вводить символы (лучше английского алфавита) НЕ СОВПАДАЮЩИЕ по написанию с символами русского алфавита (или наоборот). При таком порядке набора исключаются вероятные ошибки последующего отбора (фильтрации строк) по соответствующим признакам реквизита документа. Другими словами: не советуем ставить признак «о» (или «м», или «ок», или «р»…), потому что в одной строке может потом оказаться «русская о», а в другой строке – «английская о», и по фильтру такие строки одновременно не выберутся.

5. Методы поиска и отбора

«Microsoft Excel» позволяет строить и развивать гибкую систему элементов среды в ДОКУМЕНТООБОРОТЕ. Это особенно заметно при решении задач поиска информации или составления справок и сводок.

Унификация при занесении информации в систему ДОКУМЕНТООБОРОТ позволяет вести отбор по «Предмету документа» (столбец может называться «О чем»): Например, легко выбрать все поступившие в подразделение служебные «документы-справки, касающиеся микросхем 3456НР», установив в столбце «О чем» отбор по фильтру «содержит»-«справка»&«содержит»–«3456НР», а в колонке «признак папки» задать фильтрацию только служебных документов («содержит»-«s»). Среди отобранных справок можно отфильтровать только те, которые датированы маем, либо, которые поступили в подразделение в апреле, установив фильтр по»дате документа» или по «дате получения документа» (дате регистрации в подразделении). Далее можно оставить на экране лишь те справки, которые зарегистрированы в делопроизводстве предприятия (то есть прошедшие официальную регистрацию в учреждении) – установив фильтр на «рег. № по делопроизводству»–«не пусто». И т.д.

Устанавливая нужные отборы по фильтрам в разных столбцах можно с легкостью получить сводку по необходимым параметрам или реквизитам, результат которой состоит всего из нескольких записей (документов), выведенных на экран, а при необходимости, и на печать. Среди отобранных документов легко найти нужный простым просмотром и, одновременно, получить всю информацию о том, где он находится (копии, оригинал, электронный вид), кто ответственный, в какой стадии отработки находится документ, … и какой номер имеет при регистрации в учреждении (в делопроизводстве).

Объединив все «ВХОДЯЩИЕ» (или исх.служебные, или…) всех годов в отдельный лист в файл «для сводок» (импортируя соответствующие файлы-оглавления и установив «обновлять» при открытии этого «списка для сводок» – см. рисунок 4), легко найти, следуя такому же принципу установки

фильтров, множество документов, удовлетворяющих нужным критериям поиска.

Обнаружив нужный документ, смотрим, в каком году (см. рисунок 4, строку «Имя») документ зарегистрирован и под каким номером. Открываем соответствующую папку и извлекаем искомый

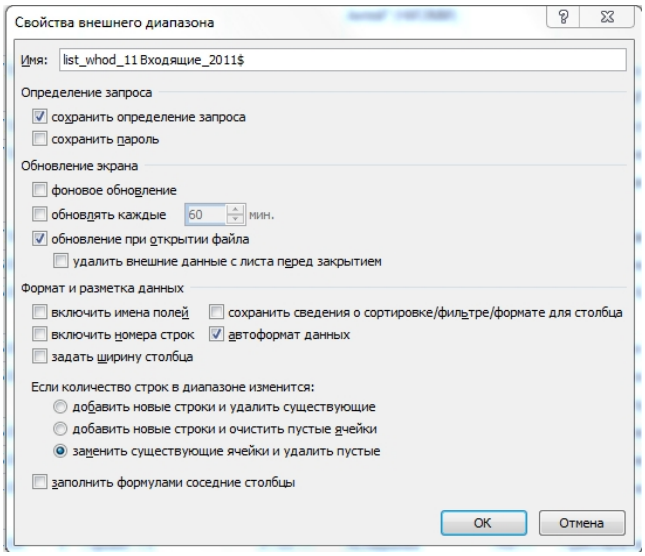


Рис. 4. Свойства внешнего диапазона данных

документ.

Эффективно организованная система поиска документа позволяет находить документ, обладая минимальной информацией о нём. Проста система составления сводок по различным статусам и атрибутам документов.

Заключение

Информационная среда организации, как правило, имеет в неоднородную программную среду. На многих предприятиях существует большое число разрозненных информационных систем, постепенно приобретаемых или разрабатываемых в процессе работы предприятия. Это вызвано тем, что не существует единой системы, которая бы покрыла все функциональные потребности. Зачастую эти системы не совместимы между собой по описанию и способам хранения данных или набору предоставляемых сервисов. Не существует общего способа решения задачи интеграции информационных сред. Как и не существует универсальной программы, удовлетворяющей всем требованиям и специфическим запросам отдельно взятого подразделения. В зависимости от ситуации, наиболее удачными решениями оказываются различные сочетания принципов взаимодействия и передачи данных между системами. Но главной проблемой таких систем является недостаточная развитость имеющихся средств интеграции.

Использование системы ДОКУМЕНТООБОРОТ, основанной на «Microsoft Excel», с легкостью позволяет объединять данные, полученные из разрозненных систем, в единое информационное пространство. Кроме того, система ДОКУМЕНТООБОРОТ подвижна и способна быстро настраиваться на изменения, которых требуют часто поступающие указания «свыше».

В отделе и на местах постоянно изыскиваются способы для улучшения контроля рабочих процессов. С другой стороны, повышаются требования к ведению учетной информации со стороны «системы качества». Организация системы ДОКУМЕНТООБОРОТ позволяет сразу вводить в действие предложения по её усовершенствованию. А гибкая структура системы позволяет постоянно её модифицировать силами пользователей в соответствии с возникающими потребностями. Созданная структура, удовлетворяя требованиям делопроизводства учреждения, в то же время способна на достаточно высоком уровне поддерживать документооборот подразделения. При этом она организационно проста, наглядна, легка в применении; на поиск и выдачу справочных, статистических и других данных уходит минимальное количество времени, всем заинтересованным лицам обеспечен доступ к информации об имеющихся материалах; проводится процедура идентификации материалов и устанавливаются связи с делопроизводством предприятия; устанавливаются связи между документами во времени и по смыслу; отслеживаются выполнения запросов заинтересованных сторон; ведётся регистрация результатов предпринятых действий; минимальны затраты на поддержание системы в рабочем состоянии.

A method of integration of document processing and of electronic document interchange for an institution and its subdivision

G.L.Levchenkova

Abstract. In this article, the questions associated with the convenience of using the electronic information environment of an institution's subdivision (electronic document interchange [1]) are considered. The requirements for the institution's record keeping are taken into account.

Keywords: workflow, document management system, records management, record keeping.

Литература

1. Г.Л.Левченкова, И.В. Холмов Система ДОКУМЕНТООБОРОТ. Организация документооборота и учета на рабочих местах и в подразделениях организаций // *под редакцией академика РАН В.Б.Бетелина*, Москва, НИИСИ РАН, 2006, С. 4-75

2. ГОСТ Р 7.0.8-2013 «Система стандартов по информации, библиотечному и издательскому делу. Делопроизводство и архивное дело. Термины и определения» // Москва. Стандартинформ, 2014 http://www.internet-law.ru/gosts/gost/56742/ (10.09.14)

3. ГОСТ Р 53898-2013 «Системы электронного документооборота. Взаимодействие систем управления документами. Технические требования к электронному сообщению». // Москва. Стандартинформ, 2014 http://www.internet-law.ru/gosts/gost/56114/ (10.09.14)

4. А. М. Гудов, С.Ю. Завозкин Об одной модели оптимизации документопотоков, реализуемой при создании системы электронного документооборота // Вычислительные технологии, 2006, Т. 11, С. 53 - 65

5. В.Л.Арлазаров, Н.Е. Емельянов Системы обработки документов. Основные компоненты. // Сборник трудов ИСА РАН «Управление информационными потоками», 2002. С. 3-20

6. А.Н..Гавердовский Концепция построения систем автоматизации документооборота. Открытые системы, 1997, №1 http://www.osp.ru/os/1997/01/29.htm (10.09.14)

Моделирование системы связанных тел методом последовательных импульсов

М.В. Михайлюк[1], Е.В. Страшнов

1 *- доктор физико-математических наук*

Аннотация: Большинство методов моделирования системы многих тел с шарнирами не обладают всеми необходимыми свойствами, которые важны для задач анимации и виртуальной реальности. Для решения этой проблемы в статье предлагается эффективный метод последовательных импульсов с использованием временной когерентности. Стабилизация ограничений обеспечивается с помощью метода раздельных импульсов. Эти методы позволяют моделировать систему многих тел с шарнирами с приемлемой точностью в режиме реального времени.

Ключевые слова: система многих тел, голономные связи, метод раздельных импульсов, временная когерентность

Введение

В системах виртуальной реальности важной задачей является моделирование реалистичного поведения динамических объектов в сцене на основе законов физики. Одной из таких задач является моделирование динамики системы связанных (соединённых шарнирами) тел. Примерами таких систем являются манипуляторы, части механизмов с шарнирами, входящих в состав других механизмов, маятники, двери и т.п. Задачи и технологии виртуальной реальности накладывают ограничения на время проведения расчётов алгоритмов физики, поэтому требуются алгоритмы, которые позволят проводить все вычисления в реальном времени. Помимо этого, алгоритмы моделирования шарнирно-связанных тел должны удовлетворять требованию универсальности, означающему их способность моделировать различные структуры механических систем, в том числе и с заранее неизвестной структурой и со структурой, которая возникает динамически в процессе моделирования. Эти алгоритмы должны также обладать свойством робастности, обеспечивающим устойчивое поведение механической системы.

Методы моделирования динамики многих тел с шарнирами можно условно разделить на методы, основанные на использовании обобщённых координат и методы, основанные на использовании полного (может быть избыточного) набора координат для каждого тела.

Среди первой группы методов выделяется алгоритм Фезерстоуна [1], обеспечивающий линейное время вычислений. В работе [2] данный алгоритм расширен для моделирования контактного взаимодействия между телами. К недостаткам данной группы методов

я в использовании импульсов вместо сил и в том, что вместо полной системы уравнений рассматриваются небольшие системы уравнений для импульсов на основе локальной информации. Преимуществом методов, основанных на использовании полного набора координат, является то, что они обладают свойством универсальности, а недостатком этих методов является необходимость стабилизации ограничений, чего не относится то, что для них требуется вывод реккурентных соотношений, для которых важен порядок, в котором шарниры соединяют тела. При моделировании шарнирно-связанных тел иногда приходится обрабатывать структуру механической системы, которая содержит кинематические цепи. Анализ подобных структур часто затруднителен, а иногда невозможен, и, кроме того, иногда накладываются специальные требования, при которых необходимо выделить замыкающий шарнир, что требует расширения таких алгоритмов на случай наличия замкнутых цепей. Также стоит отметить, что такие алгоритмы не позволяет моделировать ограничения, сформированные только в терминах скоростей, например, шестерёночные связи.

Характерной особенностей второй группы методов является использование ограничений, описывающих механические связи. В работе [3] используется метод штрафов, идея которого состоит в применении штрафных сил, пропорциональных ошибкам ограничений. Данный алгоритм зарекомендовал себя, как не требующий значительных вычислительных затрат и как относительно простой в реализации, но при этом требующий аккуратной регулировки параметров и не обладающий свойством робастности. В [4] описывается метод с использованием множителей Лагранжа, при реализации которого требуется решение системы линейных уравнений большой размерности, которая в общем случае является полностью заполненной. Только при определённой последовательности обработки, как описано в [5], можно достичь линейного времени вычислений, используя прямые методы решения системы линейных уравнений. В работах [6], [7] и [8] предлагаются методы, основанные на импульсах. Их отличие от методов [4] заключаетс

требуется в методах, основанных на использовании обобщённых координат и в методе штрафов.

В данной статье за основу берётся метод последовательных импульсов [9], [10], который состоит в последовательном составлении уравнений для импульсов и их последующем применении. Метод последовательных импульсов обладает свойством универсальности. В качестве стабилизации ограничений использу-

ется метод раздельных импульсов, который позволяет обеспечить свойство робастности. В работе будет показано, что метод последовательных импульсов сходится, а для улучшения сходимости будет использовано свойство временной когерентности, которое означает, что состояние механической системы от кадра к кадру меняется незначительно.

1. Представление задачи в виде системы линейных уравнений

Динамика n свободных твёрдых тел описывается с помощью уравнений Ньютона-Эйлера

$$m_i\dot{\bar{v}}_i = \bar{f}_i^e,\ I_i\dot{\bar{\omega}}_i = \bar{\tau}_i^e,\ i = 1,\ldots,n,$$

где для i-го тела m_i - масса тела, I_i - тензор инерции тела. Обозначим через $\bar{V}_i(t) = \left(\bar{v}_i(t), \bar{\omega}_i(t)\right)$ вектор обобщенных (линейных и угловых) скоростей, а через $\bar{F}_i^e = \left(\bar{f}_i^e, \bar{\tau}_i^e\right)$ - вектор внешних сил и моментов. Для определения положения и ориентации i-го твёрдого тела используются кинематические соотношения вида $\dot{\bar{X}}_i = H_i(\bar{X}_i, \bar{V}_i)$, где H_i - линейная функция относительно $\bar{V}_i$, $\bar{X}_i(t)$ - вектор координат i-го тела (для задания положения твёрдого тела можно использовать координаты центра масс, а для ориентации - кватернион).

Шарниры накладывают ограничения на свободное движение твёрдых тел, которые они соединяют. Данные ограничения описываются с помощью *голономных связей* – механических связей, накладывающих ограничения на координаты твёрдых тел. В общем случае, для j-го шарнира, соединяющего тела i_1 и i_2 связи (ограничения) можно записать в виде

$$\bar{G}_j(\bar{X}_{i1}, \bar{X}_{i2}) = \bar{0}, \qquad (1)$$

где $\dim(\bar{G}_j) = r_j$ - число степеней свободы, которые ограничивает j - ый шарнир. Если одно из тел неподвижно (например, к стене на петлях подвешена дверь), то в уравнении (1) будут координаты только подвижного тела (двери).

Предполагается, что компоненты (1) являются дважды непрерывно дифференцируемыми функциями. Дифференцируя (1) по времени, получим ограничения в терминах скоростей:

$$\dot{\bar{G}}_j = J_{j,i1}\bar{V}_{i1} + J_{j,i2}\bar{V}_{i2} = \bar{0}, \qquad (2)$$

где $J_{j,i1}$, $J_{j,i2}$ - матрицы-якобианы размерности $r_j \times 6$.

Ограничения (1) и (2) для всех шарниров можно записать в следующем виде:

$$\bar{G} = \begin{bmatrix} \bar{G}_1 \\ \cdots \\ \bar{G}_m \end{bmatrix} = \bar{0},\ J\bar{V} = \begin{bmatrix} J_{11} & \cdots & J_{1n} \\ \vdots & \ddots & \vdots \\ J_{m1} & \cdots & J_{mn} \end{bmatrix} \begin{bmatrix} \bar{V}_1 \\ \cdots \\ \bar{V}_n \end{bmatrix} = \bar{0}, \qquad (3)$$

где m - количество шарниров в механической системе, $\bar{V} = (\bar{V}_1, \ldots, \bar{V}_n)$ - вектор обобщённых скоростей, n - количество тел в механической системе.

Блок J_{ji} матрицы J соответствует якобиану ограничения (2), если j - ый шарнир соединяет i - ое тело, иначе $J_{ji} = \Theta$ (нулевая матрица).

Согласно принципу освобождаемости от связей, рассматривая несвободные тела, как свободные, необходимо в уравнения Ньютона-Эйлера добавить силы реакции связей $\bar{F}^c$, которые обеспечивают выполнение ограничений, то есть

$$M\dot{\bar{V}} = \bar{F}^e + \bar{F}^c, \qquad (4)$$

где $M = diag(M_1, \ldots, M_n)$ - блочно-диагональная матрица масс для всей механической системы,

$M_i = \begin{bmatrix} m_i E_3 & \Theta \\ \Theta & I_i \end{bmatrix}$ - матрица масс i - го тела.

Известно [10], что работа сил реакции связей на любом действительном перемещении равна нулю, и сами силы могут быть представлены в виде

$$\bar{F}^c = J^T\bar{\lambda}' = \begin{bmatrix} J_1^T & \cdots & J_m^T \end{bmatrix} \begin{bmatrix} \bar{\lambda}_1' \\ \vdots \\ \bar{\lambda}_m' \end{bmatrix}, \qquad (5)$$

где J - якобиан ограничений всей механической системы, $J_j = \begin{bmatrix} J_{j1} & \cdots & J_{jn} \end{bmatrix}$ - якобиан ограничений для j - го шарнира, $\bar{\lambda}' = \left(\bar{\lambda}_1', \ldots, \bar{\lambda}_m'\right)$ - неизвестные множители Лагранжа, $\bar{\lambda}_j'$ - неизвестные множители Лагранжа для j - го шарнира.

Рассматривая уравнение движения (4) в дискретные моменты времени, аппроксимируем вектор обобщённых скоростей следующим образом:

$\dot{\bar{V}} \approx \dfrac{\bar{V}(t+\Delta t) - \bar{V}(t)}{\Delta t}$, где Δt - шаг моделирования. Добавляя (5) в уравнения (4) с заменами $\bar{\lambda} = \bar{\lambda}'\Delta t$ и, учитывая ограничения (3) для скоростей в момент времени $t + \Delta t$, получим систему линейных уравнений:

$$\begin{aligned} &\bar{V}(t+\Delta t) = \bar{V}(t) + M^{-1}\bar{F}^e\Delta t + M^{-1}J^T\bar{\lambda}; \\ &J\bar{V}(t+\Delta t) = \bar{0}. \end{aligned} \qquad (6)$$

В данной формулировке задача (6) состоит в определении обобщённых скоростей $\bar{V}(t+\Delta t)$ и множителей $\bar{\lambda}$. Множители $\bar{\lambda}$ будем называть импульсами и моментами импульсов реакций связей.

2. Описание метода последовательных импульсов

Суть метода последовательных импульсов состоит в том, что мы не решаем (6) непосредственно, а последовательно проходим все шарниры, формируя и при-

меняя импульсы только на основе ограничений (2), не

Метод условно можно разбить на два этапа. На первом этапе для всех тел вычисляются предварительные скорости $\overline{V}'(t+\Delta t)$, обусловленные применением всех внешних сил по формулам

$$\overline{V}'(t+\Delta t)=\overline{V}(t)+M^{-1}\overline{F}^{e}\Delta t. \qquad (7)$$

На втором этапе выполняется итерационный цикл, на каждом шаге итерации которого для каждого шарнира вычисляются импульсы (призванные приблизиться к выполнению ограничений (2)), новые обобщенные скорости $\overline{V}'(t+\Delta t)$ тел и общие накопленные импульсы. Цикл прекращает работу, когда для всех ограничений шарниров будет выполнен критерий окончания.

Второй этап основан на уравнениях для импульсов по ограничениям (2). Данные уравнения учитывают только влияние импульсов $\Delta\overline{\lambda}_j$, которые должны обеспечивать выполнение равенств вида (2) в момент времени $t+\Delta t$. Если рассмотреть ограничения только в одном шарнире (например, в j - м шарнире, соединяющем тела с номерами i_1 и i_2), то аналогично уравнениям (6), получим:

$$\overline{V}(t+\Delta t)=\overline{V}'(t+\Delta t)+M^{-1}J_j^T\Delta\overline{\lambda}_j;$$
$$J_j\overline{V}(t+\Delta t)=\overline{0}.$$

Так как якобиан J_j содержит только две ненулевые матрицы, $J_{j,i1}$ и $J_{j,i2}$, то данные уравнения будут содержать только скорости для тел с номерами i_1 и i_2:

$$J_{j,i1}\overline{V}_{i1}(t+\Delta t)+J_{j,i2}\overline{V}_{i2}(t+\Delta t)=\overline{0};$$
$$\overline{V}_{i1}(t+\Delta t)=\overline{V}'_{i1}(t+\Delta t)+M_{i1}^{-1}J_{j,i1}^T\Delta\overline{\lambda}_j; \qquad (8)$$
$$\overline{V}_{i2}(t+\Delta t)=\overline{V}'_{i2}(t+\Delta t)+M_{i2}^{-1}J_{j,i2}^T\Delta\overline{\lambda}_j.$$

Подставляя последние два уравнения в первое, получим систему из r_j уравнений с r_j неизвестными $\Delta\overline{\lambda}_j$:

$$(J_{j,i1}M_{i1}^{-1}J_{j,i1}^T+J_{j,i2}M_{i2}^{-1}J_{j,i2}^T)\Delta\overline{\lambda}_j=$$
$$-J_{j,i1}\overline{V}'_{i1}(t+\Delta t)-J_{j,i2}\overline{V}'_{i2}(t+\Delta t). \qquad (9)$$

Решая эту систему, получаем импульсы $\Delta\overline{\lambda}_j$, которые используем для вычисления новых скоростей по формулам:

$$\overline{V}_{i1}(t+\Delta t)=\overline{V}'_{i1}(t+\Delta t)+M_{i1}^{-1}J_{j,i1}^T\Delta\overline{\lambda}_j;$$
$$\overline{V}_{i2}(t+\Delta t)=\overline{V}'_{i2}(t+\Delta t)+M_{i2}^{-1}J_{j,i2}^T\Delta\overline{\lambda}_j; \qquad (10)$$
$$\overline{V}'_{i1}(t+\Delta t)=\overline{V}_{i1}(t+\Delta t);$$
$$\overline{V}'_{i2}(t+\Delta t)=\overline{V}_{i2}(t+\Delta t).$$

Отметим, что вычисленные импульсы $\Delta\overline{\lambda}_j$ составляют только часть импульсов $\overline{\lambda}_j$, а именно, учитывая взаимосвязи между шарнирами.

$\overline{\lambda}_j=\sum_p\Delta\overline{\lambda}_j$, где p - номер итерации. Таким образом, импульсы $\overline{\lambda}_j$ накапливают значения импульсов $\Delta\overline{\lambda}_j$ от каждой итерации метода, поэтому будем их в дальнейшем называть *суммарными*.

Итерации метода последовательных импульсов можно заканчивать, если компоненты всех вычисляемых импульсов малы, т.е. не превосходят некоторого заданного ε:

$$\Delta\lambda_{jk}<\varepsilon,\ k=1,\ldots,r_j. \qquad (11)$$

Важнейшим свойством моделирования динамики систем многих тел с шарнирами является то, что скорости тел невелики и, как следствие, состояние механической системы между шагами моделирования меняется незначительно. Это свойство называется временной когерентностью и позволяет использовать суммарные импульсы $\overline{\lambda}_j$ с предыдущего шага моделирования. Поэтому между первым и вторым этапами можно добавить еще один этап, на котором вычисляются новые скорости, обусловленные импульсами $\overline{\lambda}_j$, накопленными на предыдущем шаге (так называемый горячий старт).

На практике проще вместо составления системы (9) составлять уравнения для компонентов импульсов $\Delta\lambda_{jk}$:

$$(J_{jk,i1}M_{i1}^{-1}J_{jk,i1}^T+J_{jk,i2}M_{i2}^{-1}J_{jk,i2}^T)\Delta\lambda_{jk}=$$
$$-J_{jk,i1}\overline{V}'_{i1}(t+\Delta t)-J_{jk,i2}\overline{V}'_{i2}(t+\Delta t), \qquad (12)$$

где $k=1,\ldots,r_j$.

В этом случае процесс вычисления $\Delta\lambda_{jk}$ будет сходиться медленнее, однако сами вычисления будут выполняться быстрее, т.к. вместо решения системы уравнений (9) мы сразу получаем значение $\Delta\lambda_{jk}$.

3. Сходимость метода последовательных импульсов

Покажем, что метод последовательных импульсов для данной задачи совпадает с методом Гаусса-Зейделя для решения СЛАУ относительно импульсов $\overline{\lambda}$ при одновременном учёте всех ограничений для шарниров вида (2). Для этого первое уравнение из (6) подставим во второе уравнение. В итоге получим систему относительно импульсов $\overline{\lambda}$:

$$JM^{-1}J^T\overline{\lambda}=-J\overline{V}(t)-JM^{-1}\overline{F}^{e}\Delta t.$$

Полученную систему можно представить в следующем виде

$$\sum_{l=1}^{m}A_{jl}\overline{\lambda}_l=\overline{b}_j,\ j=1,\ldots,m,$$

где $\overline{\lambda}_j$ - импульсы j - го шарнира,

$A_{jl} = \sum_{k=1}^{n} J_{jk} M_k^{-1} J_{lk}^T$, $\bar{b}_j = -J_j \bar{V}(t) - J_j M^{-1} \bar{F}^e \Delta t$.

Перепишем данную систему в следующем виде

$$A_{jj}\bar{\lambda}_j = -\sum_{l=1}^{j-1} A_{jl}\bar{\lambda}_l - \sum_{l=j+1}^{m} A_{jl}\bar{\lambda}_l + \bar{b}_j,\ j = 1,\ldots,m. \quad (13)$$

Систему уравнений будем решать методом Гаусса-Зейделя, где $\bar{\lambda}_j^{(p)}$ - решение для j-го шарнира на p-ой итерации.

Перепишем систему (13) в форме метода Гаусса-Зейделя:

$$A_{jj}\bar{\lambda}_j^{(p+1)} = A_{jj}\bar{\lambda}_j^{(p)} - \sum_{l=1}^{j-1} A_{jl}\bar{\lambda}_l^{(p+1)} - \sum_{l=j}^{m} A_{jl}\bar{\lambda}_l^{(p)} + \bar{b}_j.$$

Покажем, что полученная система уравнений для определения $\bar{\lambda}_j^{(p+1)}$ совпадает с системой уравнений (9). Для этого выразим A_{ij} и $\bar{b}_j$ обратно через матрицы масс, якобианы и скорости, получим систему относительно импульсов $\bar{\lambda}_j^{(p+1)}$:

$$\sum_{k=1}^{n} J_{jk} M_k^{-1} J_{jk}^T \bar{\lambda}_j^{(p+1)} = \sum_{k=1}^{n} J_{jk} M_k^{-1} J_{jk}^T \bar{\lambda}_j^{(p)} - \sum_{l=1}^{j-1}\sum_{k=1}^{n} J_{jk} M_k^{-1} J_{lk}^T \bar{\lambda}_l^{(p+1)} - \sum_{l=j}^{m}\sum_{k=1}^{n} J_{jk} M_k^{-1} J_{lk}^T \bar{\lambda}_l^{(p)} - J_j(\bar{V}(t) + M^{-1}\bar{F}^e \Delta t).$$

Перепишем полученную систему, группируя элементы под общую сумму по всем телам механической системы:

$$\sum_{k=1}^{n} J_{jk} M_k^{-1} J_{jk}^T (\bar{\lambda}_j^{(p+1)} - \bar{\lambda}_j^{(p)}) = -\sum_{k=1}^{n} J_{jk} (M_k^{-1} \sum_{l=1}^{j-1} J_{lk}^T \bar{\lambda}_l^{(p+1)} + M_k^{-1} \sum_{l=j}^{m} J_{lk}^T \bar{\lambda}_l^{(p)} + M_k^{-1}\bar{F}_k^e \Delta t + \bar{V}_k(t)).$$

Слагаемые $M_k^{-1}\bar{F}_k^e \Delta t + \bar{V}_k(t)$ представляют собой изменение скорости k-го тела под действием внешних сил, которое соответствует первому шагу метода последовательных импульсов (см. (7)). Слагаемые в правой части $M_k^{-1}\sum_{l=1}^{j-1} J_{lk}^T \bar{\lambda}_l^{(p+1)}$ и $M_k^{-1}\sum_{l=j}^{m} J_{lk}^T \bar{\lambda}_l^{(p)}$ представляют собой изменение скорости k-го тела под действием импульсов $\bar{\lambda}_l^{(p+1)}$ и $\bar{\lambda}_l^{(p)}$, которое соответствует второму шагу метода последовательных импульсов (см. (10)), так как $\bar{\lambda}_l^{(p+1)} = \sum_{s=0}^{p+1} \Delta\bar{\lambda}_l^{(s)}$, $\bar{\lambda}_l^{(p)} = \sum_{s=0}^{p} \Delta\bar{\lambda}_l^{(s)}$ и в силу аддитивности изменения скорости тела под действием импульса.

Поэтому для j-го шарнира правая часть полученной системы представляет собой вид ограничения (2) после применения внешних сил и вычисленных импульсов, которые в методе последовательных импульсов формируют ограничение (2) в терминах текущих скоростей $\bar{V}'(t+\Delta t)$:

$$\sum_{k=1}^{n} J_{jk} M_k^{-1} J_{jk}^T (\bar{\lambda}_j^{(p+1)} - \bar{\lambda}_j^{(p)}) = -J_j \bar{V}'(t+\Delta t).$$

Учитывая, что только два якобиана в общем якобиане J_j не равны нулю и $\Delta\bar{\lambda}_j = \bar{\lambda}_j^{(p+1)} - \bar{\lambda}_j^{(p)}$, полученная система уравнений для определения $\bar{\lambda}_j^{(p+1)}$ полностью совпадает с системой (9). Поэтому накопленный импульс $\bar{\lambda}_j^{(p+1)}$ совпадает с решением СЛАУ методом Гаусса-Зейделя на $p+1$-ой итерации.

Аналогичные рассуждения можно провести и для уравнений (12).

Таким образом, метод последовательных импульсов совпадает с методом Гаусса-Зейделя. Следовательно, достаточные условия сходимости метода последовательных импульсов совпадают с достаточными условиями метода Гаусса-Зейделя.

Достаточным условием сходимости метода Гаусса-Зейделя является условие положительной определённости матрицы системы. Так как тензоры инерции является положительно-определёнными матрицами, то все блоки матрицы A_{ij} является положительно-определёнными. Следовательно, метод последовательных импульсов сходится из-за сходимости метода Гаусса-Зейделя.

4. Стабилизация ограничений на основе метода раздельных импульсов

Вследствие того, что импульсы в (9) (или (12)), вычисляются так, чтобы обеспечить выполнимость ограничений только по скоростям и в связи с ошибками интегрирования уравнения для координат тел, в процессе моделирования связи (1) будут нарушаться. Чтобы обеспечить выполнение связей, требуется проводить так называемую стабилизацию.

Самым распространённым методом стабилизации является метод Баумгарте [11], который состоит в том, что ошибки связей добавляются в ограничения по скоростям, и тогда ограничения (2) принимают вид

$$\dot{\bar{G}}_j = J_{j,i1}\bar{V}_{i1} + J_{j,i2}\bar{V}_{i2} = -ERP\frac{\bar{G}_j}{\Delta t}, \quad (14)$$

где ERP - параметр уменьшения ошибки, который принимает значения от 0 до 1.

Точным решением этого уравнения является

$$\bar{G}_j(\tau) = \bar{G}_j(t)\exp\left(-ERP\frac{\tau - t}{\Delta t}\right), \tau \geq t.$$

Отсюда видно, что ошибки связей для таких ограничений будут экспоненциально убывать от времени τ, причем скорость убывания будет зависеть от параметра ERP.

Поведение ограничения $\overline{G}_j(t)$ можно проанализировать, аппроксимируя производную следующим образом $\dot{\overline{G}}_j \approx \frac{\overline{G}_j(t+\Delta t)-\overline{G}_j(t)}{\Delta t}$. Тогда решение $\overline{G}_j(t)$ на очередном шаге $t+\Delta t$ примет вид

$$\overline{G}_j(t+\Delta t) = (1-ERP)\overline{G}_j(t).$$

При параметре уменьшения ошибки ERP равном 1 ошибка $\overline{G}_j(t)$ будет полностью компенсирована, при меньшем параметре только часть ошибки будет компенсирована.

Недостатком данного метода является то, что силы реакции связей, которые обеспечивают выполнимость уравнения (14), теперь совершают работу, т.е. нарушают идеальность связей. Фактически это означает, что помимо сил реакций связей добавляются фиктивные силы, которые отвечают за стабилизацию ограничений, и эти силы изменяют энергию механической системы, что может привести к нарушению стабильности моделирования. В ряде случаев для адекватного моделирования механической системы требуется более аккуратный выбор параметра ERP, тем самым заранее неизвестно, какой параметр выбрать для конкретной сцены. Часто это приводит к нереалистичному поведению динамики системы многих тел.

Альтернативой методу Баумгарте является метод раздельных импульсов. В методе раздельных импульсов рассматриваются два вида импульсов (и, соответственно, два вида скоростей). Один вид импульсов (скоростей) обеспечивает выполнение ограничений вида (2) (как описано в главе 2), а второй вид импульсов (так называемые *псевдоимпульсы*, а также *псевдоскорости*) участвует только в стабилизации ограничений. Таким образом, ограничения вида (14) для стабилизации записываются только для псевдоскоростей $\overline{V}^*(t+\Delta t)$:

$$J_{j,i1}\overline{V}^*_{i1} + J_{j,i2}\overline{V}^*_{i2} = -ERP\frac{\overline{G}_j}{\Delta t}. \qquad (15)$$

В итерационном цикле метода последовательных импульсов после вычисления импульсов $\Delta\overline{\lambda}$ и скоростей $\overline{V}(t+\Delta t)$ вычисляются псевдоимпульсы $\Delta\lambda^*$ (на основе уравнений (15) и уравнений, аналогичных (9) или (12)), которые применяются для изменения только псевдоскоростей $\overline{V}^*(t+\Delta t)$. Псевдоскорости не должны накапливаться в шагах моделирования, поэтому в начале каждого шага они обнуляются. В качестве критерия окончания всех итераций для $\Delta\lambda^*$ также можно использовать неравенства, аналогичные (11). После выхода из цикла скорости $\overline{V}(t+\Delta t)$ и $\overline{V}^*(t+\Delta t)$ используются для вычисления новых координат и ориентаций тел в механической системе на основе кинематических уравнений.

Данный метод является более робастным, чем метод Баумгарте. Сходимость псевдоимпульсов в методе раздельных импульсов доказывается аналогично сходимости обычных импульсов.

5. Формирование импульсов для удовлетворения механических связей некоторых типов шарниров

Для описания относительного движения тел, соединённых шарниром, введём две системы координат, одна из которых жёстко связана с i_1 - м телом, а другая – с i_2 - м телом. Орты системы координат, связанной с i_1 - м телом будем обозначать через $\left\{\overline{n}_0^{i1}, \overline{n}_1^{i1}, \overline{n}_2^{i1}\right\}$, а с i_2 - м телом - через $\left\{\overline{n}_0^{i2}, \overline{n}_1^{i2}, \overline{n}_2^{i2}\right\}$. Мы предполагаем, что в начальный момент времени эти две системы координат совпадают и находятся в точке расположения шарнира. Эти системы координат (в отличие от локальных систем координат тел) будем называть шарнирными. При последующем моделировании относительного движения тел, их начала координат и направления осей могут разойтись. Обозначим через $\overline{r}_P^{i1}$ и $\overline{r}_P^{i2}$ радиус-векторы этих начал в мировой системе координат, а через $\overline{r}_C^{i1}$ и $\overline{r}_C^{i2}$ - радиус-векторы центров масс этих тел. Центры масс тел в общем случае не совпадают с точкой расположения шарнира.

В статье [3] описан вывод ограничений вида (1) и (2). Здесь мы опишем, как составлять уравнения для импульсов вида (9) или (12) и псевдоимпульсов для сферического, осевого и призматического шарнира.

5.1. Формирование импульсов для сферического шарнира

Сферический шарнир является вращательным шарниром, при котором соединяемые тела могут свободно вращаться относительно друг друга, но не могут осуществлять поступательное движение друг относительно друга.

Для сферического шарнира ограничения (1) имеют вид

$$\overline{G} = \overline{r}_P^{i1} - \overline{r}_P^{i2} = \overline{0}. \qquad (16)$$

Дифференцируя (16), получим ограничения в терминах скоростей, которые для момента времени $t+\Delta t$ имеют вид

$$\overline{v}_{i1}^P(t+\Delta t) - \overline{v}_{i2}^P(t+\Delta t) = \overline{0},$$

где $\overline{v}_{i1}^P$ и $\overline{v}_{i2}^P$ - скорости начал шарнирных систем координат.

Вычисляя скорости начал через линейные и угловые скорости тел, получим:

$$\overline{v}_{i1}(t+\Delta t) + \overline{\omega}_{i1}(t+\Delta t)\times(\overline{r}_P^{i1} - \overline{r}_C^{i1}) -$$

$$\overline{v}_{i2}(t+\Delta t)-\overline{\omega}_{i2}(t+\Delta t)\times(\overline{r}_P^{i2}-\overline{r}_C^{i2})=\overline{0}.$$

Преобразуем данное ограничение к виду (2):

$$\begin{aligned}&\left[E_3 \quad -skew(\overline{r}_{i1})\right]\overline{V}_{i1}(t+\Delta t)+\\&\left[-E_3 \quad skew(\overline{r}_{i2})\right]\overline{V}_{i2}(t+\Delta t)=\overline{0},\end{aligned} \tag{17}$$

где E_3 - единичная матрица размера 3x3, $skew(*)$ - кососимметричная матрица размера 3x3, $\overline{r}_{i1}=\overline{r}_P^{i1}-\overline{r}_C^{i1}$, $\overline{r}_{i2}=\overline{r}_P^{i2}-\overline{r}_C^{i2}$.

Для выполнения ограничения (16) аналогично (8) сформируем импульс $\Delta\overline{\lambda}_j$, который изменяет линейные и угловые скорости тел, соединённых сферическим шарниром, следующим образом:

$$\overline{v}_{i1}(t+\Delta t)=\overline{v}'_{i1}(t+\Delta t)+m_{i1}^{-1}\Delta\overline{\lambda}_j;$$
$$\overline{\omega}_{i1}(t+\Delta t)=\overline{\omega}'_{i1}(t+\Delta t)+I_{i1}^{-1}skew(\overline{r}_{i1})\Delta\overline{\lambda}_j;$$
$$\overline{v}_{i2}(t+\Delta t)=\overline{v}'_{i2}(t+\Delta t)-m_{i2}^{-1}\Delta\overline{\lambda}_j;$$
$$\overline{\omega}_{i2}(t+\Delta t)=\overline{\omega}'_{i2}(t+\Delta t)-I_{i2}^{-1}skew(\overline{r}_{i2})\Delta\overline{\lambda}_j.$$

Подставляя эти выражения в (17), получаем для искомого импульса $\Delta\overline{\lambda}_j$ следующую систему уравнений:

$$\begin{aligned}&((m_{i1}^{-1}+m_{i2}^{-1})E_3-skew(\overline{r}_{i1})I_{i1}^{-1}skew(\overline{r}_{i1})-\\&skew(\overline{r}_{i2})I_{i2}^{-1}skew(\overline{r}_{i2}))\Delta\overline{\lambda}_j=\\&-(\overline{v}'^{P}_{i1}(t+\Delta t)-\overline{v}'^{P}_{i2}(t+\Delta t)).\end{aligned} \tag{18}$$

Эта система из 3 уравнений с 3 неизвестными импульсами $\Delta\overline{\lambda}_j$ легко решается путём обращения матрицы. Вычисленный импульс добавляется в суммарный импульс и будет использован на следующем шаге моделирования.

Аналогично формируется система уравнений для псевдоимпульса $\Delta\overline{\lambda}_j^*$:

$$\begin{aligned}&((m_{i1}^{-1}+m_{i2}^{-1})E_3-skew(\overline{r}_{i1})I_{i1}^{-1}skew(\overline{r}_{i1})-\\&skew(\overline{r}_{i2})I_{i2}^{-1}skew(\overline{r}_{i2}))\Delta\overline{\lambda}_j^*=\\&-(\overline{v}'^{P,*}_{i1}(t+\Delta t)-\overline{v}'^{P,*}_{i2}(t+\Delta t))-ERP\frac{\overline{G}}{\Delta t}.\end{aligned} \tag{19}$$

Эта система из 3 уравнений с 3 неизвестными импульсами $\Delta\overline{\lambda}_j^*$ также легко решается путём обращения матрицы.

Далее, вычисляются скорости $\overline{V}(t+\Delta t)$ и $\overline{V}^*(t+\Delta t)$, которые используются для вычисления новых координат и ориентаций тел в механической системе на основе кинематических уравнений.

5.2. Формирование импульсов для осевого шарнира

Осевой шарнир является вращательным шарниром, который, в отличие от сферического шарнира, позволяет вращаться друг относительно друга соединяемым телам только вокруг одной оси $\overline{n}_0$.

На поступательные степени свободы накладываются ограничения вида (16) и (17), которые обеспечиваются с помощью импульсов, вычисленных через системы уравнений (18) и (19).

Ограничения вида (1), обеспечивающие вращение только вокруг одной оси имеют вид

$$\begin{aligned}&G_1=\overline{n}_1^{i1}\cdot\overline{n}_0^{i2}=0;\\&G_2=\overline{n}_2^{i1}\cdot\overline{n}_0^{i2}=0.\end{aligned} \tag{20}$$

Дифференцируя (20), получим ограничения в терминах скоростей, которые для момента времени $t+\Delta t$ имеют вид

$$\begin{aligned}&(\overline{\omega}_{i1}(t+\Delta t)-\overline{\omega}_{i2}(t+\Delta t))\overline{u}_1=0;\\&(\overline{\omega}_{i1}(t+\Delta t)-\overline{\omega}_{i2}(t+\Delta t))\overline{u}_2=0,\end{aligned} \tag{21}$$

где $\overline{u}_1=\overline{n}_0^{i2}\times\overline{n}_1^{i1}$, $\overline{u}_2=\overline{n}_0^{i2}\times\overline{n}_2^{i1}$. При дифференцировании мы использовали формулу Бура [12]

$$\dot{\overline{a}}=\overline{\omega}\times\overline{a}$$

для абсолютной производной в мировой системе координат и свойство смешанного произведения

$$\overline{a}(\overline{b}\times\overline{c})=\overline{b}(\overline{c}\times\overline{a})=\overline{c}(\overline{a}\times\overline{b}).$$

Чтобы обеспечить (21), уравнения вида (12) для определения импульсов $\Delta\lambda_{j,1}$ и $\Delta\lambda_{j,2}$ примут вид

$$\begin{aligned}&\overline{u}_k^T(I_{i1}^{-1}+I_{i2}^{-1})\overline{u}_k\Delta\lambda_{j,k}=\\&(\overline{\omega}'_{i2}(t+\Delta t)-\overline{\omega}'_{i1}(t+\Delta t))\overline{u}_k,\end{aligned}$$

где $k=1,2$.

Полученные импульсы добавляем в суммарные импульсы для использования их на следующем шаге моделирования.

Для стабилизации ограничений (20) сформируем псевдоимпульсы $\Delta\lambda_{j,1}^*$ и $\Delta\lambda_{j,2}^*$ на основе уравнений следующего вида

$$\begin{aligned}&\overline{u}_k^T(I_{i1}^{-1}+I_{i2}^{-1})\overline{u}_k\Delta\lambda_{j,k}^*=\\&(\overline{\omega}'^{*}_{i2}(t+\Delta t)-\overline{\omega}'^{*}_{i1}(t+\Delta t))\overline{u}_k-ERP\frac{G_k}{\Delta t},\end{aligned}$$

где $k=1,2$.

Далее, вычисляются скорости $\overline{V}(t+\Delta t)$ и $\overline{V}^*(t+\Delta t)$, которые используются для вычисления новых координат и ориентаций тел в механической системе на основе кинематических уравнений.

5.3. Формирование импульсов для призматического шарнира

Призматический шарнир является поступательным шарниром, который разрешает двум телам двигаться линейно друг относительно друга только в одном на-

правлении $\overline{n}_0$. Таким образом, вращательное движение в шарнире отсутствует.

Сформируем ограничения вида (1) на поступательные степени свободы

$$G_1 = (\overline{r}_P^{i1} - \overline{r}_P^{i2})\overline{n}_1^{i1} = 0;$$
$$G_2 = (\overline{r}_P^{i1} - \overline{r}_P^{i2})\overline{n}_2^{i1} = 0. \quad (22)$$

Дифференцируя (22) по времени сформируем ограничения вида (2) в терминах скоростей, которые для момента времени $t + \Delta t$ имеют вид

$$(\overline{v}_{i1}(t+\Delta t) - \overline{v}_{i2}(t+\Delta t))\overline{n}_k^{i1} +$$
$$\overline{\omega}_{i1}(t+\Delta t)\overline{\xi}_k^{i1} - \overline{\omega}_{i2}(t+\Delta t)\overline{\xi}_k^{i2} = 0,$$

где $\overline{\xi}_k^{i1} = (\overline{r}_P^{i2} - \overline{r}_C^{i1}) \times \overline{n}_k^{i1}$, $\overline{\xi}_k^{i2} = (\overline{r}_P^{i2} - \overline{r}_C^{i2}) \times \overline{n}_k^{i1}$, $k = 1, 2$.

Для того чтобы обеспечить ограничения (23) сформируем импульсы $\Delta\lambda_{j,1}$ и $\Delta\lambda_{j,2}$ на основе уравнений (12):

$$(m_{i1}^{-1} + m_{i2}^{-1} + \overline{\xi}_k^{i1,T} I_{i1}^{-1} \overline{\xi}_k^{i1} + \overline{\xi}_k^{i2,T} I_{i2}^{-1} \overline{\xi}_k^{i2})\Delta\lambda_{j,k} =$$
$$(\overline{v}'_{i2}(t+\Delta t) - \overline{v}'_{i1}(t+\Delta t))\overline{n}_k^{i1} -$$
$$\overline{\omega}'_{i1}(t+\Delta t)\overline{\xi}_k^{i1} + \overline{\omega}'_{i2}(t+\Delta t)\overline{\xi}_k^{i2},$$

где $k = 1, 2$.

Для стабилизации ограничений (22) сформируем псевдоимпульсы $\Delta\lambda_{j,1}^*$ и $\Delta\lambda_{j,2}^*$ на основе уравнений следующего вида

$$(m_{i1}^{-1} + m_{i2}^{-1} + \overline{\xi}_k^{i1,T} I_{i1}^{-1} \overline{\xi}_k^{i1} + \overline{\xi}_k^{i2,T} I_{i2}^{-1} \overline{\xi}_k^{i2})\Delta\lambda_{j,k}^* =$$
$$(\overline{v}_{i2}^*(t+\Delta t) - \overline{v}_{i1}^*(t+\Delta t))\overline{n}_k^{i1} -$$
$$\overline{\omega}_{i1}'^*(t+\Delta t)\overline{\xi}_k^{i1} + \overline{\omega}_{i2}'^*(t+\Delta t)\overline{\xi}_k^{i2},$$

где $k = 1, 2$.

Также требуется сформулировать ограничения на вращательные степени свободы так, чтобы не было относительного вращения между соединяемыми телами. Подобные ограничения на основе (1) примут вид

$$G_3 = \overline{n}_1^{i1} \cdot \overline{n}_0^{i2} = 0;$$
$$G_4 = \overline{n}_2^{i1} \cdot \overline{n}_0^{i2} = 0; \quad (23)$$
$$G_5 = \overline{n}_1^{i1} \cdot \overline{n}_2^{i2} = 0.$$

Дифференцируя (23) по времени получим ограничения в терминах скоростей вида (2), которые для момента времени $t + \Delta t$ имеют вид

$$(\overline{\omega}_{i2}(t+\Delta t) - \overline{\omega}_{i1}(t+\Delta t))\overline{u}_3 = 0;$$
$$(\overline{\omega}_{i2}(t+\Delta t) - \overline{\omega}_{i1}(t+\Delta t))\overline{u}_4 = 0; \quad (24)$$
$$(\overline{\omega}_{i2}(t+\Delta t) - \overline{\omega}_{i1}(t+\Delta t))\overline{u}_5 = 0,$$

где $\overline{u}_3 = \overline{n}_0^{i2} \times \overline{n}_1^{i1}$, $\overline{u}_4 = \overline{n}_0^{i2} \times \overline{n}_2^{i1}$, $\overline{u}_5 = \overline{n}_2^{i2} \times \overline{n}_1^{i1}$.

Для того, чтобы обеспечить ограничения (24), на основе уравнений (12) сформируем импульсы $\Delta\lambda_{j,3}$, $\Delta\lambda_{j,4}$ и $\Delta\lambda_{j,5}$ с помощью уравнений вида

$$\overline{u}_k^T (I_{i1}^{-1} + I_{i2}^{-1})\overline{u}_k \Delta\lambda_{j,k} =$$
$$(\overline{\omega}'_{i1}(t+\Delta t) - \overline{\omega}'_{i2}(t+\Delta t))\overline{u}_k,$$

где $k = 3, 4, 5$.

Для стабилизации ограничений (23) сформируем псевдоимпульсы $\Delta\lambda_{j,3}^*$, $\Delta\lambda_{j,4}^*$ и $\Delta\lambda_{j,5}^*$ на основе уравнений следующего вида

$$\overline{u}_k^T (I_{i1}^{-1} + I_{i2}^{-1})\overline{u}_k \Delta\lambda_{j,k}^* =$$
$$(\overline{\omega}_{i1}'^*(t+\Delta t) - \overline{\omega}_{i2}'^*(t+\Delta t))\overline{u}_k - ERP\frac{G_k}{\Delta t},$$

где $k = 3, 4, 5$.

Вычисленные импульсы $\Delta\lambda_{j,k}$, $k = 1, \ldots, 5$ добавляем в суммарные импульсы $\lambda_{j,k}$ для использования на очередном шаге моделирования.

Далее, вычисляются скорости $\overline{V}(t+\Delta t)$ и $\overline{V}^*(t+\Delta t)$, которые используются для вычисления новых координат и ориентаций тел в механической системе на основе кинематических уравнений.

6. Результаты моделирования

Апробация описанных выше алгоритмов была проведена на PC с процессором Intel Pentium 4 с частотой 3.4 ГГц.

В качестве первого примера рассмотрим цепь из пяти осевых шарниров (рис. 1), где первый шарнир соединяет первое тело в цепи с неподвижной платформой. Твёрдые тела имеют цилиндрическую форму с высотой 80 см и радиусом 10 см, массу $m_i = 20$ кг, главные моменты инерции $I_{x,i} = 0.133$ кг×м^2, $I_{y,i} = I_{z,i} =$ 0.667 кг×м^2. Таким образом, для цепи получаем 25 ограничений вида (1) и (2).

Будем моделировать движение цепи в течение 10 секунд. На рис. 1 показано промежуточное положение цепи в ходе моделирования.

Для постоянного шага моделирования $\Delta t = 1$ мс зависимости количества итераций от момента времени при заданной точности $\varepsilon = 10^{-3}$ и $\varepsilon = 10^{-5}$ представлены на рис. 2. Среднее количество итераций в первом случае составляет примерно 3, а во втором случае примерно 9.

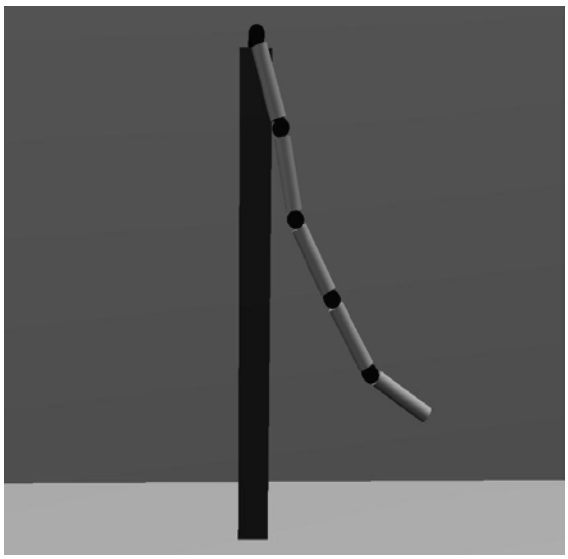


Рис. 1. Цепь из пяти осевых шарниров

Из-за свойства когерентности для соседних промежутков времени сила реакции связей меняется незначительно. Так как импульс характеризует действие силы на некотором промежутке времени, то при одинаковой силе импульс будет пропорционален этому промежутку. Поэтому для моделирования с переменным шагом мы будем масштабировать накопленные импульсы перед применением на каждом шаге следующим образом

$$\bar{\lambda}_j^{\alpha} = \frac{\Delta t_k}{\Delta t_{k-1}} \bar{\lambda}_j^{\alpha}, \alpha \in \{r, s, t\},$$

где Δt_k - переменный шаг моделирования. Шаг моделирования уменьшается или увеличивается в зависимости от того, успевает ли моделирование отобразить реальный процесс движения (шаг подстраивается под реальное время).

Зависимость количества итераций от момента времени при моделировании с переменным шагом и заданной точностью $\varepsilon = 10^{-3}$ и $\varepsilon = 10^{-5}$ представлены на рис. 3. Среднее количество итераций в первом случае составляет примерно 10 итераций, а во втором случае - 33 итерации.

Моделирование с динамическим изменением шага показало, что при пониженной точности потребуется максимум 40 итераций, а при повышенной точности достаточно 80 итераций.

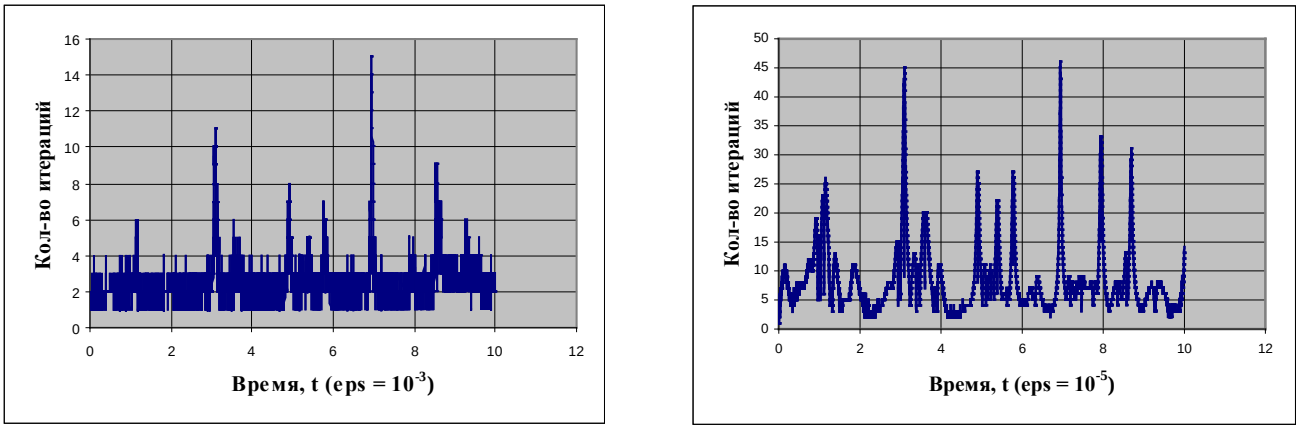


Рис. 2. Зависимость количества итераций от времени при шаге моделирования $\Delta t = 1$ мс

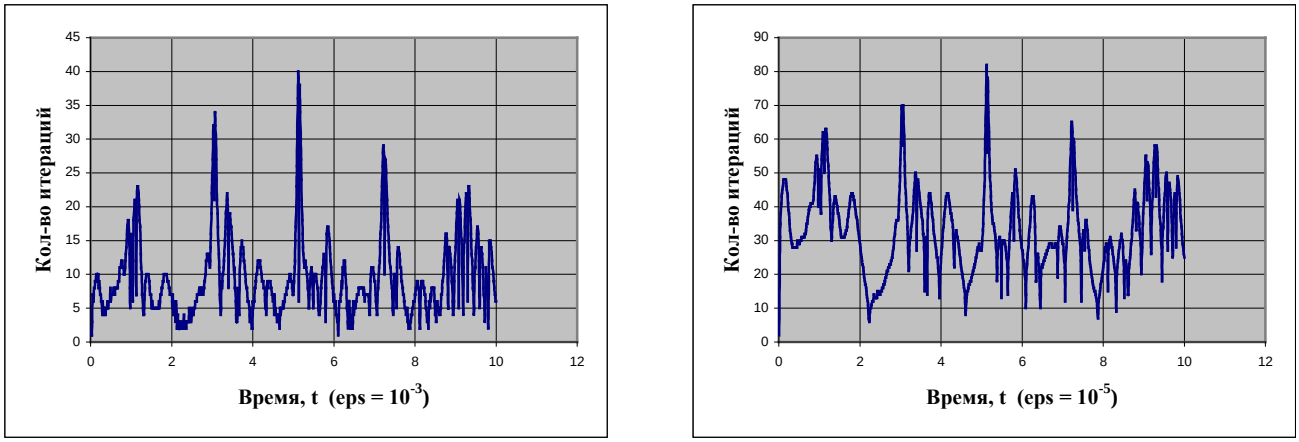


Рис. 3. Зависимость количества итераций от времени при переменном шаге моделирования

Заключение

В данной статье описан метод последовательных импульсов для моделирования шарнирно связанных тел. Для стабилизации ограничений использован метод раздельных импульсов, который является модификацией метода стабилизации Баумгарте. На примерах сферического, осевого и призматического шарниров описан вывод соотношений для импульсов и псевдоимпульсов. Реализованный метод обладает свойством универсальности, что позволяет моделировать произвольную конфигурацию механической системы. Стабилизация ограничений на основе метода раздельных импульсов обладает свойством робастности, что даёт возможность моделировать механическую систему без подкачки энергии. Для ускорения сходимости используется свойство временной когерентности, что позволяет моделировать в режиме реального времени с точностью, допустимой для реалистичного поведения механической системы.

Simulation of articulated multibody system using sequential impulses method

M.V. Mikhaylyuk, E.V. Strashnov

Abstract: Most of simulation methods for articulated multibody system don't have all required properties that are important for animation and virtual reality. To solve this problem the article presents an effective method of sequential impulses with temporal coherence. Constraint stabilization is provided by means of split impulses method. These methods allow us to simulate articulated multibody system in real time with admissible accuracy.

Keywords: multibody system, holonomic constraints, split impulses method, temporal coherence

Литература

1. R. Featherstone. Robot Dynamics Algorithms. Kluwer international series in engineering and computer science: Robotics. Kluwer Academic Publishers, 1987.

2. V. Kokkevis. Analytical Constraints for Articulated Body Dynamics, Lecture notes for the Course "Introduction to Articulated Rigid Body Dynamics", ACM SIGGRAPH, 2005.

3. М.В.Михайлюк, Е.В.Страшнов. Использование метода штрафов при моделировании шарнирно связанных твёрдых тел. Mathematica Montisnigri, Vol. XXVII, Подгорица, 2013, с. 91-106.

4. A. Witkin, D. Baraff. Physically based modeling: Principles and practice. Constrained dynamics. Course Notes, 1997.

5. D. Baraff. Linear-Time Dynamics using Lagrange Multipliers, In SIGGRAPH '96: Proceeding of the 23rd annual conference on Computer Graphics and interactive techniques, ACM Press, pp. 137-146, 1996.

6. R. Weinstein, J. Teran, and R. Fedkiw. Dynamic simulation of articulated rigid bodies with contact and collision. In IEEE Transactions on Visualization and Computer Graphics, volume 12, pages 365-374, 2006.

7. J. Bender, A. Schmitt. Fast dynamic simulation of multi-body systems using impulses. In Virtual reality Interactions and Physical Simulations (VRIPhys) (Madrid (Spain), Nov., 2006), pp. 81-90.

8. E. Catto. Iterative dynamics with temporal coherence. In Game Developer Conference, pages 1-24, 2005.

9. E. Catto. Modelling and solving constraints. Game Developers conference 2009, Slides, 2009.

10. A. A. Shabana. Computational Dynamics, Third edition, John Wiley & Sons Inc., 2010.

11. J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. Computer methods in applied mechanics and engineering, 1(1): 1-16, 1972.

12. Курс теоретической механики: Учебник для вузов / В.И. Дронг, В. В. Дубинин, М.М. Ильин и др.; под общ. ред. К. С. Колесникова, 3-е изд. стереотип. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2005. – 736.

СЦЕНАРНОЕ ИССЛЕДОВАНИЕ УЯЗВИМОСТИ СЛОЖНЫХ ОРГАНИЗАЦИОННО-ТЕХНИЧЕСКИХ СИСТЕМ

Н.О. Пономарев, Д.А. Кононов [1], Д.А. Швецов, Р.О. Пономарев

1 – доктор технических наук

Аннотация: Рассмотрены возможности сценарного подхода для исследования проблемы уязвимости и управления безопасностью сложных организационно-технических систем. Предложена схема моделирования уязвимости в условиях распространения возмущений, вызванных различными видами угроз в организационно-технической системе на операторном графе, а также классификация уязвимостей. Рассмотрены принципы исследования уязвимости в Инет-социальных сетях.

Ключевые слова: сценарный подход, уязвимость технических систем.

Введение

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

Надежность – системный параметр, свойство системы сохранять в течение заданного промежутка времени значение параметров, характеризующих проектный ЦРФ системы. Это свойство характеризует эффективность проектирования системы [1,2].

Надёжность объектов связывают с недопустимостью отказов в работе. Это есть понимание надёжности в «узком» смысле – свойство объекта сохранять работоспособное состояние в течение некоторого времени или некоторой наработки. Иначе говоря, надёжность объекта заключается в отсутствии непредвиденных недопустимых изменений качества его функционирования в процессе проектном режиме эксплуатации и хранения. Надёжность тесно связана с различными сторонами процесса эксплуатации. Надёжность в «широком» смысле – комплексное свойство, которое в зависимости от назначения объекта и условий его эксплуатации может включать свойства безотказности, долговечности, ремонтопригодности, сохраняемости и т.п., а также определённое сочетание этих свойств.

Для количественной оценки надёжности используют так называемые единичные показатели надёжности (характеризуют только одно свойство надёжности) и комплексные показатели надёжности (характеризуют несколько свойств надёжности).

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

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

В соответствии со стратификацией, предложенной в [3–5], ОТС – представляет собой сложную систему (СС) и является первой по иерархии пассивно-активной подсистемой социально-экономической системы (СЭС): здесь определен не только ЦРФ, но и активны субъекты действия (СД), а также присутствуют риски проектирования и эксплуатации типа «человеческий фактор». Свойства надежности, стойкости и живучести здесь носят специфический характер по сравнению с техническими и технологическими системами, поскольку могут практически зависеть от целеполагания и внутреннего состояния указанных субъектов.

Дуальное понятие к указанным свойствам следует считать свойство уязвимости (неуязвимости) системы.

Уязвимость – системный параметр, характеризующий возможность нанесения описываемой системе повреждений любой природы, нарушающих ЦРФ. Априори характер, степень и возможности устранения указанных повреждений зависят от созданной конструкции, условий функционирования (эксплуатации), а также средств воздействия. Таким образом, уязвимость непосредственно связана с надежностью, стойкостью и живучестью ОТС.

В компьютерной безопасности термин «уязвимость» (англ. vulnerability) используется для обозначения недостатка в системе, используя который можно намеренно нарушить её целостность и вызвать неправильную работу. Уязвимость может быть результатом ошибок программирования, недостатков, допущенных при проектировании системы, ненадежных паролей, вирусов и других вредоносных программ, скриптовых и SQL-инъекций. Ряд уязвимостей известны только

теоретически, другие активно используются и имеют известные эксплойты.

Обычно уязвимость позволяет атакующему «обмануть» приложение: заставить его совершить действие, на которое у того не должно быть прав: в систему внедряют программу данных или код в так, что программа воспримет их как «свои». Ряд уязвимостей появляются из-за недостаточной проверки данных, вводимых пользователем, позволяют вставить в интерпретируемый код произвольные команды (SQL-инъекция, XSS, SiXSS). Другие уязвимости появляются из-за более сложных проблем, пример запись данных в буфер без проверки его границ (переполнение буфера).

Для коммуникационной сети (как и для других сложных систем, представляемых в виде графов) численный расчет ее надежности может оказаться задачей, требующей значительных временных ресурсов. По сути, построение дерева отказов для коммуникационной сети сводится к простому перебору всех возможных вариантов неполучения информационного сигнала, передаваемого от одной вершины к другой.

Исследование «мест уязвимости» системы, изменение режима функционирования которых приводит к существенному отклонению от ЦРФ, до сих пор практически не изучалось.

Для описания и моделирования сложно структурированных систем используют матричный анализ, теорию графов, ряд других методов.

Модельное представление структуры сложной системы в виде графа – общепризнанный подход. В такой модели в вершинах графа сосредоточены измеряемые величины (характеристики) системных элементов. Структура системы и взаимодействие элементов при функционировании системы представляют в виде ориентированного графа. Каждым вершине и дуге графа соответствуют параметры и функционалы, адекватно описывающие процессы функционирования элементов исследуемой (моделируемой) системы. Таким образом, текущее состояние системы – значения вершин и структуры системы в текущий момент времени *t*.

Динамика изменения состояния системы в указанной модели может быть задана принципиально двумя различными способами: импульсом (мгновенным возмущением, изменением) значения параметров в заданных вершинах и/или изменением структуры системы (матрицы смежности орграфа), а также их комбинацией. Вектор начального импульса распространяется по структуре, изменяя параметры вершин. В системах с изменяющейся структурой целесообразно вести контроль изменений структуры для формирования спектра соответствующих свойств и характеристик. Достижение этой цели лежит в русле решения задач *реконфигурации или управления структурной динамикой*.

В [1,2] предложены модели распространения возмущений по ОТС, на основе которых разработаны методы их сценарного и индикаторного анализа.

В настоящей работе предложены схема моделирования и исследования проблемы уязвимости для разработки моделей и методов поиска мест (траекторий) уязвимости в ОТС и обеспечения безопасности, основанные на применении графовых моделей различного типа и методов нелинейного программирования.

Исследование уязвимости компонентов СС проводится средствами сценарного анализа. Как известно, под сценарным исследованием понимают способ изучения СС, когда основным средством исследования является построение и анализ спектра сценариев в различных ее стратах, а целью исследования – синтез сценария с заданными свойствами. Формализация соответствующей системы моделей рассмотрена в ряде работ [1–7].

1. Модель и классификация уязвимости сложной системы

Разработку модели уязвимости проведем на исходном формализованном языке исследования сложной системы, предложенном в [1, 4, 5]. Основные его компоненты фиксируют следующую иерархию понятий.

Модель 1. Компоненты сложной системы.

1) субстрат системы (набор элементов) X;
2) структура системы (связи между элементами) $R^{(X)}$;
3) внутренний концепт системы (набор свойств и характеристик) $P^{(X)}$;
4) системный элемент (внутренний) $E^{(X)}(B, r, p)$, где $B \subseteq X$, $r \in R^{(X)}$, $p \in P^{(X)}$;
5) внутреннее состояние системы (набор системных элементов и отношений между ними) υ;
6) субстрат окружения: набор элементов Y;
7) структура окружения (связи между элементами) $R^{(Y)}$;
8) набор свойств и характеристик окружения $P^{(Y)}$;
9) системный элемент внешней среды $E^{(Y)}(B, r, p)$, где $B \subseteq Y$, $r \in R^{(Y)}$, $p \in P^{(Y)}$;
10) состояние окружения системы (формируется по той же схеме) χ;
11) расширенное состояние системы $\boldsymbol{z}=(\upsilon,\chi)$;
12) характеристики внешней среды системы $\zeta^{(Y)}$;
13) совместные характеристики системы и внешней среды $\zeta^{(X,Y)}$,
14) концепт системы (назначение + интересы +…) П.

Тогда формальная система S – набор отношений между этими компонентами [5].

Временные связи между компонентами задает динамическая модель DS поведения системы S.

Пусть задан временной отрезок $\Delta=[t_0, T]$.

Определение 1. Последовательность $\Re(\Delta)=\{\boldsymbol{z}(t),\ t\in\Delta\}$ экспертно-значимых событий назовем пошаговым сценарием поведения системы S на горизонте сценария Δ.

Отметим, что при $t_0=T$ сценарий представляет собой текущее расширенное состояние системы. Общая методология сценарного исследования сложных систем предложена в [5]. Экспертно-значимое событие (ЭЗС) определяет лицо, принимающее решение (ЛПР).

Качество функционирования СС, т.е. соответствие процесса функционирования ее назначению, а также безопасности, определим областью целевого назначения и безопасности (ОЦНБ) $Q(\boldsymbol{z},t)$, зависящей от ее расширенного состояния.

Общее определение безопасности и качества функционирования, фиксирующее установки проектного режима, представляет собой условие [5]

(1) $\mathbf{z}(t)\in Q(\mathbf{z},t)$ при $t\in\Delta$.

Условия (1) устанавливают в зависимости от предметной области, объекта исследования, характера и условий функционирования и т.п.

Определение 2. Сценарий $\Re(\Delta)$ назовем сценарием безопасности, если для всех $t\in\Delta$ выполнено (1).

На множестве $M\Re(\Delta,\sigma)$ сценариев

(2) $M\Re(\Delta,\sigma)=\{\Re(\Delta,\sigma)\ \sigma\in\Omega\}$

введем меру $\rho(\Re(\Delta,\sigma), Q)$ близости сценария $\Re(\Delta,\sigma)$ множеству Q, величину отклонения

(3) $\eta_s(t)=\rho(\Re(\Delta,\sigma), Q, t)$

в момент времени t, а также условия допустимого отклонения в виде

(4) $\rho(\Re(\Delta,\sigma), Q)\le\varepsilon$,

где $\varepsilon\ge 0$ – допустимое отклонение от условия (1). Ряд таких характеристик указаны в [11].

Для формирования оценок отклонения определим следующие характеристики сценария $\Re(\Delta)$ развития ситуаций на горизонте $\Delta=[t_0,T]$:

– текущее удаление сценария $\Re$ от целевого вектора **a**

$$d_E^{(t)}(\Re,\mathbf{a})=\rho_E\left(\mathbf{z}(t),\mathbf{a}\right)=\left\|\mathbf{z}(t)-\mathbf{a}\right\|_E;$$

характеристика может быть использована для мониторинга фактического отклонения текущего расширенного состояния системы от ЦРФ в ходе реализации сценария и/или являться текущей оценкой близости границы безопасности функционирования системы;

– текущее угловое удаление сценария $\Re(\Delta)$ от целевого направления **b**

$$d_a^{(t)}(\Re,\mathbf{b})=\rho_a\left(\mathbf{z}(t),\mathbf{b}\right)=\left\|\frac{\mathbf{z}(t)}{\|\mathbf{z}(t)\|_E}-\frac{\mathbf{b}}{\|\mathbf{b}\|_E}\right\|_E;$$

характеристика может быть использована для мониторинга фактического отклонения текущего расширенного состояния системы от желательного направления развития в ходе реализации сценария и/или являться текущей оценкой близости направления;

– минимальное удаление сценария $\Re(\Delta)$ от вектора **a** на горизонте Δ

$$d_E^{\min}(\Re,\mathbf{a},\Delta)=\min_{t\in\Delta} d_E^{(t)}(\Re,\mathbf{a},\Delta);$$

характеристика может быть использована как аналитическая (целевая) для решения задачи о достижении сценарием ЦРФ и/или являться промежуточной (заключительной) оценкой уязвимости СС;

– минимальное угловое удаление сценария $\Re(\Delta)$ от направления **b** на горизонте Δ

$$d_a^{\min}(\Re,\mathbf{b},\Delta)=\min_{t\in\Delta} d_a^{(t)}(\Re,\mathbf{b},\Delta);$$

характеристика может быть использована как аналитическая (целевая) для решения задачи о достижении сценарием желательного направления развития и являться промежуточной (заключительной) оценкой уязвимости СС;

– максимальное удаление сценария $\Re(\Delta)$ от вектора **a** на горизонте Δ

$$d_E^{\max}(\Re,\mathbf{a},\Delta)=\max_{t\in\Delta} d_E^{(t)}(\Re,\mathbf{a},\Delta);$$

характеристика может быть использована как аналитическая (целевая) для решения задачи о достижении сценарием ЦРФ и/или являться промежуточной (заключительной) оценкой уязвимости СС;

– максимальное угловое удаление сценария $\Re(\Delta)$ от направления **b** на горизонте Δ

$$d_a^{\max}(\Re,\mathbf{b},\Delta)=\max_{t\in\Delta} d_a^{(t)}(\Re,\mathbf{b},\Delta);$$

характеристика может быть использована как аналитическая (целевая) для решения задачи о достижении сценарием желательного направления развития и являться промежуточной (заключительной) оценкой уязвимости СС;

– множество периодов выхода сценария $\Re(\Delta)$ за ε-окрестность вектора **a**

$$T_E^{(t,out)}(\Re,\mathbf{a},\Delta,\varepsilon)=\underset{t\in\Delta}{Arg}\left(d_E^{(t)}(\Re,\mathbf{a},\Delta)\ge\varepsilon\right);$$

– множество периодов пребывания сценария $\Re(\Delta)$ в ε-окрестности вектора **a**

$$T_E^{(t,in)}(\Re,\mathbf{a},\Delta,\varepsilon)=\underset{t\in\Delta}{Arg}\left(d_E^{(t)}(\Re,\mathbf{a},\Delta)\le\varepsilon\right);$$

характеристики могут быть использованы как целевые для решения задачи о максимальном ущербе при реализации сценария возмущений и являются промежуточной (заключительной) оценкой уязвимости СС;

– первый момент выхода сценария $\Re(\Delta)$ за ε-окрестность вектора **a**

$$T_E^{(\min,out)}(\Re,\mathbf{a},\Delta,\varepsilon)=\min_{t\in\Delta}\left(T_E^{(t,in)}(\Re,\mathbf{a},\Delta,\varepsilon)\right);$$

– последний момент пребывания сценария $\Re(\Delta)$ в ε-окрестности вектора **a**

$$T_E^{(\max,in)}(\Re,\mathbf{a},\Delta,\varepsilon)=\max_{t\in\Delta}\left(T_E^{(t,in)}(\Re,\mathbf{a},\Delta,\varepsilon)\right);$$

характеристики могут быть использованы как целевые для решения задачи о времени выхода из строя СС.

Можно ввести аналогичные угловые характеристики, фиксирующие удаленность сценария от заданной ε-конической окрестности и их производные.

Связующим понятием между характеристиками качества, безопасности, стойкости, живучести и уязвимостью системы следует считать понятие угрозы. Угрозой $\gamma\in\Gamma^{(У)}$ назовем фактор (явление), реализация которого может привести к гипотетической возможности реализации нежелательных явлений (ситуаций). При описании СС в виде формального системного объекта под угрозой будем понимать совокупность факторов системы и внешней среды, могущих привести с точки зрения ЛПР к существенному ухудшению состояния ее параметров, в том числе НШС, выходящих за рамки (1).

Реализация угрозы $\gamma\in\Gamma^{(У)}$ по отношению к некоторой компоненте S_α априори приводит к нарушению ЦРФ и возмущенному сценарию $\Re(\Delta,\gamma)$ поведения (функционирования), отличному от $\Re(\Delta)$.

Определение 3. Скажем, что на горизонте Δ система S обладает свойством сценарной стойкости по отношению к угрозе $\gamma\in\Gamma^{(У)}$, если сценарий $\Re(\Delta,\gamma)$ является сценарием безопасности.

Определение 4. Скажем, что на горизонте Δ система S обладает свойством сценарной стойкости по отношению к системе угроз $\Gamma^{(У)}$, если для каждого $\gamma\in\Gamma^{(У)}$ сценарий $\Re(\Delta,\gamma)$ является сценарием безопасности.

Возможность реализации угрозы, связанной с определенными компонентами исследуемой системы, представляет собой ее *место уязвимости*.

В рамках предложенного формализма местом уязвимости могут быть указанные компоненты, их комбинация, а также элементы динамической модели *DS*. С точки зрения теории управления непосредственное воздействие ЛПР может быть оказано на компоненты внутреннего состояния, в то время как опосредованным влиянием оно может быть оказано и на компоненты внешнего и совместного окружения.

Модель 1 и система введенных определений дают возможность сформулировать общую концепцию уязвимости сложной системы.

Модель 2. Концептуальная модель уязвимости сложной системы.

Пусть заданы

– набор компонентов S_α $\alpha\in\hat{A}$ сложной системы S в текущий момент времени t;

– множество угроз $\Gamma^{(Y,\,\alpha)}$, которые могут быть реализованы по отношению к компоненте S_α $\alpha\in\hat{A}$;

– множество возможных ремонтов $\Gamma^{(P,\,\alpha)}$, которые могут быть реализованы по отношению к компоненте S_α $\alpha\in\hat{A}$;

– меры удаленности $\rho_\alpha(\boldsymbol{z}(t), C_\alpha)$ компоненты S_α СС от целевого режима ее функционирования C_α;

– допустимая граница удаленности ε_α компоненты S_α СС от C_α;

– мера удаленности $\rho_s(Q, \rho_\alpha)$ от ОЦНБ;

– величина ущерба $W(\Re(\Delta,\gamma), Q)$ при удаленности сценария $\Re(\Delta,\gamma)$ от ОЦНБ Q.

Определение 5. Компоненту S_α системы S назовем локально уязвимой, если найдется угроза $\gamma\in\Gamma^{(Y,\,\alpha)}$, реализация которой нарушает ее стойкость, т.е.

(5) $\rho_\alpha(\Re(\Delta,\gamma), Q)> \varepsilon_\alpha$,

Уязвимость компонентов системы можно сравнивать по коэффициентам уязвимости.

Определение 6. Коэффициентом $C_\alpha^{(v)}(\gamma, \Delta)$ уязвимости компоненты S_α СС при применении угрозы $\gamma\in\Gamma^{(Y,\alpha)}$ на временном отрезке Δ назовем величину ущерба $W(\Re(\Delta,\gamma), Q)$.

Таким образом могут быть исчислены величины

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

(6) $C_\alpha^{(v,\,min)}(\gamma, \Delta)= \min\limits_{\gamma\in\Gamma^{(Y,\alpha)}} C_\alpha^{(v)}(\gamma, \Delta)$

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

(7) $C_\alpha^{(v,\,max)}(\gamma, \Delta)= \max\limits_{\gamma\in\Gamma^{(Y,\alpha)}} C_\alpha^{(v)}(\gamma, \Delta)$

– минимального коэффициента уязвимости для компонентов системы

(8) $C^{(v,\,min)}(\gamma, \Delta)= \min\limits_{\alpha\in\hat{A}} C_\alpha^{(v,\,min)}(\gamma, \Delta)$

– максимального коэффициента уязвимости для компонентов системы

(9) $C^{(v,\,max)}(\gamma, \Delta)= \max\limits_{\alpha\in\hat{A}} C_\alpha^{(v)}(\gamma, \Delta)$

На основе этих характеристик можно ввести

Определение 7. Компоненту S_α сложной системы S назовем системно уязвимой, если найдется угроза $\gamma\in\Gamma^{(Y,\,\alpha)}$, реализация которой нарушает ее стойкость системы S.

Классификацию типов уязвимости СС проведем по следующим признакам:

– характер повреждений;

– степень повреждений;

– ремонтоспособность;

– способы уязвимости (способы реализации угроз).

Характер внутренних повреждений

По характеру повреждений выделим:

– элемента системы $x_i\in X$;

– конструктивного решения (структуры системы) $R(X)$;

– концепта системы, т.е. заданных (проектируемых) свойств, в том числе целевого процесса функционирования (ЦРФ) $P(X)$;

– системных элементов, в том числе подсистем рассматриваемой системы $ME^{(X)}(B, r, p)$;

– внутреннего состояния системы, т.е. набора системных элементов и связей между ними;

– комплексные повреждения.

По степени повреждений выделим:

– сильные повреждения;

– средние повреждения;

– несущественные повреждения.

По ремонтоспособности выделим:

– фатальные нарушения;

– нарушения, требующие существенного ремонта;

– нарушения, требующие незначительного ремонта;

– нарушения, не требующие ремонта.

По способам уязвимости выделим:

– уязвимости при однократной точечной реализации угрозы;

– уязвимости при однократной групповой реализации угрозы;

– уязвимости при однократной комплексной реализации угроз;

– уязвимости при многоразовой точечной реализации угрозы;

– уязвимости при многоразовой групповой реализации угрозы;

– уязвимости при многоразовой комплексной реализации угроз.

Определения указанных типов уязвимости могут быть достаточно просто формализованы на основе предложенной методологии уязвимости.

В литературе известны способы борьбы с уязвимостью элементов системы, в частности, основанных на понятии отказоустойчивости, реконфигурации и ремонтоспособности.

Так, в качестве функциональных элементов (ФЭ) сложной системы рассматривали многофункциональные унифицированные (однородные) вычислительные средства. Для уменьшения уязвимости предлагалась процедура ее реконфигурации, заключавшаяся в следующем. Множество решаемых задач разбивается на группы задач с близкими (одинаковыми) характеристиками. Каждая группа задач решается на одном ФЭ. При отказе ФЭ выполняемая им группа задач передается на ФЭ, где решаются задачи с наиболее низкими приоритетами. Если не удается решить задачи объединенных групп, то задачи группы с низкими приоритетами снимаются с решения.

Рассматривался вариант, когда ФЭ взаимодействуют посредством некоторой телекоммуникационной подсистемы. Каждый ФЭ содержит процессоры, оперативное и долговременное запоминающее устройство, соответствующие интерфейсы, взаимодействует с внешней средой (объектами управления, операторами и др.) с помощью выделенных для этого аппаратно-программных средств. Для начального состояния осуществляется построение плана распределения задач и информационных потоков с учетом технологических, технических, стоимостных, временных, ресурсных и т.п. ограничений. В случае перехода СС (вызванного отказом некоторой совокупности ФЭ) в НШС осуществляется перераспределение решаемых ею задач между работоспособными ФЭ. Обеспечение требуемого уровня отказоустойчивости СлО осуществляется путем итогового размещения в различных ФЭ резервных копий алгоритмов решаемых задач.

Анализ показывает, что при решении задачи реконфигурации функционирования СС в рамках указанных процедур, как правило, требуется сформировать множество промежуточных состояний, переход в которые приводит к потере управления рассматриваемыми объектами, либо множества состояний, вероятности перехода в которые из начального состояния не менее заданной величины. При указанных предпосылках задачу реконфигурации СС для устранения или уменьшения уязвимости в условиях действия случайных возмущений целесообразно представлять марковским процессом с использованием подходов, описанных в [12].

Введенные понятия рассмотрим на примере модели распространения возмущений по структуре сложной организационно-технической системы.

2. Сценарная модель распространения возмущения

В настоящем разделе рассмотрим модель распространения возмущений по структуре ОТС, на основе которой возможно провести сценарное исследование ее уязвимости.

Сценарная модель распространения возмущений предполагает возможность определения связей между ресурсами работоспособности элементов ОТС.

Модель 3. Структурная модель ОТС.

Потенциалом или ресурсом работоспособности элемента системы $x_i \in X$ ($i \in \{1,2\ldots n\}=\tilde{N}$) в момент времени t назовем величину $v_i(t)$, характеризующую способность элемента выполнять заданные функции в соответствии с установленным целевым режимом его функционирования. Соединение элементов в единую систему задается ее структурой, которая задается матрицей смежности $A(t)$ в момент времени t. Элемент $a_{ij}(t)$ $i,j \in \tilde{N}$ характеризует долю изменения ресурса работоспособности j-го элемента системы при совместной работе с i-м ее элементом.

Изменение ресурса работоспособности может происходить по следующим причинам:

– совместный режим функционирования со смежным элементом;

– естественный износ элемента;

– внешние воздействия, в том числе:

– негативные внешние воздействия,

– восстановление ресурса на основе его ремонта;

– изменение режима совместного функционирования смежных элементов.

Совместный режим функционирования j-го элемента системы при совместной работе с i-м ее элементом изменяет его ресурс на величину $a_{ij}(t)v_i(t)$.

Естественный износ элемента $x_i \in X$ моделирует петля: $a_{ii}(t) \neq 0$. Внешние воздействия моделируют импульсы, изменяющие значение ресурса работоспособности. Изменение режима совместного функционирования смежных элементов моделирует изменение матрицы смежности.

Для сценарного исследования ОТС на орграфе оперирующая сторона располагает следующими средствами мониторинга в момент времени t:

– обнаруживать *накопленные* изменения состояния системы в виде k-шагового импульсного процесса изменений (ИПИ)

(10) $\mathbf{Im}(t,k)=(I_j(\tau)\ 1\le j\le n;\ t\le\tau\le t+k)=(\mathbf{I}(\tau)\ t\le\tau\le t+k)$;

– обнаруживать *накопленные* изменения структуры в виде k-шагового структурного процесса изменений (СПИ)

(11) $\Phi(t,k)=(A(\tau)\ t\le\tau\le t+k)$;

– обнаруживать *накопленные* изменения в виде комплексного процесса изменений (КПИ)

(12) $K(t,k)=(\mathbf{Im}(t,k), \Phi(t,k))$.

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

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

регистрировать k-шаговый структурный процесс возмущений (СПВ)

(14) $\delta\Phi(t,k)=(\delta A(\tau)\ t\le\tau\le +k)$;

а также регистрировать k-шаговый комплексный процесс возмущений (КПВ)

(15) $\delta K(t,k)=(\delta\mathbf{Im}(t,k), \delta\Phi(t,k))$.

В соответствии со схемой преобразования состояния в операторных графах динамику системы определяет соотношение:

(16) $\mathbf{v}(t)=\mathbf{v}(t-1)+\mathbf{Im}(t)$ при $t=1, 2,\ldots$

или

(17) $\mathbf{Im}(t)=A(t-1)\mathbf{Im}(t-1)+\delta\mathbf{Im}(t)$ при $t=1, 2,\ldots$

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

Преобразование состояния происходит по следующему алгоритму в момент времени t:

– на вход вычисления подаются: предыдущие состояние $\mathbf{v}(t-1)$, накопленный импульс $\mathbf{Im}(t-1)$, а также матрица смежности $A(t\text{-}1)$;

– вносится текущий импульс (импульсное изменение) $\delta\mathbf{Im}(t)$;

– вычисляется текущий накопленный импульс (17);

– по правилу (16) вычисляется текущее состояние $\mathbf{v}(t)$;
– вносится структурное аддитивное управление $\delta A(t)$ и матрица смежности преобразуется по правилу:
(18) $A(t)=A(t–1)+\delta A(t)$;
– проверяется условие завершенности горизонта сценария $t=T$:
в случае $t<T$ проводится очередной шаг;
в случае $t=T$ процесс завершается.

Пусть задан комплексный процесс возмущений (15). Тогда в соответствии с указанным алгоритмом преобразования состояний системы (16)–(17) может быть получена последовательность состояний $V(\delta K(t,k))$
(19) $V(\delta K(t,k))=\{\mathbf{v}(t)$ при $t=0,1, 2,\ldots\}$,
которую назовем *k-шаговым детерминированным сценарием* развития системы (16)–(17), соответствующим комплексному процессу возмущений $\delta K(t,k)$.

Выход величины ресурса работоспособности $v_i(t)$ за рамки условия (1) характеризует *i*-элементную стойкость ОТС. На определении характеристики «временная граница стойкости» $T_i^{(re)}(K(t,k))$ элемента может быть построена матрица взаимных временных отказов $T_{ij}^{(re)}$ работоспособности элементов $x_i\in X$, т.е. времени отказа элемента $x_i\in X$ при отказе элемента $x_j\in X$ в момент времени t_0.

Действительно, представим комплексный процесс возмущений $\delta K(t_0,k)$ при отказе элемента $x_j\in X$ в виде автономного импульсного процесса, запущенного в момент времени t_0:
(20) $\delta K(t_0,k)=(\delta\mathbf{Im}(t_0,k), \delta\Phi(t_0,k))$,
$\delta\mathbf{Im}(t_0,k)=(\delta\mathbf{I}^{(j)}(t_0), \mathbf{0},\ldots\mathbf{0})$, $\delta\Phi(t_0,k)=0$ при $t= t_0,\ldots, t_0,+k$,
где $\delta\mathbf{I}^{(j)}{}_j(t_0)= v_j(t_0)–a_j(t_0)\pm \varepsilon_j(t_0)$, $\delta\mathbf{I}^{(j)}{}_p(t_0)=0$ при $p\neq j$, t_0 – момент утери работоспособности, $a_j(t_0)$ – проектный режим, $\varepsilon_j(t_0)$ – граница работоспособности, $v_j(t_0)$ – потенциал работоспособности элемента $x_j\in X$ в момент отказа, $\mathbf{0}$ – нулевые *n*-мерные векторы, $\mathbf{v}(t_0)$ – состояние системы в момент отказа t_0. Пусть
(21) $[\mathbf{v}(t_0)+A^k\delta\mathbf{I}^{(j)}(t_0)]_i\geq a_j+\varepsilon_i$, $T_{ij}^{(re+)}=\min k$,

Так, формальное определение характеристик «временная граница стойкости» и «**w**-ремонтоспособность» в указанной работе основано на предположениях, что определение границ допустимости рассогласования $\varepsilon^a(t)$ между ЦРФ $\mathbf{a}(t)$ и текущим состоянием $\mathbf{v}(t)$ работоспособности ОТС сформулировано виде условий:
(22) $\rho(\mathbf{v}(t)–\mathbf{a}(t))\leq\varepsilon^a(t)$при $t=1, 2,\ldots$,
где ρ – заданная метрика, определяет временные границы стойкости системы по отношению к комплексным возмущениям. В частности, при задании границ допустимости по каждому элементу системы
(23) $\boldsymbol{\varepsilon}^a(t)=\{\varepsilon^a{}_i(t), i=1,\ldots,n\}$ при $t=1, 2,\ldots$,
временные границы стойкости системы по отношению к комплексным возмущениям должны удовлетворять условиям:
(24) $\delta\mathbf{a}(t)=\mathbf{v}(t)–\mathbf{a}(t)$,
(25) $|\delta\mathbf{a}_i(t)|\leq\varepsilon^a{}_i(t)$ при $t=1, 2,\ldots, i=1,\ldots,n$.

В [1] формализованы понятия *сценарной* угрозы, стойкости, ремонтоспособности и живучести ОТС. Это позволяет рассматривать задачи исследования и управления безопасностью ее функционирования, определяя, в частности, различные сценарные характеристики рассмотренных понятий, а также критерии безопасности $Q(\mathbf{v},t)$.

Выход величины ресурса работоспособности $v_i(t)$ за рамки условия (24) характеризует *i*-элементную стойкость ОТС. На определении характеристики «временная граница стойкости» $T_i^{(re)}(K(t,k))$ элемента может быть построена матрица взаимных временных отказов $T_{ij}^{(re)}$ работоспособности элементов $x_i\in X$, т.е. времени отказа элемента $x_i\in X$ при отказе элемента $x_j\in X$ в момент времени t_0.

Действительно, представим комплексный процесс возмущений (15) при отказе элемента $x_j\in X$ в виде автономного импульсного процесса, запущенного в момент времени t_0:
(26) $\delta K(t_0,k)=(\delta\mathbf{Im}(t_0,k), \delta\Phi(t_0,k))$,
$\delta\mathbf{Im}(t_0,k)=(\delta\mathbf{I}^{(j)}(t_0), \mathbf{0},\ldots\mathbf{0})$, $\delta\Phi(t_0,k)=0$ при $t= t_0,\ldots, t_0,+k$,
где $\delta\mathbf{I}^{(j)}{}_j(t_0)= v_j(t_0)–a_j(t_0)\pm \varepsilon_j(t_0)$, $\delta\mathbf{I}^{(j)}{}_p(t_0)=0$ при $p\neq j$, t_0 – момент утери работоспособности, $a_j(t_0)$ – проектный режим, $\varepsilon_j(t_0)$ – граница работоспособности, $v_j(t_0)$ – потенциал работоспособности элемента $x_j\in X$ в момент отказа, $\mathbf{0}$ – нулевые *n*-мерные векторы, $\mathbf{v}(t_0)$ – состояние системы в момент отказа t_0. Пусть
(27) $[\mathbf{v}(t_0)+A^k\delta\mathbf{I}^{(j)}(t_0)]_i\geq a_j+\varepsilon_i$, $T_{ij}^{(re+)}=\min k$,
(28) $[\mathbf{v}(t_0)+A^k\delta\mathbf{I}^{(j)}(t_0)]_i\leq a_j–\varepsilon_i$, $T_{ij}^{(re–)}=\min k$.
Тогда
(29) $T_{ij}^{(re)}=\min(T_{ij}^{(re–)}, T_{ij}^{(re+)})$.

Отметим, что при отсутствии структурной связи между элементами $x_i\in X$ и $x_j\in X$, т.е. пути из x_j в x_i, величина $T_{ij}^{(re)}=\infty$. В таком случае угроза их взаимного отказа исключена. Следовательно, в терминах исходных определений реализация отказа элемента $x_j\in X$ является $T_{ij}^{(re)}$-угрозой отказа элемента $x_i\in X$ [1].

Уязвимость для рассмотренной модели может быть рассмотрена не основе ряда определений, вытекающих из результатов работы [10].

Пусть фиксированы
– момент времени начала возмущения t_0,
– ЦРФ $\mathbf{a}(t_0)$ в момент времени t_0,
– состояние системы $\mathbf{v}(t_0)$ в момент времени t_0,
– состояние структуры $A(t_0)$ в момент времени t_0,
– момент времени мониторинга возмущения t,
– ЦРФ $\mathbf{a}(t)$ в момент времени t,
– состояние системы $\mathbf{v}(t)$ в момент времени t,
– вектор допустимых отклонений от ЦРФ $\boldsymbol{\varepsilon}(t)$ в момент времени t,
– элемент $x_j\in X$.

Тогда может быть определен вектор $\delta\mathbf{a}(t)$ (24), который назовем *полем изменений ЦРФ*. Множество
(30) $V^a(t)=\{i\in N \mid \delta\mathbf{a}_i(t)\neq 0)\}$,
т.е. набор номеров элементов, ЦРФ которых изменен, назовем *областью изменений*, множество
(31) $V^a{}_д(t, \boldsymbol{\varepsilon}(t))=\{i\in N \mid$ выполнено (25)$\}$
область допустимых изменений, а множество
(32) $V^a{}_у(t, \boldsymbol{\varepsilon}(t))=N\backslash V^a{}_д(t, \boldsymbol{\varepsilon}(t))$
областью уязвимости в момент времени t. Элемент $x_i\in X$, для которого $i\in V^a{}_у(t, \boldsymbol{\varepsilon}(t))$ называется уязвимым в момент времени t.

Теорема 1. Пусть в момент времени t зафиксирован КПВ $K^{(B)}(\Delta)$ (15), начатый в момент начала возмущения t_0 и определенный на временном сегменте $\Delta=[t_0,$

T]. Тогда поле изменений ЦРФ в момент времени t задает соотношение

(33) $\mathbf{\delta a}(t)=[\sum_{\tau=0}^{t}\prod_{r=\tau}^{t}[\sum_{l=1}^{t-r}\delta A(t_0+l)+A(t_0)]\mathrm{I}^{(0)}(\tau+t_0)]+\mathbf{\delta a}(t_0)$,

где $I^{(0)}(t)=I^{(\mathrm{B},0)}(t)$; $\delta A(t)=\delta A^{(\mathrm{B})}(t)$ при $t\in\Delta$; или

(34) $\mathbf{\delta a}(t)=\sum_{\tau=0}^{t}\prod_{r=\tau}^{t}[\delta_{(t-r)}A+\mathrm{A}^{t-\tau}(t_0)]\mathrm{I}^{(0)}(t_0+\tau)]+\mathbf{\delta a}(t_0)$,

где $\delta_{(t-r)}A=\sum_{l=1}^{t-r}\delta A(t_0+l)$, т.е. изменения матрицы $A(t_0)$, накопленные к моменту времени t-r.

Доказательство следует из Утверждения 4 в [11] и определения поля изменений ЦРФ.

В рамках предложенных формальных определений и коэффициентов уязвимости можно формализовать
– уязвимость элемента $x_i\in X$ системы – $i\in V^{\mathrm{a}}{}_{\mathrm{y}}(t, \mathbf{\varepsilon}(t))$;
– уязвимость конструктивного решения (структуры системы) – изменение матрицы A;
– уязвимость процесса функционирования (ЦРФ);
– степень и типы уязвимости.

3. Уязвимость Инет-социальных сетей

На вербальном уровне под Инет-социальной сетью понимают систему социального взаимодействия, при котором коммуникация обеспечивается средствами сети Интернет [9]. Там же предложена методология исследования, основанная на ее представлении как сложной стратифицированной системы.

Область описания объекта исследования подвергают стратификации на основе различных признаков. Для социально-экономической системы выделяют следующие страты: технологическая, организационная, правовая, экономическая, социальная, культурологическая.

I. Технологическая страта представляет собой, с одной стороны, комплекс технических и программных средств, используемых для коммуникации Участниками сети, с другой – правила, приемы и способы взаимодействия элементов этого комплекса (компьютерной сети), протоколы передаваемых данных, способов организации и взаимосвязи всех элементов системы.

Обычно под компьютерной сетью понимают совместное подключение компьютеров к единому каналу передачи данных, а также систему взаимосвязанных аппаратных и программных компонентов, осуществляющих обработку информации и взаимодействующих с другими подобными системами. Аппаратные компоненты сети включают компьютеры и коммуникационное оборудование, программные компоненты – сетевые операционные системы и сетевые приложения, описывающие способы передачи, хранения и обработки данных.

II. Организационная страта – это формализованное описание организационной и функциональной структур, определяемых информационными отношениями между элементами и управленческими связями. По своим полномочиям Участники социальной сети значительно отличаются.

Особенностью сетей является децентрализованность их структуры, при которой каждый Участник, представленный в виде аккаунта, является самостоятельной вершиной. При регистрации аккаунта в Инет-социальной сети пользователь обязан указать свои идентификационные данные в виде e-mail, имени и иногда номера телефона. Авторизация производится с помощью указания электронного адреса и пароля.

III. Правовая страта описывает законодательное регулирование деятельности людей в социальных сетях. Участники руководствуются законодательством в области информационных технологий и общим законодательством.

IV. Экономическая страта описывает отношения, которые регулируются категорией стоимости. В инет-социальной сети Участниками осуществляется рекламная деятельность, организация реализации товаров и услуг, предоставление информационных услуг и реализация информационных продуктов.

V. К социальной страте относится совокупность необходимых данных, описывающих протекание социальных процессов. Социальная страта показывает социальную структуру Участников социальных сетей.

VI. Культурологическая страта описывает отношения элементов социальной сети с точки зрения специфики этнических, исторических, религиозных и других культурологических особенностей.

В каждой из перечисленных страт понятие уязвимости имеет свои особенности в зависимости от того, каков предмет деятельности в её рамках. Определим понятие уязвимости в каждой из перечисленных страт и выделим характеристики, по которым можно отслеживать нарушение целевого режима функционирования (ЦРФ).

I. В технологической страте нарушение ЦРФ будет характеризоваться невозможностью коммуникации между элементами Инет-социальной сети. Для идентификации нарушения ЦРФ в данной страте можно использовать следующие характеристики: физические нарушения и отключения средств передачи данных, сохранность сетей передачи данных, нарушения работоспособности Интернета и т.п.

Таким образом, уязвимость Инет-социальной сети в технологической страте – это возможность нанесения ей повреждений, приводящих к невозможности коммуникации между её элементами при нарушении физической неисправности.

II. В организационной страте уязвимость Инет-социальной сети – это возможность нанесения ей повреждений, приводящих к разрушению организационной и функциональной структуры, управленческих связей сети. В Инет-социальной сети это проявляется, например, в невозможности зарегистрироваться или войти в систему.

III. В правовой страте уязвимость Инет-социальной сети – это возможность нанесения ей повреждений, приводящих к отсутствию правовых норм поведения. Это может произойти, например, при крушении государства Участников Инет-социальной сети, когда ни один закон не будет иметь силу.

IV. В экономической страте уязвимость Инет-социальной сети – это возможность нанесения повреждений, приводящих к прекращению обмена стоимо-

стью. Такая ситуация будет характеризоваться отсутствием в торговой, рекламной деятельности и деятельности по предоставлению платных услуг.

V. В социальной страте уязвимость Инет-социальной сети – это возможность нанесения повреждений, приводящих к нарушениям социальной структуры. Например, это возможно при прекращении дружеских отношений между участниками Инет-социальной сети, при утере ими таких социальных признаков как уровень образования или дохода.

VI. В культурологической страте уязвимость Инет-социальной сети – это возможность нанесения повреждений, приводящих к прекращению деятельности Участников в сфере культурологической коммуникации, в том числе ликвидация групп по интересам и т.п.

Scenic vulnerability studies complex organizational and technical systems

N.O Ponomarev, D.A Kononov, D.A Shvetsov, P.O Ponomarev

Abstract: The possibility of the scenario approach for the study of problems of vulnerability and security management of complex organizational and technical systems. A scheme for modeling vulnerability to the spread of disturbances caused by various types of threats in the organizational and technical system on the operator graph, as well as the classification of vulnerabilities .. The principles of vulnerability research in the Internet and social networks.

Keywords: vulnerability, technical system

Литература

1. Д.А.Кононов, А.А.Кочкаров, Н.О.Пономарев. Стойкость сложных технических систем: сценарный взгляд на проблему // Труды НИИСИ РАН. – М.: 2012, с. 80-86.
2. В.В.Кульба, Д.А.Кононов, С.А.Косяченко, А.А.Кочкаров, Д.С.Сомов. Использование сценарного и индикаторного подходов для управления живучестью, стойкостью и безопасностью сложных технических систем // Научное издание. – М.: ИПУ РАН, 2011.
3. В.В.Кульба, Д.А.Кононов, С.А.Косяченко, А.Н.Шубин. Методы формирования сценариев развития социально-экономических систем /Серия «Системы и проблемы управления». – М.: СИНТЕГ, 2004, 296 с.
4. Модели и методы анализа и синтеза сценариев развития социально-экономических систем. Кн. 1 /Под редакцией чл.-корр. РАН В.Л.Шульца, д.т.н., проф. В.В.Кульбы В.В. – М.: Наука, 2012.
5. Д.А.Кононов. Модели и методы анализа и синтеза сценариев развития социально-экономических систем. Диссертация на соискание ученой степени доктора технических наук – М.: ИПУ РАН, 2010, 432 с.
6. В.В.Кульба, Д.А.Кононов, С.А.Косяченко, А.А.Кочкаров, Д.С.Сомов. Использование сценарного и индикаторного подходов для управления живучестью, стойкостью и безопасностью сложных технических систем /Научное издание. – М.: ИПУ РАН, 2011. – 116с.
7. В.В.Кульба, В.Л.Шульц, А.Б.Шелков, И.В.Чернов, Д.А.Кононов, Д.С.Сомов. Управление безопасностью и живучестью объектов инфраструктуры железнонорожного транспорта на основе индикаторного подхода //Экономика, тренды и управление. 2013. № 2, с. 1-107.
8. Д.А.Новиков. Сетевые структуры и организационные системы. – М.: ИПУ РАН, 2003. – 102 с.
9. Д.А.Кононов, В.В.Муромцев, Д.А.Швецов. Информационные технологии формирования «мягкой силы» // Проблемы управления безопасностью сложных систем. XXI Международная конференция. Материалы конференции. – М.: ИПУ РАН. 2013.
10. В.В.Кульба, Д.А.Кононов, И.В.Чернов, П.Е.Рощин, О.А.Шулигина. Сценарное исследование сложных систем: анализ методов группового управления // Управление большими системами. Специальный выпуск 30.1 «Сетевые модели в управлении» – М.: ИПУ РАН, 2010. с.154-186.
11. С.А.Власов, Д.А.Кононов, В.В.Кульба. Сценарный анализ группового управления // Информационные технологии и математическое моделирование систем 2009-2010. Труды международной научно-технической конференции. – М.: Учреждение Российской академии наук Центр информационных технологий в проектировании РАН, 2010. с. 246-268.
12. М.Лоэв. Теория вероятностей. – М.: ИЛ. 1962.

Применение теневых карт для моделирования теней в виртуальных 3D сценах в реальном времени

А.В. Мальцев

кандидат физико-математических наук

Аннотация: В статье предлагаются методы и алгоритмы моделирования теней в трехмерных виртуальных сценах, содержащих направленные и всенаправленные источники света, с использованием различных типов теневых карт: планарных, параболических и каскадных. Детально рассматривается процесс создания и применения таких карт в масштабе реального времени. Для реализации алгоритмов используются современные средства визуализации, включая шейдеры и механизм FBO (framebuffer objects) для прямой визуализации в текстуру.

Ключевые слова: моделирование теней, масштаб реального времени, виртуальные сцены, системы визуализации.

Введение

Среди множества графических эффектов, которые позволяют сделать визуализируемые трехмерные виртуальные сцены более реалистичными, тени являются одними из наиболее существенных. Объекты, освещаемые лучами солнечного света, естественными или искусственными источниками, отбрасывают тени на окружающие их предметы и поверхности. Это неотъемлемая часть нашего мира. Поэтому для правильного восприятия визуализируемой виртуальной среды очень важно наличие в ней теней. Существует несколько технологий для моделирования теней в трехмерных сценах, например, теневые объемы и теневые карты. Мы в данной работе остановимся на подходе, основанном на применении теневых карт, поскольку он независим от геометрии сцены и хорошо подходит для реализации теней в масштабе реального времени даже в высокополигональных сценах.

В статье будут рассмотрены методы и алгоритмы моделирования в реальном времени теней в трехмерных виртуальных сценах с использованием различных типов теневых карт: планарных, параболических и каскадных. Каждый из этих типов имеет свою область для эффективного применения. Так, планарные теневые карты хорошо подходят для виртуальных сцен, представляющих замкнутые (например, помещения внутри зданий) или ограниченные открытые (небольшой участок открытой местности) пространства с использованием направленных источников света. В случае обширных открытых местностей или ландшафтов и все тех же направленных источников освещения для качественного моделирования теней в режиме реального времени наиболее подходящими являются каскадные теневые карты. В свою очередь, параболический тип текстур теней эффективен для всенаправленных источников, излучающих свет по всем направлениям.

1. Планарные теневые карты.

Моделирование теней в трехмерных виртуальных сценах с направленными источниками света с помощью планарных теневых карт состоит из двух этапов. На первом этапе производится формирование буфера глубины виртуальной сцены из позиции источника света. Этот буфер сохраняется в виде текстуры специального формата в градациях серого, которую принято называть *планарной теневой картой*, или сокращенно – *теневой картой*. На втором этапе происходит визуализация объектов сцены из позиции наблюдателя. При этом для каждой видимой точки каждого объекта выполняется специальный тест глубины. А именно, если глубина рассматриваемой точки относительно источника света больше соответствующего значения глубины, взятого из теневой карты, то данная точка находится в тени; если же эти значения приблизительно равны, то точка считается освещенной источником света.

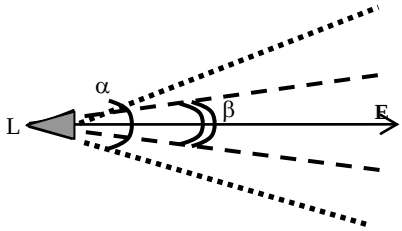


Рис. 1. Источник света «фара»

Здесь и далее при моделировании теней от направленных источников света мы ограничимся двумя видами таких источников: spotlight («фары») и target direct (прожекторы, освещающие параллельными лучами). Коротко остановимся на их параметрах. Действие spotlight задается с помощью углов α и β, где α – это угол раствора конуса, в пределах которого распространяется освещение, а β - так называемый угол «горячего пятна», определяющий конус, внутри которого освещение от источника максимально (рис. 1). При этом должно соблюдаться условие $0 < \beta \le \alpha < 180°$. Вне конуса с углом α освещение отсутствует, а между конусами оно равномерно изменяется от 0 до 1. Аналогично для target direct определены радиусы R и r

цилиндров ($0 < r \le R$). Внутри цилиндра радиуса r освещение максимально (т.е. равно 1), вне цилиндра радиуса R освещение отсутствует (равно 0), а между цилиндрами – равномерно изменяется от 0 до 1 (рис. 2).

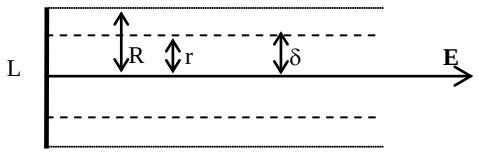


Рис. 2. Источник света «прожектор»

При рассмотрении источников описанного вида, в отличие от полусферических и всенаправленных (речь о них пойдет в п.2), для моделирования теней достаточно построения одного стандартного буфера глубины (теневой карты), плоскость которого перпендикулярна вектору **E** направления освещения.

Для реализации первого этапа моделирования, генерации теневой карты, необходимо создать фиктивную камеру C_L в точке расположения источника света и направить ось -Z (вектор взгляда камеры) ее видовой системы LVCS так же, как вектор **E**. После этого с помощью стандартных функций OpenGL задается вид проецирования: перспективное (функция gluPerspective) с углом раствора камеры, равным параметру α, и величиной aspect = 1 для источника типа spotlight или ортографическое (функция glOrtho) с отсекающими плоскостями left и bottom, равными –R, a right и top, равными +R, для target direct. Далее отключается запись в буфер цвета, включается буфер глубины и осуществляется визуализация сцены. Необходимо отметить, что в дальнейшем, на этапе применения теневой карты, нам потребуются матрицы видового (M_{light}) и проекционного (M_{proj}) преобразований, использованные для фиктивной камеры C_L, поэтому их нужно сохранить.

В результате описанных действий текущий буфер глубины будет содержать требуемую нам информацию, которую надо одним из доступных способов (копирование, присоединение) перенести в текстуру, имеющую формат GL_DEPTH_COMPONENT (данный формат доступен через расширение OpenGL GL_ARB_depth_texture) и являющуюся собственно теневой картой.

Для применения сгенерированной теневой карты необходимо во фрагментном шейдере реализовать специальный тест глубины пикселей относительно источника света. Рассмотрим его суть на простом примере. Пусть имеется точка P, принадлежащая поверхности некоторого объекта виртуальной сцены, и пусть известны ее координаты в объектной системе координат (СК) OCS. Для проведения теста глубины с использованием теневой карты необходимо найти координаты этой точки P в СК нормализованного объема видимости (LNDCS) для фиктивной камеры C_L. Формула преобразования координат точки P имеет вид:

$$P_{LNDCS} = M_{proj} \cdot M_{light} \cdot M_{model} \cdot P_{OCS}, \qquad (1)$$

где M_{model} – матрица модельного преобразования, переводящая координаты фрагмента из объектной системы координат (OCS) в мировую (WCS), M_{light} - матрица видового преобразования, переводящая координаты фрагмента из мировой СК в видовую систему (LVCS) фиктивной камеры C_L, «привязанной» к источнику, M_{proj} – матрица, задающая проекционное преобразование для камеры C_L. Матрицы M_{light} и M_{proj} были вычислены и сохранены нами на этапе построения теневой карты. Необходимо отметить, что после умножения на матрицу M_{proj}, мы получаем для P_{LNDCS} однородные координаты (x, y, z, w), где $w \neq 1$. Поэтому следует поделить все координаты P_{LNDCS} на w.

Поскольку координаты точек в системе LNDCS по всем трем осям находятся в отрезке [-1, 1], а для проведения теста затененности фрагмента с использованием теневых карт нужны текстурные координаты, соответствующие отрезку [0, 1], выполним преобразование:

$$P_{tex} = M_{bias} \cdot P_{LNDCS}, \text{ где } M_{bias} = \begin{pmatrix} 0.5 & 0 & 0 & 0.5 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \qquad (2)$$

Пусть $P_{tex} = (x_p, y_p, z_p, 1)$. Заметим, что z_p соответствует расстоянию (глубине) от точки P до источника освещения. Поскольку теневая карта представляет собой буфер глубины виртуальной сцены из позиции источника света, то в каждом ее пикселе записано расстояние от источника освещения до ближайшей точки какого-либо объекта по направлению, соответствующему данному пикселу. Для определения освещенности точки из подготовленной теневой карты по паре координат (x_p, y_p) выбирается записанное в ней значение глубины z_{map} и сравнивается со значением z_p.

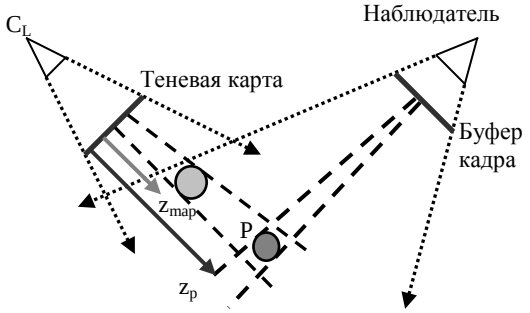


Рис. 3. Объект в тени

При этом возможно два варианта:

- $z_{map} < z_p$, что соответствует ситуации, когда точка

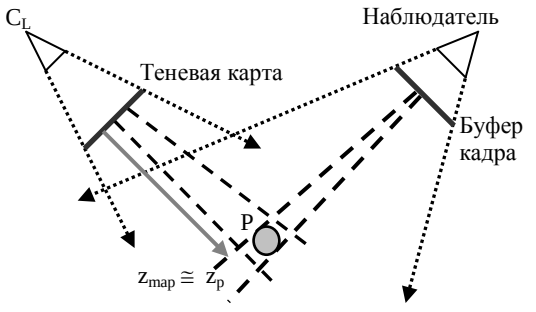


Рис. 4. Освещенный объект

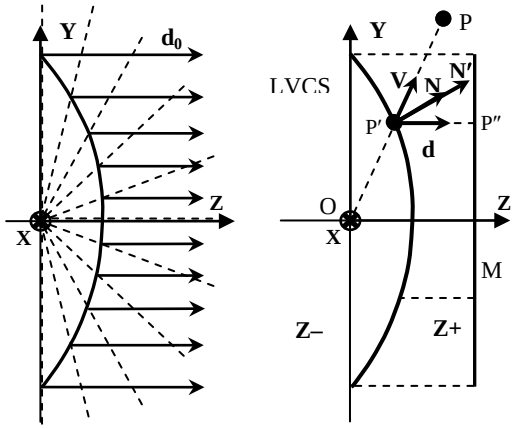
P закрыта от попадания в нее прямых лучей из источника участком поверхности какого-то объекта в сцене (рис. 3). Другими словами, точка P находится в тени;

- $z_{map} \cong z_p$ (z_p и z_{map} приблизительно эквивалентны), то есть в теневой карте записана с некоторой погрешностью глубина самой точки P (рис. 4), а следовательно P освещается источником.

2. Параболические теневые карты.

В реальности, кроме направленных источников света, существуют и преобладают такие, которые испускают в окружающую среду лучи по всем направлениям. Это так называемые *всенаправленные* источники (omnidirectional light sources). Также можно выделить источники, именуемые *полусферическими* (hemispherical light sources), свет от которых распространяется не во все пространство, а в полупространство. Так как в данных случаях нет определенного направления освещения, то реализовать эффект теней, используя одну теневую карту в прежнем ее понимании, невозможно.

Чтобы решить эту проблему, мы предлагаем применять параболическое отображение, при котором на плоскость двумерной текстуры можно отобразить целое полупространство. Такая текстура называется *параболической картой* окружающей среды. При использовании данного подхода для полусферических источников света будет достаточно всего лишь одной параболической теневой карты, а для всенаправленных – двух, в совокупности составляющих так называемую *двойную параболическую карту*.



а) б)

Рис. 5. Параболическое отображение

Далее рассмотрим принципы и методы генерации параболических теневых карт, а также реализацию с их помощью теней для всенаправленных и полусферических источников света в режиме реального времени.

2.1. Параболические карты окружающей среды

Параболическую карту окружающей среды можно получить путем проецирования изображения с поверхности идеально отражающего параболоида, задаваемого функцией

$$f(x,y) = \frac{1}{2} - \frac{1}{2}(x^2 + y^2), \text{ где } x^2 + y^2 \le 1, \qquad (3)$$

лучами, параллельными оси Z, на плоскость квадратной текстурной карты M || XY с длиной стороны, равной двум, и центром на оси Z (рис. 5б). Текстура M называется *параболической картой*. Параболоид (3) выбран таким образом, что его фокус находится в точке O = (0,0,0), то есть в начале системы координат. Поэтому, исходя из свойств параболоида, все падающие на него из полупространства Z+ лучи, сходящиеся в фокальной точке, отражаются в одном и том же направлении $\mathbf{d_0}$ = (0,0,1) (рис. 5а). Следовательно, для любой точки P ∈ Z+ можно найти ее отображение P″ на карте M, испустив луч PO. Это означает, что одна параболическая карта M полностью отображает полупространство Z+. Для отображения Z– можно воспользоваться параболоидом $-f(x,y)$ с направляющим вектором отражения $\mathbf{d_1}$ = (0,0,-1) или инвертировать пространство относительно плоскости XY и применить параболоид (3). Совокупность параболических карт для Z+ и Z– называется *двойной параболической картой* окружающей среды.

Пусть P′ – точка пересечения поверхности параболоида (3) с лучом PO, выходящим из произвольной точки P ∈ Z+ (рис. 5б). Наша основная задача заключается в нахождении для точки P ее отображения в текстурной карте M, т.е. координат P''_x, P''_y проекции точки P′ на плоскость M. Так как проецирование осуществляется лучами, параллельными оси Z, то координаты P''_x, P''_y точки P″ на текстуре M будут совпадать с координатами P'_x, P'_y точки P′.

Определим два касательных вектора $\mathbf{T_x}$, $\mathbf{T_y}$ к поверхности параболоида (3) в точке P′ = (x, y, f(x,y)):

$$\boldsymbol{T}_x = \frac{\partial P'}{\partial x} = \left(1, 0, \frac{\partial f(x,y)}{\partial x}\right) = (1, 0, -x),$$

$$\boldsymbol{T}_y = \frac{\partial P'}{\partial y} = \left(0, 1, \frac{\partial f(x,y)}{\partial y}\right) = (0, 1, -y).$$

Вычислив векторное произведение $\mathbf{T_x}$ и $\mathbf{T_y}$, получим нормаль к поверхности параболоида (3) в точке P′:

$$\boldsymbol{N} = [\boldsymbol{T}_x \times \boldsymbol{T}_y] = (x, y, 1) = (P'_x, P'_y, 1). \qquad (4)$$

С другой стороны, сумма единичного вектора $\mathbf{V}$ направления из начала системы координат в точку P и вектора отражения $\mathbf{d_0}$ даст вектор $\mathbf{N}'$ (рис. 5б), сонаправленный с нормалью $\mathbf{N}$, но отличающийся от нее длиной:

$$\boldsymbol{N}' = \boldsymbol{V} + \boldsymbol{d}_0 = (V_x, V_y, V_z + 1),$$

где $\boldsymbol{V} = \frac{1}{\sqrt{P_x^2 + P_y^2 + P_z^2}}(P_x, P_y, P_z)$.

Следовательно, нормаль $\mathbf{N}$ в точке P′ можно получить, поделив вектор $\mathbf{N}'$ на его z-координату:

$$\boldsymbol{N} = \frac{\boldsymbol{N}'}{N'_z} = \left(\frac{V_x}{V_z + 1}, \frac{V_y}{V_z + 1}, 1\right). \qquad (5)$$

Сравнивая формулы (4) и (5), получим

$$P''_x = P'_x = \frac{V_x}{V_z + 1} = \frac{P_x}{P_z + \sqrt{P_x^2 + P_y^2 + P_z^2}},$$
$$P''_y = P'_y = \frac{V_y}{V_z + 1} = \frac{P_y}{P_z + \sqrt{P_x^2 + P_y^2 + P_z^2}}. \quad (6)$$

Отметим, что точка P′ на параболоиде (как и точка P″ на текстурной карте M) соответствует любой точке полупространства Z+, лежащей на луче OP′.

2.2. Генерация параболических карт теней.

Теперь рассмотрим процесс генерации двойной параболической карты для всенаправленного источника света. Пусть LVCS – его система координат. В общем случае в качестве LVCS можно принять любую правостороннюю ортонормированную СК с центром в точке расположения источника. Однако для упрощения вычислений удобнее выбрать LVCS так, чтобы оси координат были сонаправлены с осями мировой системы WCS. Тогда матрица перехода из WCS в LVCS будет иметь вид

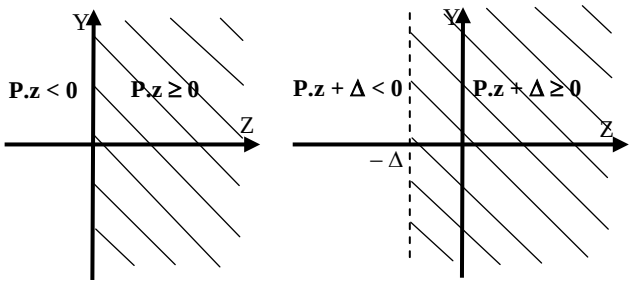
$$M_{light} = \begin{pmatrix} 1 & 0 & 0 & -L_x \\ 0 & 1 & 0 & -L_y \\ 0 & 0 & 1 & -L_z \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

где L_x, L_y, L_z – координаты источника света в СК WCS. Плоскость XY системы LVCS делит все пространство, окружающее источник света, на два полупространства: Z+ и Z– (рис. 5б). Для каждого из полупространств необходимо построить свою параболическую теневую карту. Рассмотрим как это можно сделать с помощью шейдеров на примере полупространства Z+ .

Пусть P – некая точка (вершина объекта) с координатами, представленными в локальной СК OCS объекта, которому принадлежит эта точка. Найдем ее координаты в системе LVCS:

$$P_{LVCS} = M_{light} \cdot M_{model} \cdot P_{OCS}, \quad (7)$$

где M_{light} – матрица перехода из мировой системы WCS в LVCS, описанная выше, и M_{model} – модельная



а) б)

Рис. 6. Границы отбраковки точек из Z+

матрица, осуществляющая преобразование из локальной СК OCS в мировую WCS.

Для устранения ошибочных затенений при создании параболической карты необходимо отсечь от обработки точки, не принадлежащие рассматриваемому полупространству Z+. Тогда условием отбраковки точки P будет $P_{LVCS,z} < 0$ (рис. 6а).

При практической реализации данного метода из-за погрешностей вычислений производится неточная отбраковка точек вблизи плоскости XY. Для устранения этого недостатка сдвинем границу отбраковки на некоторую малую величину $\Delta > 0$ (рис. 6б) т.е. запишем условие отбраковки точки в виде $P_{LVCS,z} + \Delta < 0$.

Если было установлено, что точка P принадлежит рассматриваемому полупространству, необходимо найти по формуле (6) ее образ $P'' = (P''_x, P''_y)$ на плоскости параболической карты. Также требуется вычислить длину отрезка OP, то есть расстояние $R = \sqrt{P_x^2 + P_y^2 + P_z^2}$ от точки P до начала координат O системы LVCS, в котором находится фокус параболоида (3). Фактически, R определяет глубину точки относительно источника по направлению OP (рис. 5б). Это значение должно быть сохранено в параболической теневой карте, но прежде его надо преобразовать к интервалу [0,1]. Введем для источника максимальное z_{far} и минимальное z_{near} расстояния отбрасывания тени, при этом $z_{far} > z_{near} > 0$, а также некоторую малую поправку глубины z_{bias}, устраняющую артефакты при дальнейшем использовании генерируемой карты. Тогда

$$R' = \frac{R - z_{near}}{z_{far} - z_{near}} + z_{bias} \quad (8)$$

Найденную для точки P тройку значений (P''_x, P''_y, R′) подадим на выход вершинного шейдера, а во фрагментном шейдере для каждого визуализируемого фрагмента рассчитаем значение R_f по формуле (8) и установим его в качестве глубины. Как уже было сказано ранее, все точки полупространства, лежащие на луче OP (рис. 5б) будут ассоциироваться с одной и той же точкой на параболоиде, а значит с одним и тем же текселом в теневой карте. Поэтому с помощью стандартного теста глубины при визуализации карты в этот тексел будет записано наименьшее из всех существующих значений R_f, то есть расстояние до точки, находящейся ближе всего к источнику освещения по направлению OP.

Итак, мы рассмотрели случай генерации параболической теневой карты для полупространства Z+. Карту для полупространства Z– можно построить практически аналогично. Для этого нужно умножить на –1 координату z точки P после проведения теста отбраковки. Такое умножение на коэффициент –1 переносит все точки полупространства Z– в полупространство Z+ и, наоборот, все точки из Z+ в Z–, то есть инвертирует пространство относительно плоскости XY системы координат LVCS, что сводит построение параболической теневой карты для полупространства Z– к уже рассмотренному выше алгоритму.

Отметим, что построение теневой карты для полусферического источника света является частным случаем рассмотренной выше схемы, так как требуется генерация параболической текстуры только для одного из полупространств Z+ или Z–, в зависимости от системы координат LVCS. Однако выбор системы

LVCS усложняется тем фактом, что освещаемое источником полупространство должно полностью совпасть с одним из полупространств Z+ или Z–. Этого, скорее всего, не произойдет при выборе в качестве LVCS системы координат, оси которой сонаправлены с осями мировой СК WCS. В данном случае нужно выбрать систему LVCS так, чтобы ее центр находился в точке размещения источника света, плоскость XY была параллельна основанию источника, а ось Z была направлена в освещаемое полупространство. Матрица M_{light} будет иметь более сложный вид, чем рассмотренная выше, но формула (7) сохранится.

Для генерации теневой карты и визуализации сцены в режиме реального времени, целесообразно применение так называемого прямого рендеринга в текстуру с использованием современных программно-аппаратных средств, а именно технологии FBO (framebuffer objects). FBO – это расширение OpenGL, которое обеспечивает простой интерфейс для отрисовки в контексты, отличные от буферов GL, предоставленных оконной системой. Более подробно про использование технологии FBO можно прочитать в [1] и [2].

2.3. Реализация теней от полусферических и всенаправленных источников света.

Для применения сгенерированной теневой карты при моделировании теней от всенаправленного (или полусферического) источника света необходимо во фрагментном шейдере реализовать тест глубины пикселей относительно источника света. По своей сути он аналогичен тесту, рассмотренному в п.1, однако имеет особенности в части нахождения текстурных координат фрагмента, используемых для обращения к теневой карте. Рассмотрим эти особенности на примере точки P с координатами в объектной СК OCS, принадлежащей поверхности некоторого объекта виртуальной сцены, освещаемой всенаправленным источником.

Чтобы выполнить тест глубины фрагмента, соответствующего точке P, необходимо найти образ $P'' = (P''_x, P''_y)$ этой точки на плоскости параболической теневой карты и расстояние R между P и началом координат O системы LVCS источника света, то есть длину отрезка OP. Также требуется определить, какую из двух текстур, входящих в состав двойной параболической карты, использовать для рассматриваемой точки.

Вычислим координаты (x,y,z,w) точки P в системе LVCS источника света по формуле (7). Тогда

$$R = \sqrt{x^2 + y^2 + z^2}$$

Так как в теневой карте все записанные расстояния имеют значения от 0 до 1, необходимо преобразовать R к отрезку [0,1] по формуле (8), получая R′. При этом, если параметр z_{bias} уже был учтен в процессе построения теневой карты, то в данном случае целесообразно задать его равным 0.

Выбор одной из двух параболических карт (Z+ или Z–) осуществляется путем сравнения значения вычисленной выше координаты z точки P_{LVCS} с нулем. Если $z < 0$, то точка P_{LVCS} находится в полупространстве Z– и используется текстура Z–, и, наоборот, если $z \geq 0$, то задействована текстура Z+. Отметим, что в отличие от случая генерации параболической теневой карты, здесь требуется разделение полупространств Z+ и Z– именно плоскостью XY, то есть без учета каких-либо погрешностей.

Если было установлено, что точка $P_{LVCS} \in Z–$, то требуется умножить ее координату z на коэффициент -1, чтобы производить дальнейший расчет аналогично случаю $P_{LVCS} \in Z+$. Далее по формуле (6) находим координаты (P''_x, P''_y) отображения точки P_{LVCS}. Числа P''_x и P''_y фактически являются текстурными координатами для выбранной параболической теневой карты. Однако их значения лежат в отрезке [-1,1], поэтому надо преобразовать P''_x и P''_y к отрезку [0,1]: $s = 0.5 \cdot P''_x + 0.5$, $t = 0.5 \cdot P''_y + 0.5$. Для определения наличия или отсутствия тени в точке P, по паре координат (s, t) из теневой карты выбирается записанное в ней значение расстояния (глубины) R_{map} до точки, ближайшей к источнику освещения по лучу OP и сравнивается со значением R′ (полностью аналогично сравнению параметров z_{map} и z_p в п.1).

3. Каскадные теневые карты.

В случае довольно протяженной трехмерной виртуальной сцены (например, обширной местности или ландшафта), пространство которой требуется осветить направленным источником света, использование планарной теневой карты будет давать изображения с достаточно сильным эффектом «ступенчатости» тени, так называемым алиасингом. Это обусловлено рядом причин.

Во-первых, теневая карта имеет определенное разрешение в пикселах, следовательно, чем больший участок сцены она охватывает, тем меньше становится ее детализация. То есть мелкие, по сравнению с общим размером сцены, детали отображаются на такой текстуре в один или несколько текселов, что в свою очередь приводит к очень грубой границе тени от таких объектов или вовсе отсутствию тени. Увеличение детализации до необходимого уровня можно было бы обеспечить, существенно повысив разрешение текстуры для теневой карты. Но современные графические адаптеры ограничивают максимальное разрешение текстур. К тому же, время создания теневой карты с большим разрешением для высокополигональной виртуальной сцены не позволит проводить визуализацию такой сцены в реальном времени.

Во-вторых, при просмотре сцены из виртуальной камеры, имеющей перспективную проекцию, алиасинг тени резко возрастает от дальних объектов к ближним, если детализация для них в карте теней одинакова. Это происходит, поскольку объекты, находящиеся около ближней плоскости отсечения камеры занимают на изображении большее число пикселов, чем те, которые располагаются около дальней. Поэтому при одной и той же детализации теневой текстуры одному пикселу дальнего объекта будет соответствовать несколько текселов в этой текстуре, а нескольким пикселам ближнего объекта – один тексел текстуры.

Решить описанную проблему для направленных источников света, освещающих параллельными лучами, можно с помощью технологии *каскадных теневых карт*. Она основана на построении для источника нескольких планарных карт теней с различной детализацией (для близких от наблюдателя объектов – с большей, для дальних – с меньшей).

Реализация такого подхода состоит в разбиении пирамиды видимости виртуальной камеры (наблюдателя) плоскостями (параллельными ближней и дальней плоскостям отсечения) на некоторое число усеченных пирамид, для каждой из которых строится собственная планарная теневая карта (рис. 7). При этом получается, что, во-первых, мы генерируем

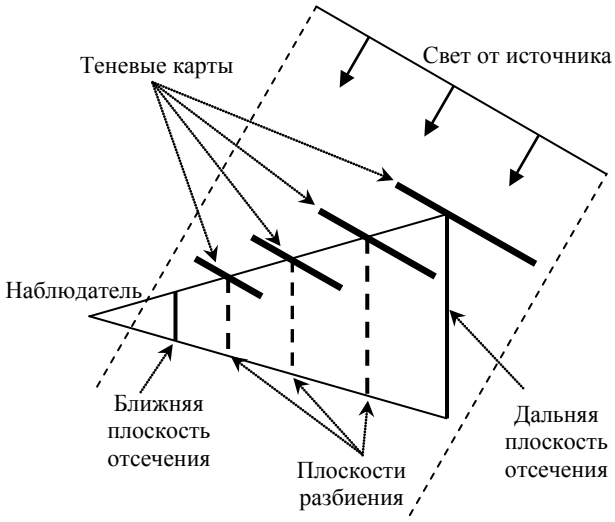


Рис. 7. Каскадные теневые карты

текстуру теней не для всей сцены, а только для той части, которую видит наблюдатель. Во-вторых, деление на подобласти позволяет обеспечить для каждой из них теневую карту с требуемой детализацией. Соответственно, даже для небольших объектов в ближнем поле зрения наблюдателя будут визуализироваться тени без артефактов.

3.1. Выделение подобластей в пирамиде видимости

Вначале рассмотрим задачу разбиения усеченной пирамиды видимости F_{cam} виртуальной камеры плоскостями, параллельными ближней и дальней плоскостям отсечения, на N частей. Тогда общее число разделяющих плоскостей, включая ближнюю и дальнюю плоскость отсечения, будет равно N+1. Существует несколько подходов к решению этой задачи: равномерное, логарифмическое и смешанное разбиения.

При равномерном разбиении F_{cam} разделяющие плоскости располагаются так, чтобы расстояния по оси Z видовой СК VCS между ними были одинаковыми. Тогда расстояние от наблюдателя до i-ой разделяющей плоскости будет равно

$$d_i = d_n + \frac{\left(d_f - d_n\right)}{N} i, \;\; i \in [0, N], \qquad (9)$$

где d_n, d_f – расстояния соответственно до ближней и дальней отсекающих плоскостей. Равномерное разбиение, как показано в [3], отличается сильным увеличением перспективного алиасинга тени на объектах, находящихся близко к наблюдателю.

Логарифмическое разбиение [3] основано на идее достижения константного значения ошибки ρ перспективного алиасинга для объектов, которые находятся в F_{cam} между ближней и дальней плоскостями отсечения. При этом $\rho = \ln\left(d_f / d_n\right)$. Тогда i-ая разделяющая плоскость будет находиться на расстоянии

$$d_i = d_n \left(\frac{d_f}{d_n}\right)^{i/N}, \; i \in [0, N]. \qquad (10)$$

При использовании логарифмического разбиения области между ближайшими к наблюдателю разделяющими плоскостями получаются слишком узкими и, следовательно, содержащими мало объектов. В результате этого эффективность работы алгоритма моделирования теней снижается.

Смешанное разбиение основано на сочетании двух предыдущих способов. Для нахождения d_i в этом случае возьмем линейную комбинацию правых частей равенств (9) и (10) с некоторым вещественным весовым коэффициентом $\lambda \in (0, 1)$:

$$d_i = \lambda d_n \left(\frac{d_f}{d_n}\right)^{i/N} + (1-\lambda)\left(d_n + (i/N)\left(d_f - d_n\right)\right), \quad i \in [0, N]. \qquad (11)$$

В большинстве случаев коэффициент λ можно задавать равным 0.5. Однако в некоторых виртуальных сценах для получения лучшего результата разбиения может потребоваться подбор λ опытным путем.

Задавая количество областей N, на которое будет разделена усеченная пирамида видимости F_{cam}, следует придерживаться приемлемого для графического приложения соотношения скорости визуализации и качества получаемых теней. Так, при $N < 3$ результат моделирования теней не будет значительно отличаться от использования одной планарной теневой карты. При слишком же больших значениях N существенно упадет скорость визуализации виртуальной сцены, поскольку в таком случае потребуется генерировать множество теневых карт, а, следовательно, и производить множество визуализаций сцены. Для получения качественных теней в режиме реального времени целесообразно устанавливать N в пределах 3-5.

Расстояния до всех плоскостей разбиения, посчитанные по одной из формул (9)-(11), следует запомнить в массиве D расстояний (так, что $D[i] = d_i$), поскольку они потребуются в дальнейшем, как на этапе генерации теневых карт, так и на этапе визуализации сцены с применением этих карт.

3.2. Создание текстур каскадных теневых карт

Найденные в пункте 3.1 разделяющие плоскости разбивают пирамиду видимости наблюдателя на N усеченных пирамид, для каждой из которых нам необходимо сгенерировать теневую карту. Рассмотрим процесс создания такой карты для одной из усеченных

пирамид F_i, образованной разделяющими плоскостями i и i+1. Предположим, что виртуальная сцена содержит один направленный источник света типа «прожектор».

Как и в предыдущих случаях генерации теневых карт для направленных источников света, разместим в точке расположения источника фиктивную виртуальную камеру C_L с видовой системой координат LVCS и направлением взгляда (ось -Z), совпадающим с направлением освещения. При визуализации трех-

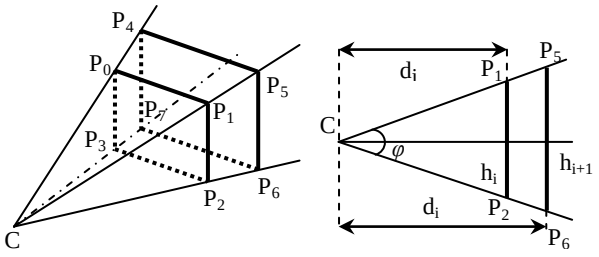


Рис. 8. Определение параметров усеченной пирамиды

мерной сцены с использованием этой камеры будем получать теневую карту в буфере глубины видеоадаптера. Поскольку рассматриваемый источник освещает параллельными лучами, выберем для созданной фиктивной камеры ортографическое проецирование. Далее необходимо правильно задать параметры этого проецирования.

Вначале вычислим координаты восьми вершин усеченной пирамиды F_i в видовой СК LVCS камеры C_L. Для этого найдем половины длин сторон прямоугольников $P_0P_1P_2P_3$ и $P_4P_5P_6P_7$, являющихся пересечениями плоскостей i и i+1 с пирамидой видимости наблюдателя (рис. 8). Используя известные значения угла φ вертикального раствора камеры и отношения

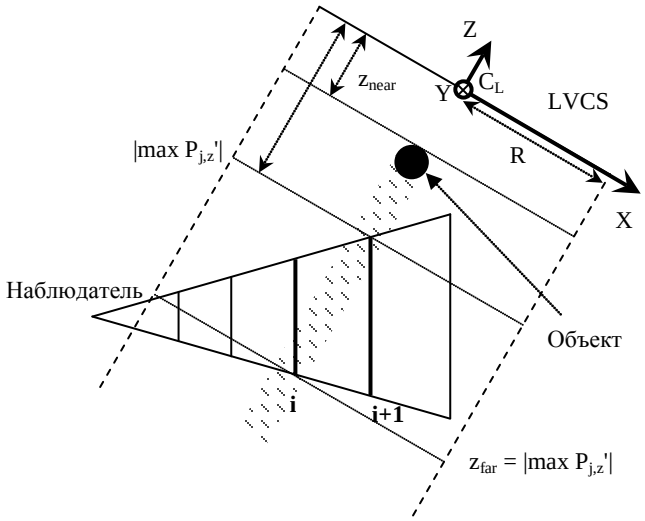


Рис. 9. Корректировка ближней плоскости отсечения

k_{aspect} ширины кадра к его высоте, а также расстояние d_i, найденное в п. 3.1, получим для $P_0P_1P_2P_3$:

$$h_i = \frac{P_1P_2}{2} = d_i \tan\frac{\varphi}{2}, \quad w_i = \frac{P_0P_1}{2} = h_i k_{aspect}.$$

Аналогично рассчитываются h_{i+1} и w_{i+1} для $P_4P_5P_6P_7$. Тогда искомые координаты вершин F_i в видовой системе VCS наблюдателя будут: $P_0(-w_i, h_i, -d_i, 1)$, $P_1(w_i, h_i, -d_i, 1)$, $P_2(w_i, -h_i, -d_i, 1)$, $P_3(-w_i, -h_i, -d_i, 1)$, $P_4(-w_{i+1}, h_{i+1}, -d_{i+1}, 1)$ и т.д. Минус в z-компоненте появляется потому, что взгляд камеры (наблюдателя) направлен в отрицательном направлении оси Z ее системы координат, а из равенств (9)-(11) было рассчитано абсолютное значение расстояния d_i. Перевод получившихся координат в систему LVCS осуществляется по формуле $P_j' = M_{light} \cdot M_{cam} \cdot P_j$, где M_{light} – матрица преобразования из мировой СК (WCS) в LVCS, M_{cam} – матрица преобразования из СК наблюдателя (VCS) в WCS.

Далее для задания ближней и дальней отсекающих плоскостей области видимости фиктивной камеры C_L, «привязанной» к источнику света, нужно выбрать из полученных восьми точек в СК LVCS максимальное и минимальное значения z-координат. При этом $|\max P_{j,z}'|$ будет соответствовать ближней отсекающей плоскости, а $|\min P_{j,z}'|$ – дальней (поскольку взгляд камеры направлен вдоль оси -Z ее СК). Однако необходимо учесть, что между полученной ближней отсекающей плоскостью и источником света, могут находиться объекты, невидимые наблюдателем (рис. 9). Информация о таких объектах должна попасть в создаваемую теневую карту, поскольку они будут отбрасывать тени в область, ограниченную усеченной пирамидой F_i. Следовательно, нужно расширить область видимости камеры C_L путем смещения ближней плоскости отсечения к источнику до такого значения z_{near}, пока все объекты между источником и усеченной пирамидой F_i не попадут в поле видимости фиктивной камеры.

Остальные плоскости отсечения (левая, правая, верхняя, нижняя) определяются исходя из радиуса R (рис. 9) цилиндрического светового потока от источника света (см. п.1). Зная положения отсекающих плоскостей, мы можем получить матрицу $M_{Lprj,i}$ проекционного преобразования для камеры C_L, например, с помощью функции glOrtho графической библиотеки OpenGL.

Далее необходимо выделить из спроецированной области участок, охватывающий рассматриваемую область F_i. Для этого умножим рассчитанные ранее в системе LVCS вершины P_j' ($j \in [0,7]$) усеченной пирамиды F_i на матрицу $M_{Lprj,i}$, получая точки P_j'' в СК нормализованного объема видимости. Найдем для получившихся точек $x_{min} = \min P_{j,x}''$, $x_{max} = \max P_{j,x}''$, $y_{min} = \min P_{j,y}''$, $y_{max} = \max P_{j,y}''$. Чтобы перейти от всего объема видимости, задаваемого матрицей $M_{Lprj,i}$, к выделяемому участку применим последовательно преобразования переноса и растяжения пространства, задаваемые соответственно матрицами

$$M_{T,i} = \begin{pmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ и } M_{S,i} = \begin{pmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

где $T_x = -0.5(x_{max} + x_{min})$, $T_y = -0.5(y_{max} + y_{min})$, $S_x = 2/(x_{max} - x_{min})$, $S_y = 2/(y_{max} - y_{min})$. Тогда матрица перехода будет иметь вид

$$M_{crop,i} = M_{S,i} M_{T,i} = \begin{pmatrix} S_x & 0 & 0 & S_x T_x \\ 0 & S_y & 0 & S_y T_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

В результате общая матрица проекционного преобразования для усеченной пирамиды F_i будет $M_{proj,i} = M_{crop,i} \cdot M_{Lprj,i}$. Установим ее в качестве текущей проекционной матрицы и визуализируем сцену из фиктивной камеры C_L с записью только буфера глубины и отключенным выводом в буфер кадра. Матрицу $M_{proj,i}$ следует сохранить в памяти, поскольку она потребуется при использовании подготовленной теневой карты.

Итак, мы получили в буфере глубины теневую карту для усеченной пирамиды F_i. Карты для остальных усеченных пирамид создаются аналогичным образом. Необходимо отметить, что в том случае, если виртуальная сцена содержит несколько направленных источников света, то каждый из них будет иметь свою собственную карту теней для каждой из N пирамид.

3.3. Применение каскадных теневых карт для моделирования теней

При визуализации виртуальной сцены из позиции наблюдателя во фрагментном шейдере необходимо определить для каждого рассматриваемого фрагмента, к какой из N усеченных пирамид видимости он относится. Для этого сравним расстояние d_{frag} от точки P, соответствующей фрагменту, до наблюдателя (в видовой системе координат VCS) с ранее записанными в массив D расстояниями от наблюдателя до разделяющих плоскостей. Значение d_{frag} определяется по формулам:

$$d_{frag} = \left| P_{VCS,z} \right|, \; P_{VCS} = M_{MV} \cdot P_{OCS},$$

где P_{VCS}, P_{OCS} – координаты точки P в видовой и объектной СК соответственно, M_{MV} – текущая модельно-видовая матрица. Если $D[i] \le d_{frag} < D[i+1]$, где $i \in [0, N]$, то фрагмент находится в i-ой усеченной пирамиде и для него необходимо использовать соответствующую ей теневую карту T_i. Для проведения теста глубины фрагмента относительно источника воспользуемся подходом, описанным в п.1. Найдем координаты точки P в СК LNDCS для камеры C_L, «привязанной» к источнику, по формуле (1), подставив в качестве M_{proj} матрицу $M_{proj,i}$, вычисленную на этапе генерации теневой карты T_i (п.3.2). Чтобы получить текстурные координаты $P_{tex} = (x_p, y_p, z_p, 1)$ для чтения и анализа данных из текстуры T_i, воспользуемся формулой (2). И, наконец, сравним глубину z_p рассматриваемого фрагмента со значением z_{map}, полученным из T_i по координатам (x_p, y_p). Если $z_{map} < z_p$, то точка P, а значит, и соответствующий ей фрагмент находятся в тени.

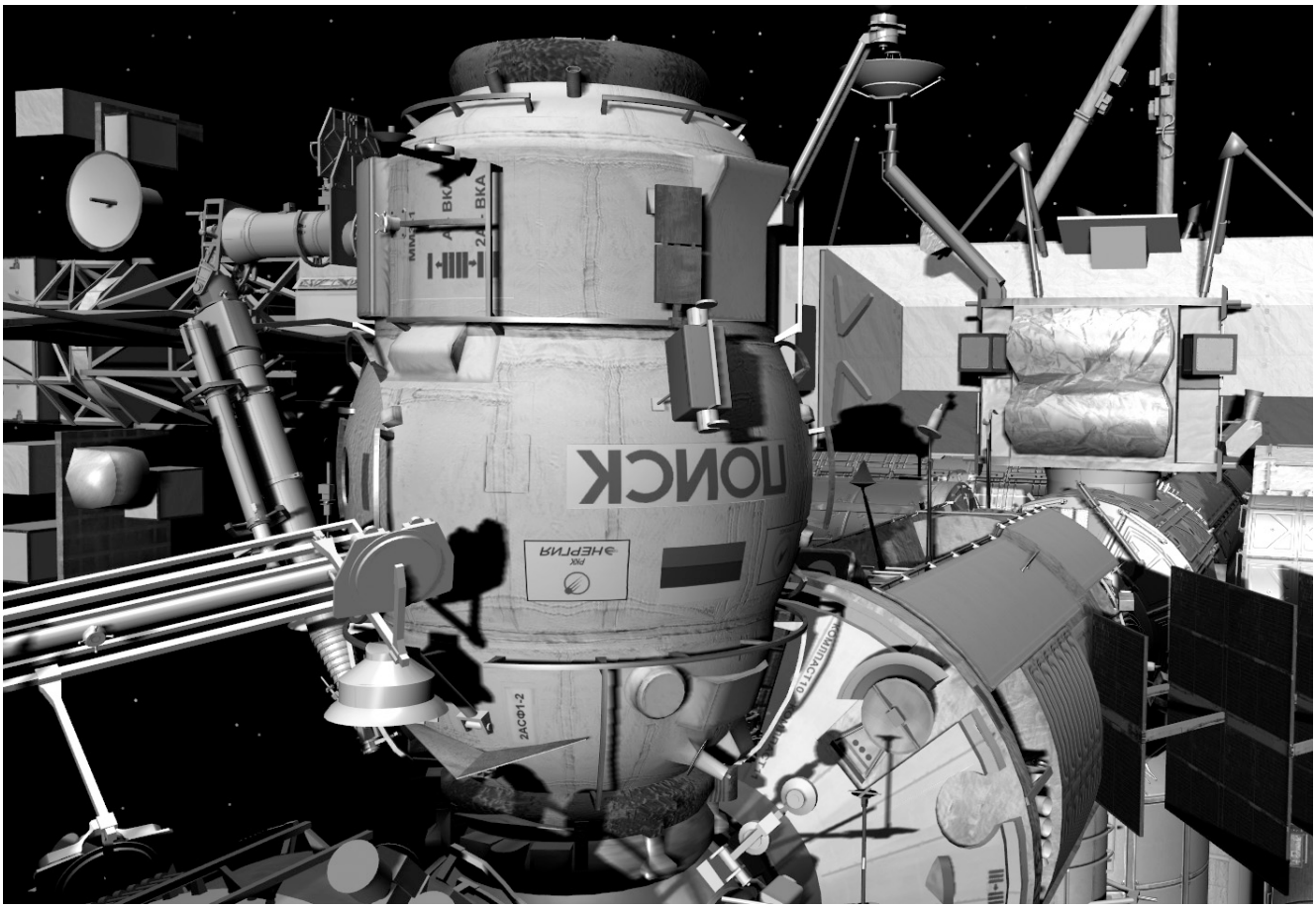


Рис. 10. Моделирование светотеневой обстановки на МКС

На рисунке 10 представлен результат моделирования светотеневой обстановки на внешней поверхности Международной космической станции, основанного на описанных выше методах и алгоритмах.

Данная работа выполняется при поддержке РФФИ, грант № 12-07-00256.

Using of shadow maps for real-time shadow simulation in 3D virtual scenes

A.V. Maltsev

Abstract: At the paper methods and algorithms are proposed for shadow simulation in three-dimensional virtual scenes containing directional and omni-directional light sources, by using different types of shadow maps (planar, parabolic and cascaded maps). The process of such map creation and application in real time mode are considered in detail. To implement algorithms modern visualization tools are used, including shaders and FBO (framebuffer objects) for direct visualization to a texture.

Keywords: shadow simulation, real time mode, virtual scenes, visualization systems

Литература

1. А.В. Мальцев, М.В. Михайлюк Реализация теней для направленных источников света в 3D сценах в реальном режиме времени // Интеллектуальные и адаптивные роботы. – 2009. – № 1-2. – с. 103-108.

2. http://www.opengl.org/wiki/Framebuffer_Object#Multisampling_Considerations (обращение: 27.03.2014)

3. Fan Zhang, Hanqiu Sun, Leilei Xu, Lee Kit Lun. Parallel-Split Shadow Maps for Large-Scale Virtual Environments // In Proceedings of ACM International Conference on Virtual Reality Continuum and Its Applications. – 2006. – p. 311-318.

Моделирование освещения Земли в реальном времени с учетом атмосферы

П. Ю. Тимохин

Аннотация: В статье описывается метод моделирования реалистичного освещения виртуальной модели Земли для систем виртуального окружения и видео тренажерных комплексов. Предложенный метод основан на физически правдоподобном моделировании прохождения света через атмосферу с помощью уравнения переноса излучения и позволяет синтезировать реалистичные изображения земной поверхности для произвольного взаимного расположения виртуального наблюдателя и источника освещения. В работе также описан подход, позволяющий вычислять освещенность Земли в режиме реального времени, основанный на использовании таблиц с предрассчитанными значениями сложно вычисляемых компонентов уравнения переноса.

Ключевые слова: визуализация, моделирование освещения Земли, режим реальное время.

Введение

Одной из актуальных задач современной компьютерной графики является моделирование освещения Земли для различного времени суток (день, рассвет/закат, сумерки, ночь) в режиме реального времени – с частотой синтеза кадров не менее 25 раз в секунду. Такая визуализация активно востребована при моделировании открытых трехмерных пространств во многих современных системах виртуального окружения и видео тренажерных комплексах наземного и аэрокосмического назначения [1].

Наилучшее визуальное сходство с реальным прототипом позволяет достигнуть физически корректное моделирование освещения на основе уравнения переноса излучения. До недавнего времени такой подход не позволял синтезировать изображения в реальном времени из-за его высокой вычислительной сложности, однако с развитием современных программных и аппаратных средств визуализации ситуация стала заметно меняться. Появилась возможность значительно ускорять вычисления за счет их распараллеливания между ядрами графического процессора (GPU), а также возможность быстрой обработки массивов произвольных вещественных данных, записанных в виде текстур, что дало мощный стимул для разработки новых, физически обоснованных, методов и алгоритмов реалистичной визуализации атмосферы [2], [3].

В данной работе рассматривается метод, позволяющий моделировать освещение Земли для произвольного времени суток на основе решения упрощенного уравнения переноса излучения, а также описывается подход, позволяющий вычислять освещенность Земли в режиме реального времени, основанный на использовании текстур с предварительно вычисленными таблицами значений наиболее затратных компонентов уравнения.

1. Постановка задачи

Пусть имеется трехмерная виртуальная сцена, содержащая модель земной поверхности, окруженной атмосферой, и источник освещения, имитирующий Солнце (рис. 1). В данной работе Земля моделируется сферой радиуса R_G, а атмосфера – внешней сферической оболочкой радиуса R_A. Источник освещения излучает в направлении $\vec{S}$ (лучи полагаются параллельными ввиду высокой удаленности Земли от Солнца) свет с интенсивностью I_S на границе входа в атмосферу. Вне земной поверхности, в точке с координатами P_V, располагается виртуальная камера, направленная вдоль вектора $\vec{V}$. Рассматриваемая задача состоит в вычислении изображения виртуальной сцены на картинной плоскости виртуальной камеры с учетом атмосферной оболочки.

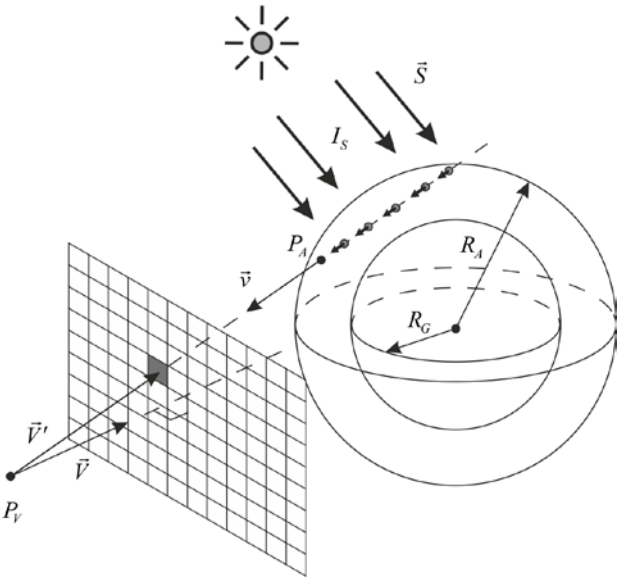


Рис. 1. Схема исходной виртуальной сцены

Рассмотрим случай, когда виртуальная камера расположена вне атмосферы (см. рис. 1). Проведем луч $\vec{V}'$ из точки P_V через некоторый пиксел картинной плоскости камеры. Из рисунка 1 видно, что когда луч $\vec{V}'$ пересекает атмосферную оболочку, цвет пиксела будет зависеть от атмосферы. Пусть P_A – точка первого пересечения луча $\vec{V}'$ с атмосферой (в случае, когда камера располагается внутри атмосферы, точка P_A совпадает с положением P_V камеры). Так как с оптической точки зрения атмосфера является средой, способной переносить излучение (participating medium),

т.е. менять в каждой своей точке его направление и спектр, то цвет рассматриваемого пиксела можно вычислить как результирующую интенсивность I_V всех лучей, прошедших через точку P_A в направлении $\vec{v} = -\vec{V}'$. В данной работе компоненты цвета пиксела моделируются тремя интенсивностями $I_V(\lambda)$ для длин волн, соответствующих красному (λ_R), зеленому (λ_G) и синему (λ_B) спектральному цвету видимого диапазона. Таким образом, чтобы вычислить необходимое изображение, требуется вычислить три интенсивности I_V для всех пикселов.

2. Вычисление интенсивности I_V

Рассмотрим вычисление одной компоненты интенсивности, соответствующей красному спектральному цвету (остальные будут вычисляться аналогично). Для простоты будем обозначать эту компоненту через I_V. Как отмечалось выше, интенсивность I_V определяется на основе пересечения луча $\vec{V}'$ с атмосферной оболочкой (см. рис. 1). Если такого пересечения нет, то I_V полагается нулевой. Ненулевой I_V будет в двух случаях: 1) когда луч $\vec{V}'$ пересекает только атмосферную оболочку планеты (случай "атмосферы над горизонтом"), и, 2) когда луч $\vec{V}'$ проходит через атмосферу и пересекает земную поверхность (случай "атмосфера над земной поверхностью").

Вначале рассмотрим случай прохождения луча $\vec{V}'$ через атмосферу над горизонтом. Обозначим P_B точку выхода луча $\vec{V}'$ из оболочки атмосферы (рис. 2).

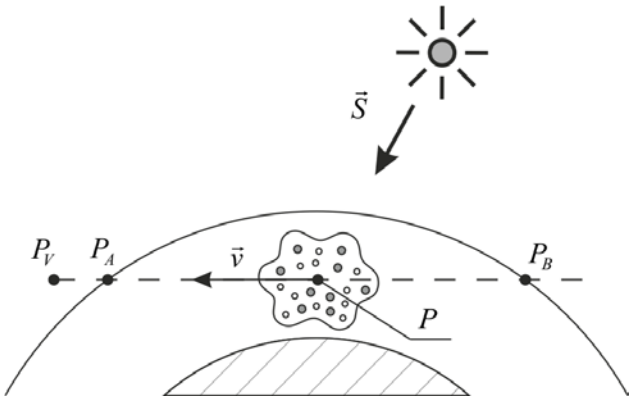


Рис. 2. Участок атмосферы над горизонтом

Пусть P – некоторая точка отрезка P_AP_B. Взаимодействие света с атмосферой в точке P определяется взаимодействием с частицами атмосферы в данной точке. В видимом диапазоне решающее влияние на солнечное излучение в атмосфере оказывают молекулы воздуха и аэрозольные частицы [4]. Ввиду того, что эти частицы крайне малы и расположены по отношению друг к другу случайным образом, взаимодействие света с атмосферой в точке P можно рассматривать как взаимодействие с элементарным объемом атмосферы (рис. 2), описывающим совокупное вероятностное взаимодействие с частицами. Каждый такой элементарный объем изменяет интенсивность I_V в точке P на некоторую величину I_v; тогда I_V можно определить как результат таких изменений вдоль отрезка P_AP_B. Рассмотрим, в результате каких процессов формируется I_v.

При прохождении через атмосферу интенсивность излучения может остаться без изменения (рис. 3, а), ослабиться в результате поглощения энергии луча аэрозольными частицами (рис. 3, б) и рассеяния наружу (out-scattering) молекулами воздуха и аэрозольными частицами (рис. 3, в) [4], а также усилиться за счет лучей, рассеянных другими элементарными объемами внутрь данного (in-scattering) (рис. 3, г).

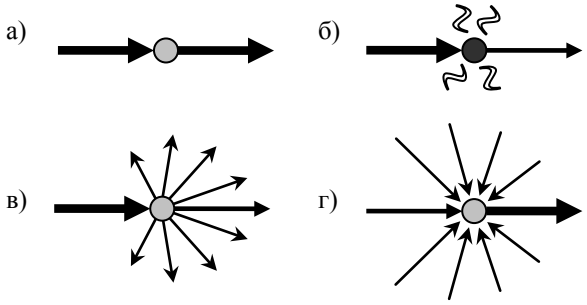


Рис. 3. Моделируемые типы взаимодействия луча с элементарным объемом атмосферы

С учетом описанных явлений величина I_v в каждый следующий момент времени определяется суммой приобретенной и потерянной интенсивности света и может быть представлена в следующем виде:

$$I_v = I_{v,ab} + I_{v,out} + I_{v,in} \qquad (1)$$

где $I_{v,ab}$ – часть интенсивности I_v, потерянная в результате поглощения, а $I_{v,out}$ – часть интенсивности I_v, потерянная в результате рассеяния в сторону, $I_{v,in}$ - часть интенсивности I_v, приобретенная за счет лучей, рассеянных внутрь объема. Рассмотрим более подробно эти составляющие.

2.1. Дифференциальное ослабление света

Дифференциальное ослабление света можно рассматривать как суммарную долю от интенсивности I_V, которая расходуется на рассеяние и поглощение в элементарном объеме, т.е. сумму первых двух слагаемых равенства (1). Для молекулярного и аэрозольного рассеяния такая доля определяется объемными коэффициентами σ_M и σ_A рассеяния, а для аэрозольного поглощения – объемным коэффициентом k_A поглощения (в данной работе принимается $k_A \approx 0.11\sigma_A$). Коэффициенты σ_M и σ_A вычисляются с помощью теории рассеяния Релея и теории Ми соответственно, и задаются для высоты на уровне моря в следующем виде:

$$\sigma_{M,0} = \frac{8\pi^3\left(n_e{}^2 - 1\right)^2}{3N_M\lambda^4}, \quad \sigma_{A,0} = \frac{8\pi^3\left(n_e{}^2 - 1\right)^2}{3N_A}, \qquad (2)$$

где n_e – показатель преломления света в атмосфере Земли на уровне моря, а N_M и N_A – количественные концентрации усредненных молекул и аэрозольных частиц в атмосфере Земли на уровне моря. В силу того, что плотность атмосферы неоднородна, коэффициенты σ_M и σ_A в произвольной точке атмосферы будут иметь значения, отличные от $\sigma_{M,0}$ и $\sigma_{A,0}$. С визуальной точки зрения наиболее заметно изменение плотности атмосферы в зависимости от высоты. Такую зависимость можно смоделировать с помощью коэффициентов ρ_M и ρ_A:

$$\rho_M(h) = e^{-h/H_M},\ \rho_A(h) = e^{-h/H_A},$$

где h - высота над поверхностью Земли, H_M, H_A – высота гипотетической однородной атмосферной оболочки, состоящей из частиц соответствующего типа ($H_M \approx 8$ км, $H_A \approx 1{,}2$ км). Тогда объемные коэффициенты σ_M и σ_A можно вычислить как

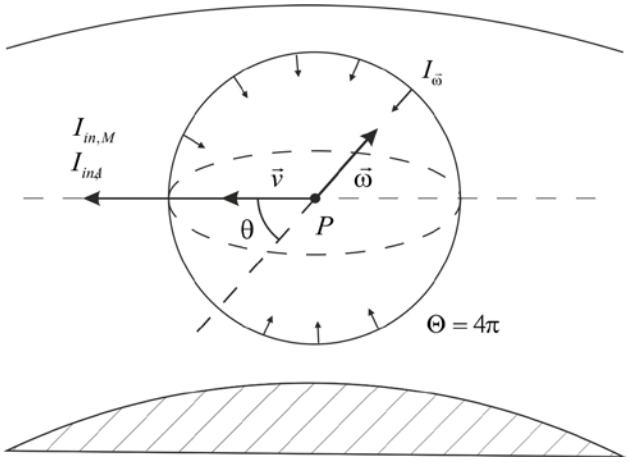
$$\sigma_M(h) = \sigma_{M,0} \cdot \rho_M(h),\quad \sigma_A(h) = \sigma_{A,0} \cdot \rho_A(h), \qquad (3)$$

(для сокращения записи далее h опускается), а общее дифференциальное ослабление света в точке P вычислить следующим образом:

$$I_{v,ab} + I_{v,out} = -(\sigma_M + \sigma_A + k_A) \cdot I_V. \qquad (4)$$

2.2. Дифференциальное усиление света

Как отмечалось выше, усиление света в точке P происходит за счет лучей, рассеянных внутрь элементарного объема. Такие лучи могут прийти в точку P со всех направлений, образуя сферу возможных направлений (рис. 4).



.

Рис. 4. Дифференциальное усиление света

При рассеянии света на частицах атмосферы возникает интерференция, в результате чего лучи рассеиваются в точке P анизотропно, т.е. с преобладанием одних направлений над другими, образуя профиль вероятностей рассеяния (индикатрису рассеяния). Для молекул воздуха и аэрозольных частиц такие индикатрисы рассеяния моделируются с помощью фазовых функций F_M и F_A. На практике в качестве таких фазовых функций используются аналитические приближения, которые хорошо описывают экспериментальные данные о реальной атмосфере. Для молекулярного рассеяния в качестве такого приближения используется фазовая функция Релея:

$$F_M = F_R = \frac{3}{4}\left(1 + \cos^2\theta\right), \qquad (5)$$

где θ - угол между направлением падения луча и направлением, в котором этот луч рассеивается. Как видно из рисунка 5, молекулы воздуха с большей вероятностью рассеивают свет в направлениях близких к прямому и обратному и с меньшей вероятностью – в поперечном направлении.

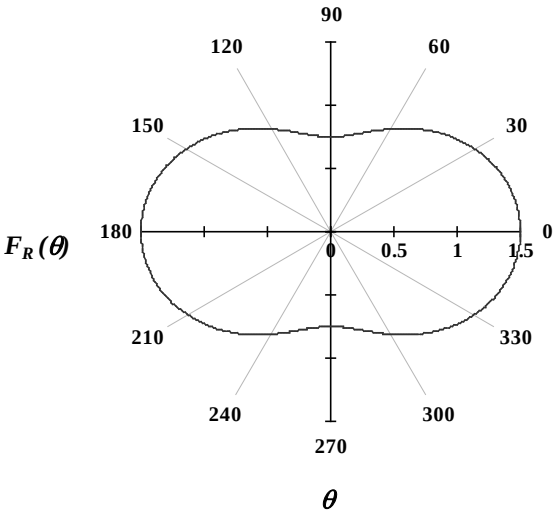


Рис. 5. Фазовая функция Релея

Для моделирования аэрозольного рассеяния Ми распространено использование фазовой функции F_{HGC} Хеньи-Гринстейна с улучшением Корнета-Шанкса [5]:

$$F_A = F_{HGC} = \frac{3\left(1-g^2\right)}{2\left(2+g^2\right)} \frac{\left(1+\cos^2\theta\right)}{\left(1+g^2-2g\cos\theta\right)^{3/2}}, \qquad (6)$$

где $g \in [-1,1]$ – коэффициент асимметрии, характеризующий степень асимметрии индикатрисы рассеяния. Величина g связана с "пасмурностью" атмосферы. В данной работе используется коэффициент $g = 0.76$ для стандартной модели безоблачной атмосферы. При таком коэффициенте индикатриса фазовой функции (6) имеет ярко выраженную направленность в сторону распространения излучения (рис. 6).

На практике, выполнение возведения в степень 3/2 в знаменателе функции (6) является вычислительно затратной операцией, поэтому в данной работе предлагается заменить функцию (6) на более простую в вычислении модифицированную фазовую функцию Шлика:

$$F_A = F_{HGC} \approx F_{MS} = \frac{3\left(1+\cos^2\theta\right)}{2\left(2+g^2\right)} \cdot F_S, \qquad (7)$$

где $F_S = \dfrac{1-k^2}{\left(1-k\cdot\cos\theta\right)^2}$ – фазовая функция Шлика [6], а $k \approx 1.55g - 0.55g^3$. Из рисунка 6 видно, что профиль

предложенной функции практически идентичен профилю функции (6).

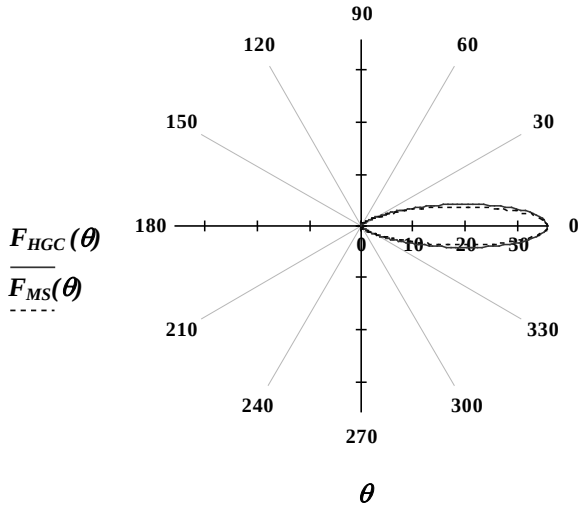


Рис. 6. Сравнение фазовой функции Хеньи-Гринстейна (F_{HGC}) и модифицированной функции Шлика (F_{MS})

Смоделировав таким образом фазовые функции F_M и F_A, вычислим интенсивности $I_{in,M}$ и $I_{in,A}$ луча, рассеянного молекулами воздуха и аэрозольными частицами в направлении $\vec{v}$. Пусть $\vec{\omega}$ - некоторое направление из точки P (см. рис. 4), а $I_{\vec{\omega}}$ - интенсивность луча, пришедшего в точку P по направлению $-\vec{\omega}$. Тогда $I_{in,M}$ и $I_{in,A}$ вычисляются как

$$I_{in,M} = \sigma_M \cdot F_M(\vec{\omega}) \cdot I_{\vec{\omega}},$$
$$I_{in,A} = \sigma_A \cdot F_A(\vec{\omega}) \cdot I_{\vec{\omega}},$$

а общее дифференциальное усиление света в точке P как

$$\begin{aligned} I_{v,in} &= \int_\Theta I_{in,M} d\vec{\omega} + \int_\Theta I_{in,A} d\vec{\omega} = \\ &= \sigma_M \int_\Theta F_M(\vec{\omega}) I_{\vec{\omega}} d\vec{\omega} + \sigma_A \int_\Theta F_A(\vec{\omega}) I_{\vec{\omega}} d\vec{\omega} \end{aligned}, \quad (8)$$

где $\Theta = 4\pi$ - телесный угол, соответствующий сфере возможных направлений падения лучей в точку P.

2.3. Уравнение переноса излучения

Подставив (4) и (8) в выражение (1) получим:

$$\begin{aligned} I_v &= \sigma_M \int_\Theta F_M(\vec{\omega}) I_{\vec{\omega}} d\vec{\omega} + \sigma_A \int_\Theta F_A(\vec{\omega}) I_{\vec{\omega}} d\vec{\omega} - \\ &- (\sigma_M + \sigma_A + k_A) \cdot I_V \end{aligned}, \quad (9)$$

где коэффициенты σ_M, σ_A вычисляются согласно (3), а фазовые функции F_M и F_A - согласно (5) и (7) соответственно. Полученное выражение является частным случаем интегро-дифференциального уравнения объемного рендеринга с неизвестным $I_{\vec{\omega}}$ (см. [7]), ввиду чего для (9) можно записать следующую интегральную форму, описывающую искомую интенсивность I_V:

$$\begin{aligned} I_V &= T(P_A, P_B) I_{V,0} + \\ &+ \int_{P_A}^{P_B} T(P_A, P) \left(\sigma_M \int_\Theta F_M(\vec{\omega}) I_{\vec{\omega}} d\vec{\omega} + \right. \\ &\left. + \sigma_A \int_\Theta F_A(\vec{\omega}) I_{\vec{\omega}} d\vec{\omega} \right) ds, \end{aligned} \quad (10)$$

где $I_{V,0}$ – интенсивность света, исходящего из точки P_B в направлении $\vec{v}$ (вычисление $I_{V,0}$ приводится ниже), $T(P_A, P) \in (0,1]$ - коэффициент пропускания (transmittance) атмосферы для отрезка от точки P_A до точки $P = P_A + s \cdot (-\vec{v})$, где $s \in [0, L]$, а L - длина отрезка $P_A P_B$. Коэффициент $T(P_A, P)$ пропускания описывает количество света, которое без потерь доходит по прямой от точки P_A до точки P и вычисляется как

$$T(P_A, P) = \exp(-\tau_{P_A P}), \quad (11)$$

где $\tau_{P_A P}$ - оптическая глубина отрезка $P_A P$:

$$\tau_{P_A P} = \int_{P_A}^{P} (\sigma_M + \sigma_{A,} + k_A) ds.$$

В данной работе для случая "атмосферы над горизонтом" $I_{V,0}$ вычисляется как

$$I_{V,0} = \begin{cases} I_S, & \text{если } \vec{v} = -\vec{S}, \\ 0, & \vec{v} \neq -\vec{S}. \end{cases}$$

Для случая "атмосфера над земной поверхностью" вычисление $I_{V,0}$ приводится ниже.

2.4. Вычисление света, отраженного от земной поверхности

Рассмотрим случай, когда луч $\vec{V}'$ пересекает земную поверхность (рис. 7). Обозначим через P_G точку пересечения луча $\vec{V}'$ с земной поверхностью. В рассматриваемом случае точка P_G также является обозначенной ранее точкой P_B выхода луча $\vec{V}'$ из оболочки атмосферы, ввиду чего в данном случае интенсивность $I_{V,0}$ из уравнения (10) является интенсивностью I_G света, отраженного от земной поверхности в точке P_G в направлении $\vec{v}$. Рассмотрим более подробно вычисление I_G.

В данной работе земная поверхность моделируется непрозрачной поверхностью, которая полностью отражает весь падающий свет одинаково во всех направлениях (поверхность Ламберта). Отражение света от такой поверхности можно смоделировать с помощью следующей двулучевой функции f_G отражательной способности (Bidirectional Reflectance Distribution Function, BRDF):

$$f_G = \frac{\rho_G}{\pi}, \quad (12)$$

где $\rho_G \in [0,1]$ - коэффициент диффузного отражения (альбедо) в точке P_G. Функция f_G связана с интенсивностью I_G следующим соотношением:

$$f_G = \frac{dI_G}{dE_G}, \quad (13)$$

где E_G - освещенность точки P_G. Из-за рассеяния света в атмосфере точка P_G освещается фактически из

множества направлений, образующих полусферу (см. рис. 7).

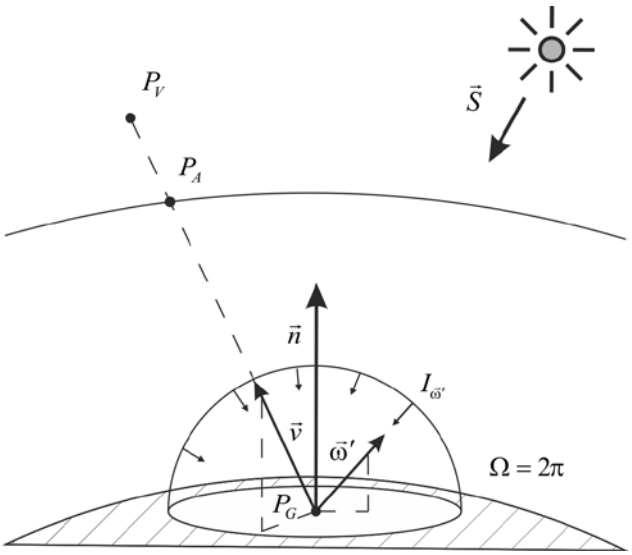


Рис. 7. Моделирование отражения света от земной поверхности

Обозначим через $\vec{\omega}'$ некоторое направление из точки P_G в такой полусфере, а $I_{\vec{\omega}'}$ - интенсивность луча, пришедшего в точку P_G по направлению $-\vec{\omega}'$. Тогда освещенность E_G можно вычислить как

$$E_G = \int_\Omega I_{\vec{\omega}'}(\vec{n}, \vec{\omega}')\, d\vec{\omega}', \qquad (14)$$

где $\vec{n}$ - нормаль к земной поверхности в точке P_G, а $\Omega = 2\pi$ - телесный угол, соответствующий полусфере. Подставив (12) и (14) в (13), получим соотношение, из которого I_G вычисляется как

$$I_G = \frac{\rho_G}{\pi} \int_\Omega I_{\vec{\omega}'}(\vec{n}, \vec{\omega}')\, d\vec{\omega}'. \qquad (15)$$

2.5. Решение уравнения переноса излучения

Из полученного в разделе 2.3 уравнения (10), описывающего искомую интенсивность I_V видно, что I_V зависит от интенсивности $I_{\vec{\omega}}$, которая по определению вычисляется аналогично I_V и так же зависит от интенсивностей лучей, рассеянных в атмосфере и т.д. Такое уравнение не имеет аналитического решения. Один из подходов к решению данной задачи состоит в решении уравнения численными методами, например, методом Монте-Карло. Такой подход обеспечивает вычисление освещенности атмосферы с высокой точностью, однако ценой огромных вычислительных затрат, что не позволяет выполнять визуализацию сцены в режиме реального времени. Другой подход, используемый в данной работе, состоит в упрощении уравнения (10), которое не вносит заметного визуального ущерба, но позволяет существенно ускорить вычисление, обеспечивая при этом весь спектр моделируемых физических явлений.

В рамках используемого подхода все рассеянные в атмосфере лучи рассматриваются с помощью порядков рассеяния. Лучи 0-ого порядка на всем пути следования не меняют своего направления (прямой свет, zero scattering), лучи 1-го порядка меняют направление один раз (однократное рассеяние, single scattering), лучи более высоких порядков (пример такого луча приведен на рисунке 7) претерпевают несколько актов рассеяния (многократное рассеяние, multiple scattering).

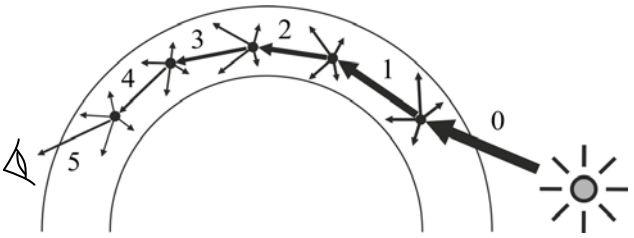


Рис. 8. Пример луча 5-го порядка рассеяния

Учет многократно рассеянных лучей очень важен при моделировании освещения атмосферы, т.к. позволяет наблюдать освещенность атмосферы в отсутствие прямой видимости Солнца (сумерки). Однако экспериментальные исследования показывают, что свет очень высоких порядков (выше 5-го-6-го) рассеяния визуально практически не заметен, поэтому в уравнении (10) им можно пренебречь, ограничив количество учитываемых порядков рассеяния. Обозначим интенсивности I_G, I_S и I_V, в которых учитываются лучи только k-го порядка рассеяния, как I_G^k, I_S^k и I_V^k, имеющие следующий вид:

$$I_G^k = \frac{\rho_G}{\pi} \int_\Omega I_V^{k-1}(\vec{n} \cdot \vec{\omega}')\, d\vec{\omega}',$$

$$I_S^k = \int_{P_A}^{P_B} T_{P_A P} \left(\sigma_M \int_\Theta F_M(\vec{\omega}) I_V^{k-1} d\vec{\omega} + \right.$$

$$\left. + \sigma_A \int_\Theta F_A(\vec{\omega}) I_V^{k-1} d\vec{\omega} \right) ds,$$

$$I_V^0 = T_{P_A P_B} \cdot I_{V,0},$$

$$I_V^k = I_G^k + I_S^k,$$

где для случая "атмосфера над земной поверхностью" в качестве P_B берется точка P_G (см. раздел 2.4). Тогда решение уравнения (10) для обоих случаев можно записать следующим образом:

$$I_V \approx I_V^K = I_V^0 + I_V^1 + \ldots + I_V^K = \sum_{k=0}^{K} I_V^k \qquad (16)$$

где K – количество учитываемых порядков рассеяния.

В полученном выражении вычисление компонент T, $I_G^* = \sum_{k=1}^{K} I_G^k$ и $I_S^* = \sum_{k=1}^{K} I_S^k$ обладает высокими вычислительными затратами. Существенно ускорить их вычисление можно, приняв следующие допущения. Из-за ослабления света в атмосфере интенсивность рассеянных лучей убывает с увеличением порядка рассеяния. В виду этого прямой и однократно рассеянный свет преобладает в визуальном восприятии освещенности атмосферы, а многократно рассеянный свет (обладающий существенно меньшей интенсивностью) воспринимается больше как фоновая компонента. Учитывая это, свойство симметричности сферы, а также принятое ранее допущение, что плотность атмосферы изменяется только в зависимости от высоты, компоненты T, I_G^*, I_S^* можно вычислить для выборочных значений P_V, $\vec{v}$, $\vec{S}$,

охватывающих весь рабочий диапазон этих параметров и записать в виде таблиц $\hat{T}$, $\hat{I}_G^*$ и $\hat{I}_S^*$. Тогда выражение (16) можно представить в следующем виде:
образом:

$$I_V \approx \hat{T}\left(P_V, \vec{v}, \vec{S}\right) \cdot I_{V,0} + I_G^0 + \\ + \hat{I}_G^*\left(P_V, \vec{v}, \vec{S}\right) + \hat{I}_S^*\left(P_V, \vec{v}, \vec{S}\right). \quad (17)$$

Вычисление (17) выполняется в два этапа. На этапе предварительной обработки вычисляются таблицы $\hat{T}$, $\hat{I}_G^*$, $\hat{I}_S^*$, а на этапе визуализации для каждого пиксела изображения Земли из данных таблиц выполняется выборка (с интерполяцией) значений, соответствующих текущим параметрам P_V, $\vec{v}$, $\vec{S}$ и вычисляется I_V согласно (17).

Выражение (17) позволяет вычислять I_V для каждого пиксела синтезируемого изображения независимо, что позволяет распараллелить такие вычисления между ядрами графического процессора и обеспечить моделирование освещения атмосферы в режиме реального времени. Для этого в данной работе таблицы $\hat{T}$, $\hat{I}_G^*$, $\hat{I}_S^*$ на этапе предобработки записываются в текстурные карты, которые на этапе визуализации обрабатываются с помощью разработанной фрагментной шейдерной программой, реализующей вычисление выражения (17) для всех пикселов изображения атмосферы. На рисунке 9 приведены результаты моделирования освещения виртуальной модели Земли для различных высот наблюдения. Работа была выполнена при финансовой поддержке РФФИ в рамках гранта №13-07-00674.

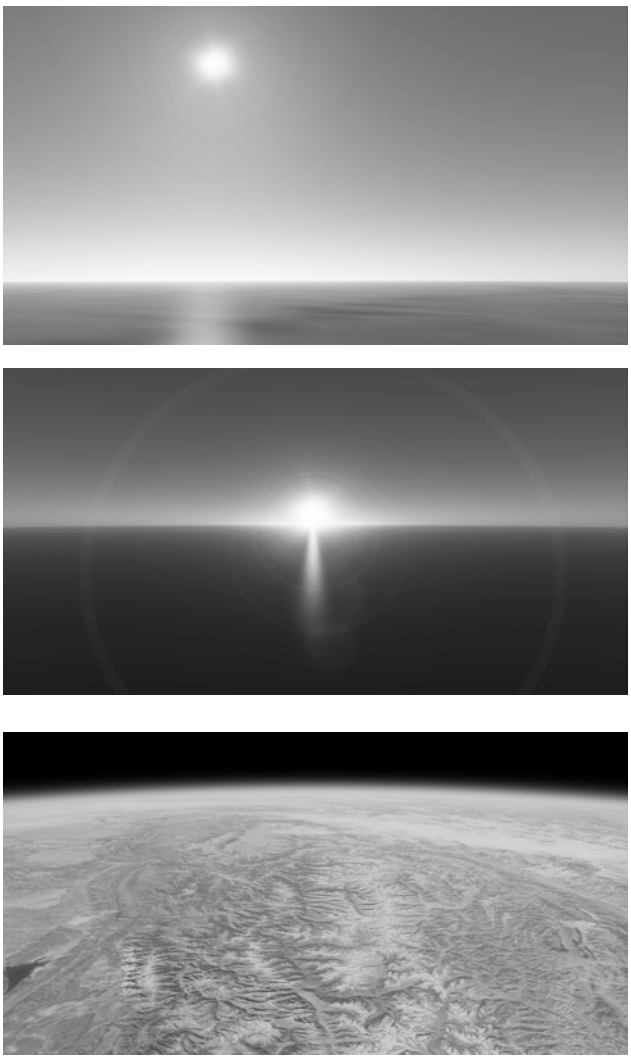


Рис. 9. Результаты моделирования освещения Земли для различных высот наблюдения

Real-time virtual Earth illumination simulation given the atmosphere

P.Yu. Timokhin

Abstract. The paper presents a method of simulation of the Earth illumination for virtual reality systems and video simulators. The proposed method is based on physically plausible simulation of light traversing the atmosphere by means of light transport equation, and allows generating realistic Earth images for arbitrary mutual "viewer-light source" arrangement. The paper also describes an approach which allows computing Earth lighting in real-time, based on using precomputed tables for costly transport equation terms.

Keywords: visualization, the Earth, the daily lighting, atmosphere, real time.

Литература

1. М.В.Михайлюк, М.А.Торгашев. Система визуализации "GLView" для имитационно-тренажерных комплексов подготовки космонавтов. // Пилотируемые полеты в космос, № 4, 2013, с. 60-74.

2. E. Bruneton, F. Neyret. Precomputed atmospheric scattering // Eurographics Symposium on Rendering 2008, v.27, № 4.

3. O. Elek. Rendering parametrizable planetary atmospheres with multiple scattering in real-time. // In Proceedings of the Central European Seminar on Computer Graphics, 2009.

4. Ю.М.Тимофеев, А.В.Васильев. Теоретические основы атмосферной оптики. – СПб.: Наука, 2003.

5. W.M. Cornette, J.G. Shanks. Physical reasonable analytic expression for the single-scattering phase function. Applied Optics, 1992, v. 31, № 16, pp.3152- 3160.

6. P. Blasi, B.Le Saec, C. Schlick. A rendering algorithm for discrete volume density objects. // Computer Graphics Forum, 1993, v. 12, № 3, pp. 201–210.

7. E. Cerezo, F. Pérez, X. Pueyo etc. A survey on participating media rendering techniques. // Visual Computer, 2005, v. 21, № 5, pp. 303-328.

Оптимальная коррекция директивных интервалов в задаче построения многопроцессорного расписания с нефиксированными параметрами

М.Г. Фуругян

кандидат физико-математических наук

Аннотация: Рассматривается задача составления допустимого расписания с прерываниями в многопроцессорной системе для случая, когда заданы директивные интервалы, а длительности выполнения работ линейно зависят от количества выделенного им дополнительного ресурса. В случае, когда при заданном количестве дополнительного ресурса допустимого расписания не существует, рассматривается задача оптимальной коррекции директивных интервалов.

Ключевые слова: директивные интервалы, многопроцессорное расписание

1. Постановка задачи

Рассматривается вычислительная система, состоящая из m идентичных процессоров, и множество работ $N = \{1, 2, ..., n\}$. Предполагается, что в каждый момент времени каждый процессор может выполнять не более одной работы, а каждая работа выполняется не более чем одним процессором. При выполнении работ допускаются прерывания и переключения с одного процессора на другой. Предполагается, что прерывания и переключения не сопряжены с временными затратами. Для работы $i \in N$ установлен директивный интервал $(b_i, f_i]$ (т.е. работа i может выполняться только в этом интервале). Помимо процессоров в системе имеется дополнительный ресурс невозобновляемого типа. Суммарное количество этого ресурса составляет R. Если работе i выделено r_i единиц дополнительного ресурса, то ее длительность составляет $t_i = d_i - a_i r_i$, где

$$r_i \in [0, \bar{r}_i], i = 1,...,n, \quad (1)$$

$$\sum_{i \in N} r_i \le R, \quad (2)$$

$a_i, d_i, \bar{r}_i -$ заданные величины, $a_i > 0$, $d_i > 0$, $0 \le \bar{r}_i < d_i / a_i$. Таким образом, $t_i \in [d_i - a_i \bar{r}_i, d_i]$ причем $d_i - a_i \bar{r}_i > 0$. Требуется найти такое распределение ресурсов $(r_1^0, r_2^0, ..., r_n^0)$, при котором существует допустимое расписание (каждая работа выполняется в своем директивном интервале). При этом распределение ресурсов $(r_1^0, r_2^0, ..., r_n^0)$ должно удовлетворять ограничениям (1), (2). В случае, когда указанного распределения ресурсов не существует, требуется определить директивные интервалы $(b_i, f_i']$, при которых указанное распределение ресурсов существует и величина $\max_{i \in N}(f_i' - f_i)$ принимает минимальное значение.

Подобная задача для случая, когда дополнительный ресурс отсутствует, рассматривалась в [1] и была сведена к задаче о максимальном потоке в сети специального вида. Задача с произвольными процессорами и произвольными директивными интервалами рассматривалась в [2], а задача с произвольными процессорами и одинаковыми директивными интервалами – в [3]. В обеих этих работах дополнительный ресурс не рассматривался. Задача минимизации времени выполнения сетевого комплекса работ, когда длительности выполнения работ являются функциями от вектора распределения ресурсов, а число процессоров не ограничено, рассматривалась в [4] и была сведена к задаче нелинейного программирования. Задача с дополнительным ресурсом и произвольными директивными интервалами рассматривалась в [5] и была сведена к задаче о потоке минимальной стоимости.

Рассмотрим некоторые примеры подобных задач. В первом примере длительность t_i задания i зависит от величины энергопотребления r_i для этого задания: $t_i = d_i - a_i r_i$. Общий объем энергоресурсов для всех заданий ограничен величиной R. Во втором примере каждому заданию i выделяются дополнительные средства r_i на приобретение специализированных процессоров (которые могут быть использованы только для выполнения конкретного задания), что снижает объем работы основных процессоров по выполнению данного задания и его длительность. В третьем примере, аналогичном второму, помимо основных станков (процессоров) имеется денежный фонд в размере R для приобретения дополнительных станков или найма рабочей силы для выполнения работы i, что снижает объем работы, выполняемой основными станками.

2. Задача с общим директивным интервалом

В этом разделе предполагается, что директивные интервалы всех работ совпадают. Без ограничения общности можно считать, что они совпадают с интервалом $(0, F]$. Предположим также, что величины R и F фиксированы. Как следует из [1], необходимым и достаточным условием существования допустимого рас-

писания в рассматриваемой задаче является выполнение неравенства $\sum_{i\in N} t_i \le mF$, или

$$\sum_{i\in N}(d_i - a_i r_i) \le mF\,,\quad \sum_{i\in N} a_i r_i \ge \sum_{i\in N} d_i - mF.$$

Пусть $\sum_{i\in N} d_i - mF = B$. Тогда задача заключается в поиске такого распределения ресурса (r_1, r_2, …, r_n), которое удовлетворяет системе ограничений

$$\sum_{i\in N} a_i r_i \ge B,$$
$$\sum_{i\in N} r_i \le R\,, \qquad (3)$$
$$r_i \in [0,\ \bar{r}_i\ ],\ i = 1, 2, \ldots, n.$$

Таким образом, допустимое расписание существует тогда и только тогда, когда задача линейного программирования (3) имеет решение. Если задача (3) имеет решение $(r_1^0, r_2^0, ..., r_n^0)$, то определив $t_i = d_i - a_i r_i^0,\ \ i \in N,$ и применив алгоритм упаковки [1], найдем допустимое расписание. Вычислительная сложность алгоритма Кармаркара решения задачи линейного программирования для случая, когда число ограничений не превосходит числа переменных n, составляет $O(n^{4,5}\ \log_2(nT))$, где T – максимальное значение числового параметра. В задаче (3) n переменных и $n + 2$ ограничения. Вводя две фиктивные переменные, получаем, что вычислительная сложность решения задачи (3) составляет $O(n^{4,5}\ \log_2(n\,T_1))$, где T_1 – максимальное из чисел a_i, d_i / a_i ($i \in N$), B и R. Сложность алгоритма, описанного в [3] – $O(n)$.

Определим минимальное количество $R_{\min}$ дополнительного ресурса, при котором для заданного директивного срока F допустимое расписание существует. Рассмотрим задачу линейного программирования:

$$R \to \min_{(r_1,\ldots,r_n,R)},$$
$$\sum_{i\in N} a_i r_i \ge B\,,$$
$$\sum_{i\in N} r_i \le R\,, \qquad (4)$$
$$r_i \in [0,\ \bar{r}_i\ ],\ i = 1, 2, \ldots, n,$$
$$R \ge 0.$$

Задача (4) имеет решение. Каждому ее решению $(r_1^0, r_2^0, ..., r_n^0)$, $R_{\min}$ соответствует допустимое расписание, а R_{min} – минимально допустимое количество дополнительного ресурса. Вычислительная сложность решения задачи (4) составляет $O(n^{4,5}\ \log_2(n\,T_2))$, где T_2 – максимальное из чисел a_i, d_i / a_i ($i \in N$) и B.

3. Минимизация общего директивного срока

Теперь будем предполагать, что количество ресурса R в системе фиксировано, а директивный срок может изменяться, за счет чего будет достигаться существование допустимого расписания. Как и в разд. 2, предполагается, что директивный интервал каждой работы совпадает с интервалом (0, F], а величина F минимизируется. В этом случае решается следующая задача линейного программирования:

$$F \to \min_{(r_1,\ldots,r_n,F)},$$
$$\sum_{i\in N} a_i r_i + mF \ge \sum_{i\in N} d_i,$$
$$\sum_{i\in N} r_i \le R, \qquad (5)$$
$$r_i \in \left[0, \bar{r}_i\right] i \in N.$$

Задача (5) имеет решение. Каждому ее решению $(r_1^0, r_2^0, ..., r_n^0)$, $F_{\min}$ соответствует допустимое расписание, а $F_{\min}$ – минимально допустимый директивный срок при фиксированной величине ресурса R в системе. Применив алгоритм упаковки [1], получим окончательное решение в виде допустимого расписания. Вычислительная сложность решения задачи (5) составляет $O(n^{4,5}\ \log_2(n\,T_3))$, где T_3 – максимальное из чисел a_i, d_i / a_i ($i \in N$), $\sum_{i\in N} d_i$, m и R.

4. Сведение исходной задачи к задаче о потоке минимальной стоимости

4.1. Произвольные директивные интервалы

В общем случае, когда директивные интервалы произвольные, по аналогии с тем, как это сделано в [1], построим потоковую сеть $G = (V, A)$ (рис. 1) с источником s и стоком t (V – множество узлов сети, A – множество дуг). Пусть $y_0 < y_1 < ... < y_p$ ($p < 2n$) – все различные по величине значения b_i, $f_i (i \in N)$. Определим $V = \{s, t, I_1, I_2, ..., I_p, w_1, ..., w_n\}$, где узел I_j соответствует интервалу $(y_{j-1}, y_j]\ (j = 1,2,..., p)$, а узел w_i – работе $i \in N$. Множество дуг A сети G определим следующим образом:
$(s, I_j)(j = 1,2,..., p); (w_i, t)(i \in N); (I_j, w_i)$ в случае, если $(y_{j-1}, y_j] \subseteq (b_i, f_i]$ (отметим, что для любых j ($1 \le j \le p$) и $i \in N$ интервал $(y_{j-1}, y_j]$ либо не пересекается с интервалом $(b_i, f_i]$, либо целиком лежит в нем); определим также возвратную дугу (t, s). Пусть $\Delta_j = y_j - y_{j-1}$ – длина интервала I_j. Для каждой дуги определим три параметра: L, U, C, где L – нижняя граница потока по дуге, U – верхняя граница потока по дуге, C – стоимость единицы потока по дуге. Параметры дуг сети G указаны в табл. 1.

Таблица 1. Параметры дуг сети G

Дуга	***L***	***U***	***C***
(s, I_j)	0	$m\Delta_j$	0
(I_j, w_i)	0	Δ_i	0
(w_i, t)	$d_i - a_i \overline{r_i}$	d_i	$-1/a_i$
(t, s)	0	$\sum_{i \in N} d_i$	0

Л е м м а. Для существования допустимого расписания необходимо и достаточно существование в сети G циркуляции g, стоимость которой

$$c(g) \le R - \sum_{i \in N} d_i / a_i . \qquad (6)$$

Доказательство леммы содержится в [5]. Там же описан алгоритм решения задачи, основанный на этой лемме.

Отметим, что задача с одинаковыми директивными интервалами является частным случаем задачи с произвольными директивными интервалами и поэтому также может быть сведена к задаче о потоке минимальной стоимости. Следовательно, для случая с одинаковыми директивными интервалами справедливо утверждение леммы и сохраняет силу описанный в [5] алгоритм решения задачи. Однако, размерность сети G при этом уменьшается. А именно, в этом случае $p = 1$.

двух или более работ их директивные интервалы совпадают, то эти работы можно заменить одной работой с тем же директивным интервалом и длительностью, равной сумме длительностей этих работ.) Построим для этого случая потоковую сеть и будем обозначать ее G^* (рис. 2). Заметим, что $I_1 = (0, f_1], I_2 = (f_1, f_2], \ldots, I_n = (f_{n-1}, f_n]$. Таким образом, особенностью данной сети является то, что из узла I_j исходят дуги, ведущие в узлы $\{w_j, w_{j+1}, \ldots, w_n\}$. Пусть g – циркуляция минимальной стоимости в сети G^*, а $c(g)$ – ее стоимость. Обозначим потоки по дугам $(s, I_j), (I_j, w_i)$ и (w_i, t) через $g(s, I_j)$, $g(I_j, w_i)$ и $g(w_i, t)$ соответственно.

Предположим, что условие (6) леммы не выполнено, т.е. $c(g) > R - \sum_{i \in N} d_i / a_i$. Поскольку общее количество ресурса фиксировано, то добиться выполнения неравенства (6) за счет увеличения величины R нельзя. Будем добиваться выполнения неравенства (6) за счет увеличения директивных сроков f_i, $i \in N$. Из постановки задачи следует, что правые границы всех директивных интервалов следует увеличивать на одну и ту же величину. Увеличим их на величину $\varepsilon > 0$: $f_i' = f_i + \varepsilon$. Тогда будем иметь следующие длины Δ_j^* интервалов I_j: $\Delta_1^* = \Delta_1 + \varepsilon$, $\Delta_j^* = \Delta_j$, $(j \ge 2)$. Следовательно, после такой коррекции всех директивных интервалов увеличатся пропускные способности дуг (s, I_1) и (I_1, w_i), $i \in N$. Это означает возможность увеличения потока по этим дугам, а также по дугам

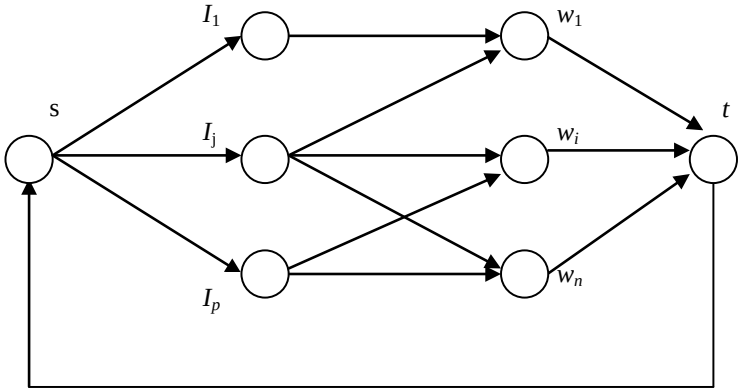


Рис.1. Потоковая сеть G

4.2. Директивные интервалы с общим начальным сроком

4.2.1. Построение потоковой сети

В этом разделе будем предполагать, что директивные интервалы имеют вид $(0, f_i]$, $i \in N$. Без ограничения общности можно считать, что $f_i < f_{i+1}$ при всех $i \in N$, т.е. работы упорядочены по возрастанию правых границ их директивных интервалов. (Если для

(w_i, t) и (*t*, *s*). Поскольку стоимость потока по дугам (w_i, t) – величина отрицательная, то увеличение потока по ним снижает его стоимость. (Отметим, что если поток по всем дугам (w_i, t) будет равен его верхней границе, т.е. величине d_i, то условие (6) будет выполнено.)

4.2.2. Сведение задачи оптимальной коррекции директивных интервалов к задаче линейного программирования

Пусть $\delta = c(g) - \left[R - \sum_{i \in N} \frac{d_i}{a_i} \right]$. В дальнейшем будем рассматривать сеть G^*. Будем увеличивать потоки по дугам $(s, I_1), (I_1, w_i)$, (w_i, t) и (*t*, *s*). Потоки по всем остальным дугам сети G^* изменять не будем. Пусть

$$g^*(w_i, t) = g(w_i, t) + x_i \le d_i,\ i \in N,$$
$$g^*(t, s) = g(t, s) + \sum_{i \in N} x_i \le \sum_{i \in N} d_i.$$

Отметим, что последнее неравенство является следствием предыдущего. Поскольку ненулевую стоимость потока имеют только дуги (w_i, t), то общая стоимость потока изменится на величину $\sum_{i \in N} x_i \cdot (-1/a_i)$. Следовательно, из неравенства (6) получаем, что для существования допустимого расписания должно выполняться соотношение $c(g) - \sum_{i \in N} x_i / a_i \le R - \sum_{i \in N} d_i / a_i$, откуда следует еще одно ограничение на x_i:

$$\sum_{i \in N} x_i / a_i \ge \delta.$$

Таким образом, минимизируя ε при указанных ограничениях, получаем следующую задачу линейного программирования:

$$\min_{x_1, \ldots, x_n, \varepsilon} \varepsilon,$$
$$\sum_{i \in N} x_i - m\varepsilon \le m\Delta_1 - g(s, I_1),$$
$$x_i - \varepsilon \le \Delta_1 - g(I_1, w_i),\ i \in N, \qquad (7)$$
$$x_i \le d_i - g(w_i, t),\ i \in N,$$
$$\sum_{i \in N} x_i / a_i \ge \delta,$$
$$\varepsilon > 0,\ x_i \ge 0,\ i \in N.$$

Решение задачи (7) позволит оптимальным образом скорректировать директивные интервалы. А именно, все директивные сроки f_i будут увеличены на минимальную величину ε так, что допустимое расписание будет существовать. Для построения допустимого расписания следует применить алгоритм, описанный в [5]. Вычислительная сложность решения задачи (7) составляет $O(n^{4,5} \log_2(n T_4))$, где T_4 – максимальное из чисел $1/a_i$, d_i ($i \in N$), $\sum_{i \in N} (d_i / a_i - R)$ и $m\Delta_1$.

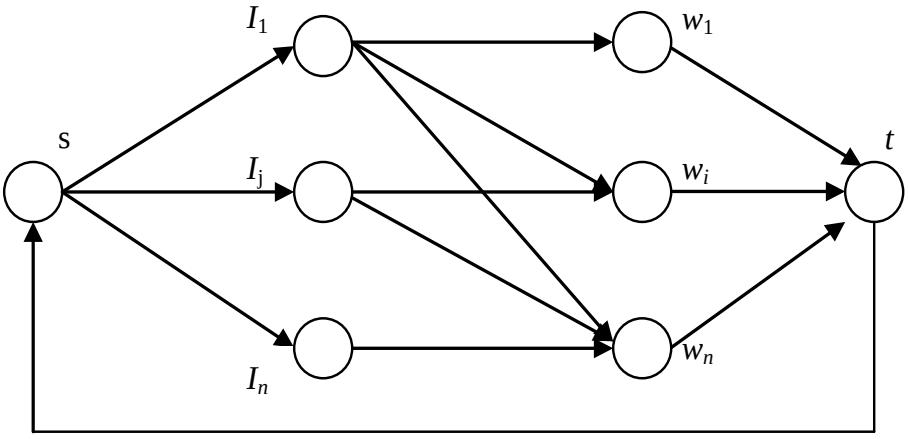


Рис. 2. Потоковая сеть *G**

g^* – модифицированная циркуляция в сети G^* и $g^*(I_1, w_i) = g(I_1, w_i) + x_i$, $i \in N$, где x_i – величина, на которую увеличен поток по дуге (I_1, w_i). В силу условия сохранения потока в узлах сети модифицированные потоки по остальным дугам сети G^* будут следующими:

$$g^*(s, I_1) = g(s, I_1) + \sum_{i \in N} x_i,$$
$$g^*(w_i, t) = g(w_i, t) + x_i,\ i \in N,$$
$$g^*(t, s) = g(t, s) + \sum_{i \in N} x_i.$$

Поскольку величины $x_i \ge 0$ при всех $i \in N$ и ограничения снизу на потоки g по всем дугам сети G^* выполнены, то и ограничения снизу на потоки g^* по всем дугам сети G^* также выполнены. Выпишем ограничения сверху на потоки g^* по дугам сети G^*, поток по которым увеличивается:

$$g^*(s, I_1) = g(s, I_1) + \sum_{i \in N} x_i \le m(\Delta_1 + \varepsilon);$$
$$g^*(I_1, w_i) = g(I_1, w_i) + x_i \le \Delta_1 + \varepsilon,\ i \in N;$$

4.2.3. Последовательное изменение потоков

Рассмотрим второй способ оптимальной коррекции директивных интервалов. Пусть, по-прежнему, условие (6) не выполнено, а правые границы f_i директивных интервалов увеличены на ε, т.е. $f_i' = f_i + \varepsilon$. Как отмечалось выше, это дает возможность увеличивать в сети G^* потоки по циклам $(s,\ I_1,\ w_i,\ t,\ s)$, $i \in N$. Ниже предлагается эвристический алгоритм, позволяющий определить, существует ли в этом случае допустимое расписание. Алгоритм выдает значение $A_1 = 1$, если допустимое расписание существует, и $A_1 = 0$ в противном случае. Алгоритм основан на последовательном выборе указанных циклов с минимальной стоимостью единицы потока.

Алгоритм 1.

1. Положить $\alpha = m\Delta_1 - g(s, I_1)$,
$\beta = \sum_{i \in N} d_i - g(t, s)$.

2. Положить $i_0 = \min_{i \in N}(-1/a_i)$.

3. Положить
$\gamma = \min(\alpha; \Delta_{i_0} - g(I_1, w_{i_0}); d_{i_0} - g(w_{i_0}, t); \beta)$.

4. Положить $N = N / \{i_0\}$.

5. Положить $g(s, I_1) = g(s, I_1) + \gamma$;
$g(I_1, w_{i_0}) = g(I_1, w_{i_0}) + \gamma$; $g(w_{i_0}, t) = g(w_{i_0}, t) + \gamma$;
$g(t, s) = g(t, s) + \gamma$.

6. Если $N \neq \varnothing$, то перейти на шаг 1. В противном случае перейти на шаг 7.

7. Если условие (6) выполнено, то положить $A_1 = 1$ (допустимое расписание существует), в противном случае положить $A_1 = 0$ (допустимого расписания не существует). Алгоритм завершает работу.

Дадим некоторые пояснения к алгоритму 1. На шаге 1 определяются остаточные пропускные способности дуг (s, I_1) и $(t,\ s)$. На шаге 2 определяется дуга (w_{i_0}, t) с минимальной стоимостью единицы потока. На шаге 3 определяется максимально допустимая величина γ увеличения потока по циклу $(s,\ I_1,\ w_{i_0},\ t,\ s)$. На шаге 4 из множества N исключается работа i_0. На шаге 5 производится увеличение потока по дугам цикла $(s,\ I_1,\ w_{i_0},\ t,\ s)$ на величину γ. На шаге 6 проверяется условие завершения алгоритма, а на шаге 7 – условие существования допустимого расписания.

Пусть E – это максимальная величина, на которую можно увеличить правые границы директивных интервалов (эта величина определяется из физической постановки задачи). Приведем описание эвристического алгоритма нахождения минимальной величины A_2, на которую можно увеличить правые границы директивных интервалов, чтобы допустимое расписание существовало. Пусть τ – это максимально допустимая погрешность вычисления величины A_2.

Алгоритм 2.

1. Положить $\varepsilon = E$.

2. Применить алгоритм 1. Если $A_1 = 0$, то коррекция директивных интервалов невозможна. Алгоритм завершает работу. Если $A_1 = 1$, то перейти на шаг 2.

3. Положить $E_1 = 0,\ E_2 = E$.

4. Положить $\varepsilon = (E_1 + E_2)/2$.

5. Применить алгоритм 1. Если $A_1 = 0$, то положить $E_1 = \varepsilon$. Если $A_1 = 1$, то положить $E_2 = \varepsilon$.

6. Если $E_2 - E_1 \le \tau$, то положить $A_2 = E_2$; алгоритм завершает работу. Если $E_2 - E_1 > \tau$, то перейти на шаг 4.

Дадим некоторые пояснения к алгоритму 2. На каждом шаге алгоритма величины E_1 и E_2 определены таким образом, что при $\varepsilon = E_1$ допустимого расписания не существует, а при $\varepsilon = E_2$ допустимое расписание существует. Если $E_2 - E_1 \le \tau$, то в качестве приближенного решения выбирается значение $\varepsilon = E_2$ (шаг 6). При этом погрешность вычисления не превосходит заданной величины τ. Если же $E_2 - E_1 > \tau$, то определяется середина отрезка $[E_1, E_2]$ (шаг 4) и вычисления продолжаются.

Отметим, что вычислительная сложность алгоритма 1 составляет $O(n)$, а число обращений к алгоритму 1 при работе алгоритма 2 – $O(\log_2 E)$. Таким образом, вычислительная сложность алгоритма 2 составляет $O(n \log_2 E)$.

Optimal deadline correction in multiprocessor scheduling with not fixed parameters

M.G.Furugian

Abstract: The problem of deadline interruption scheduling in multiprocessor system in the case of execution times of jobs linearly depend on additional resource is studied. In the case of feasible schedule not exist the problem of optimal deadline correction is studied.

Keywods: deadline, multiprocessor scheduling

Литература

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

2. *A. Federgruen, H. Groenevelt.* Preemptive Scheduling of Uniform Machines by Ordinary Network Flow Technique". // Management Sci. – 1986, v. 32, №.3, March , pp. 812 – 829.

3. *T.Gonzales, S. Sahni.* Preemptive scheduling of uniform processor systems // Journal of the Association for Computing Machinery. – 1978, v. 25, № 1,. pp. 92 – 101.

4. *Э.Г.Давыдов.* Исследование операций. М.: Высшая школа, 1990.

5. *М.Г.Фуругян.* Планирование вычислений в многопроцессорных системах с дополнительным ресурсом. // Труды НИИСИ РАН, 2012, т. 2, № 1, с. 60 – 66.

Методы расчета безбликовых оптических систем визуализации глазного дна

В.К. Салахутдинов, А.Г.Прозорова, Б.В.Глухоедов, Е.А.Монакова и Джассер Дорошенко

Аннотация: В работе представлены результаты разработки фундус камеры с безбликовой оптической системой, в которой используются многолинзовые объективы со светосилой (D/F ≤ 1) существенно меньше единицы и реализуется освещение глазного дна через зрачок глаза менее 3 мм. При этом основное внимание уделено методу расчету безбликовой оптической системы. Основная идея данной работы состоит в том, что геометрия оптических элементов системы такова, что блик в виде паразитного отражения от оптической поверхности одного из элементов фокусируется на малоразмерном поглощающем экране, расположенном на другой оптической поверхности.

Ключевые слова: оптическая система, глазное дно

Введение

Одним из основных диагностических приборов современной офтальмологии является фундус камера, которая позволяет бесконтактно регистрировать изображение тканей глазного дна через зрачок глаза. Особенность оптической системы фундус-камеры в том, что одни и те же ее оптические элементы одновременно используются для освещения тканей глазного дна и для регистрации изображения. Так как интенсивность освещающего светового пучка примерно в 10^6 раз превосходит интенсивность регистрируемого изображения, то возникает проблема устранения бликов от оптических поверхностей. Для устранения бликов практически все современные фундус камеры реализуют оптическую схему Nordenson Zeiss, которая была предложена еще в 1925 году и предполагает применение однолинзового объектива со светосилой (D/F ≥ 1) больше единицы. Проблема в том, что для эффективного применения в телемедицине фундус камеры должны обеспечивать высокое разрешение, которое практически невозможно реализовать на базе оптической схемы Nordenson Zeiss из-за больших аберраций суперсветосильного однолинзового объектива. Попытки уменьшить аберрации за счет применения адаптивной оптики на практике оказались малоэффективными. Кроме того, оптическая система Nordenson Zeiss не позволяет реализовать освещение глазного дна при размере зрачка глаза менее 3-4 мм, что также ограничивает ее применение в телемедицине.

В работе представлены результаты разработки фундус камеры с безбликовой оптической системой, в которой используются многолинзовые объективы со светосилой (D/F ≤ 1) существенно меньше единицы и реализуется освещение глазного дна через зрачок глаза менее 3 мм. При этом основное внимание уделено методу расчету безбликовой оптической системы.

Основная идея данной работы состоит в том, что геометрия оптических элементов системы такова, что блик в виде паразитного отражения от оптической поверхности одного из элементов фокусируется на малоразмерном поглощающем экране, расположенном на другой оптической поверхности.

1. Оптическая система

Оптическая схема представлена на рис.1. и представляет собой афокальную систему из двух объективов (O_1 и O_2), каждый из которых состоит из двух линз. В переднем фокусе объектива расположен зрачок глаза, а в заднем фокусе объектива расположен точечный источник света для освещения глазного дна. Для устранения бликов в центрах поверхностей оптических элементов расположены малоразмерные светопоглощающие экраны.

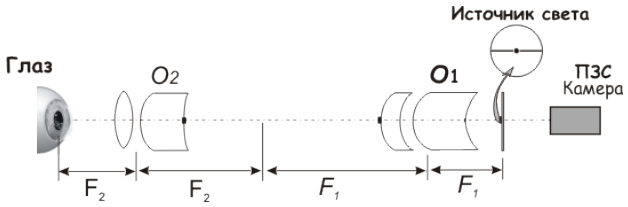


Рис. 1. Оптическая схема фундус камеры.

Для регистрации изображения глазного дна оптическая система фокусирует излучение источника света на зрачке исследуемого глаза и как следствие, освещает глазное дно. Одновременно, оптическая система переносит изображение глазного дна на ПЗС камеру, которая его регистрирует.

Для устранения бликов оптические элементы выполнены и расположены так, что блик от r_i оптической поверхности фокусируется на малоразмерном поглощающем экране, который расположен в центре другой (r_j) оптической поверхности. На приведенном примере (рис.2) блик от поверхности R поглощается экраном A , расположенным на поверхности D.

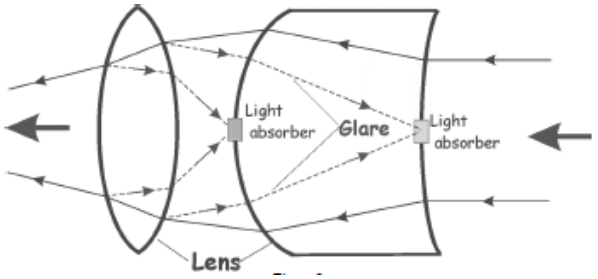


Рис. 2. Пример поглощения блика

2. Метод расчета безбликовой системы

Для расчета описанной схемы воспользуемся методами матричной оптики [4]. Для этого представим схему (рис.1) в виде рис3.

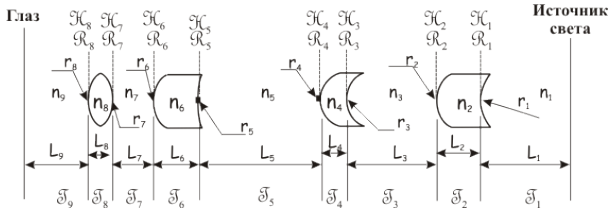


Рис. 3. Адаптированная оптическая схема фундус-камеры для расчета.

Где L_i – длина i-го оптического пути, n_i – показатель преломления, r_i – радиус кривизны i-ой оптической поверхности,

$$\Im_i = \begin{bmatrix} 1 & \frac{L_i}{n_i} \\ 0 & 1 \end{bmatrix},$$

$$\Re_i = \begin{bmatrix} 1 & 0 \\ -(n_{i+1}-n_i)/r_i & 1 \end{bmatrix}, K_i = \begin{bmatrix} 1 & 0 \\ 2n_i/r_i & 1 \end{bmatrix}.$$

Можно показать, что требование афокальности соответствует условиям $m^1_{12}=0$ и $m^1_{21}=0$, где m^1_{12}, m^1_{21} – элементы матрицы вида

$$M^1 = (T_9 R_8 T_8 R_7 T_7 R_6 T_6 R_5 T_5 R_4 T_4 R_3 T_3 R_2 T_2 R_1 T_1)$$

и трех дополнительным условиям –

$$m^{01}_{22}=0$$

для $M^{01} = (R_4 T_4 R_3 T_3 R_2 T_2 R_1 T_1)$,

$$m^{11}_{11}=0$$

для $M^{11} = (T_9 R_8 T_8 R_7 T_7 R_6 T_6 R_5)$ и

$$L_5 = \frac{m^{11}_{22}}{m^{11}_{21}} - \frac{m^{01}_{11}}{m^{01}_{21}}$$

А для устранения бликов необходимо выполнение совокупности условий:

$$(m^2_{12}=0)\wedge(m^3_{12}=0)\wedge(m^4_{12}=0)\wedge(m^5_{12}=0)\wedge(m^6_{12}=0)\wedge(m^7_{12}=0)\wedge(m^8_{12}=0)\wedge(m^8_{12}=0),$$

где m^2_{12} – элемент матрицы вида $M^2=(T_1 K_1 T_1)$,

m^3_{12} – элемент матрицы вида

$$M^3=(T_2 K_2 T_2 R_1 T_1)\cup(T_1 R_1 T_2 K_2 T_2 R_1 T_1)$$

m^4_{12} – элемент матрицы вида

$$M^4=(T_3K_3T_3R_2T_2R_1T_1)\cup(T_2R_2T_3K_3T_3R_2T_2R_1T_1)\cup(T_1R_1T_2R_2T_3K_3T_3R_2T_2R_1T_1)$$

m^5_{12} – элемент матрицы вида

$$M^5=(T_4 K_4 T_4 R_3 T_3 R_2 T_2 R_1 T_1)\cup(T_4 K_4 T_4 R_3 T_3 R_2 T_2 R_1 T_1)\cup(T_1R_1T_2R_2T_3K_3T_3R_2T_2R_1T_1).$$

Аналогично формулируются условия для m^6_{12}, m^7_{12}, m^8_{12}, m^9_{12}.

Видно, что задача расчета сводится к решению линейной системы, в которой число неизвестных больше числа уравнений. Можно показать, что эти уравнения независимы, то есть множество решения является подпространством в пространстве переменных. Из множества допустимых решений целесообразно выбрать вариант, который обладает максимальной устойчивостью. Основная трудность в таком подходе связана с большим объемом требуемых вычислений.

С целью снижения вычислительной сложности задачи и эффективного выбора наилучшего решения была создана библиотека прецедентов. Численные эксперименты показали, что использование библиотеки сокращает время вычислений в среднем в 10 – 12 раз.

Результаты расчета рассмотренной (Рис.1., Рис.3.) оптической системы представлены в таблице 1

L_1, mm	r_1 mm	L_2, mm	r_2 mm	L_3, mm	r_3 mm	L_4, mm	r_4 mm
n=1,0		1,51/ BK7		n=1,0		1,51/ BK7	
56,41	56,41	32,1	47,0	10,0	62,0	10,0	40,0

L_5, mm	r_5 mm	L_6, mm	r_6 mm	L_7, mm	r_7 mm	L_8, mm	r_8 mm	L_9, mm
n=1,0		1,51/ BK7		n=1,0		1,51/ BK7		n=1,0
147,0	407,0	25,0	50,0	11,0	-	5,0	60,0	43,50

					140,5			

Видно, что светосила оптических элементов не превышает 0.3.

3. Эксперимент

Рассчитанная (Таблица 1) оптическая система была реализована в виде экспериментального макета. В качестве точечного источника света использовался торец световолокна диаметром 9мкм, который закреплен на светопоглощающей площадке диаметром 650 мкм. С помощью сплиттера в световолокно вводилось излучение лазеров с λ=530нм и λ=650нм.. Диаметр светового пятна осветителя на поверхности роговицы глаза составлял менее 500 мкм. Диаметры светопоглотителей, которые были закреплены на центрах оптических поверхностей линз, составлял от 400 до 700 мкм

На рис.4 представлено изображение глазного дна, полученное с помощью Фундус камеры Zeiss FF-450 и с помощью экспериментального макета.

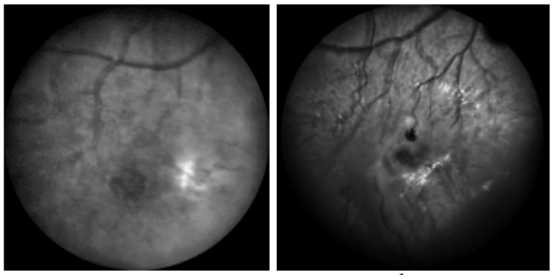


Рис. 4. Изображение глазного дна с помощью Фундус камеры (левый) и с помощью экспериментального макета (правый).

Видно, что разрешающая способность экспериментального макета заметно выше, чем у одного из лучших промышленных изделий. Также видно, что на изображении глазного дна, полученном с помощью макета, виден артефакт, который обусловлен наличием в световой апертуре поглотителей.

Заключение

Очевидна тривиальность того, что представлено после изложения общей идеи. Тем не менее, мы сочли целесообразным включить это в работу, чтобы показать возможность практической реализации. Что касается очевидных недостатков (артефакта в виде небольшого темного пятна в центре каждого фрагмента регистрируемого изображения и хроматических аберраций), то первый полностью устраняется внесением незначительных изменений в программу сшивки отдельных малоапертурных фрагментов в общее широкоапертурное изображение, а второй – за счет применения спектрально-селективной регистрации (получение одного цветного снимка из нескольких кадров, которые сняты при различной длине волны освещения). Кроме того, при спектрально-селективной регистрации цветного изображения можно использовать B/W камеру, чувствительность которой на порядок выше, чем у цветной и за счет этого понизить световое воздействие на сетчатку глаза. В экспериментах, с помощью B/W камеры нами было получено цветное изображение глазного дна с разрешением Full HD при диаметре зрачка примерно 1.5 мм и освещенностью тканей глазного дна в пределах санитарных норм. Мы полагаем, что рассмотренное решение позволяет реализовать регистрацию изображения глазного дна без применения медикаментозных средств и более высоким разрешением. Это принципиально важно для развития телемедицины.

Glare-free optical system for fundus visualization

V. Salakhutdinov, A.Prozorova, B.Glukhoedov, E.Monakova, J. Doroshenko

Abstract: The paper describes results of development of fundus-camera with non-glare optical scheme. The scheme is based on multiple lenses with a light gathering power (D/F ≤ 1) substantially less than one. Illumination of fundus can be provided through eye's pupil less than 3 mm. And much attention was directed to method of calculation of the nonglare optical scheme. The key idea is that geometry of optic elements of the system provides that glare in the form of ghost reflection from optical surface of one element focuses on a small-size absorbing screen located on another optical surface. Shows the possibility of implementation and the experimental results. During experiments with B/W camera we got Full HD color image of fundus having the eye's pupil diameter of 1.5 mm and illumination of fundus tissue in accordance with sanitary rules.

Keywords: optical system, the ocular fundus

Литература

1. Jack Kanski and Brad Bowling. Clinical Ophthalmology: A Systematic Approach. Elsevier (2011)

2. Patent US # 20130301005. Fundus camera and control method for the fundus camera (2013)

3. Arthur J. Bedell. The Newest Model of the Nordenson Zeiss Photographing Ophthalmoscope. Trans. Am. Ophthalmol. Soc., v. 36, pp. 278–279 (1938)

4. Donald T. Miller. Coherence gating and adaptive optics in the eye. Proceedings of SPIE, 49 (56), pp. 65-72 (2003)

5. Nathan Doble, Donald T Miller, Geunyoung Yoon, and David R Williams. Requirements for discrete actuator and segmented wavefront correctors for aberration compensation in two large populations of human eyes. Applied Optics, 46 (20), (2007)

6. A.Garrard and J.M.Burch, Introduction to matrix methods in optics. A Wiley - Interscience Publication John Wiley & Sons, London New-York-Sydney-Toronto, 40 (81), (1975)

7, Francois C Delori, Robert H Webb, and David H Sliney. Maximum permis-sible exposures for ocular safety (ANSI 2000), with emphasis on ophthalmic devices. J. of the Optical Society of America, v. 24, № 5, pp. 1250-1265 (2007).

Обработка коллизий виртуальных объектов с помощью метода последовательных импульсов

А.М. Трушин

Аннотация: Виртуальные динамические объекты в трехмерных сценах компьютерных тренажерных систем могут сталкиваться друг с другом. Задание реалистичной реакции (обработки) на столкновения (коллизии) является важной и неотъемлемой частью моделирования динамики виртуальных объектов. В настоящей работе предлагается метод обработки коллизий объектов, основанный на ряде известных методов (последовательные импульсы, разделенные импульсы, «горячий» старт) и являющийся объединением данных методов.

Ключевые слова: виртуальные сцены, коллизии, метод последовательных импульсов.

Введение

В процессе моделирования динамики твердые тела (объекты) могут сталкиваться. Моделирование столкновений (коллизий) виртуальных объектов состоит из двух частей: определение коллизий и обработка коллизий.

Для определения коллизий объекты произвольной формы окружаются (аппроксимируются) простыми геометрическими телами (прямоугольные параллелепипеды, сферы, цилиндры и т.д.) и далее ищутся пересечения этих тел. Глобально алгоритмы определения коллизий разделяются на два типа: априорные и апостериорные. В первом случае алгоритм предсказывает коллизию до столкновения объектов, во втором – определяет ее уже по факту столкновения. В данной работе подразумевается использование второго типа алгоритмов. Существует множество апостериорных алгоритмов для определения коллизий, наиболее распространенные из которых: алгоритмы, основанные на теореме о разделяющей оси [1], [2]; алгоритмы, основанные на алгоритме Гилберта-Джонсона-Кёрти [3]; алгоритмы, основанные на понятии диаграмм Вороного [4], [5] и т.д.

Для обработки коллизии необходимо сформировать реакции для всех столкнувшихся виртуальных объектов. В общем виде такая реакция разделяется на две составляющие:

– изменение скоростей (линейных и угловых) столкнувшихся объектов;

– коррекция положения и ориентации объектов для устранения их взаимного проникновения.

В первом разделе данной статьи рассматриваются существующие подходы к обработке коллизий при помощи ограничений. Во втором разделе дается постановка задачи в виде ограничений, основанных на скоростях движения объектов. В третьем разделе ограничения на скоростях движения объектов переформулируются с учетом применяемых для обеспечения выполнения ограничений импульсов и рассматривается метод последовательных импульсов.

1. Обработка коллизий при помощи ограничений

В общем случае коллизия (столкновение) двух объектов характеризуется набором троек параметров $T_i = (\vec{P}_i, \vec{N}_i, D_i)$, где $i = 1,..,k_j$, $\vec{P}_i$ – i-я точка пересечения двух объектов, $\vec{N}_i$ – единичный вектор нормали столкновения в точке $\vec{P}_i$, а D_i – глубина проникновения вдоль нормали $\vec{N}_i$. При рассмотрении коллизий n пар объектов всего будет $K = \sum_{j=1}^{n} k_j$ троек параметров коллизии. Обработка всех коллизий виртуальных объектов сводится к последовательной циклической обработке всех этих K троек параметров, чем и занимается часть динамической библиотеки, называемая солвером (от англ. Solver).

Будем называть один из объектов пары первым объектом, а другой – вторым и договоримся задавать направление нормали $\vec{N}_i$ так, чтобы она всегда была направлена из второго объекта в первый. При таком задании нормали первый объект должен расталкиваться в направлении $\vec{N}_i$, а второй в направлении $-\vec{N}_i$.

Рассмотрим некоторую тройку $T = (\vec{P}, \vec{N}, D)$. Для устранения пересечения объектов в точке $\vec{P}$ необходимо так изменить их положение и ориентацию, чтобы раздвинуть их в точке $\vec{P}$ на расстояние D вдоль оси $\vec{N}$ (см. рис. 1). Кроме того, также необходимо изменить скорости объектов. Изменение скоростей объектов происходит из-за силы удара, действующей вдоль нормали $\vec{N}$, а также из-за силы трения, действующей в тангенциальной плоскости – плоскости, перпендикулярной нормали $\vec{N}$ и проходящей через точку $\vec{P}$.

Одним из распространенных методов обработки коллизий является задание так называемых ограничений [6], которые вводятся на положение и/или движение объектов. В дальнейшем объекты могут

двигаться только так, чтобы не нарушать эти ограничения.

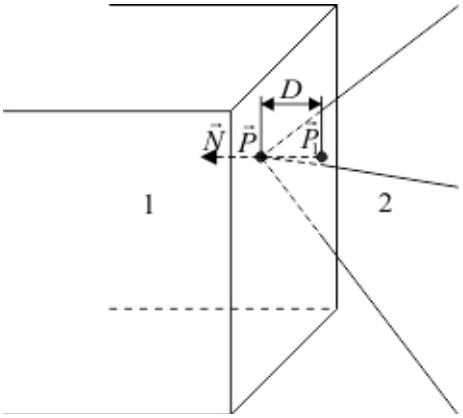


Рис. 1. Параметры столкновения объектов

Для каждой тройки $T = (\vec{P}, \vec{N}, D)$ параметров коллизий вводятся ограничения вида $C(S(t)) \geq 0$, где $S(t)$ – обобщенные координаты положения и ориентации тел. Существует три типа ограничений:

– ограничения $C(S(t)) \geq 0$, основанные на положении и ориентации объектов (position-based methods);

– ограничения $\dot{C}(S(t), \vec{V}(t)) = J\vec{V} \geq 0$, основанные на скоростях движения объектов (impulse/velocity-based methods);

– ограничения $\ddot{C}(S(t), \vec{V}(t), \dot{\vec{V}}(t)) = J\dot{\vec{V}} + \dot{J}\vec{V} \geq 0$, основанные на ускорениях движущихся объектов (force/acceleration-based methods).

Обозначим через $\vec{P}_1$ точку $\vec{P} - D\vec{N}$. Тогда ограничение для коллизии можно записать в виде

$$C = (\vec{P}_1 - \vec{P}) \cdot \vec{N} \geq 0. \quad (1)$$

При выполнении этого ограничения объекты не будут пересекаться. Продифференцировав (1) по времени t и пренебрегая изменением нормали $\vec{N}$, получим:

$$\dot{C} = (\vec{v}_{P_1} - \vec{v}_P) \cdot \vec{N} \geq 0, \quad (2)$$

где $\vec{v}_{P_1} = \vec{v}_1 + [\vec{\omega}_1, \vec{r}_1]$ и $\vec{v}_P = \vec{v}_2 + [\vec{\omega}_2, \vec{r}_2]$ – скорости движения точек $\vec{P}_1$ и $\vec{P}$ – соответственно, $(\vec{v}_1, \vec{\omega}_1)$ и $(\vec{v}_2, \vec{\omega}_2)$ – векторы линейных и угловых скоростей объектов, а $\vec{r}_1 = \vec{P}_1 - \vec{c}_1$ и $\vec{r}_2 = \vec{P} - \vec{c}_2$ – радиус-векторы от центров масс $\vec{c}_1$ и $\vec{c}_2$ объектов до точек $\vec{P}_1$ и $\vec{P}$ соответственно. Неравенство (2) характеризует скорость ограничения (1). Если неравенство не выполнено, то это означает, что глубина проникновения тел при движении будет увеличиваться. Воспользовавшись свойством смешанного произведения $[\vec{a}, \vec{b}] \cdot \vec{c} = [\vec{c}, \vec{a}] \cdot \vec{b} = [\vec{b}, \vec{c}] \cdot \vec{a}$, получим:

$$\dot{C} = \vec{v}_1 \cdot \vec{N} + [\vec{r}_1, \vec{N}] \cdot \vec{\omega}_1 - (\vec{v}_2 \cdot \vec{N} + [\vec{r}_2, \vec{N}] \cdot \vec{\omega}_2) \geq 0$$

Представим левую часть этого неравенства в виде:

$$\dot{C} = J\vec{V}, \quad (3)$$

где $J = \left(\vec{N}, [\vec{r}_1, \vec{N}], -\vec{N}, -[\vec{r}_2, \vec{N}]\right)$ – матрица Якоби, а $\vec{V} = (\vec{v}_1, \vec{\omega}_1, \vec{v}_2, \vec{\omega}_2)$ – вектор текущих обобщенных скоростей двух объектов.

В зависимости от знаков C и $\dot{C}$ можно классифицировать тип коллизии:

- если $C < 0$, то объекты пересекаются;
- если $C = 0$, то объекты касаются друг друга;
- если $C > 0$, то объекты не пересекаются;
- если $\dot{C} < 0$, то объекты движутся навстречу друг другу (проникновение увеличивается);
- если $\dot{C} = 0$, то объекты находятся в состоянии контакта (проникновение не изменяется);
- если $\dot{C} > 0$, то объекты движутся друг от друга (проникновение уменьшается).

Для установления факта контакта обычно используют пороговые значения. Все коллизии, для которых $|\dot{C}| < restingContactVelocity$, считаются контактами.

Рассмотрим подробнее три вида ограничений. Идея позиционных ограничений $C(S(t)) \geq 0$ состоит в том, что если объекты пересекаются, то нужно изменить их положения и ориентации так, чтобы объекты перестали пересекаться. У этого метода существуют очевидный минус: при столкновении объекты должны изменять не только свои положения и ориентации, но и скорости движения. Поэтому для обработки коллизий данный метод применяется редко.

Для обеспечения выполнения ограничений $\ddot{C}(S(t), \vec{V}(t), \dot{\vec{V}}(t)) = J\dot{\vec{V}} + \dot{J}\vec{V} \geq 0$, основанных на ускорениях, применяются ограничивающие силы (constraint forces) [6], [7]. Распишем второй закона Ньютона для системы объектов в векторном виде:

$$M\dot{\vec{V}} = J^T\vec{\lambda} + \vec{f}_{ext},$$

где M – диагональная матрица масс объектов, $\vec{\lambda}$ – искомые ограничивающие силы, $\vec{f}_{ext}$ – внешние силы, действующие на систему. Подставив $\dot{\vec{V}}$ в ограничения для $\ddot{C}$, получим:

$$\ddot{C} = JM^{-1}J^T\vec{\lambda} + JM^{-1}\vec{f}_{ext} + \dot{J}\vec{V} \geq 0$$

Сделаем замены $A = JM^{-1}J^T$, $\vec{b} = JM^{-1}\vec{f}_{ext} + \dot{J}\vec{V}$. Тогда выполнение ограничений $\ddot{C} \geq 0$ сводится к решению задачи линейного дополнения (LCP – Linear complementarity problem) [8]:

$$\ddot{C} = A\vec{\lambda} + \vec{b} \geq 0. \quad (4)$$

Величина $\ddot{C} = A\vec{\lambda} + \vec{b}$ является относительным ускорением объектов в направлении нормали столкновения. Условие $\ddot{C} \geq 0$ означает, что объектам запрещается ускоряться вдоль нормали внутрь друг друга. $\vec{\lambda} \geq 0$ означает, что ограничивающие силы могут действовать только в направлении, соответствующем разрешению коллизии, а при условии $\ddot{C}\vec{\lambda} = 0$ ограничивающие силы будут действовать только до тех пор, пока объекты пересекаются.

LCP (4), основанное на ускорении, всегда имеет решение в случае отсутствия силы трения. В случае же наличия силы трения LCP может не иметь решения даже для одной коллизии в системе. Это так называемая проблема Пенелеве [9]. Она была сформулирована еще в 1895 и иллюстрирует, что закон Кулона-Амонтона не всегда применим при моделировании некоторых ситуаций реального мира. В связи с этим ограничения, основанные на ускорениях, не всегда будут иметь решения, а значит давать реалистичную картину при моделировании столкновений объектов.

Выполнение ограничений $\dot{C}(S(t),\vec{V}(t)) = J\vec{V} \geq 0$, основанных на скоростях движения объектов, обеспечивается с помощью применения импульсов. Расчетом таких импульсов занимается импульсный солвер. Вначале выведем все ограничения, накладываемые на пару столкнувшихся объектов для одной тройки $T = (\vec{P}, \vec{N}, D)$ параметров их коллизии.

2. Постановка задачи

Рассмотрим составление ограничения удара без учета сил трения для одной тройки $T = (\vec{P}, \vec{N}, D)$ пары столкнувшихся объектов. При абсолютно упругом ударе (деформация отсутствует) для столкнувшихся объектов выполняются законы сохранения импульса и момента импульса:

$$m_1\vec{v}_1 + m_2\vec{v}_2 = m_1\vec{v}_1' + m_2\vec{v}_2'$$
$$I_1\vec{\omega}_1 + I_2\vec{\omega}_2 = I_1\vec{\omega}_1' + I_2\vec{\omega}_2',$$

где m_1 и m_2 – массы столкнувшихся объектов, I_1 и I_2 – матрицы их тензоров инерции, $(\vec{v}_1, \vec{\omega}_1)$ и $(\vec{v}_2, \vec{\omega}_2)$ – векторы линейных и угловых скоростей объектов до удара, а $(\vec{v}_1', \vec{\omega}_1')$ и $(\vec{v}_2', \vec{\omega}_2')$ – после удара. Здесь и далее договоримся обозначать штрихами значения соответствующих параметров в конце текущего шага моделирования, то есть на момент времени $t + \Delta t$, где t – время начала моделирования текущего кадра, Δt – время, потраченное на моделирование текущего кадра. Отсюда изменения скоростей движения тел:

$$m_1(\vec{v}_1' - \vec{v}_1) = -m_2(\vec{v}_2' - \vec{v}_2)$$
$$I_1(\vec{\omega}_1' - \vec{\omega}_1) = -I_2(\vec{\omega}_2' - \vec{\omega}_2)$$

Иными словами, тела обмениваются импульсами и моментами импульсов, направленными в противоположные стороны. В реальности при столкновении тел происходит упругий удар. Он описывается при помощи так называемого коэффициента восстановления. Коэффициент восстановления в теории удара является величиной, определяющей, какая доля начальной относительной скорости тел восстанавливается к концу удара. Для тройки параметров $T = (\vec{P}, \vec{N}, D)$ он равен отношению относительной скорости объектов вдоль нормали $\vec{N}$ в точке $\vec{P}$ после удара к относительной скорости до удара, взятому со знаком минус (поскольку нормальная составляющая вектора относительной скорости при ударе изменяет направление):

$$e = -\frac{\vec{v}_{отн}'^{P} \cdot \vec{N}}{\vec{v}_{отн}^{P} \cdot \vec{N}},$$

Отсюда:

$$(\vec{v}_1'^{P} - \vec{v}_2'^{P}) \cdot \vec{N} = -e(\vec{v}_1^{P} - \vec{v}_2^{P}) \cdot \vec{N}, \qquad (5)$$

где $\vec{v}_1^{P} = \vec{v}_1 + [\vec{\omega}_1, \vec{r}_1]$, $\vec{v}_2^{P} = \vec{v}_2 + [\vec{\omega}_2, \vec{r}_2]$ – скорости движения объектов в точке $\vec{P}$ до удара, $\vec{v}_1'^{P}$, $\vec{v}_2'^{P}$ – после удара, а $\vec{r}_1 = \vec{P} - \vec{c}_1$ и $\vec{r}_2 = \vec{P} - \vec{c}_2$ – радиус-векторы от центров масс $\vec{c}_1$ и $\vec{c}_2$ столкнувшихся объектов до точки $\vec{P}$. Значение $e = 0$ характеризует абсолютно неупругий удар, а $e = 1$ – абсолютно упругий. Уравнение удара (5) описывает столкновение пары объектов, но не подходит при описании множественных одновременных столкновений [10]. На практике в случае удара ограничение строят на том основании, что после удара тела должны разлетаться со скоростями не меньшими, чем те, что получаются в случае их одиночного столкновения [11]:

$$(\vec{v}_1'^{P} - \vec{v}_2'^{P}) \cdot \vec{N} + e(\vec{v}_1^{P} - \vec{v}_2^{P}) \cdot \vec{N} \geq 0$$

По аналогии с выводом (3) из (2) получаем ограничение удара:

$$\dot{C}' = J\vec{V}' + eJ\vec{V} \geq 0, \qquad (6)$$

где $\vec{V}'$ – вектор обобщенных скоростей двух объектов после удара.

Как уже было сказано, обработка столкновений двух объектов заключается в формировании реакции объектов на коллизию, при которой они изменяют свои положения и ориентации и скорости. Скорости изменяются под действием силы удара, действующей вдоль нормали $\vec{N}$, а так же под действием сил трения в тангенциальной плоскости. Коррекция положения и ориентации объектов связана с тем, что при использовании апостериорных методов определения коллизий происходит взаимное проникновение объектов между шагами моделирования ввиду дискретности вычислений, когда каждый шаг просчета динамики осуществляется через некоторый промежуток времени Δt.

При упругом ударе ($e \neq 0$) условимся пренебрегать действием силы трения. Взаимное проникновение объектов исчезнет в процессе их последующего разлета после удара. Поэтому для моделирования упругого удара достаточно обеспечить выполнение ограничения (6).

При контакте ($|\dot{C}| < restingContactVelocity$) тел необходимо учитывать действие силы трения, а так же объекты необходимо выталкивать друг из друга для коррекции их взаимного проникновения. Рассмотрим пример, когда один объект проник внутрь другого и скользит вдоль его поверхности (см. рис. 2). Так как первый объект скользит параллельно поверхности второго, то. $\vec{v}_1 \cdot \vec{N} = 0$. Поскольку второе тело не движется, то вектор обобщенных скоростей $\vec{V} = (\vec{v}_1, \vec{0}, \vec{0}, \vec{0})$. Согласно (3) получаем $\dot{C} = J\vec{V} = \vec{v}_1 \cdot \vec{N} = 0$, т.е. объекты находятся в состоянии контакта (взаимное проникновение не изменяется). В этом случае для моделирования реальной картины

необходимо препятствовать дальнейшему проникновению объектов друг в друга (например, под действием силы тяжести), устранить их взаимное проникновение и препятствовать проскальзыванию объектов. Для этого составляются ограничения на контакт, выталкивание и силу трения соответственно.

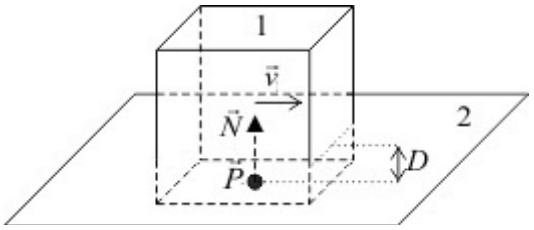


Рис. 2. Скольжение 1-го объекта по поверхности 2-го

Ограничение на контакт составляется на основании запрета дальнейшего проникновения объектов друг в друга, т.е. вводится ограничение на относительную скорость их движения в точке $\vec{P}$ вдоль нормали $\vec{N}$:

$$\dot{C}' = (\vec{v}_1'^{P} - \vec{v}_2'^{P}) \cdot \vec{N} \geq 0$$

Опять по аналогии вывода (3) из (2) получаем ограничение контакта:

$$\dot{C}' = J\vec{V}' \geq 0\,, \qquad (7)$$

где $\vec{V}'$ – вектор новых обобщенных скоростей двух объектов после обработки контакта.

Выталкивание объектов друг из друга реализуется через так называемые псевдоскорости. Псевдоскорости – искусственно введенные мгновенные скорости объектов. Они не равны нулю только в том случае, когда объекты выталкиваются друг из друга. Чтобы вытолкнуть объекты друг из друга, необходимо сообщить им такую относительную псевдоскорость, которая бы устранила их взаимное проникновение в точке $\vec{P}$ на глубину D вдоль нормали $\vec{N}$ за время моделирования Δt. Выталкивание объектов друг из друга аналогично расталкиванию объектов (удар и контакт) происходит вдоль нормали $\vec{N}$. Поэтому по аналогии с (3) их относительная псевдоскорость характеризуется величиной $J\vec{V}_{ps}$. Тогда:

$$J\vec{V}_{ps}' = \frac{D}{\Delta t},$$

где $\vec{V}_{ps}'$ – вектор полученных объектами обобщенных псевдоскоростей. Придание такой относительной псевдоскорости $J\vec{V}_{ps}'$ двум объектам вытолкнет тела друг из друга за время моделирования Δt. На практике в таком виде псевдоскорости применяются редко. Дело в том, что мгновенное выталкивание тел за один шаг моделирования может приводить к их резким скачкам при высокой степени влияния тел друг на друга (например, в случае, когда множество тел покоится друг на друге). Поэтому дополнительно вводят ERP (Error Reduction Parameter) – параметр уменьшения ошибки. Смысл ERP заключается в том, насколько быстро будет устраняться взаимное проникновение тел. ERP изменяется в интервале $(0,1]$ и устанавливается опытным путем (обычно значения ERP устанавливают в пределах $[0.6, 0.8]$). Тогда необходимая для выталкивания относительная псевдоскорость пары объектов равна:

$$J\vec{V}_{ps}' = ERP \cdot \frac{D}{\Delta t} \qquad (8)$$

Как и в случае с ударом, формулировка необходимой псевдоскорости описывает выталкивание пары объектов, но не подходит при описании множественных одновременных выталкиваний. Поэтому на практике при составлении ограничения на выталкивание применяют неравенство по аналогии с (6). Тогда ограничение на выталкивание имеет вид:

$$\dot{C}' = J\vec{V}_{ps}' - ERP \cdot \frac{D}{\Delta t} \geq 0 \qquad (9)$$

Как известно, сила трения препятствует относительному движению объектов в плоскости их соприкосновения. Существуют два вида силы трения: статическая и динамическая. Первая препятствует началу относительного движения объектов из состояния покоя, вторая действует на уже движущиеся соприкасающиеся объекты. По закону Кулона-Амонтона $F_{fr} = \mu N$, где μ – коэффициент трения, N – сила реакции опоры. При этом коэффициенты статического и динамического трения не равны (а следовательно не равны и силы трения), но этим фактом обычно пренебрегают. Сила трения направлена против относительного движения объектов в плоскости соприкосновения. Поскольку нормаль $\vec{N}$ перпендикулярна этой плоскости, тангенциальная составляющая $\vec{v}_{отн\_N\perp}^{P}$ относительной скорости движения двух объектов в точке $\vec{P}$ равна $\vec{v}_{отн\_N\perp}^{P} = \vec{v}_{отн}^{P} - (\vec{v}_{отн}^{P} \cdot \vec{N})\vec{N}$. Обозначим направление движения тел в тангенциальной плоскости (плоскости соприкосновения) единичным вектором $\vec{t} = \vec{v}_{отн\_N\perp}^{P} \big/ |\vec{v}_{отн\_N\perp}^{P}|$. Тогда по аналогии с заменами в (3) тангенциальная составляющая относительной скорости движения объектов в точке $\vec{P}$ характеризуется величиной $J_{fr}\vec{V}$, где $J_{fr} = \left(\vec{t}, [\vec{r}_1, \vec{t}], -\vec{t}, -[\vec{r}_2, \vec{t}]\right)$. В случае статического трения тела должны перестать проскальзывать относительно друг друга в плоскости соприкосновения. То есть ограничение для статического трения имеет вид:

$$\dot{C}' = J_{fr}\vec{V}' = 0\,, \qquad (10)$$

где $\vec{V}'$ – вектор новых обобщенных скоростей двух объектов после обработки силы трения. В случае динамического трения известно лишь то, что объекты под действием силы трения уменьшат относительную скорость движения в плоскости соприкосновения. В данном случае ограничение на скоростях можно описать только в виде:

$$\dot{C}' = J_{fr}\vec{V}' \neq 0$$

Знак ограничения динамического трения зависит от направления действия силы трения относительно оси $\vec{t}$. Поскольку сила трения действует противоположно относительной скорости движения, то знак ограничения будет обратным по отношению к знаку

силы трения. Ограничение на скоростях для силы трения в общем виде можно выписать только используя непосредственно импульс самой силы трения, изменяющий скорости скольжения объектов. Сделаем это далее при описании постановки задачи с использованием не только скоростей объектов, но и применяемых импульсов.

Задачей импульсного солвера является расчет таких импульсов, которые нужно применить к двум столкнувшимся телам в точке $\vec{P}$, чтобы выполнялись все описанные ограничения для каждой такой тройки $T = (\vec{P}, \vec{N}, D)$ параметров столкновения двух объектов.

3. Импульсный солвер

Ограничения, основанные на скоростях, так же, как ограничения, основанные на ускорениях, могут быть переформулированы в виде LCP по аналогии с (4). Для этого запишем второй закон Ньютона в импульсном виде для случая только поступательного движения для пары столкнувшихся объектов, к которым применяется некоторый импульс $J^T\lambda$ (вывод LCP для общего случая поступательного и вращательного движения для каждого вида ограничений оставим на главы подробного разбора каждого из ограничений):

$$M(\vec{V}' - \vec{V}) = J^T\lambda + \vec{f}_{ext}\Delta t,$$

где M – диагональная матрица масс двух объектов, $\vec{V}$ и $\vec{V}'$ – вектора обобщенных скоростей до и после применения импульса $J^T\lambda$, $\vec{f}_{ext}$ – суммарная внешняя сила, действующая на объекты. Выразим вектор новых обобщенных скоростей двух тел после применения импульса $J^T\lambda$:

$$\vec{V}' = M^{-1}J^T\lambda + M^{-1}\vec{f}_{ext}\Delta t + \vec{V}$$

Подстановкой вектора $\vec{V}'$ в ограничения удара (6), контакта (7) и выталкивания (9) можно получить формулировку задачи в виде LCP. Распишем подробнее формулировку LCP на примере ограничения удара (6).

$$\dot{C}' = J\vec{V}' + eJ\vec{V} = JM^{-1}J^T\lambda + JM^{-1}\vec{f}_{ext}\Delta t + J\vec{V} + eJ\vec{V}$$

Сделаем замены $A = JM^{-1}J^T$ и $b = JM^{-1}\vec{f}_{ext}\Delta t + J\vec{V} + eJ\vec{V}$. Тогда LCP удара после применения импульса λ:

$$\dot{C}' = A\lambda + b \qquad (11)$$

При этом:

- $\dot{C}' \geq 0$ по условию ограничения;
- $\lambda \geq 0$ так как импульсы от удара должны действовать только в направлении, соответствующем разрешению коллизии;
- $\dot{C}'\lambda = 0$ так как импульс действует только до того момента, пока объекты движутся в направлении увеличения их взаимного проникновения.

Рассмотрим происхождения последнего условия $\dot{C}'\lambda = 0$ подробнее. Текущую скорость ограничения характеризует $\dot{C}$ из (3). В случае, если $\dot{C} < 0$, рассчитывается такой импульс λ, применение которого делает $\dot{C}' = 0$. Если же $\dot{C} \geq 0$, то ограничение уже выполнено, поэтому никакой импульс применять не надо и $\lambda = 0$. Поэтому $\dot{C}'\lambda = 0$.

LCP для контакта и выталкивания выводится аналогичным образом и имеет тот же вид, что и (11), только для контакта $b = JM^{-1}\vec{f}_{ext}\Delta t + J\vec{V}$, а для выталкивания $b = JM^{-1}\vec{f}_{ext}\Delta t + J\vec{V} - ERP \cdot D/\Delta t$.

Аналогичное представление ограничения силы трения подстановкой $\vec{V}'$ в ограничение (10) соответствует значению $b = JM^{-1}\vec{f}_{ext}\Delta t + J\vec{V}$. При этом ограничение силы трения нельзя переформулировать в виде LCP, потому что в случае с силой трения на импульс λ накладывается более сложное условие, чем просто $\lambda \geq 0$. Максимальное значение импульса ограничено импульсом максимальной силы трения. Т.е. $|\lambda| <= \lambda_{fr}$, где $\lambda_{fr} = F_{fr}\Delta t = \mu N \Delta t$. Поэтому ограничение на силу трения формулируется в виде так называемого смешанного LCP (MLCP – mixed LCP):

$$\begin{cases} \dot{C}' = A\lambda + b \\ \begin{bmatrix} -\lambda_{fr} < \lambda < \lambda_{fr}, \dot{C}' = 0 \\ \lambda = -\lambda_{fr}, \dot{C}' > 0 \\ \lambda = \lambda_{fr}, \dot{C}' < 0 \end{bmatrix} \end{cases} \qquad (12)$$

Решение MLCP для трения заключается в начальном поиске импульса λ, при котором текущая скорость ограничения обнуляется, т.е. $\dot{C}' = 0$. Если этот импульс попадает в границы максимальных импульсов силы трения, то он применятся и тогда $\dot{C}' = 0$. Если нет, то λ устанавливается в пограничное значение и применяется.

Таким образом, для одной тройки $T = (\vec{P}, \vec{N}, D)$ ограничения на удар, контакт и выталкивание формулируются и решаются в виде LCP, а для трения формулируются и решаются в виде MLCP.

Для одной тройки $T = (\vec{P}, \vec{N}, D)$ пары столкнувшихся объектов общий вид солвера имел бы вид:

Если $|\dot{C}|$ > *restingContactVelocity*, то

Вычисление и применение импульса удара

Иначе

Вычисление и применение импульса контакта

Вычисление и применение импульса выталкивания

Вычисление и применение импульса трения

Но в действительности же пара объектов может иметь несколько точек столкновения $\vec{P}$ одновременно (на рис. 2 это могут быть, например, 4 вершины нижней грани первого объекта) или же несколько тел могу сталкиваться одновременно. Соответственно может существовать множество различных ограничений одновременно. Более того эти ограничения могут быть взаимосвязаны. Например, в случае, когда множество объектов лежат друг на друге. Все ограничения будут представлять собой систему из n LCP и MLCP. В упомянутом примере покоя

множества тел друг на друге n может доходить до нескольких тысяч. Прямое решение такой системы заняло бы в этом случае очень длительное время, что сказалось бы на общем времени Δt моделирования одного шага просчета динамики, а соответственно на стабильности системы. Поэтому на практике применяют итеративные способы решения таких систем. Одним из таких методов является проекционный метод Гаусса-Зейделя (Projected Gauss-Seidel). Идея состоит с том, что LCP и MLCP решаются последовательно друг за другом. Для каждого LCP и MLCP вычисляется импульс $\Delta\lambda$, необходимый для выполнения рассматриваемого ограничения. Импульс $\Delta\lambda$ применяется к двум телам согласно рассматриваемому ограничению, за счет чего тела изменяют свои текущие скорости движения. Таким образом, изменяются начальные условия (импульс $\Delta\lambda$ учитывается посредством изменения скоростей) для остальных ограничений, в которых могут участвовать эти тела. По мере повторения такого процесса для всей системы, получаемые скорости объектов приближаются к искомым значениям, удовлетворяющим решению всей системы (такой процесс называют сходящимся). Проекционный метод Гаусса-Зейделя, применяемый в импульсных солверах, называется методом последовательных импульсов. Стоит отметить, что вычисляемые импульсы $\Delta\lambda$ составляют в этом случае только часть от суммарных применяемых импульсов λ. Если для LCP или MLCP нарушается условие на импульс λ, то импульс $\Delta\lambda$ корректируется так, чтобы выполнить условие на суммарный импульс. Такую коррекцию рассмотрим позднее при подробном разборе расчета каждого вида применяемых импульсов $\Delta\lambda$.

Таким образом, общий вид солвера для всех n ограничений системы приобретет вид:

Пока НЕ (условие выхода)
  Цикл по i от 0 до n
    Если $|\dot{C}[i]|>$ *restingContactVelocity*, то
      Вычисление и применение импульса удара
    Иначе
      Вычисление и применение импульса контакта
      Вычисление и применение импульса выталкивания
      Вычисление и применение импульса трения
  Конец цикла

Чем больше итераций выполняется в солвере, тем ближе получаемые в нем значения к истинным. Условием выхода (критерием завершения расчетов в солвере) могут быть:

– общее количество итераций солвера. Экспериментальным методом можно подобрать такое общее количество итераций солвера, которого будет всегда достаточно для получения реалистичной визуальной картинки даже при большом числе n в взаимосвязанных ограничений в системе (например, в случае покоя множества тел друг на друге);

– малость текущих применяемых импульсов во всех структурных блоках солвера. Все импульсы λ_i, применяемые в блоках удара, контакта, трения и псевдоимпульсов по модулю меньше какой-то малой величины (см. далее расчет импульсов λ во всех блоках солвера);

– общее время расчетов в солвере.

Рассмотрим подробно вычисление импульсов удара, контакта, выталкивания и трения одной тройки $T=(\vec{P},\vec{N},D)$ параметров коллизии. Для всех троек параметров всех коллизий системы метод вычисления и применения импульсов будет идентичным. Различаться будут лишь исходные данные, которыми будут служить характеристики объектов, а именно: векторы текущих линейных скоростей ($\vec{v}_1$ и $\vec{v}_2$), векторы текущих угловых скоростей ($\vec{\omega}_1$ и $\vec{\omega}_2$), массы (m_1 и m_2), матрицы тензоров инерции (I_1 и I_2), векторы положения центров масс ($\vec{c}_1$ и $\vec{c}_2$). Векторы и матрицы задаются в мировой системе координат. Напомним, что при использовании методов последовательных импульсов будут вычисляться импульсы $\Delta\lambda$, составляющие часть суммарного импульса λ.

3.1. Вычисление импульса удара

Выведем LCP удара в общем виде с учетом поступательного и вращательного движения. При ударе к столкнувшимся телам применяется импульс $\vec{p}$ в точке $\vec{P}$, удовлетворяющий закону удара (5). Как уже было описано, при ударе тела обмениваются импульсами, направленными в противоположные стороны. Поскольку нормаль $\vec{N}$ направлена из второго тела в первое, то первое тело надо расталкивать в направлении $\vec{N}$, а второе – в направлении $-\vec{N}$. Обозначив через $\Delta\lambda$ величину искомого импульса, получим, что для первого тела импульс равен $\vec{p}=\Delta\lambda\vec{N}$, а для второго – $\vec{p}=-\Delta\lambda\vec{N}$. Из формул $\vec{p}=m\vec{v}$, $\vec{L}=[\vec{r},\vec{p}]$, $\vec{L}=I\vec{\omega}$ рассчитаем линейные и угловые скорости движения двух тел после приложения к ним противоположно направленных импульсов $\vec{p}$:

$$\vec{v}_1'=\vec{v}_1+\frac{\vec{p}}{m_1}=\vec{v}_1+\frac{\Delta\lambda\vec{N}}{m_1} \qquad \vec{v}_2'=\vec{v}_2+\frac{-\vec{p}}{m_2}=\vec{v}_2+\frac{-\Delta\lambda\vec{N}}{m_2}$$

$$\vec{\omega}_1'=\vec{\omega}_1+\mathrm{I}_1^{-1}[\vec{r}_1,\vec{p}]=\vec{\omega}_1+\mathrm{I}_1^{-1}[\vec{r}_1,\Delta\lambda\vec{N}]$$

$$\vec{\omega}_2'=\vec{\omega}_2+\mathrm{I}_2^{-1}[\vec{r}_2,-\vec{p}]=\vec{\omega}_2+\mathrm{I}_2^{-1}[\vec{r}_2,-\Delta\lambda\vec{N}],$$

где $\vec{r}_1=\vec{P}-\vec{c}_1$ и $\vec{r}_2=\vec{P}-\vec{c}_2$.

Прежде, чем подставлять значения новых линейных и угловых скоростей тел в уравнение удара (5), произведем его преобразование, воспользовавшись свойством смешанного произведения $([\vec{a},\vec{b}],\vec{c})=([\vec{c},\vec{a}],\vec{b})=([\vec{b},\vec{c}],\vec{a})$:

$$(\vec{v}_1'^{P}-\vec{v}_2'^{P})\cdot\vec{N}+e(\vec{v}_1^{P}-\vec{v}_2^{P})\cdot\vec{N}=$$

$$=(\vec{v}_1'+[\vec{\omega}_1',\vec{r}_1]-\vec{v}_2'-[\vec{\omega}_2',\vec{r}_2])\cdot\vec{N}+$$

$$+e(\vec{v}_1+[\vec{\omega}_1,\vec{r}_1]-\vec{v}_2-[\vec{\omega}_2,\vec{r}_2])\cdot\vec{N}=$$
$$=\vec{N}\cdot\vec{v}_1'+[\vec{r}_1,\vec{N}]\cdot\vec{\omega}_1'-\vec{N}\cdot\vec{v}_2'-[\vec{r}_2,\vec{N}]\cdot\vec{\omega}_2'+$$
$$+e(\vec{N}\cdot\vec{v}_1+[\vec{r}_1,\vec{N}]\cdot\vec{\omega}_1-\vec{N}\cdot\vec{v}_2-[\vec{r}_2,\vec{N}]\cdot\vec{\omega}_2)=0$$

Теперь подставим значения новых линейных и угловых скоростей тел после применения импульса $\vec{p}$:

$$\vec{N}\cdot(\vec{v}_1+\frac{\Delta\lambda\vec{N}}{m_1})+[\vec{r}_1,\vec{N}]\cdot(\vec{\omega}_1+\mathrm{I}_1^{-1}[\vec{r}_1,\Delta\lambda\vec{N}])-$$
$$-\vec{N}\cdot(\vec{v}_2+\frac{-\Delta\lambda\vec{N}}{m_2})-[\vec{r}_2,\vec{N}]\cdot(\vec{\omega}_2+\mathrm{I}_2^{-1}[\vec{r}_2,-\Delta\lambda\vec{N}])+$$
$$+e(\vec{N}\cdot\vec{v}_1+[\vec{r}_1,\vec{N}]\cdot\vec{\omega}_1-\vec{N}\cdot\vec{v}_2-[\vec{r}_2,\vec{N}]\cdot\vec{\omega}_2)=0$$

Сделаем замены $\vec{c}_1=\vec{N}$, $\vec{c}_2=-\vec{N}$, $\vec{d}_1=[\vec{r}_1,\vec{c}_1]$ и $\vec{d}_2=[\vec{r}_2,\vec{c}_2]$:

$$\vec{c}_1\cdot(\vec{v}_1+\frac{\Delta\lambda\vec{c}_1}{m_1})+\vec{d}_1\cdot(\vec{\omega}_1+\Delta\lambda\,\mathrm{I}_1^{-1}\vec{d}_1)+$$
$$+\vec{c}_2\cdot(\vec{v}_2+\frac{\Delta\lambda\vec{c}_2}{m_2})+\vec{d}_2\cdot(\vec{\omega}_2+\Delta\lambda\,\mathrm{I}_2^{-1}\vec{d}_2)+$$
$$+e(\vec{c}_1\cdot\vec{v}_1+\vec{d}_1\cdot\vec{\omega}_1+\vec{c}_2\cdot\vec{v}_2+\vec{d}_2\cdot\vec{\omega}_2)=$$
$$=\vec{\mathrm{c}}_1\cdot\vec{v}_1+\Delta\lambda\cdot\frac{\vec{\mathrm{c}}_1\cdot\vec{\mathrm{c}}_1}{m_1}+\vec{d}_1\cdot\vec{\omega}_1+\Delta\lambda\vec{d}_1\cdot\mathrm{I}_1^{-1}\vec{d}_1+$$
$$\vec{\mathrm{c}}_2\cdot\vec{v}_2+\Delta\lambda\cdot\frac{\vec{\mathrm{c}}_2\cdot\vec{\mathrm{c}}_2}{m_2}+\vec{d}_2\cdot\vec{\omega}_2+\Delta\lambda\vec{d}_2\cdot\mathrm{I}_2^{-1}\vec{d}_2+$$
$$+e(\vec{c}_1\cdot\vec{v}_1+\vec{d}_1\cdot\vec{\omega}_1+\vec{c}_2\cdot\vec{v}_2+\vec{d}_2\cdot\vec{\omega}_2)$$

Поскольку нормаль контакта задается единичным (по условию) вектором $\vec{N}$, то $(\vec{\mathrm{c}}_1,\vec{\mathrm{c}}_1)=(\vec{\mathrm{c}}_2,\vec{\mathrm{c}}_2)=1$. Сделаем замену $a=\frac{1}{m_1}+\vec{d}_1\cdot\mathrm{I}_1^{-1}\vec{d}_1+\frac{1}{m_2}+\vec{d}_2\cdot\mathrm{I}_2^{-1}\vec{d}_2$ и $b=(\vec{\mathrm{c}}_1,\vec{v}_1)+(\vec{d}_1,\vec{\omega}_1)+(\vec{\mathrm{c}}_2,\vec{v}_2)+(\vec{d}_2,\vec{\omega}_2)$. Тогда закон удара с учетом применяемого импульса $\Delta\lambda$ будет иметь вид:

$$a\Delta\lambda+(e+1)b=0$$

Отсюда импульс удара равен:

$$\Delta\lambda=-\frac{(1+e)\cdot b}{a}$$

С учетом корректировки текущего применяемого импульса $\Delta\lambda$ так, чтобы суммарный импульс $\lambda=\sum\Delta\lambda\geq 0$, итоговое вычисление импульса удара имеет вид:

$$\Delta\lambda=-\frac{(1+e)\cdot b}{a}$$
$$\lambda=\lambda+\Delta\lambda \qquad (13)$$
Если $\lambda<0$, то
$$\Delta\lambda=\Delta\lambda-\lambda$$
$$\lambda=0$$

Стоит отметить, что математически более точно перед началом всех итераций посчитать величину eb из (13), так как это величина, которая характеризует скорость объектов после удара (правая часть уравнения удара (5)). Она зависит только от начальных условий, но значения b по мере обработки всех ограничений, в которых участвуют данные тела, может изменяться. Так же значение b может изменяться по мере обработки рассматриваемого ограничения в итерациях в солвере.

После применения импульса $\Delta\lambda$ скорость двух столкнувшихся объектов (текущая скорость рассматриваемого ограничения удара) будет равна:

$$\dot{C}=a\Delta\lambda+(e+1)b=0$$

Возможна ситуация, когда обработка в рамках текущей итерации метода последовательных импульсов других ограничений, в которых участвуют рассматриваемые объекты, нарушит условие $\dot{C}=0$. Если на следующей итерации станет $\dot{C}\geq 0$, то тела итак будут двигаться в сторону разрешения коллизии, поэтому будет $\Delta\lambda=0$. Если же получится $\dot{C}<0$, то снова будет считаться импульс $\Delta\lambda$, необходимый для восстановления условия $\dot{C}=0$. И так далее до достижения одного из критериев выхода из итераций. Тогда итоговое значение скорости ограничения удара в конце итераций будет равно:

$$\dot{C}'=a\lambda+(e+1)b\geq 0,$$

что соответствует LCP удара, расписанному с учетом применяемых импульсов из ограничения удара (6).

3.2. Вычисление импульса контакта

Контакт двух объектов характеризуется отсутствием движения объектов в сторону увеличения проникновения, т.е. текущая скорость ограничения $|\dot{C}|<restingContactVelocity$. Но когда объекты переходят из состояния удара в состояние контакта? Пусть объект падает на землю под действием силы тяжести. При этом согласно (5) при каждом столкновении с землей скорость объекта будет уменьшаться пропорционально e $(e\neq 1)$. В какой-то момент скорость тела станет маленькой и выполнится $|\dot{C}|<restingContactVelocity$. *restingContactVelocity* не должна быть маленькой, иначе возможна ситуация, когда $|\dot{C}|>restingContactVelocity$ будет всегда и переход от удара к контакту осуществляться не будет. Связано это с тем, что за шаг моделирования объекты получают мгновенный прирост скорости под действием внешних сил. Например, под действием силы тяжести (самой большой постоянной внешней силы при моделировании динамики твердых тел) прирост скорости будет $\Delta\vec{v}=-\vec{g}\Delta t$. При $\Delta t=40мс$ (максимальное время просчета одного шага, при котором моделирование осуществляется в реальном времени – 25 кадров в секунду) прирост скорости по модулю равен $|\Delta\vec{v}|\approx 10\cdot 0,04=0,4м/с$. Если выбрать $restingContactVelocity<0.4м/с$, то очевидно, что в этом случае переход от удара к контакту осуществляться не будет. То есть тело, лежащее на земле, будет постоянно получать импульс удара и изменять направление своей скорости на противоположное пропорционально коэффициенту восстановления e. Это приведет к тому, что покоящиеся тела будут постоянно дрожать. Выбор $restingContactVelocity>0.4м/с$ гарантирует переход

от состояния удара к контакту даже при максимальном времени $\Delta t = 40мс$ моделирования в реальном времени. Наглядным критерием выбора *restingContactVelocity* является выбор высоты падения h (пути, который «пройдет» объект под действием силы тяжести). Из физики свободного падения известно, что при падении с высоты h, тело развивает конечную скорость $\vec{v} = \sqrt{2\vec{g}h}$. Выбрав высоту $h = 1см = 0.01м$, при котором падение тела визуально будет незаметным, получим конечную скорость $\vec{v} \approx \sqrt{2 \cdot 10 \cdot 0.01} = 0.44м/с$. Такое значение удовлетворяет условию, описанному выше, поэтому зададим $restingContactVelocity = 0.44м/с$. Тогда критерий перехода от удара к контакту будет:

$$|\dot{C}| < 0.44$$

Итак, известно, что тела хоть и движутся относительно друг друга вдоль нормали коллизии $\vec{N}$, но с очень маленькой скоростью. Тогда рассчитаем такой импульс контакта $\vec{p}_к = \Delta\lambda_c \vec{N}$, который обнулит эту скорость. Такой импульс будет соответствовать импульса удара двух тел с коэффициентом восстановления ноль. Подставив $e = 0$ в (13), получим выражение для вычисления импульса контакта:

$$\begin{aligned} &\Delta\lambda_c = -\frac{b}{a} \\ &\lambda_c = \lambda_c + \Delta\lambda_c \\ &\text{Если } \lambda_c < 0 \text{, то} \\ &\quad \Delta\lambda_c = \Delta\lambda_c - \lambda_c \\ &\quad \lambda_c = 0, \end{aligned} \qquad (14)$$

где a и b совпадают с (13). Применение посчитанных в рамках итераций солвера импульсов $\Delta\lambda_c$ будет соответствовать применению суммарного импульса контакта $\lambda_c = \sum \Delta\lambda_c$. Тогда итоговое значение скорости ограничения контакта в конце итераций будет равно:

$$\dot{C}' = a\lambda_c + b \geq 0,$$

что соответствует LCP контакта, расписанному с учетом применяемых импульсов из ограничения контакта (7).

После решения LCP контакта объекты перестанут двигаться относительно друг друга вдоль нормали коллизии $\vec{N}$. Но в этом случае объекты все еще будут проникать друг в друга на глубину D вдоль нормали коллизии $\vec{N}$. За коррекцию их взаимного проникновения отвечают так называемые псевдоимпульсы (импульсы выталкивания).

3.3. Вычисление импульса выталкивания

Идея псевдоимпульсов в том, что, если два объекта проникают друг в друга на глубину D вдоль нормали $\vec{N}$, то их нужно вытолкнуть друг из друга так, чтобы они перестали пересекаться. При этом, как уже было сказано, псевдоимпульсы не изменяют скорости тел, а влияют только на так называемые псевдоскорости. То есть по сути псевдоимпульсы посредством передачи через псевдоскорости изменяют только положение и ориентацию объектов на текущем шаге моделирования. Этот механизм разделения скоростей от импульсов удара/контакта и псевдоимпульсов получил название разделенных импульсов. Вывод псевдоимпульса аналогичен выводу импульса удара только с использованием вместо закона удара (5) условия выталкивания (8). Поэтому опустим подробное описание, а выпишем итоговое выражение для псевдоимпульса:

$$\Delta\lambda_{ps} = \frac{-b + ERP \cdot D/\Delta t}{a},$$

где изменяется только значение $b = J\vec{V}_{ps} = (\vec{c}_1, \vec{v}_{1\_ps}) + (\vec{d}_1, \vec{\omega}_{1\_ps}) + (\vec{c}_2, \vec{v}_{2\_ps}) + (\vec{d}_2, \vec{\omega}_{2\_ps})$, где $\vec{V}_{ps} = (\vec{v}_{1\_ps}, \vec{\omega}_{1\_ps}, \vec{v}_{2\_ps}, \vec{\omega}_{2\_ps})$ – вектор обобщенных псевдоскоростей. Коррекция применяемого псевдоимпульса для учета условия неотрицательности суммарного псевдоимпульса аналогична описанным ранее импульсам удара и контакта и имеет вид:

$$\begin{aligned} &\Delta\lambda_{ps} = \frac{-b + ERP \cdot D/\Delta t}{a} \\ &\lambda_{ps} = \lambda_{ps} + \Delta\lambda_{ps} \\ &\text{Если } \lambda_{ps} < 0 \text{, то} \\ &\quad \Delta\lambda_{ps} = \Delta\lambda_{ps} - \lambda_{ps} \\ &\quad \lambda_{ps} = 0 \end{aligned}$$

LCP выталкивания с учетом применения суммарного псевдоимпульса имеет вид:

$$\dot{C}'_{ps} = a\lambda_{ps} + b - ERP\frac{D}{\Delta t} \geq 0$$

3.4. Вычисление импульса трения

Напомним, что ограничение на силу трения формулируется в виде MLCP (12). Разберем его подробнее. При моделировании силы трения применяют подход, при котором вначале рассчитывается сила F_{fr}^{∞}, которую нужно приложить, чтобы прекратить относительное движение объектов в плоскости соприкосновения (часто ее называют бесконечной силой трения). Поскольку данная статья посвящена рассмотрению ограничений на скоростях, то считается импульс бесконечной силы трения $\lambda_{fr}^{\infty} = F_{fr}^{\infty}\Delta t$. Для нахождения импульса бесконечной силы трения воспользуемся ограничением (10) силы трения покоя. Заметим, что выполнение ограничения (10) соответствует выполнению ограничения (7) контакта, переписанного в виде уравнения. На уровне импульсов переход от уравнения к неравенству означает, что если суммарный импульс контакта ограничивался $\lambda_c \geq 0$, то для суммарного импульса бесконечной силы трения λ_{fr}^{∞} ограничения на его допустимые значения отсутствуют. То есть, если суммарный импульс контакта препятствует движению объектов вдоль нормали $\vec{N}$ только в сторону дальнейшего проникновения, то для

суммарного импульса бесконечной силы трения движение запрещается вдоль оси трения в обе стороны. Как было показано при выводе ограничения (10) ось трения задается вектором $\vec{t} = \vec{v}^P_{отн\_N\perp} / |\vec{v}^P_{отн\_N\perp}|$, где $\vec{v}^P_{отн\_N\perp} = \vec{v}^P_{отн} - (\vec{v}^P_{отн} \cdot \vec{N})\vec{N}$. Обозначим рассчитываемую часть суммарного импульса бесконечной силы трения как $\Delta\lambda_{fr}$. Учитывая описанное выше и выражение (14) для импульса контакта, импульс бесконечной силы трения равен:

$$\Delta\lambda_{fr} = -\frac{b}{a}, \qquad (15)$$

где значение a совпадает с (14), а для $b = (\vec{c}_1, \vec{v}_1) + (\vec{d}_1, \vec{\omega}_1) + (\vec{c}_2, \vec{v}_2) + (\vec{d}_2, \vec{\omega}_2)$ $\vec{d}_1$ и $\vec{d}_2$ так же равны $\vec{d}_1 = [\vec{r}_1, \vec{c}_1]$, $\vec{d}_2 = [\vec{r}_2, \vec{c}_2]$, но $\vec{c}_1 = \vec{t}$ и $\vec{c}_2 = -\vec{t}$. Применение суммарного импульса бесконечной силы трения $\lambda^\infty_{fr} = \sum \Delta\lambda_{fr}$ будет аналогично применению бесконечной силы трения – какими бы не были входные параметры, тела никогда не будут проскальзывать друг относительно друга в точке $\vec{P}$ вдоль оси $\vec{t}$. Но сила трения является конечной величиной. Из закона Кулона-Амонтона известно, что $F_{fr} = \mu N$. Величина импульса такой силы трения равна $F_{fr}\Delta t = \mu N \Delta t$. Сила реакции опоры N при контакте объектов считается через импульс контакта как $N = \lambda_c / \Delta t$. Тогда $F_{fr}\Delta t = \mu N \Delta t = \mu\lambda_c$ – величина максимального импульса трения для двух контактирующих объектов с коэффициентом трения μ. Ограничение по модулю для величины суммарного импульса трения $\lambda_{fr} = \sum \Delta\lambda_{fr}$ можно описать как $|\lambda_{fr}| = \min(|-b/a|, \mu\lambda_c)$. Тогда применяемый импульс трения считается в следующем виде:

$$\Delta\lambda_{fr} = -\frac{b}{a}$$
$$\lambda_{fr} = \lambda_{fr} + \Delta\lambda_{fr}$$
Если $\lambda_{fr} > \mu\lambda_c$, то
$$\Delta\lambda_{fr} = \Delta\lambda_{fr} + \mu\lambda_c - \lambda_{fr}$$
$$\lambda_{fr} = \mu\lambda_c,$$
Иначе Если $\lambda_{fr} < -\mu\lambda_c$
$$\Delta\lambda_{fr} = \Delta\lambda_{fr} - \mu\lambda_c - \lambda_{fr}$$
$$\lambda_{fr} = -\mu\lambda_c$$

На практике сформулированное одноосевое трение используют редко. Это связано с тем, что при решении всех ограничений системы для ударов/контактов и трения (псевдоимульсы сюда не входят, потому что они меняют псевдоскорости, а не реальные скорости объектов) скорость двух тел в точке $\vec{P}$ может изменяться не только вдоль рассматриваемых осей (вдоль $\vec{N}$ и $\vec{t}$ для удара/контакта и трения соответственно), но и в перпендикулярных им направлениях. Это часто приводит к нестабильности системы, особенно когда ограничения в высокой степени связаны друг с другом (например, в случае покоя множества объектов друг на друге). Исходя из этого, применяют двухосевую силу трения, дополнительно учитывая направление $\vec{t}_\perp$, перпендикулярное $\vec{t}$ и $\vec{N}$ ($\vec{t}_\perp = [\vec{N}, \vec{t}]$). Более того, силу трения необходимо учитывать даже в случае, когда в начале итераций солвера объекты не движутся друг относительно друга в точке $\vec{P}$, т.е. $|\vec{v}^P_{отн\_N\perp}| = 0$. В этом случае первую ось трения $\vec{t}$ выбирают из векторного произведения нормали $\vec{N}$ и одного из двух заранее выбранных перпендикулярных между собой единичных векторов $\vec{x}$ и $\vec{y}$ ($\vec{x} \perp \vec{y}$). Алгоритм выбора осей трения следующий:

Если $|\vec{v}^P_{отн\_N\perp}| \neq 0$, то
$$\vec{t} = \vec{v}^P_{отн\_N\perp} / |\vec{v}^P_{отн\_N\perp}|$$
Иначе Если $|[\vec{N}, \vec{x}]| \neq 0$, то
$$\vec{t} = [\vec{N}, \vec{x}] / |[\vec{N}, \vec{x}]|$$
Иначе
$$\vec{t} = [\vec{N}, \vec{y}] / |[\vec{N}, \vec{y}]|$$
Конец Если
$$\vec{t}_\perp = [\vec{N}, \vec{t}]$$

Стоит отметить, что ввиду того, что ограничение на трение распространяется на всю плоскость трения (плоскость, задающуюся векторами $\vec{t}$ и $\vec{t}_\perp$), не важно, в каком порядке брать векторные произведения (не важно, какую тройку векторов правую или левую представляют собой векторы $\vec{N}$, $\vec{t}$ и $\vec{t}_\perp$). Расчет величины применяемого импульса $\Delta\lambda_{fr_\perp}$ и суммарного импульса $\lambda_{fr_\perp}$ трения вдоль оси $\vec{t}_\perp$ аналогичен (15):

$$\Delta\lambda_{fr_\perp} = -\frac{b}{a}$$
$$\lambda_{fr_\perp} = \lambda_{fr_\perp} + \Delta\lambda_{fr_\perp},$$

где a, b, $\vec{d}_1$ и $\vec{d}_2$ совпадают с (15), а $\vec{c}_1 = \vec{t}_\perp$ и $\vec{c}_2 = -\vec{t}_\perp$. Тогда суммарный импульс трения вдоль двух осей $\vec{t}$ и $\vec{t}_\perp$:

$$\vec{p}_{fr} = \lambda_{fr}\vec{t} + \lambda_{fr_\perp}\vec{t}_\perp$$

В случае с одноосевым трением ограничение суммарного импульса трения λ_{fr} обеспечивалось коррекцией одного применяемого импульса $\Delta\lambda_{fr}$. В случае двухосевого трения ограничивать надо сумму двух суммарных импульсов λ_{fr} и $\lambda_{fr_\perp}$ коррекцией двух применяемых импульсов $\Delta\lambda_{fr}$ и $\Delta\lambda_{fr_\perp}$. В связи с этим вычисление применяемых импульсов $\Delta\lambda_{fr}$ и $\Delta\lambda_{fr_\perp}$ и выполнение условия на ограничение суммарного импульса трения усложняется:

$$\Delta\lambda_{fr} = -\frac{b}{a}$$
$$\lambda_{fr} = \lambda_{fr} + \Delta\lambda_{fr}$$
$$\Delta\lambda_{fr_\perp} = -\frac{b}{a}$$
$$\lambda_{fr_\perp} = \lambda_{fr_\perp} + \Delta\lambda_{fr_\perp}$$

Если $|\vec{p}_{fr}| = \sqrt{\lambda_{fr}^2 \vec{t}^{\,2} + \lambda_{fr_\perp}^2 \vec{t}_\perp^{\,2}} > \mu\lambda_c$, то

Если $|\vec{p}_{fr}| \neq 0$, то

$$\vec{r} = \frac{\vec{p}_{fr}}{|\vec{p}_{fr}|}\mu\lambda_c$$

Иначе

$$\vec{r} = \vec{0}$$

Конец Если

$$\lambda_{fr\_old} = \lambda_{fr}$$

$$\lambda_{t\_old_\perp} = \lambda_{t_\perp}$$

$$\lambda_{fr} = \vec{r} \cdot \vec{t}$$

$$\lambda_{t_\perp} = \vec{r} \cdot \vec{t}_\perp$$

$$\Delta\lambda_{fr} = \Delta\lambda_{fr} + \lambda_{fr} - \lambda_{fr\_old}$$

$$\Delta\lambda_{fr_\perp} = \Delta\lambda_{fr_\perp} + \lambda_{t_\perp} - \lambda_{t\_old_\perp}$$

Конец Если

3.5. «Горячий» старт

Количество итераций, выполняемых в солвере, характеризует сходимость системы в целом. Говорят, что чем больше итераций требуется в солвере до получения малых значений применяемых импульсов, тем дольше система сходится. Хорошим примером сложной в плане сходимости системы является уже упоминавшийся случай покоя множества тел друг на друге. В такой системе подавляющее число ограничений связано друг с другом, т.е. применение импульсов для одной пары тел влияет на исходные данные для других пар. Существует метод, называемый WarmStarting («горячий» старт), который существенно улучшает сходимость импульсного солвера. Его суть заключается в том, что на каждом новом шаге моделирования расчеты по возможности надо начинать не с нуля, а со значений предыдущего шага. Для этого для всех ограничений системы запоминаются суммарные примененные импульсы. На следующем шаге моделирования расчет импульсов в неизменившихся ограничениях начинается с применения суммарного импульса, записанного с прошлого шага моделирования. При этом суммарный импульс текущего шага моделирования инициализируется значением суммарного импульса предыдущего шага. Под неизменившимися ограничениями понимаются ограничения, в которых не сильно изменяются параметры тройки $T = (\vec{P}, \vec{N}, D)$, а точнее точка $\vec{P}$ и нормаль $\vec{N}$. Для точки $\vec{P}$ это означает, что положение точки в пространстве на текущем шаге моделирования не сильно отличается от ее положения на прошлом шаге. Если $\vec{P}$ – положение точки на текущем шаге моделирования, а $\vec{P}_{old}$ – запомненное положение с предыдущего шага, то условие несильного изменения точки в пространстве можно записать как:

$$|\vec{P} - \vec{P}_{old}| < \varepsilon_P,$$

где ε_P – допустимое изменение положения точки $\vec{P}$ между кадрами. В случае с нормалью $\vec{N}$ условие несильного изменения между кадрами означает небольшое отклонение нормали $\vec{N}$ текущего шага моделировании от нормали $\vec{N}_{old}$ предыдущего шага. Это условие можно посчитать как:

$$|\vec{N} \cdot \vec{N}_{old} - 1| < \varepsilon_N,$$

где ε_N – допустимый угол отклонения нормали между кадрами в радианах. В случае выполнения условий на точку $\vec{P}$ и нормаль $\vec{N}$ применяют запомненные суммарные импульсы прошлого шага моделирования для всех ограничений тройки $T = (\vec{P}, \vec{N}, D)$, а так же инициализируют текущие значения суммарных импульсов значениями запомненных суммарных импульсов. За счет этого существенно улучшается сходимость системы ввиду того, что обеспечивается непрерывность расчетов импульсов для неизменяющихся ограничений в нескольких идущих подряд шагах моделирования.

Заключение

Предложенный метод обработки коллизий на основе импульсов включает в себя несколько существующих методов: метод последовательных импульсов, метод разделенных импульсов, технология «горячего» старта. Данный метод был реализован и протестирован на наборе тестовых сцен с разнообразными ситуациями при столкновении объектов. Были проверено соответствие результатов моделирования с реальной картиной для абсолютно упругих и неупругих центральных и нецентральных ударов, одновременных столкновений и т.д. Отдельное внимание было уделено тестированию сложной ситуации покоя множества объектов друг на друге. В этом случае за счет применения «горячего» старта удалось достичь общей сходимости системы для большого количества объектов с соблюдением условия моделирования в реальном времени.

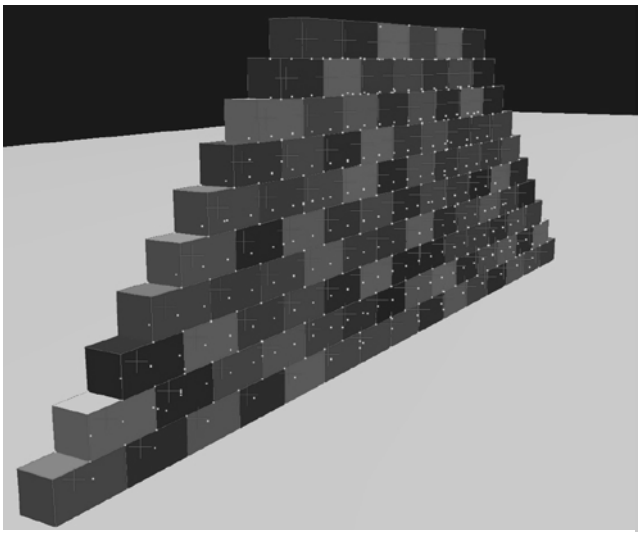


Рис. 3. Стенка из 105 прямоугольных параллелепипедов. Отображены точки контакта и центры масс

Сцена с пирамидальной стенкой из 105 покоящихся друг на друге прямоугольных параллелепипедов (см. рис. 3), для которых опре-

делялось около 400 троек $T = (\vec{P}, \vec{N}, D)$, успешно моделировалась в реальном режиме времени.

При этом для каждой из таких троек в случае покоя создавались ограничения на контакт, силу трения и выталкивание. Число ограничений в такой системе превышает 1000, причем все эти ограничения тесно связаны друг с другом. При расчете такой системы ограничений накладывалось условие, согласно которому на работу солвера отводилось 5 мс. За такое довольно небольшое время расчетов в солвере стенка из кубиков не разваливалась. А за счет применения «горячего» старта с течением некоторого времени система сходилась, после чего время расчета падало практически до нуля. Так что данный метод показал хорошие результаты моделирования в реальном режиме времени даже при поиске решений для выполнения большого количества тесно связанных друг с другом ограничений.

Работа выполнена при поддержке РФФИ, грант № 13-07-00708.

Collision response of virtual objects using the method of sequential impulses

A.M. Trushin

Abstract: Virtual dynamic objects in 3D scenes of computer simulation systems may collide with each other. Specifying a realistic reaction (response) on collisions is an important and essential part of the simulation of virtual objects' dynamics. In this paper we propose a method for collision response of objects based on a number of known methods (sequential impulses, separated impulses, warmstarting), connected together.

Keywords: virtual scenes, collisions, method of sequential impulses

Литература

1. Hyperplane separation theorem. Wikipedia. The Free Encyclopedia. Available from: http://en.wikipedia.org/wiki/Hyperplane_separation_theo-rem.

2. М.В. Михайлюк, А.М. Трушин Расчет коллизий прямоугольных параллелепипедов в задачах динамики – Труды НИИСИ РАН, 2012, т. 2 № 2, с. 51-59.

3. Gilbert–Johnson–Keerthi distance algorithm. Wikipedia. The Free Encyclopedia. Available from: http://en.wikipedia.org/wiki/Gilbert-Johnson-Keerthi_distance_algorithm.

4. Voronoi diagram. Wikipedia. The Free Encyclopedia. Available from: http://en.wikipedia.org/wiki/Voronoi_diagram.

5. B. Mirtich V-Clip: Fast and Robust Polyhedral Collision Detection. Available from: http://www.merl.com/publications/docs/TR97-05.pdf

6. A. Witkin An Introduction to Physically Based Modeling: Constrained Dynamics. Available from: http://www.cs.cmu.edu/~baraff/pbm/constraints.pdf.

7. D. Baraff Linear-Time Dynamics using Lagrange Multipliers. Available from: http://www.cs.cmu.edu/~baraff/papers/sig96.pdf.

8. Linear complementarity problem. Wikipedia. The Free Encyclopedia. Available from: http://en.wikipedia.org/wiki/Linear_complementarity_problem.

9. D. Stewart. Rigid-Body Dynamics with Friction and Impact. Available from: https://homes.cs.washington.edu/~todorov/courses/amath533/Stewart00.pdf.

10. W. Stronge Unraveling paradoxical theories for rigid body collisions. ASME Journal of Applied Mechanics, 1991, v. 58, pp. 1049-1055.

11. D. Baraff. Analytical methods for Dynamic Simulation of Non-penetrating Rigid Bodies. Available from: http://www.cs.cmu.edu/~baraff/papers/sig89.pdf.

Влияние плазмы коронного разряда на плазмонный резонанс

А.Н. Палагушкин, С.А. Прокопенко, А.П. Сергеев

Аннотация: В работе приводятся результаты экспериментальных исследований по воздействию плазмы коронного разряда в воздухе на поверхность плазмонных покрытий и влияния на их оптические параметры методом поверхностного плазмоного резонанса. Ранее, по нашим сведениям, исследования в этом направлении не проводились. Было обнаружено существенное различие при воздействии отрицательного и положительного коронных разрядов на изменение оптических свойств плазмонных покрытий. В первом случае оно приводит к необратимым, во втором к более слабым и обратимым изменениям оптических свойств покрытий. Коронный разряд создавался между иглой из W и плоской поверхностью образца при напряжениях 5.5 и 4.5 кВ, при токе 6 мкА. Измерения проведены для покрытий из слоев Ag 76 нм с защитным от окисления слоем Al_2O_3 1 нм и слоев Au 68 и 53 нм. Использовалась автоматизированная гониометрическая установка со схемой Кречмана. Изменения эффективных толщин, показателей преломления и поглощения слоев определялись путем их варьирования, по наилучшему соответствию формы теоретической угловой зависимости коэффициента отражения при плазмоном резонансе и экспериментальной кривой.

Ключевые слова: поверхностный плазмонный резонанс, SPR, наноструктуры, коронный разряд.

Введение

Рассмотрены особенности воздействия облучения плазмой коронного разряда в воздухе чувствительной поверхности плазмонной структуры, в условиях поверхностного плазмонного резонанса (SPR) [1]. Подобные исследования ранее не проводились, известно одно сообщение о применении финишной ионной очистки поверхности для регенерации SPR – биосенсоров [2]. В тоже время использование ионного облучения и коронного разряда для очистки и модификации поверхности металлов и диэлектриков хорошо известно [3,4]. Плазмонные SPR структуры, состоящие из металлодиэлектрических нанопокрытий, крайне чувствительны к их оптическим свойствам и наличию на поверхности адсорбированных из внешней среды газов и паров воды. Поэтому облучение таких покрытий плазмой коронного разряда существенно изменяет их свойства, а сам метод SPR позволяет определить эти изменения.

2. Экспериментальные исследования

Проводились при периодическом облучении SPR-покрытий плазмой отрицательного и положительного коронного разрядов на воздухе и последовательном измерении коэффициентов угловой зависимости полного отражения по схеме Кречмана на автоматизированной гониометрической установке ранее описанной в [5]. Она изготовлена на основе рентгеновского θ - 2θ гониометра (ГУР-8) с оптической схемой, состоящей из полуцилиндра и дополнительной цилиндрической оптикой, обеспечивающей плоский фронт волны на отражающем образце (Рис.1).

Использованы He-Ne (632.8 нм) и полупроводниковый (460 нм) лазеры, высоколинейный фотоприемник с преобразованием напряжения в частоту и накоплением сигнала. Установка управляется ПК, зависимость углового отражения записывается в файл.

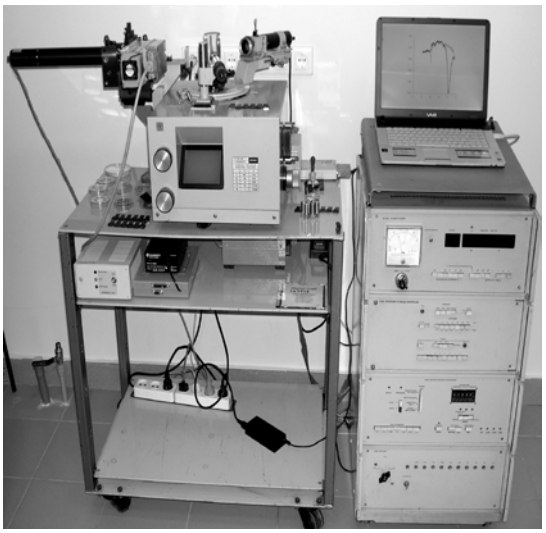


Рис.1. Гониометрическая установка

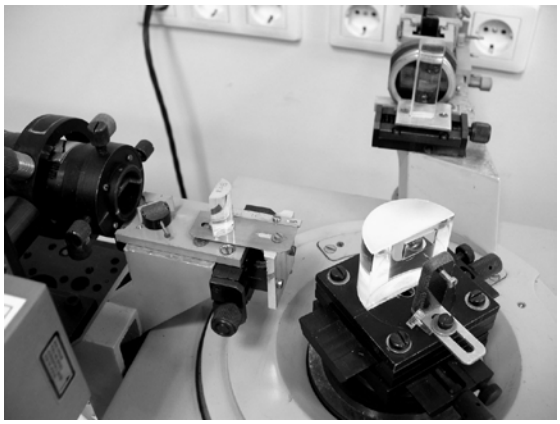


Рис.2. Столик образца с коронирующей иглой.

Против центра зоны измерения установлена изолированная игла из вольфрама, служащая для возбуждения коронного разряда (Рис.2). От стабилизированного источника высокого напряжения через ограничительный резистор подавалось напряжение между металлическим покрытием образца и иглой (на рисунке не показано). Фиксировалось напряжение, ток и время разряда. Облучение поверхности образца плазмой коронного разряда и измерение его параметров чередовались. Были проведены эксперименты с отрицательным и положительным коронным разрядом.

Затем, по методике, приведенной в [6], проводился расчет изменения эффективных оптических параметров нано слоев SPR покрытия после облучения. Приведенные в данной статье измерения выполнялись на длине волны лазера 632.8 нм.

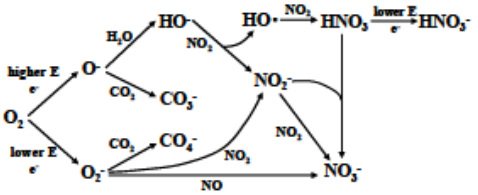
3. Изготовление образцов SPR покрытий.

SPR покрытия были изготовлены на подложках из плавленого кварца (10x20x1 мм^3) электроннолучевым нанесением в высоком вакууме (5*10^{-6} mbar) нанослоев Ag с защитным слоем Al_2O_3 с толщинами соответственно 67 нм и 1…5 нм, а также слоев Ag (64 нм) и Au (54 нм) без защиты. Слои металла и защитные покрытия наносились в одном цикле напыления на установке Leybold-Heraeus L-560Q. Скорость нанесения и толщина покрытий контролировалась кварцевым монитором Inficon.

4. Результаты измерений покрытий при воздействии коронного разряда.

Было обнаружено сильное и различающееся воздействие отрицательного и положительного коронного разряда на угловое смещение минимума отражения при SPR – резонансе. Измеряемый образец устанавливался на плоскую поверхность полуцилиндра через тонкий слой иммерсионного масла (n = 1.5135). Для определения эффективного показателя преломления среды записывалась зависимость углового отражения для чистой кварцевой подложки без покрытий. Затем устанавливался измеряемый образец с покрытием Ag (67…65 нм) и защитным слоем Al_2O_3 (~1 нм). Записывалось его начальное состояние и изменение после воздействия отрицательного коронного разряда (Рис.4).

Состав ионизированной коронным разрядом плазмы в воздухе (Рис.3) достаточно хорошо изучен [7].



(а) Отрицательная корона

(б) Положительная корона

Рис. 3. Ионные процессы при отрицательном и положительном коронном разряде в воздухе [7].

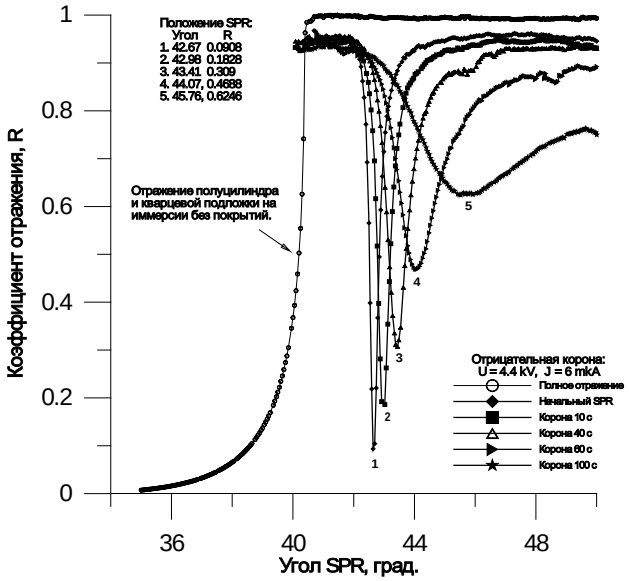


Рис. 4. Изменение SPR образца Ag(67 нм) – Al_2O_3 (~1 нм) при отрицательном коронном разряде.

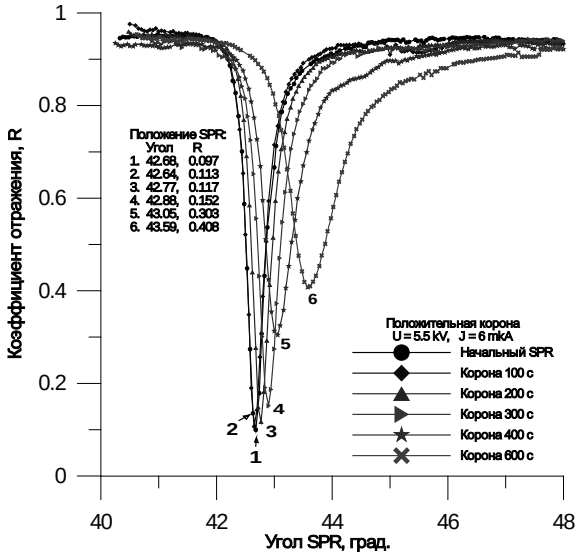


Рис.5. Изменение SPR образца Ag(67 нм) – Al_2O_3 (~1 нм) при положительном коронном разряде.

При действии отрицательной короны поверхность облучается отрицательными ионами, и происходит ее быстрое окисление и осаждение продуктов пиролиза следов примесей, присутствующих в воздухе. В разряде присутствует озон и окислы азота, которые могут взаимодействовать с поверхностью серебра через поры в защитном слое и вызывать его окисление. Это приводит к образованию окислов и азотных соединений на поверхности и уменьшению толщины слоя металла. При этом угол SPR резонанса смещается вправо, отражение возрастает, а прохождение света вне углов полного отражения увеличивается. После отключения разряда достигнутая форма SPR сохраняется. На Рис.5 показано влияние положительного коронного разряда на SPR резонанс. Воздействие положительного коронного разряда на поверхность плазмонного покрытия первоначально приводит к небольшому уменьшению угла резонанса, а затем он сдвигается в сторону больших углов, но на меньшую величину. Для этого требуется значительно большее (на порядок) время действия коронного разряда. При слабом и коротком временном воздействии разряда положение резонанса (2) Рис.5. точно воспроизводится, в то время как положение начального SPR (1) зависит от условий внешней среды (например, влажности воздуха) и толщины защитного покрытия. После прекращения разряда и выдержки на воздухе исходное положение резонансной зависимости восстанавливается.

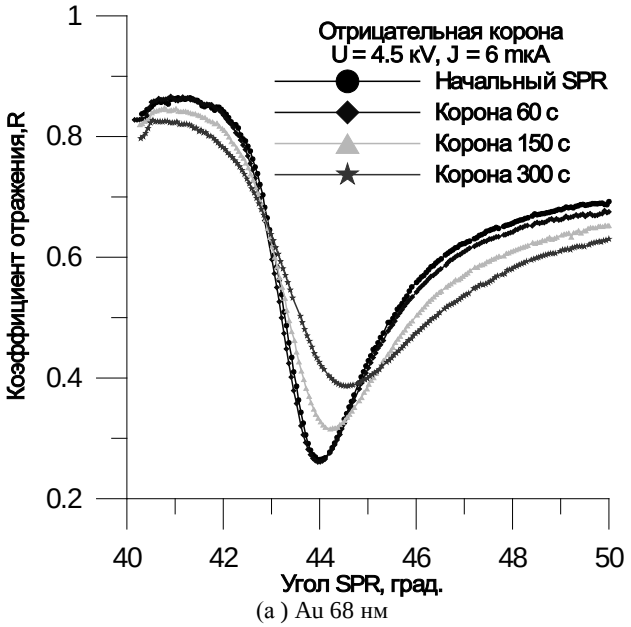
В положительной короне преобладают радикалы азота и воды, вероятно, приводящие при малой мощности разряда к обратимой эффективной десорбции и очистке поверхности защитного слоя на серебре и активации центров адсорбции. При этом его угловое положение изменяется незначительно.

Поверхность слоев Ag без защитного слоя Al_2O_3 окисляется присутствующими в разряде радикалами кислорода, слой металлического Ag становится тоньше, а на нем образуется более толстый слой окиси. Это приводит к уширению и смещению резонанса SPR в область больших углов и возрастанию отражения.

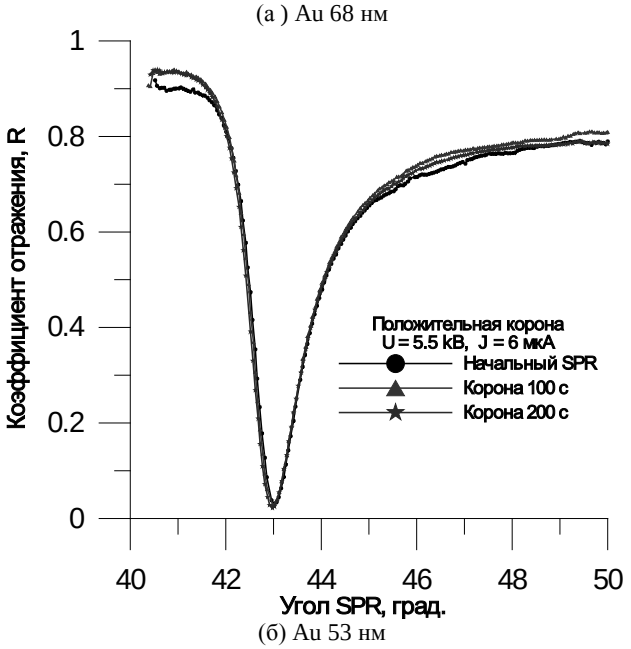
Покрытия из золота устойчивы к окислению и при воздействии коронного разряда обеих полярностей ведут себя аналогичным образом (Рис.6), причем положительная корона влияет на SPR очень слабо.

5. Результаты расчета изменения оптических параметров покрытий

Оптические свойства металлических и диэлектрических покрытий нанометровой толщины применяемых для исследований плазмонного резонанса могут существенно отличаться от свойств объемных образцов и ряда справочных данных. Они сильно зависят от используемого метода, условий нанесения, таких как температура и скорость осаждения, присутствия примесей. Причем оптимальные условия нанесения SPR покрытий отличаются от принятых в оптических и электронных технологиях.



(а) Au 68 нм



(б) Au 53 нм

Рис.6. Воздействие отрицательного (а) и положительного (б) коронного разряда на SPR в слоях золота.

Поэтому наиболее рационально измерять оптические параметры таких нанопокрытий по наилучшему совпадению расчетных и экспериментальных данных спектральной или угловой зависимости коэффициента отражения в области SPR. Коэффициент отражения определяется по формулам Френеля. Показатели преломления и поглощения слоев покрытий и их толщины вычисляются по минимальному отклонению всех экспериментальных точек угловой резонансной зависимости отражения от

расчетной теоретической, при варьировании оптических параметров покрытий.

Вычисленные значения отражают эффективные, а не абсолютные значения оптических параметров слоев и могут использоваться для расчетов, сравнения и моделирования новых конструкций плазмонных покрытий на их основе. Результаты расчетов для угловых зависимостей отражения показанных на Рис.4 приведены в Таблице 1.

Таблица 1. Изменение SPR образца Ag(67 нм) –(~1 нм) при отрицательном коронном разряде (Рис. 4).

№ графика	Толщина слоя Ag, нм	Показатель преломления Ag	Показатель поглощения Ag	Толщина слоя Al_2O_3, нм	Показатель преломления Al_2O_3	Показатель поглощения Al_2O_3	Ошибка, %
1	67.18	0.070	3.513	0.86	1.5	0.099	1.34
2	66.25	0.081	3.50	3.43	1.488	0.1	1.77
3	65.06	0.0942	3.511	7.82	1.421	0.1	1.69
4	62.13	0.1509	3.6050	14.4	1.40	0.1	1.86
5	62.02	0.1653	3.4487	32.2	1.27	0.117	2.47

Низкие исходные значения n и k для Ag получены в оптимально подобранных условиях нанесения слоев и отсутствия их окисления при защите слоем Al_2O_3**,** а также увеличения толщины слоя Ag больше оптимальной, соответствующей минимальному отражению. Для сравнения, в базе Sopra [8] приведены значения n = 0.136 и k =4.011.

При действии коронного разряда толщина слоя Ag вначале быстрее, а затем медленнее уменьшается, что вероятно вызвано его окислением через поры в защитном слое. Соответственно, преломление слоя возрастает, а его поглощение слабо меняется. Эффективная толщина защитного слоя Al_2O_3 резко возрастает, а его показатель преломления уменьшается, что связано с осаждением на его поверхности продуктов пиролиза в коронном разряде и осаждением на поверхности радикалов плазмы.

Среднеквадратичное отклонение теории от эксперимента не превышает 2.5% . Результаты расчета при действии положительной короны (Рис.5) приведены в Таблице 2.

Таблица 2. Изменение SPR образца Ag(66 нм) –(~1 нм) при положительном коронном разряде (Рис. 5).

№ графика	Толщина слоя Ag. нм	Показатель преломления Ag	Показатель поглощения Ag	Толщина слоя Al_2O_3. нм	Показатель преломления Al_2O_3	Показатель поглощения Al_2O_3	Ошибка. %
1	65.94	0.0833	3.5349	1.17	1.4998	0.008	1.63
2	66.71	0.0826	3.5343	1.12	1.4999	0.008	1.62
3	66.67	0.0887	3.5342	2.05	1.4994	0.009	1.51
4	66.38	0.100	3.5341	3.17	1.4958	0.016	2.45
5	66.52	0.1253	3.5648	5.26	1.4458	0.041	2.11
6	66.76	0.1041	3.4961	9.80	1.3935	0.099	1.71

При действии положительного коронного разряда толщина слоя Ag и его поглощение меняются слабо, а показатель преломления незначительно увеличивается. Толщина защитного слоя существенно увеличивается и возрастает его поглощение, показатель преломления уменьшается не значительно. При этом, изменение толщины значительно меньше, чем в случае отрицательной короны, если принять во внимание значительно большее время воздействия коронного разряда. Вероятно, это связано с осаждением на поверхности только положительных радикалов плазмы.

При действии отрицательного коронного разряда на слои золота также происходит образование на его поверхности дополнительного слоя из продуктов ионизированной плазмы. При расчете этот слой учитывался как слой с заранее неизвестными оптическими параметрами. Результаты расчета представлены в Таблице 3.

Таблица 3. Изменение параметров слоя Au при отрицательном коронном разряде.

№ графика	Толщина слоя Au. нм	Показатель преломления Au	Показатель поглощения Au	Толщина слоя. нм	Показатель преломления слоя	Показатель поглощения слоя	Ошибка. %
1	67.72	0.2792	2.8471	0.07	1.3138	0.09	2.17
2	56.31	0.2879	2.8612	9.12	1.2831	0.107	1.52
3	55.16	0.3240	3.2808	11.61	1.2835	0.145	1.35
4	51.73	0.3698	3.4406	14.97	1.3080	0.133	1.11

Расчет показывает, что толщина слоя золота несколько уменьшается, а его показатели преломления и поглощения возрастают. Толщина образующегося на поверхности слоя (неизвестного состава, с показателем преломления ~ n = 1.3) возрастает до 15 нм и он обладает заметным поглощением света.

Положительный коронный разряд слабо воздействует на SPR слоя Au (Таблица 4). При этом несколько уменьшается толщина слоя и его показатель преломления, возрастает поглощение. Измеренные параметры слоев Au несколько превышают значения, приведенные в базе Sopra (n = 0.178, k = 3.070).

Таблица 4. Изменение параметров слоя Au при положительном коронном разряде.

№ график а	Толщина слоя Au. нм	Показатель преломления Au	Показатель поглощения Au	Ошибк а. %
1	52.52	0.2244	3.3012	1.60
2	51.02	0.1975	3.3174	1.49
3	49.60	0.2028	3.3381	1.48

Заключение

Проведенные эксперименты и обнаруженные изменения оптических параметров плазмонных покрытий на основе Ag с защитным слоем Al_2O_3 и Au под воздействием облучения плазмой на воздухе показали, что полярность разряда по-разному влияет на их параметры. При отрицательном разряде происходит необратимое окисление поверхности слоев Ag и образование дополнительных диэлектрических слоев на поверхности Ag и Au. Положительный коронный разряд оказывает более слабое, и частично обратимое влияние на поверхность плазмонных покрытий. Для более подробного изучения этих особенностей необходимо исследовать динамику воздействия положительного коронного разряда в контролируемых средах. Поскольку влажность и микропримеси в воздухе оказывают влияние на результаты измерений, а их адсорбция на поверхности может происходить в процессе измерения. На основе использования коронного разряда может быть создан новый тип SPR газовых сенсоров. Это направление наших дальнейших исследований.

Работа выполнена по программе фундаментальных исследований ОНИТ РАН № 2.1

Influence of plasma corona on plasmon resonance

A.N.Palagushkin, C.A.Prokopenko, A.P.Sergeev

Abstract: The paper presents the results of experimental studies on the effects of the plasma corona discharge in air to the surface plasmon coatings and their influence on the optical parameters by surface plasmon resonance. Previously, as far as we know, research in this direction have not been conducted. A significant difference in the impact of negative and positive corona discharges on changes in the optical properties of plasmonic coatings was detected. In the first case it leads to irreversible, secondly to a weaker and reversible changes of optical properties of the coating. Corona discharge was generated between the tip of the W and the flat surface of the sample at voltages of 5.5 and 4.5 kV, 6 μA current. Measurements were carried out for coatings of 76 nm Ag layers with a protective layer against oxidation Al2O3 and 1 nm Au layers 68 and 53 nm. We used automated goniometer setting the Krechman scheme. Changes of effective thickness, refractive index and absorption layers are defined by their varying for best match of the theoretical angular dependence of the reflection coefficient at the plasmon resonance with the experimental dependence.

Keywords: surface plasmon resonance, SPR, nanostructures, corona discharge.

Литература

1. A.L.Mikaelian, B.V.Kryzhanovsky, A.N.Palagushkin, S.A.Prokopenko, A.P.Sergeev, F.A.Yudkin and A.N.Arlamenkov, “Sensors Using Plasmon Nanostructures”. Optical Memory and Neural Networks (Information Optics). 2005, v.14, №.4, pp.229-244.
2. http://nanospr.uservoice.com/knowledgebase/articles/175054-gold-spr-slide-surface-cleaning.
3. Peter M. Martin. Handbook of Deposition Technologies for Films and Coatings. Published by Elsevier Inc., 2010.
4. Ю.П. Райзнер. Физика газового разряда. М., «Наука», 1992 .

5. А.Л.Микаэлян, Б.В.Крыжановский, А.Н.Палагушкин, С.А. Прокопенко, А.П. Сергеев, А.Н. Арламенков. Автоматизированная установка для измерения оптических параметров нанослоев металлов методом поверхностного плазмонного резонанса. «Научная сессия МИФИ – 2007: Фотоника и информационная оптика (22-26 января 2007)», тезисы докладов.

6. A.N. Palagushkin, S.A. Prokopenko, A.P. Sergeev and A. N. Arlamenkov. Measurement of Metal Nanolayers Optical Parameters Using Surface Plasmon Resonance Method. Optical Memory and Neural Networks (Information Optics). 2007, v. 16, № 4, pp. 288-294.

7. K. Sekimoto, M. Takayama. Negative ion evolution and formation in atmospheric pressure corona discharges between point-to-plane electrodes. 30th ICPIG. August 28-th – September 2-nd, 2011. Belfast. Northern Ireland, UK.

8. SOPRA: http://www.sspectra.com/files/misc/win/sopra.exe

Компьютерное моделирование температурных полей электронных систем при неопределенности определяющих параметров

П.И. Кандалов, А.Г. Мадера[1]

1 – доктор технических наук

Аннотация. В настоящее время при моделировании температурных полей технических систем реализуется детерминированный подход, при котором принимается, что все параметры, определяющие тепловой режим системы, являются однозначными и абсолютно точно определенными. Вместе с тем реальная практика показывает, что параметры теплового режима носят неопределенный характер, принимая возможные значения внутри своих интервалов изменения, что приводит к интервальной неопределенности значений температуры в различных точках системы. В статье развивается метод моделирования температурных полей электронных систем в условиях интервальной неопределенности, который более адекватен реальности, чем детерминированный подход.

Ключевые слова: неопределенность, электронная система, температурное поле, моделирование, интервальная неопределенность

Введение

Одними из наиболее важных дестабилизирующих факторов, воздействующих на электронную систему (ЭС), являются тепловые воздействия, которые носят неоднородный, необратимый, неустранимый характер. В процессе функционирования ЭС в интегральных микросхемах (ИМС) и других полупроводниковых элементах (ППЭ) в ЭС возникают распределения температуры. Электрические (статические и динамические) параметры ИМС и ППЭ в сильной степени зависят от температуры, поэтому возникновение температурных полей в ЭС обусловливает необратимые отрицательным последствия для ЭС и ее элементов: выходу параметров ИМС, ППЭ и всей ЭС за пределы своих допустимых значений, снижению надежности, уменьшению быстродействия, помехозащищенности и т.д. [6]. Поэтому актуальной задачей является уменьшение влияния воздействий температуры на функционирование ЭС, что достигается в первую очередь снижением уровней температуры и уменьшением ее перепадов по конструкциям ИМС и ЭС. Снижение уровней температурных полей осуществляется различными техническими средствами – физическими принципами охлаждения, конструкциями теплоотводов, систем охлаждения. Все это требует для своего проектирования адекватных методов математического и компьютерного моделирования температурных полей ЭС.

Методы моделирование температурных полей технических систем различного назначения интенсивно развивается как в России [1], так и за рубежом (Beta-Soft, Mentor-Graphics / Therm & Flow, Ansys и др.). Вместе с тем, существующие в настоящее время математические и компьютерные модели, равно как и методы моделирования температурных полей, являются детерминированными [1 – 4]. Это означает, что все исходные данные, определяющие протекание теплового процесса и его характер, являются полностью определенными и однозначно известными. Между тем, такое допущение не соответствует реальности. На практике реальные параметры как самой технической системы, так режимов ее функционирования и взаимодействия со средой никогда не известны в точности и являются неопределенными [10 – 13]. В реальности параметры системы всегда изменяются в некоторых интервалах своих значений, причем какое именно значение примет тот или иной параметр в каждом конкретном случае и данной системе априори не известно. Другими словами, реальные параметры, определяющие как функционирование системы, так и протекающие в ней процессы, носят неопределенный характер. К таковым параметрам относятся параметры конструкции (зазоры между контактирующими элементами, размеры, форма, расстояния и пр.), параметры функционирования ИМС и ЭС (мощности потребления, электрические статические и динамические параметры ИМС), параметры среды (температуры, скорости теплоносителей, влажность и пр.), которые изменяются в интервалах своих допустимых значений. Поскольку параметры ЭС и ее элементов изменяются в пределах своих интервалов, то такую неопределенность мы будем далее называть интервальной неопределенностью.

Интервальная неопределенность параметров, определяющих электрический и тепловой режимы ИМС и ЭС в целом, обусловливает, в свою очередь, интервальную неопределенность выходных параметров и характеристик, приводя к интервальной неопределенности температурных полей технической системы и ее элементов. Иначе говоря, температура в каждой точке ЭС будет носить недетерминированный, неоднозначный характер и всегда будет представлять

собой интервал, внутри которого она принимает свои возможные значения, отвечающие всевозможным сочетаниям конкретных значений определяющих параметров из своих интервалов изменения. Таким образом, для адекватного моделирования температурных полей ЭС и ее элементов необходимо чтобы математические и компьютерные модели учитывали интервальную неопределенность определяющих параметров ЭС, являющихся исходными данными для моделирования.

В настоящей статье рассматривается методология компьютерного интервального моделирования температурных полей технических систем, которая

Компьютерное моделирование интервальных температурных полей

В общем случае все параметры ЭС, определяющие тепловой режим (коэффициенты теплопроводности материалов конструкции, коэффициенты теплоотдачи с поверхностей элементов и системы, температуры среды, мощности тепловыделений, контактные тепловые сопротивления и др.), являются интервально неопределенными, то есть изменяются в пределах своих интервалов, принимая внутри них с равной вероятностью любое значение,.

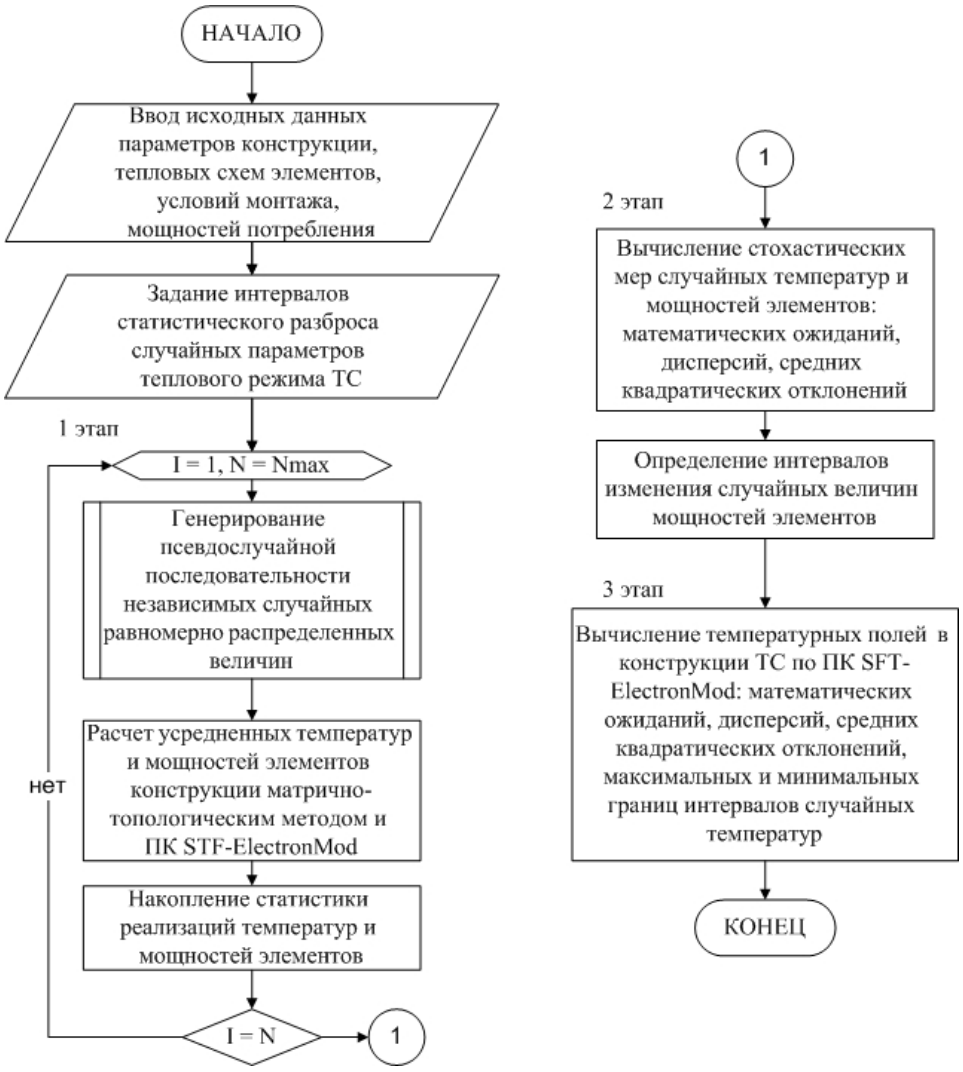


Рис. 1. Компьютерное моделирование температурных полей ЭС при интервальной неопределенности

позволяет моделировать распределения интервалов неопределенностей температур ИМС и ЭС, внутри которых могут находиться возможные реальные значения температуры в каждой точке системы. При этом параметры системы, то есть исходные данные, определяющие температурное поле, являются также неопределенными и заданными в виде интервалов своих возможных изменений.

Поскольку, параметры и характеристики технической системы носят интервально стохастический характер, то решение уравнений стохастической математической модели, описывающей температурное поле, будет также интервально стохастическим полем, то есть $T(\omega) = T(x,y,z,\omega)$, $\omega \in \Omega$, где ω – элементарные события из пространства элементарных событий Ω. Заметим, что в каждой точке технической системы температура будет изменяться в некотором интервале и иметь распределение вероятностей, вообще говоря, отличное от

равномерного. В результате решения уравнений стохастической математической модели (посредством разработанного авторами программного комплекса) для каждой реализации $\omega \in \Omega$ определяются поля статистических мер (математических ожиданий, дисперсий, средних квадратических отклонений, границ интервалов) интервально стохастического температурного поля $T(x,y,z,\omega)$ в технической системе:

• поле математических ожиданий (МО) температур $\overline{T} = \overline{T}(x, y, z)$;

• поле дисперсий $D(x,y,z)$ рассеяния случайных значений температур вокруг соответствующих точек в поле математического ожидания;

• поле средних квадратических отклонений (СКО) $\sigma(x,y,z)$ температурного поля, определяемого как $\sigma^2(x,y,z) = D(x,y,z)$;

• поля нижней $T_{\text{Н}}(x,y,z)$ и верхней $T_{\text{В}}(x,y,z)$ границ интервалов стохастического температурного поля $T(x,y,z,\omega)$, $\omega \in \Omega$, в пределах которых изменяются отдельные случайные реализации температуры для $\omega \in \Omega$ в каждой точке технической системы.

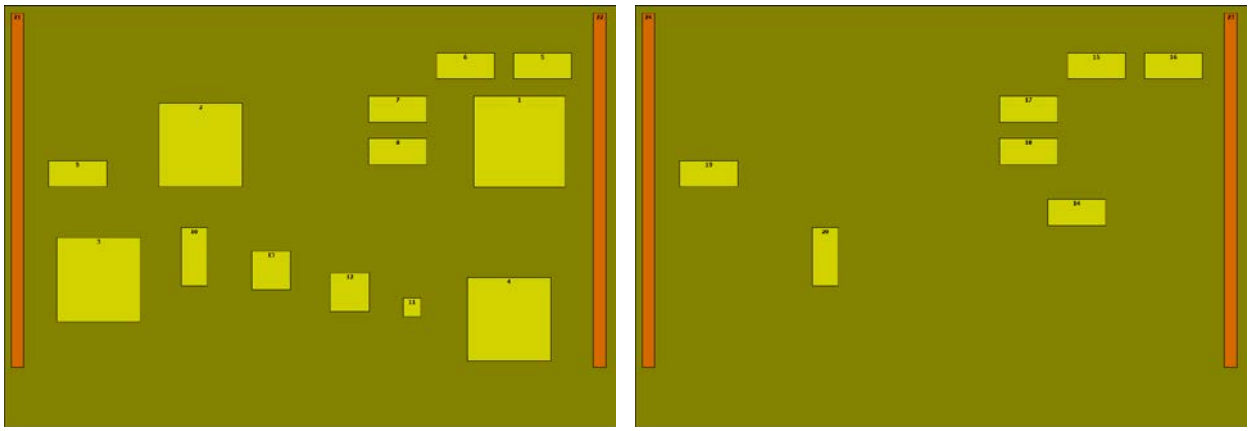
Границы интервалов температуры в каждой точке (x,y,z) технической системы $T_{\text{Н}}(x,y,z)$ и $T_{\text{В}}(x,y,z)$ зависят от доверительной вероятности $P_{\text{дов}}$, с которой интервалы $[T_{\text{Н}}(x,y,z),\ T_{\text{В}}(x,y,z)]$ покрывают возможные значения реальных температур в технической системе, имеющих место на практике. При этом нижние и верхние границы интервалов температур в точках (x,y,z) вычисляются как $T_{\text{Н}}(x,y,z) = \overline{T} - \varepsilon \cdot \sigma(x,y,z)$ и $T_{\text{В}}(x,y,z) = \overline{T} + \varepsilon \cdot \sigma(x,y,z)$.

Значения температуры $T(x,y,z,\omega)$ в различных точках технической системы (x,y,z), которые могут встретиться в реальности, будут изменяться внутри интервала $T(x,y,z,\omega) \in [T_{\text{Н}}(x,y,z),\ T_{\text{В}}(x,y,z)]$, где $\varepsilon = 1 \div 3$ – множитель, зависящий от величины принятой доверительной вероятности $P_{\text{дов}}$.

Компьютерное моделирование интервально стохастических температурных полей (рис. 1) содержит три основных этапа:

1. Задание интервально стохастических входных данных в виде своих интервалов изменения (нижних и верхних границ). По введенным данным осуществляется расчет усредненных температур и мощностей всех элементов в технической системе, с применением матрично-топологического метода и программного комплекса расчета детерминированных температурных полей STF-ElectronMod [8]. После этого методом статистических испытаний разыгрывается множество реализаций интервально стохастических входных данных и рассчитанных величин [5, 7].

2. Вычисление статистических мер усредненных температур и мощностей элементов ЭС: математических ожиданий (МО); дисперсий (D); средних квадратических отклонений (СКО); границ (нижней и верхней) интервалов, внутри которых изменяются реализации усредненных температур и мощностей; ковариационной матрицы стохастической связи между мощностями активных элементов (источников тепловыделения) и пассивных элементов (стоков тепловых потоков).



a) b)

Рис. 2. ЭМ (верхняя (a) и нижняя (b) поверхности), содержащая МПП, с установленными на ней ИМС (желтые прямоугольники) и конструктивными элементами системы (оранжевые прямоугольники)

3. Определение полей статистических мер интервально стохастических температур в технической системе: полей МО, полей D, полей СКО, полей границ интервалов, покрывающих при заданной доверительной вероятности возможные значения стохастических температур [9, 14].

Рассмотренный алгоритм положен в основу разработанного авторами программного комплекса ПК STF-ElectronMod-2014, предназначенного для анализа неопределенных интервально стохастических температурных полей технических систем и вычисления полей статистических мер неопределенных распределений температуры в достаточно сложных ЭС. Применение программного комплекса для реальных технических систем показало его высокую эффективность и адекватность. Так, для процессора с частотой 1,5 ГГц точность вычислений не превышает 3÷5% при затратах машинного времени 2÷3 минут.

Моделирования интервальных температурных полей ЭС

В качестве примера ЭС рассматривается электронный модуль (ЭМ) с многослойной печатной платой (МПП), с установленными на ней 20-ю ИМС (рис. 2).

Рассматриваемая электронная система находится в условиях свободно конвективного теплообмена с воздушной средой при нормальном давлении.

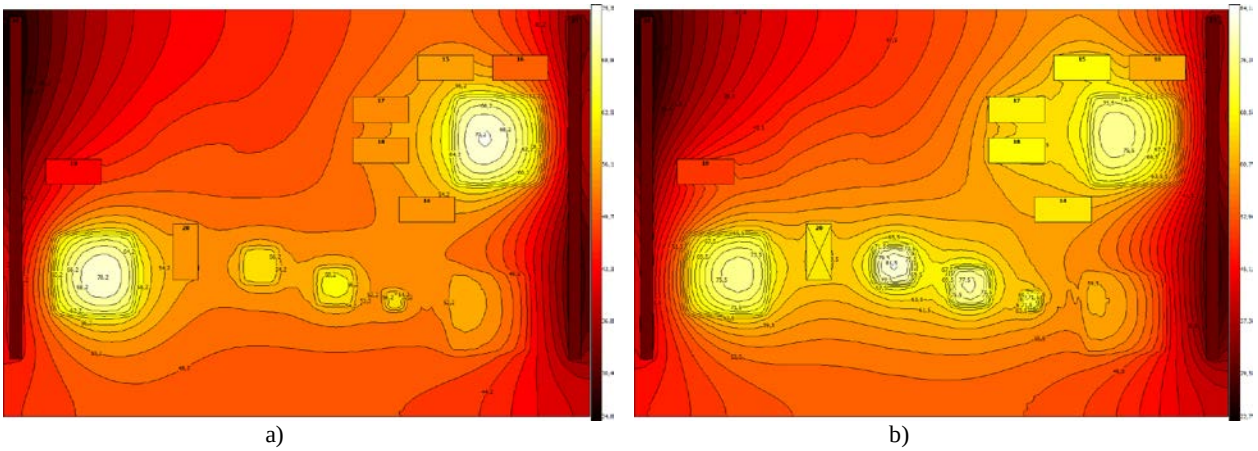
Моделирование интервального температурного поля ЭМ проводилось в условиях интервальной неопределенности мощностей потребления всех ИМС, возможные значения которых лежат в интервалах, ограниченных их минимальными и максимальными значениями (табл. 1). Отметим, что конкретные значения мощностей потребления ИМС в каждом конкретном экземпляре электронной системы априори не известны, однако можно указать диапазон (интервал) внутри которого они могут находиться. При этом всевозможные сочетания значений мощностей ИМС, которые могут реализоваться в системе, принадлежат множеству, представляющему собой прямое произведение всех множеств, каждое из которых является соответствующим интервалом изменения той или иной мощности.

Таблица 1

Минимальные и максимальные значения мощностей потребления ИМС и интервалы возможных значений средних температур корпусов ИМС

Номер ИМС на МПП (рис. 2)	Интервальная мощность ИМС, Вт		Интервал возможных значений температуры корпусов ИС, °С	Разброс средних температур корпусов ИС, %
	Минимальная	Максимальная		
1	5,5	5,8	73,0 - 80,7	10,5
2	0,4	0,65	47,8 - 55	15, 1
3	4,9	5,5	75,3 - 83,2	10,5
4	1,9	2,5	54,9 - 62	12,9
5	0,17	0,5	52,3 - 64,7	23,7
6	0,17	0,5	57,0 - 72,4	27,0
7	0,17	0,5	55,0 - 71,2	29,5
8	0,17	0,5	55,6 - 71,8	29,1
9	0,17	0,5	44,7 - 54,1	21,0
10	0,17	0,5	55,5 - 70,2	26,5
11	0,3	0,55	64,3 - 84,1	30,8
12	0,9	1,8	59,6 - 78,4	31,5
13	0,8	2,1	58,7 - 81,7	39,2
14	0,17	0,5	56,1 - 69,6	24,1
15	0,17	0,5	56,8 - 72,3	27,3
16	0,17	0,5	51,9 - 64,3	23,9
17	0,17	0,5	55,1 - 71,4	29,6
18	0,17	0,5	55,7 - 72,0	29,3
19	0,17	0,5	44,6 - 54,1	21,3
20	0,17	0,5	55,5 - 70,2	26,5

Результаты моделирования – интервалы возможных значений средних температур корпусов ИМС и визуальное представление линий уровня (изотерм) на верхней поверхности МПП электронной системы, рассчитанных для минимального и максимального значений мощностей потребления ИС,



a) b)

Рис. 3. Изотермы на верхней поверхности МПП при минимальных (a) и максимальных (b) граничных значениях мощностей потребления ИМС

приведены в табл. 1 и рис. 3. Моделирование интервальных температурных полей электронной системы показывает, что реальные значения температур в каждой точке поверхности МПП (рис. 2), как и средние температуры корпусов ИС (табл. 1), являются не точечными и однозначно определенными, а лежат внутри соответствующих интервалов. Причем на практике могут встретиться любые значения температур, заключенные внутри своих интервалов, и лежащие между рассчитанными минимальными и максимальными значениями. Анализ данных моделирования свидетельствуют, что разброс возможных значений температур, которые могут встретиться в реальности может достигать существенных значений Так, наименьший (10,5%) разброс значений температуры корпуса имеет место у ИС 1 и ИС 3 а наибольший (39,2%)– у ИС 13.

Расчет температурного поля, осуществляемый на основании однозначно и «точно» заданных исходных данных, и приводящий к однозначным и единственным значениям температур, не соответствует реальности, так как значения температур носят принципиально неопределенный характер. Поэтому детерминированный подход к моделированию температурных полей, применяемый в настоящее время, не может быть признан адекватным реальной практике. В то же время подход, при котором моделирование температурных полей проводится в условиях неопределенности исходных данных, а именно, в условиях интервальной неопределенности, когда параметры, определяющие температурные поля технических систем, задаются в виде своих возможных значений, принадлежащих соответствующим интервалам своего изменения, отвечает реальной практике и является адекватным ей.

Работа выполнена при поддержке РФФИ, грант № 12-07-00076-а

Simulation of temperature fields of electronic systems under uncertainty of determing parameters

P.I. Kandalov, A.G. Madera

Abstract. At present, the simulation of temperature fields of technical systems implemented deterministic approach in which it is assumed that all the parameters that define the thermal conditions are single-valued and exactly defined. However actual practice shows that the parameters of the thermal regime are uncertain taking the possible values within their ranges of change. This leads to an interval indeterminacy of temperatures at various points in the system. In this paper we develop a method for modeling of temperature fields of engineering systems under interval indeterminacy, which is more adequate to reality than the deterministic approach. A specific example of an interval simulation are considered.

Keywords: uncertainty, technical system, temperature, the temperature field, mathematical and simulation, interval arithmetic, interval indeterminacy

Литература

1. А.Г.Мадера, П.И.Кандалов. Матрично-топологический метод математического и компьютерного моделирования температурных полей в электронных модулях: программный комплекс STF-ElectronMod // Программные продукты и системы, 2012, № 4, с. 160–164.

2. П.И.Кандалов, А.Г.Мадера. Моделирование температурных полей в многослойных структурах // Программные продукты и системы, 2008. № 4 (84), с .46–49

3. А.Г.Мадера. Иерархический подход при тепловом проектировании электронных изделий // Программные продукты и системы, 2008. № 4 (84), с. 43–46.

4. А.Г.Мадера, П.И.Кандалов. Матрично-топологический метод математического и компьютерного моделирования температурных полей в электронных модулях: программный комплекс STF-ElectronMod // Программные продукты и системы, 2012, № 4, с. 160–164.

5. С.М.Михайлов, Г.А.Ермаков. Статистическое моделирование. М., Наука, 1982

6. Надежность технических систем: Справочник / под ред. И.А. Ушакова/. М., Радио и связь, 1985

7. И.М.Соболь. Численные методы Монте-Карло. М., Наука, 1973

8. Программный комплекс Simulation of Temperature Fields of Electronic Modules 2013 (STF-ElectronMod 2013) / П.И. Кандалов, А.Г. Мадера // Свидетельство о Гос. регистрации ПК для ЭВМ №2013615400 от 06.06.2013.

9. Н. Хастингс, Дж.Пикок. Справочник по статистическим распределениям. М., Статистика, 1980

10. J.G.Georgiadis. On the approximate solution on non-deterministic heat and mass transport problems // International Journal of Heat and Mass Transfer, 1991, v. 33, № 8, pp.2099 – 2105.

11. C.J.Keller, V.W.Antonetti. Statistical thermal design for computer electronics // Electronic Packaging and Production, 1979, v.19, № .3, pp. 55 – 62.

12. A.G.Madera. Modelling of stochastic heat transfer in a solid // Applied Mathematical Modelling, 1993, v. 17, № 12, pp. 664–668

13. A.G.Madera. Simulation of stochastic heat conduction processes // International Journal of Heat and Mass Transfer., 1994, v. 37, № 16, pp. 2571–2577

14. H. Ratschek, J.Rokne. Computer methods for the range of functions. New York, John Wiley, 1984

Моделирование характеристик транзисторных структур «германий на изоляторе»

Н.В. Масальский

кандидат физико-математических наук

Аннотация: На основе численных решений уравнения Пуассона анализируются основные электрофизические характеристики двух затворных КМОП нанотранзисторов со структурой «германий на изоляторе» и архитектурой «без перекрытия областей затвора и стока/истока». Основное внимание уделяется влиянию ряда технологических параметров на пороговые напряжения и крутизну подпорогового наклона. В анализируемых транзисторных структурах коротко-канальные эффекты проявляются в большей степени, чем в классических структурах КНИ. Нелинейный характер полученных зависимостей обусловлен экспоненциальным ростом объемного заряда как функции потенциала в рабочей области транзистора. Технологические параметры транзисторной структуры позволяют эффективно управлять рассматриваемыми ключевыми характеристиками. Дан небольшой обзор перспектив развития промышленной технологии получения монокристаллического германия.

Ключевые слова: моделирование, транзисторные структуры, германий на изоляторе

Введение

Отличительной чертой развития микроэлектроники на современном этапе является не только неуклонное снижение топологических норм, но и применение новых материалов в транзисторных структурах. В последние годы во всем мире возникла новая задача - использовать монокристаллы германия как основной материал для создания высокоэффективных микросхем [1], так как германию, как известно, присуща намного более высокая подвижность электронов и дырок по сравнению с кремнием.

Россия исторически имеет очень хорошие традиции в выращивании полупроводниковых кристаллов методом Чохральского. В наследство от Советского Союза досталась очень хорошая школа материаловедения, связанная с кремнием и германием. Базируясь на них, в настоящее время развиваются основы промышленной технологии получения бездислокационных монокристаллов германия. Природа не в состоянии создать столь совершенный бездислокационный кристалл кремния или германия. Любой, даже самый дорогой алмаз, рубин содержит множество дефектов. А современные технологии позволяют вырастить слиток кремния диаметром 200 мм и длиной до 2 м, в котором нет дислокаций. Однако, это относится только к кремнию. По физическим свойствам вырастить бездислокационный крупногабаритный монокристалл германия существенно сложнее. Несмотря на то, что кремний и германий материалы очень схожие, но у выращенного кристалла германия и диаметр меньше, и длина короче, и дислокации присутствуют.

Сегодня в России только три предприятия выращивают монокристаллы германия. Следует отметить, что и за рубежом этой технологией владеют немногие компании. Что касается сырья, то после распада Советского Союза производство германия из собственного сырья на какое-то время вообще прекратилось. В 2001 году удалось частично восстановить собственные источники сырья, все промышленные запасы которого сосредоточены в каустобиолитах (углях) трех районов: Приморском крае, Сахалинской и Читинской областях.

В настоящей работе в качестве прототипа транзисторной структуры рассматривается аналогичная традиционной структуре «кремний на изоляторе», в которой кремний заменен на германий. Однако, хорошо известно, что применение тонких слоев обычного окисла германия приводит к большим физическим и технологическим проблемам. Перспективными диэлектриками для использования в транзисторных структурах являются нитриды германия, которые показывают лучшую технологичность, чем обычные окислы германия.

Применение новых материалов приводит к изменению ключевых характеристик транзисторов. В настоящей работе анализируются на основе численных решений такие основные характеристики как пороговое напряжение и наклон подпороговой характеристики применительно к двухзатворным нанотранзисторам с нелегированным каналом с архитектурой без перекрытия областей затвора и стока/истока. Выбор такой архитектуры обусловлен тем, что ее реализация на базе технологии «кремний на изоляторе» обладает рядом ценных практических свойств [2-4].

1. Ограничения на топологические параметры транзистора

Рассматриваемый тип транзисторов, (функциональная схема приведена на рис. 1), как известно, характеризуется эффективным подавлением коротко-канальных эффектов (ККЭ), низким значением емкости и представляет практический интерес для создания низковольтных СБИС с малой потребляемой мощностью [2-4].

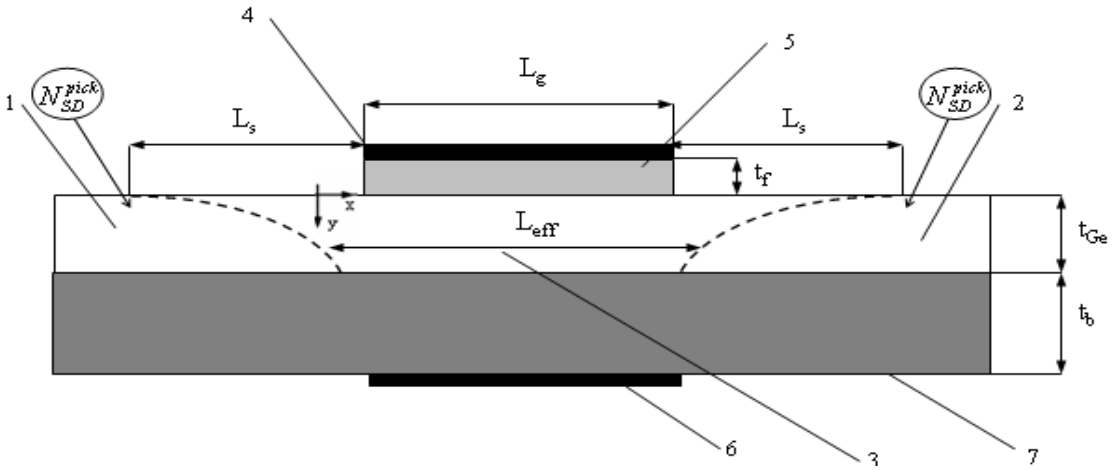


Рис. 1. Схема транзистора с архитектурой без перекрытия, где 1 – область стока, 2 – область истока, 3 - рабочая область, 4 – фронтальный затвор, 5 – фронтальный подзатворный окисел, 6 - обратный затвор, 7 - погруженный окисел. Пунктирными линиями показаны профили концентрации легирования стока/истока, L_g – длина затвора, t_{Ge} – толщина пленки германия (рабочей области), t_f – толщина окисла фронтального затвора, L_s – длина зазора, g – градиент легирования областей стока/истока, N_{SD}^{pick} - максимальная концентрация легирования областей стока/истока.

Для определения области допустимых значений (ОДЗ) топологических параметров для подавления ККЭ необходимо удовлетворить ряду следующих критериев, которые вытекают из физических ограничений, технологических и конструкционных требований [1, 2-6]. (i) $L_{eff}^2 / 2lL_g > 1$ – условие подавления ККЭ, где l - характеристическая длина. (ii) $t_{Ge}^{\min} \geq 5$ нм - 5 нм - минимальное значение толщины рабочей области, так как, такой выбор также отвечает технологическим требованиям ограничения при создании тонких пленок без дефектов. (iii) $g \geq 2$ нм/дес – технологическое требование (значения g<2 нм/дес труднореализуемы, что требует использование нестандартного техпроцесса). (iv) 5.5 нм $\leq \sqrt{2\eta g L_g / \ln(10)} \leq$ 8.5 нм – конструкционное требование, определяющие крутизну профиля примеси областей сток/исток в продольном направлении, где $\eta = L_s / L_g$.

Необходимость определения ОДЗ возникает на первоначальном этапе проекта при выборе транзисторной структуры/технологии. При этом допустимый диапазон градиента легирования должен быть более тщательно проанализирован, как этого зависит весь технологический процесс. Из условия (i) в приближении $L_{eff} / 2l = 1$ следует, что минимальная ширина относительного зазора $\eta_{\min}$, связанная с топологическими параметрами транзисторной структуры для подавления ККЭ, может быть вычислена как

$$\eta_{\min} - \sqrt{\eta_{\min}} \sqrt{\frac{2g}{L_g \ln 10}} \sqrt{N_{SD}^{norm}} = \frac{2l - L_g}{2L_g}, \quad (1)$$

где - N_{SD}^{norm} нормированное концентрации легирования областей стока/истока при которой экстрагируется канал.

Аналогично, оценку $\eta_{\min}$ следует получить из условия (iv). Совокупность этих результатов обеспечивает простой и эффективный метод оценки минимальной ширины зазора для выбранной транзисторной структуры/технологии с подавлением ККЭ. Пренебрежение одной из оценок может привести к проявлению ККЭ и возникновению дополнительного последовательного сопротивления (из-за вклада не перекрытых областей затвор-сток/исток) которое ухудшит производительность устройства [4]. Эта методика также позволяет получить эффективную оценку диапазона градиента легирования для ограничения ККЭ для данной топологии.

2. Поведение характеристической длины

Исследуем поведение характеристической длины l, которая, как известно, в модельных представлениях является индикатором проявления ККЭ в рассматриваемых транзисторных структурах [2]. Значение параметра l определяют из анализа распределения потенциала в рабочей области транзистора. Данное распределение является решением 2D уравнения Пуассона. В общем случае решения уравнения Пуассона связаны с решением характеристического уравнения для нахождения собственных значений, которое следует из условий непрерывности потенциала и электрического поля на границах рабочей области. В рассматриваемом случае следуя [6] характеристическое уравнение для собственных значений запишем, например, в виде:

$$\tan \lambda_n t_{Ge} = \frac{\varepsilon_{Ge}(C_f + C_b)\lambda_n}{\varepsilon_{Ge}^2 \lambda_n^2 - C_f C_b}, \qquad (2)$$

где ε_{Ge} - диэлектрическая проницаемость германия, t_{Ge} - толщина рабочей области транзистора, C_f – емкость фронтального затвора, C_b – емкость обратного затвора, где также показано, что величина характеристической длины связана с собственными значениями следующим выражением $l = \frac{\pi}{\lambda_1}$ [7].

Численно рассчитанные зависимости характеристической длины от толщин как рабочей области так и фронтального затвора приведены на рис. 2 и 3.

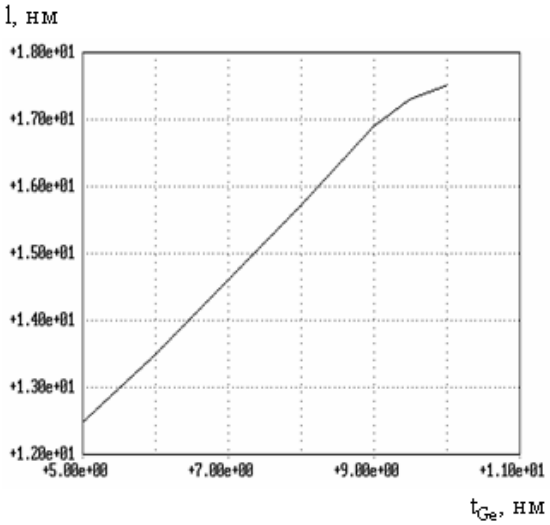


Рис. 2. Характеристическая кривая характеристической длины от толщины пленки t_{Ge} (рабочей области) для толщины подзатворного окисла 1 нм

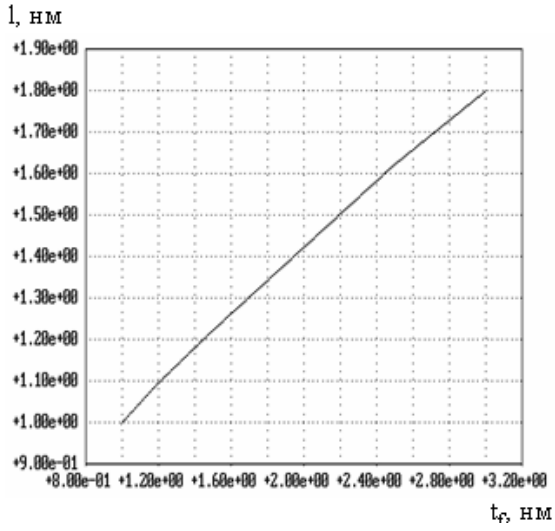


Рис. 3. Характеристическая кривая нормированной характеристической длины от толщины подзатворного окисла
для толщины пленки t_{Ge} 5 нм.
Для диапазона t_{Ge} от 5 нм до 10 нм, в относительных единицах данная зависимость инвариантна.

Следуя методики рассмотренной в [6], по двум данным зависимостям (рис. 2 и 3) можно определить собственное значение для любой комбинации t_{Ge}, t_f из области допустимых значений по правилу $l = l(t_{Ge})l(t_f)$.

Опираясь как на результаты работ [2, 6, 8], так и на представленные на рис. 2 и 3 можно оценить степень проявления ККЭ при использовании различных материалов формирующих транзисторную структуру. Случаем наиболее неуязвимым от проявления ККЭ является традиционная структура КНИ, которая при толщине рабочей области 5 нм и толщиной фронтального затвора 1 нм значение характеристической длины составляет 9.9 нм, для структуры «германий на изоляторе» 12.6 нм и КНИ с фронтальным затвором окиси гафния 13.3 нм. С точки зрения создания высокоэффективных микросхем переход на германий более перспективен по сравнению со структурами КНИ с фронтальным затвором из окиси гафния. Поскольку с учетом условия (i) из предыдущего пункта в структурах «германий на изоляторе» ККЭ будут проявляться при меньших длинах затвора, что в совокупности с более высокой подвижностью обуславливает выигрыш в быстродействии и рассеиваемой мощности [9].

3. Пороговое напряжение

В квазиклассическом приближении пороговое напряжение U_{th} определим как затворное напряжение при котором минимальная индуцированная плотность инверсионных зарядов достигает значения заряда Q_{th} адекватного равного для обеспечения проводимости (генерации канала) [2], которое аппроксимируется выражением

$$Q_{th} = \int_0^{t_{Ge}} n_i e^{q\Psi_{\min}(y)/kT} dy, \qquad (3)$$

где n_i – индуцированная плотность носителей в канале, $\Psi_{\min}$ - минимум канального потенциала, который может быть представлен в аналитическом виде [6, 8].

Следует отметить, что соотношении (3) эквивалентно методу, который применяется для экстракции порогового напряжения из результатов измерений. В приведенных допущениях пороговое напряжение для транзисторов с архитектурой «без перекрытия» можно представить в виде двух составляющих длинно-канальной и коротко-канальной компонент, по форме аналогичной для транзисторов КНИ [4].

Результаты моделирования для прототипа транзистора n-типа с длиной затвора 22 нм, толщиной рабочей области 6 нм, толщинами подзатворного и погруженного окислов 1.2 нм и 25

нм, приведенные на рис. 4, иллюстрируют зависимость U_{th} от различных значений η, g .

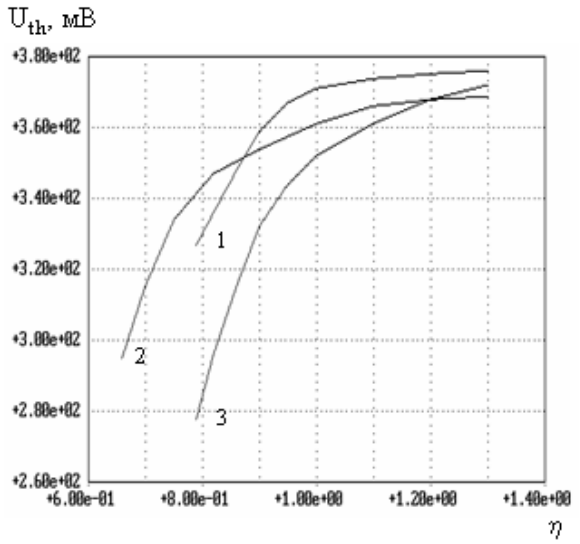


Рис. 4. Зависимость порогового напряжения U_{th} от η, где кривая 1 - g=2 нм/дек, 2 – g=3 нм/дек, 3 – g=4 нм/дек

Нелинейный характер полученных зависимостей обусловлен тем, что поскольку концентрация носителей экспоненциально зависит от уровня потенциала в рабочей области, а в свою очередь распределение потенциала существенно зависит от величины параметра η (а в общем случае и от значения характеристической длины l), то это приводит к проявлению таких нелинейностей.

Представленные данные позволяют оценить диапазон изменения порогового напряжения при переходе к более крутым профилям легирования стока/истока. В общем случае с уменьшением зазора пороговое напряжение U_{th} понижается. Снижение η определяет возрастание влияния ККЭ, которое вызывает характерное изменение U_{th}. Этот эффект roll-off усиливается с увеличением градиента легирования. При этом для больших зазоров ($\eta > 1$) он проявляется в меньшей степени, чем для $\eta < 1$. Большие значения η приводят к значениям U_{th}, которые почти независимы от η и слабо зависимы от g. В анализируемом диапазоне абсолютная величина снижения порогового напряжения (ΔU_{th}) для g=2 нм/дес составляет 49 мВ, для g=3 нм/дес ΔU_{th} =74 мВ и для g=4 нм/дес ΔU_{th} =94 мВ. Следовательно, с ростом значения градиента крутизна зависимости $U_{th}(\eta)$ увеличивается. В соответствии с критерием проявления ККЭ в рассматриваемом ОДЗ контролируется степень влияния ККЭ.

4. Подпороговый наклон

Из анализа распределения потенциала можно получить приемлемую оценку величины наклона подпороговой характеристики. Предположение, что подпороговый наклон (S-наклон) связан с концентрацией носителей в точке минимума потенциала и следовательно с самим потенциалом Ψmin то, тогда S-наклон может быть получен как

$$S = \frac{\partial U_f}{\partial \log I_{ds}} \approx \ln(10)\frac{\partial U_f}{\partial \ln(n_{\min})} = \ln(10)\frac{kT}{q}\left(\frac{\partial \Psi_{\min}}{\partial U_f}\right)^{-1} \quad (4)$$

где U_f – напряжение на фронтальном затворе, n_{min} - концентрацией носителей в точке минимума потенциала q - заряд электрона, k - постоянная Больцмана, T - температура.

Этот подход для оценки S-наклона широко использовался в нескольких предшествующих опубликованных работах [2, 4, 10, 11]. Результаты моделирования исследуемого прототипа транзистора приведенные на рис. 5 и 6, иллюстрируют зависимость S от различных значений η, g .

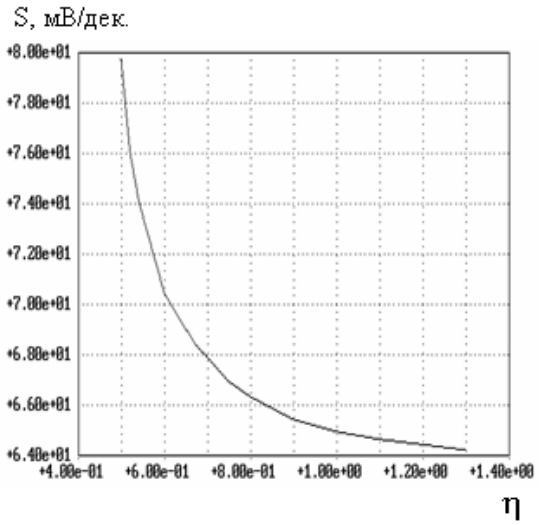


Рис. 5. Характеристическая зависимость наклона подпороговой характеристики от η для g=3 нм/дек .

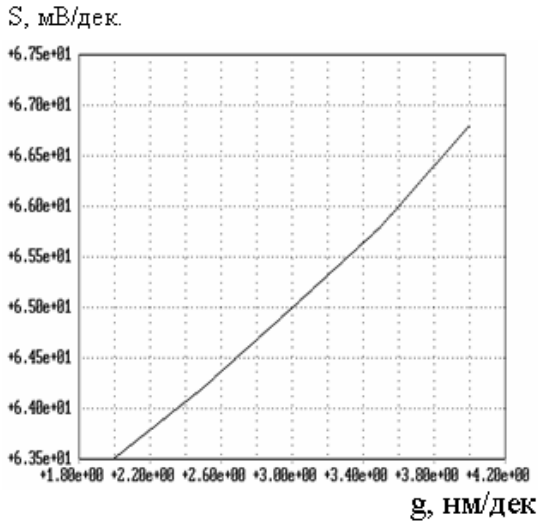


Рис. 6. Характеристическая зависимость наклона подпороговой характеристики от g для η =1.

Из результатов расчетов следует, что подпороговый наклон близкий к теоретическому приделу достигается только для случая градиента легирование

2 нм/дек и $\eta \geq 1$. Тогда как выше более высоких значений g возрастание величины S-наклона характеризуется практически линейной зависимостью. В области $\eta \geq 1$ для рассматриваемого диапазона градиентов легирования значение подпорогового наклона практически не зависит от η. Для η<1 зависимость S(η) круто возрастает начиная с некоторого значения η, характерное для конкретного значения g. Например, для g=2 нм/дек η=0.73, g=3 нм/дек η=0.84, g=4 нм/дек η=0.95.

Заключение

В работе на основе численных решений уравнения Пуассона проанализированы основные электрофизические характеристики двух затворных КМОП нанотранзисторов со структурой «германий на изоляторе» и архитектурой «без перекрытия областей затвора и стока/истока». Исследована степень влияния ряда технологических параметров на пороговые напряжения и крутизну подпорогового наклона. В общем случае нелинейный характер полученных зависимостей обусловлен экспоненциальным ростом объемного заряда как функции потенциала в рабочей области транзистора, что связано с проявлением коротко-канальных эффектов проявляются в большей степени, чем в классических структурах КНИ. В общем случае с уменьшением зазора пороговое напряжение резко понижается. Этот эффект roll-off значительно усиливается с увеличением градиента легирования областей стока и истока. Большие величины зазора приводят к значениям порогового напряжения, которые почти независимы как зазора, так и от градиента легирования. Подпороговый наклон близкий к теоретическому приделу достигается только для случая градиента легирование 2 нм/дек и большим зазором. Для маленьких зазоров подпороговый наклон круто возрастает начиная с некоторого значения, характерного для конкретного значения градиента легирования. Для выбранных топологических норм оптимизация параметров определяющих области стока и истока предоставляет дополнительную степень свободы управления, как пороговым напряжением так и наклоном подпороговой характеристик наряду с толщиными рабочей области и подзатворного окисла фронтального затвора, что важно при анализе применимости транзисторных структур «германий на изоляторе».

Modeling characteristics of transistor structures “germanium on insulator”

N.V. Masalsky

Astract: On the basis of numerical solutions of a Poisson equation the main electro-physical characteristics of double gate CMOS nanotransistors with structure "germanium on an insulator" and architecture "underlapp design" are analyzed. The main attention is paid to influence of a row of technological parameters on threshold voltages and the steepness of a subthreshold inclination. In analysable transistor structures it is short - channel effects are shown more than in classical SOI structures. Non-linear nature of the received dependences is caused by the exponential growth of a volume charge as potential functions in work area of the transistor. Technological parameters of transistor structure allow to control considered key characteristics effectively. The small review of perspectives of development of industrial technology of receiving monocrystal germanium is this.

Keywords: modeling, transistor structures, «germanium on insulator»

Литература

1. International technology roadmap for semiconductor 2012 edition. Available from: (http://public.itrs.net)

2. A.Kranti, G.Armstrong. Engineering source/drain extension regions in nanoscale double gate (DG) SOI MOSFETs: Analytical model and design considerations // *Solid-State Electronics*.- 2006, № 2 (50), pp. 437 - 447

3. A.Kranti, Y.Hao, G.Armstrong. Performance projections and design optimization of planar double gate SOI MOSFETs for logic technology applications // *Semiconductor Science and Technology*.- 2008, № 4 (23), pp. 217-224

4. Н.В.Масальский. Оптимизация параметров двух затворных суб- 20 нм КНИ КМОП транзисторов с архитектурой «без перекрытия» // *Микроэлектроника*, 2012, № 1(41), с. 57-64

5. C.Chui, F.Ito, K.Saraswat. Scalability and electrical properties of germanium oxynitride MOS dielectrics // *IEEE Electron Devices Letts*.- 2004, № 9 (25), pp. 613-615

6. Н.В.Масальский. Синтез характеристик логических вентилей на двух затворных суб-25 нм КНИ КМОП транзисторах для маломощных применений // *Нано- и микросистемная техника*. 2010, № 5(118), с. 41-46

7. X. Liang., Y.Taur. A 2-D Analytical Solution for SCEs in DG MOSFETs // *IEEE Trans Electron Devices*.- 2004, № 7 (51), pp. 1385-1391

8. Н.В.Масальский. Пороговые характеристики КНИ КМОП нанотранзисторов с высокой диэлектрической проницаемостью подзатворного диэлектрика // *Труды НИИСИ РАН* , 2012, т. 2, № 2, с. 70-77.

9. H. Shang., M.M.Frank, E.P.Gusev, J.O.Chu, S.W.Bedell, K.W.Guarini, M.Ieong. Germanium channel MOSFETs: opportunities and challenges // *IBM J. Res. Develop.*, 2006, № 4 (50), pp. 377–386.

10. Q.Chen, B.Agrawal, J.D.Meindl. A comprehensive analytical subthreshold swing (*S*) model for double-gate MOSFETs // *IEEE Trans Electron Devices*, 2002, № 6 (49), pp. 1086-1090.

11. D.Munteanu, J.L.Autran, S.Harrison. Quantum short-channel compact model for the threshold voltage in double-gate MOSFETs with high-permittivity gate dielectrics // *Journal of Non-Crystalline Solids*, 2005, № 21 (351), pp. 1911–1918.

2D математическая модель подпорогового тока суб 20 нм двухзатворных КМОП транзисторах

Е.Н. Епихин[1], Н.В. Масальский[1]

1-кандидат физико-математических наук.

Аннотация: Обсуждается 2D аналитическая модель распределения подпорогового тока для суб 20 нм двухзатворных КМОП транзисторов со структурой «кремний на изоляторе» и «германий на изоляторе». Модель получена непосредственно из решения уравнений Пуассона и Шреденгера. Результаты моделирования предсказывают, что масштабирование по толщине рабочей области приводит к резкому (на несколько порядков) ограничению тока утечки. При этом структуры германий на изоляторе характеризуются очень высоким уровнем подпорогового тока по сравнению с кремниевыми структурами. Масштабирование по длине затвора приводит к резкому возрастанию уровня подпорогового тока, что ограничивает масштабируемость транзисторных структур. На границе области масштабирования толщина рабочей области не может превышать 4 нм.

Рассматривается компактная модель для схемотехнических приложений, которая получена из 2D модели при помощи ряда физических упрощений. Она в полной мере отражает физику подпорогового тока двух затворных суб 20 нм КНИ транзисторов. Максимальная ошибка рассогласования составляет 12%. С помощью разработанной модели анализируется задача оценки влияния флуктуации топологических параметров, в частности длины затвора и толщины рабочей области, на разброс подпорогового тока. Получено хорошее согласование в пределах 10% результатов моделирования по методу Монте-Карло расчетов, выполненных по предложенной модели.

Ключевые слова: модель подпорогового тока, двухзатворный КМОП транзистор, структура «кремний на изоляторе»

Введение

Мультизатворные транзисторные архитектуры - предшественники устройств, которые должны последовательно заменить классические объемные кремниевые КМОП транзисторы при переходе к 10 нм топологическим нормам [1]. Квазипланарные структуры, такие как двух затворные КНИ, Tri-gate and Omega-FET среди самых популярных конструкций устройств, в основном из-за их относительной простоты и технологичности [2-5]. Они также отличаются расширенной возможностью использовать более толстый подзатворный диэлектрик, превосходящие коротко-канальные характеристики, улучшение мобильности в нелегированной тонкой рабочей области и устранении случайных эффектов легирующей примеси.

Агрессивно-жесткие сценарии масштабирования полупроводниковых устройств, устанавливаемые ITRS (International Technology Roadmap for Semiconductors) [1], задачу точного детального моделирования характеристик транзисторов у границы области масштабирования усложняют в разы по сравнению с моделированием транзисторных структур даже с 65 нм топологическими нормами [6]. К настоящему моменту опубликовано ряд работ связанных с моделированием двух затворных суб 20 нм КНИ КМОП транзисторов, которые посвящены как численным, так и аналитическим методам моделирования [7-11]. Точное моделирование исследуемого типа транзисторов требует самосогласованного решения уравнений Пуассона и Шреденгера на основе, например, формализма неравновесных функций Грина (NEGF), учитывающие в полной мере квантовые эффекты [10]. Но с точки зрения проектирования наноразмерных КМОП схем использование 2D NEGF метода - это эксцесс и с точки зрения сложности и с точки зрения вычислительной стоимости [10]. Поэтому, необходимы новые методы и технологии, чтобы преодолевать эти ограничения.

Одной из важнейших характеристик двух затворных суб 20 нм КНИ КМОП транзисторов является подпороговый ток (ток утечки). Известно, что физика данного устройства отличается от традиционного объемного КМОП транзистора. Два основных физических явления, которые влияют на ток утечки, являются коротко-канальные эффекты (ККЭ) и эффекты размерного квантования (ЭРК).

Тонкопленочные двух затворные транзисторы характеризуются почти полностью инвертированной рабочей областью, т.е. электрические токи всюду протекают по ультратонкому телу. Потенциал в центре рабочей области, который является областью, наименее управляемой каждым из затворов, повышается из-за влияния стока. Что эквивалентно, понижению барьера исток-сток, и утечка увеличивается. Это - классический ККЭ. В данном случае максимальный ток течет вдоль области максимального потенциала, т.е. в центре рабочей области. 2-D распределение потенциала получено как решение уравнения Пуассона [12]. Главенствующая роль в подходах моделирования отводится анализу распределения потенциала, поскольку оно определяет ключевые характеристики транзистора. Так, Нои [13] представил модель двумерного (2D) распределения потенциала в рабочей области транзистора при помощи использование метода функции Грина. Chiang [14] решил 2D уравнение Пуассона при помощи

метода суперпозиции. Несмотря на доказанную точность этих моделей, они включают бесконечные ряды, которые увеличивают математическую сложность. Напротив, модель Young' s основана на предложение параболического распределение потенциала в тонкой кремниевой пленке, что практичнее при схемотехнических расчетах.

Ограничение носителей в ультратонком теле приводит к квантованию энергетических уровней и конечной вероятности размещения этих уровней, которые определяются решением волнового уравнения. Квантованные уровни характеризуются более высокой энергии, чем классические уровни. Это увеличивает барьер исток-сток, таким образом, уменьшая утечку. Как волновые функции, так и квантованные энергетические уровни являются решениями уравнения 1D Шреденгера. Фактически, задача может быть приближена к известному случаю частицы в бесконечной потенциальной яме [15].

Как следует из сказанного выше, самосогласованное решение уравнений Пуассона и Шреденгера требуется для любого устройства, чтобы надлежащим образом объяснить классические и квантовые эффекты. В данной работе, предлагается подход аналитического 2D моделирования подпорогового тока двух затворных КНИ КМОП транзисторов, основанный на аналитическом самосогласованном решении уравнений Пуассона и Шреденгера. А также рассматриваются ее приложения. Кроме того, суб 20 нм КНИ КМОП транзистор представляет вызовы аналитическому схемотехническому моделированию, связанному с расширенной связью между контактами (сток, исток и затвор), квантовое ограничение, баллистический транспорт, затворный туннельный ток, и т.д. [5, 7, 11, 16]

1. Аналитическая модель

Обобщенная 2D аналитическая модель подпорогового тока для полностью обедненных двух затворных КНИ транзисторов с размером кластера менее 20 нм представленная в этой работе совмещает решение 2D уравнение Пуассона дополненное решением 1D уравнения Шреденгера полученное Trivedi [15], для рассматриваемого случая. Она учитывает как ККЭ так и ЭРК. Итоговое выражение для подпорогового тока транзистора в режиме слабой инверсии может быть записано как:

$$I_{sub} = \frac{\mu kT(1-e^{\frac{-qU_{ds}}{kT}})}{\int_0^{L_g} \frac{dy}{\int_{-t_s/2}^{t_s/2} n_c(x,y)dx}} \qquad (1)$$

где q – заряд электрона, μ - подвижность носителей, L_g – длина затвора, t_s - толщина рабочей области транзистора, T – температура, k – константа Больцмана, U_{ds} - напряжение сток-исток, функция $n_c(x, y)$ в интеграле представляет распределение эффективной концентрации носителей по всему объему рабочей области: $n_c(x,y) = n_c^{QM}(x)e^{-\frac{q\varphi(x,y)}{2kT}}$

Распределение потенциала в рабочей области транзистора подчиняется выражению:

$$\varphi(x,y) = (\frac{(u_{bi}+l^2A_f)(e^{\frac{L_g}{l}}-1)-U_{ds}}{e^{\frac{L_g}{l}}-e^{-\frac{L_g}{l}}}e^{-\frac{y}{l}} + \frac{(u_{bi}+l^2A_f)(e^{-\frac{L_g}{l}}-1)-U_{ds}}{e^{-\frac{L_g}{l}}-e^{\frac{L_g}{l}}}e^{\frac{y}{l}} - l^2A_f) (1+\frac{C_f}{\varepsilon_s}x-\frac{x^2}{l^2}) + x\frac{C_f}{\varepsilon_S}[-U_f+U_{FB}^f] + \frac{A_f}{2}x^2 \qquad (2)$$

и

$$A_f = \frac{eN_A}{\varepsilon_S} - 2\frac{C_f(1+\frac{C_b}{C_S})(U_f-U_{FB}^f)+C_b(U_b-U_{FB}^b)}{t_s^2(C_b+2C_s)} \qquad (3)$$

где $l = t_S\sqrt{\frac{(1+2\frac{C_S}{C_b})}{2(1+\frac{C_f}{C_S}+\frac{C_f}{C_b})}} = \frac{t_S}{\sqrt{2}}\sqrt{\frac{t_f(1+2\varepsilon_{отн}\frac{t_b}{t_S})}{t_f+t_b+\varepsilon_{отн}^{-1}t_S}}$ - характеристическая длина, $\varepsilon_{отн} = \frac{\varepsilon_S}{\varepsilon_{ox}}$, ε_s - диэлектрическая рабочей области, N_A – концентрация легирования рабочей области, C_s – емкость рабочей области, C_f – емкость фронтального затвора, C_b – емкость обратного затвора, t_f - толщина подзатворного диэлектрика фронтального затвора, t_b - толщина подзатворного диэлектрика обратного затвора, U_f - напряжение на фронтальном затворе, U_b - напряжение на обратном затворе, U_{Fb_f} - напряжение плоских зон фронтальном затворе, U_{bi} - встроенная разность потенциалов.

1D распределение эффективной концентрации носителей с учетом зонной структуры [15, 17]:

$$n_i^{QM}(x) = e^{-\frac{E_g}{2kT}}\frac{kT}{\pi\hbar^2}\sum_{ij} g_i m^* e^{-\frac{qE_{ij}}{kT}}\left|\Psi_j(x)\right|^2 \qquad (4)$$

и

$$\left|\Psi_j(x)\right|^2 = \frac{2}{t_S}\left[\sin(\frac{j\pi(x+\frac{t_S}{2})}{t_S}\right]^2 \quad (5)$$

где E_g – ширина запрещенной зоны, h – постоянная Планка, m* – эффективная масса, E_{ij} - $E_{ij} = \frac{j^2h^2}{8m_i^2t_S^2}$ - энергия собственных значений (здесь, индекс i представляет долину (продольный или поперечный), и j - индекс поддиапазона).

Следует отметить, что 2D потенциал $\varphi(x,y)$ в рабочей области получен из решения уравнения Пуассона методом разделения переменных [12]. Концентрация носителей $n_i^{QM}(x)$ вычислена исходя из плотности состояний и вероятности размещения на основе 1D решения волнового уравнения. Волновая функция Ψ (x) и энергия собственных значений E_{ij} получены из решения уравнения Шреденгера, полагая что носители ограничены в бесконечной потенциальной яме ширины t_s [15].

Подвижность носителей в КНИ нанотранзисторах является функцией электрического поля в его рабочей области. При этом считают независимое факторизованное разделение поля вдоль оси *y* и вдоль оси *x* на продольную и поперечную компоненты [18]:

$$\mu = \mu(E_x)\mu(E_y), \quad (6)$$

где $\mu(E_x)$, $\mu(E_y)$ - зависимости подвижности как функции напряженности поперечного и продольного полей соответственно. Следует отметить, что учет распределения поля в модели подвижности основан на экспериментальных данных и является эмпирическим. Тогда выражение для $\mu(E_x)$ имеет вид $\mu(E_x) = \frac{\mu_o}{1+\vartheta_E\left|\frac{\partial\phi(x,y)}{\partial x}\right|}$, где μ_o – значение подвижности в области малых полей, ϑ_E – коэффициент деградации подвижности, выбираемый из условий эксперимента. Выражение для $\mu(E_y)$ имеет вид $\mu(E_y) = \frac{\mu(E_x)E_c}{E_c+E_y} = \frac{2v_{sat}\mu(E_x)L}{2v_{sat}L+\mu(E_x)U_{ds}}$, где $E_c = \frac{2v_{sat}}{\mu_x}$ - напряженность критического поля, v_{sat} - дрейфовая скорость насыщения.

Анализ данного выражения показывает, что подвижность носителей ограничена величиной $\mu(E_x)$ при нулевом смещении транзистора. Для случая больших продольных полей подвижность нанотранзистора стремится к величине $\frac{2v_{sat}L}{U_{ds}}$, то есть не зависит от поперечного поля. Заметим, что первоначально необходимо вычислить поперечное значение подвижности, а затем - продольное.

Модельный подход, рассмотренный в этом разделе, не ограничен выбранными решениями потенциала и волновой функции и могу быть применены другие решения, такие как полученные, например, в [19]. Практически, данный подход может быть расширен на все тонко пленочные полностью обедненные устройства.

2. Результаты моделирования

Ниже приведены результата моделирования, выполненные по представленной выше модели. Они помогают проанализировать тенденции изменения подпоргового тока при масштабировании топологических параметров в соответствии с ITSR2014 [1] и использовании различных материалов рабочей области. На рис. 1-4 показаны обобщенные характеристики прототипа с длиной канала L_g = 15 нм, толщиной подзатворного диэлектрика t_f = 0.7 нм и пониженном напряжении питания U_{dd} для транзисторных структур КНИ и германий на изоляторе.

Следует отметить, что прямая зависимость подпорогового тока от смещения на контактах или от топологических параметров имеет в общем случае не монотонный характер. Вклад диффузных и туннельных токов существенно может меняться в рамках одного образца. Что в свою очередь отражается на таком ключевом параметре как пороговое напряжение. Поэтому мы представляем результаты, где такие корректировки учтены, при этом, исключая из анализа ряд других механизмов утечки, например, туннельный ток затвора.

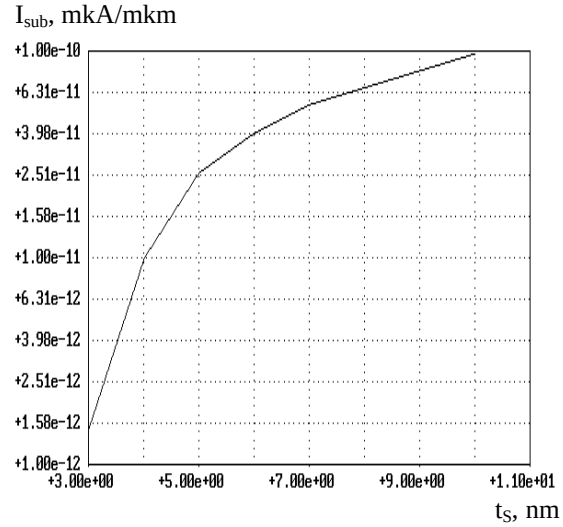


Рис. 1. Зависимость подпорогового тока КНИ от толщины t_S для Lg=15 нм, Udd=0.5 В

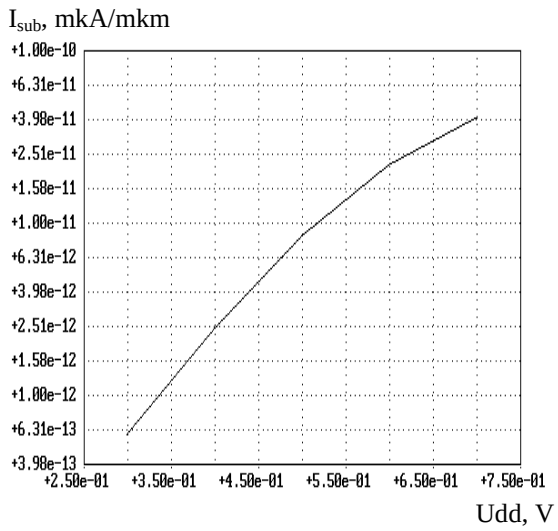


Рис. 2. Зависимость подпорогового тока КНИ от Udd для Lg=15 нм, t_S=5 нм

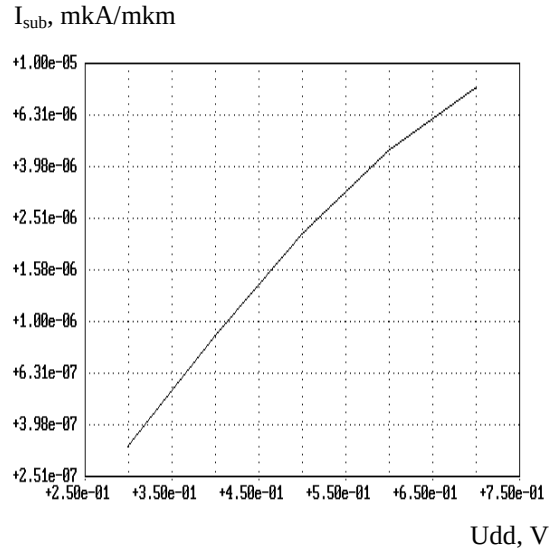


Рис. 3. Зависимость подпорогового тока от Udd, германий на изоляторе, Lg=15 нм, t_S=5 нм

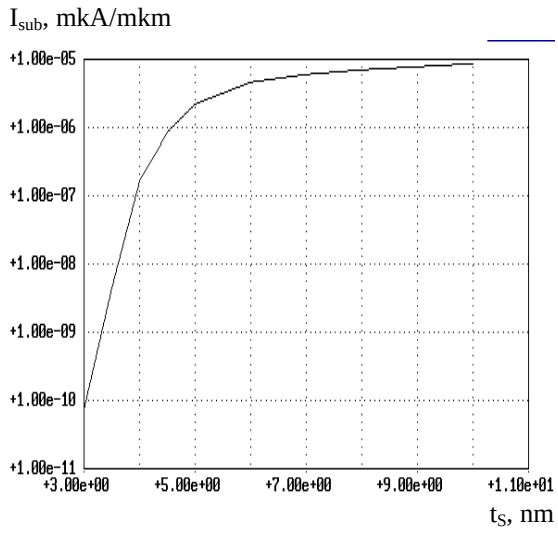


Рис. 4. Зависимость подпорогового тока от толщины t_S, германий на изоляторе, Lg=15 нм, Udd=0.5 В

Германий - материал с узкой запрещенной зоной, поэтому транзисторы на его основе имеют чрезвычайно большой ток утечки выше, чем 0.1μA/μm в отличии от кремневых структур (рис. 1 и рис. 2). Однако, если технология позволяет снизить толщину рабочей области меньше 4 нм, возможно ограничить I_{sub} ниже 0.1μA/μm. Масштабирование параметра t_s улучшает производительность устройства, подавляя ККЭ и уменьшая вклад туннельных токов утечки. Как следует из результатов моделирования, уровень I_{sub} в структурах на германии может быть снижен до 1000 раз при масштабировании t_s до 3 нм (Рис. 4), из-за увеличенной запрещенной зоны квантованием поддиапазонов [8]. Так при масштабировании t_s, ток I_{on} (ток насыщения) увеличивается в 2.5 раза, но одновременно электроны становятся более тяжелыми, что может привести к потере преимущества по сравнению с кремнием [20].

Масштабирование напряжения питания U_{dd} – положительный момент для обоих исследуемых материалов т.к. оно существенно снижает вклад туннельных токов в утечку (рис. 2 и рис. 3) посредством уменьшения максимального электрического поля в рабочей области, что также уменьшает энергию переключения. Это приведет к тому, что структуры на германии при напряжении питания U_{dd}=0.5 В будут обеспечивать более высокий ток I_{on}, чем кремниевые при U_{dd}=0.7 В. Следовательно, транзистор на германии будет использовать в идеальном случае только чуть больше половины активной энергии, расходуемой транзистором на Si. Однако, возможные большие пороговые напряжения для некоторых структур на германии [21] препятствуют тому, чтобы U_{dd} масштабировался ниже 0.4 В где они начинают управлять меньшими токами, чем в кремнии.

На рис. 5 иллюстрирует изменение подпорогового тока при масштабировании длины затвора КНИ транзисторных структур.

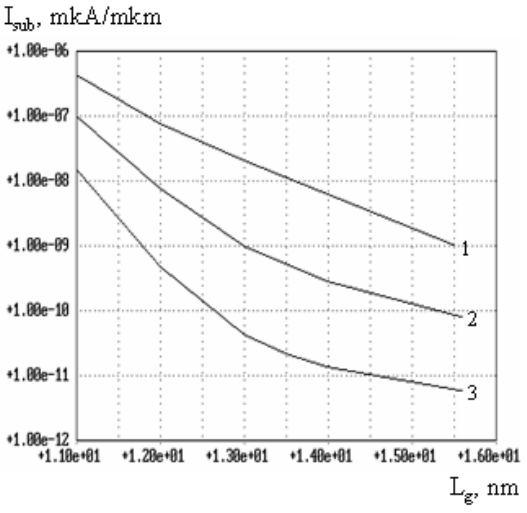


Рис. 5. Зависимость подпорогового тока КНИ от длины затвора при разной t_s , где 1- t_s =5 нм, 2- t_s =4 нм, 3- t_s =3 нм. Подзатворный диэлектрик t_f = 1 нм, U_{dd} = 0.8 В.

Теоретически масштабирование параметра L_g улучшает быстродействие электронных устройств. Однако, результаты моделирования предупреждают о негативных последствиях, связанных с резким возрастанием подпорогового тока. Ситуация на границе области масштабирования для КНИ при L_g = 11 нм соответствует случаю структур германий на изоляторе, где ограничить данный ток можно только для t_s ниже 4 нм. И если для кремния хорошо исследованы изменения зонной структуры, то зонные структуры ультра коротких транзисторов германий на изоляторе, в частности, отклонение ее от параболического профиля [22], требуют более детального изучения, с использованием атомистического моделирования.

3. Схемотехническая модель

Модельные уравнения (1)-(5) включают интегралы и сложные функциональные формы, и не могут быть использованы непосредственно, чтобы получить аналитическую модель для схемотехнических приложений. Ключевое наблюдение здесь состоит в том, что физическая зависимость I_{sub} может быть получена, считая только для потенциала в центре ультратонкого слоя (x = 0), и наверху барьера исток-сток (y = y_{top}). Это вызвано тем, что x-интеграл в (1) в основном определяется значением подынтегрального выражения при x = 0, и y-интеграл - подынтегральным выражением при y=y_{top} (вершина барьера исток-сток). Поэтому, потоки максимального тока в центре тела (дальше всего от затвора) и его величина определяются в основном самой маленькой концентрацией носителей в y-направлении, наверху барьера исток-сток [12]. Таким образом, плотность тока может быть выражена так:

$$I_{sub} = \mu kT n_i^{QM}(0) e^{\frac{q\varphi(0,y_{top})}{kT}} (1 - e^{-\frac{qU_{ds}}{kT}}) \qquad (7)$$

Важно отметить, что два члена $e^{\frac{q\varphi(0,y_{top})}{kT}}$ и $n_c^{QM}(0)$ представляют ККЭ и ЭРК соответственно. Рис. 5 иллюстрирует соответствие результатов моделирования по аналитической модели (6) и полученных из 2D моделирования для транзистора с L_g=15 нм. Максимальная ошибка рассогласования составляет 12%. Следует отметить, что в данном случае не использовались подгоночные коэффициенты. Для более длинных в широком диапазоне длин затворов и значения толщины рабочей области согласование результатов еще лучше.

Эта упрощенная модель в полной мере отражает физику подпорогового тока двух затворных КНИ транзисторов и может быть применима для схемотехнических приложений, а также для решения других задач.

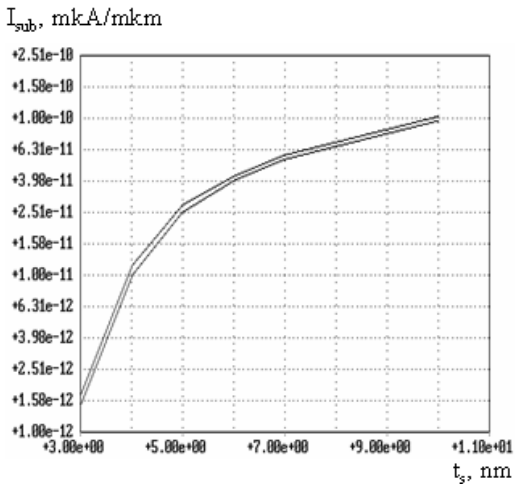


Рис. 6.Зависимость подпорогового тока КНИ от толщины t_s для L_g=15 нм, U_{dd}=0.5 В, где верхняя кривая – схемотехническая модель, нижняя – 2D модель

4. Влияние флуктуаций топологических параметров на уровень тока утечки

Не масштабируемость подпорогового тока вынуждает проанализировать задачу влияние флуктуаций топологических параметров транзисторов на уровень подпорогового тока. В контексте данной работы рассмотрим влияние двух параметров L_g и t_s на исследуемый ток. Для дальнейшего анализа будем использовать соотношение (7). Чтобы упростить процедуру исследования, мы рассматриваем следующее выражение

$$\Delta \log I_{sub} = \frac{q\Delta\varphi(0,y_{top})}{kT} + \log \frac{n_i^{QM}(0)}{n_{i_{nom}}^{QM}(0)} \qquad (8)$$

Каждое из этих слагаемых раскладываем ряд Тейлора с номинальными значениями $L_{g\,nom}$ и $t_{s\,nom}$. Например,

$$\frac{q\Delta\varphi(0,y_{top})}{kT} = f_L^1(L_g - L_{g_{nom}}) + f_s^1(t_s - t_{s_{nom}}) + \\ + \frac{f_L^2}{2!}(L_g - L_{g_{nom}})^2 + \frac{f_s^2}{2!}(t_s - t_{s_{nom}})^2 + \ldots \qquad (9)$$

где f_L^i - i-тая частная производная от $e^{\frac{q\varphi(0,y_{top})}{kT}}$ относительно L_g, $f_{t_s}^i$ - относительно t_s, и т.д. Данный подход применяется и для второго слагаемого в (8). Это приближение допустимо, потому что значения L_g и t_s, как предполагается, сконцентрированы вокруг их номинальных значений в области ограниченной 20%-ным отклонением. Изменение функции Δlog Ioff достаточно медленное, поэтому, ограничиваемся первыми пятью членами

разложения. Среднее значение случайной переменной Δlog Ioff может быть вычислено следующим образом:

$$E(\Delta \log I_{sub}) = \\ = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} dL_g dt_s p_{L_g} p_{t_s} = \sum_{i,j} \alpha_{ij} M_i(L_g) M_j(t_s) \quad (10)$$

где p_{L_g} и p_{t_s} - функции плотности вероятности, и $M_i(L_g)$ и $M_j(t_s)$ являются i-м и j-м центральными моментами распределения L_g и t_s, соответственно.

Полагаем, что распределения нормальны и независимы. Следовательно, распределение значений тока I_{sub} тоже будет нормально. Этот вывод подтверждается результатами моделирования по методу Монте-Карло [23]. Рис. 6 иллюстрирует хорошее соответствие между флуктационным распределением уровней токов I_{sub}, полученным по методу моделирования Монте-Карло и по модели (7) для демонстрационного случая. При этом значение средних (10) хорошо согласуются.

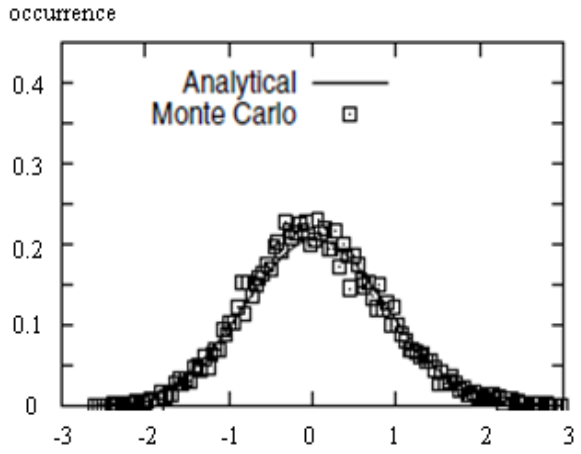


Рис. 7. Распределение флуктуации подпорогового тока для случая L_g= 25 нм, t_s=12.5 нм, 3 σ =20%, где σ - стандартное отклонение.

С другой стороны, для простоты сравнения установим, что согласование или расхождение определен как процент устройств, которые отвечают условию - попадают в интервал $I_{sub}<I_{max\ sub}<KI_{sub\ nom}$. Таблица 1 отражает оценку ошибки согласования, используя представленную модель по сравнению с моделированием по методу Монте-Карло для К = 2.

Значение обобщенного случайного параметра $I_{max\ sub}$

Материал	Параметры Прототип	$\%_{МК}$	$\%_{мод}$	Ошибка, %
Si	L_g=13 нм , t_s=3 нм	81.3	87.4	7.5
	L_g=15 нм , t_s=5 нм	83.0	88.7	6.8
	L_g=18 нм , t_s=8 нм	93.9	90.1	6.0
Ge	L_g=15 нм , t_s=5 нм	77.3	84.6	9.5
	L_g=18 нм , t_s=8 нм	76.1	83.1	9.1

Ошибка незначительна и варьируется в диапазоне до 10% – для всего диапазона длины затвора и значений толщины рабочей области для диапазона флуктуаций номинальных значений ограниченных как 3σ = 10%, так и 3σ = 20%. Анализ повторен для различных значений К, и величина ошибки остается прежней. В этом анализе мы учитываем только флуктуацию выбранных параметров и их взаимное влияние. Но подход может быть расширен, чтобы учитывать и влияние флуктуации других параметров транзистора.

Заключение

В работе представлена обобщенная 2D аналитическая модель распределения подпогогового тока для суб 20 нм двух затворных КМОП транзисторов со структурой «кремний на изоляторе» и «германий на изоляторе». Модель получена непосредственно из решения уравнений Пуассона и Шреденгера и таким образом объединяет коротко-канальные и квантовые эффекты, квазибаллистический транспорт носителей и зонную структуру материала рабочей области транзистора. Результаты моделирования выбранной архитектуры транзисторной структуры с различными материалами рабочей области (кремний и германий) предсказывают, что масштабирование по толщине рабочей области приводит к резкому (на несколько порядков) ограничению тока утечки. Однако, германий - материал с узкой запрещенной зоной характеризуется очень высоким уровнем подпорогового тока по сравнению с кремниевыми структурами. Масштабирование по длине затвора приводит к резкому возрастанию уровня подпорогового тока, что ограничивает масштабируемость транзисторных структур. На границе области масштабирования толщина рабочей области не может превышать 4 нм.

В работе предложенная компактная модель для схемотехнических приложений, которая получена из 2D модели при помощи ряда физических упрощений. Она в полной мере отражает физику подпорогового тока двух затворных суб 20 нм КНИ транзисторов и может быть применима для схемотехнических приложений, а также для решения других задач. Максимальная ошибка рассогласования составляет 12%.

С помощью компактной модели рассмотрена задача оценки влияния флуктуации топологических параметров, в частности длины затвора и толщины рабочей области, на разброс подпорогового тока. Получено хорошее согласование в пределах 10% результатов моделирования по методу Монте-Карло расчетов, выполненных по предложенной модели, что позволяет не прибегая к вычислительно-емким методам сделать приемлемый прогноз поведения электро-физических характеристик транзистора по вероятному диапазону изменения топологических параметров.

2D mathematical model of leakage current of sub 20 nm double gate CMOS transistors

E.N. Epikhine, N.V. Masalsky

Abstract: The 2D analytical distribution model of leakage current for sub 20 nm of double gate CMOS of transistors is discussed with structure "silicon on insulator" and "germanium on insulator". The model is received directly from the solution of Poisson's and Schr¨odinger's equations. Results of simulation foretell that scaling on thickness of work area brings to sharp (on some orders) to leakage current restriction. Thus structures germanium on equations insulator are characterized by very high level of leakage current in comparison with silicon structures. Scaling on length of a lock leads to sharp increase of level of leakage current that restricts scalability of transistor structures. On boundary of area of scaling thickness of work area can't exceed 4 nm.

The compact model for circuitry applications which is received from 2D model by means of a row of physical simplifications is considered. It fully reflects physics of leakage current of double gate sub 20 nm of SOI of transistors. The maximum error of a mismatch makes 12%. By means of a compact model the task of an impact assessment of fluctuation of topological parameters, in particular lengths of a lock and thickness of work area, on dispersion of leakage current is analyzed. Good coordination within 10% of results of simulation according to the Monte-Carlo method of the calculations executed on offered model is received.

Keywords: model of leakage current, double gate CMOS transistor, structure «silicon on isulator»

Литература

1. International Technology Roadmap for Semiconductors // Published online at http://public.itrs.net, 2014

2. R. Chau. Advanced depleted-substrate transistors: single-gate, double-gate and tri-gate // In *Intl. Conf. on Solid State Devices and Materials*, Nagoya, Japan, 2002, pp. 68-69

3. D. Hisamotol. FinFET – a self-aligned double-gate MOSFET scalable to 20nm // *IEEE Trans. Electron Devices*, 2000, № 12(47), pp. 2320–2325

4. F.-L. Yang. 25 nm CMOS Omega FETs // In *IEDM Dig.*, 2002, pp. 255–258.

5. A.Aniket, D.Breed, P.Kenneth, G.Roenker. Comparison of the scaling characteristics of nanoscale SOI N-channel multiple-gate MOSFETs // *Analog Integr Circ Sig* Process, 2008, № 1(56), pp. 135–141

6. Н.В.Масальский. Масштабирование характеристик двух затворных КНИ нанотранзисторов // *Труды НИИСИ*, 2011, т. 1, № 1, с. 16-19

7. L.Ge, J.Fossum. Analytical modeling of quantization and volume inversion in thin Si-film double-gate MOSFETs // *IEEE Trans. Electron Devices*, 2002, № 2(49), pp. 287-294.

8. D.Munteanu, J.Autran, X.Loussier. Quantum short channel compact modeling of drain current in double-gate MOSFET // *Solid-State Electron*., 2006, № 3(50), pp. 680-688

9. N.Barin, M.Braccioli, C.Fiegna. Analysis of scaling strategies for sub-30 nm double-gate SOI N-MOSFETs // *IEEE Trans Nanotechnol*,. 2007, № 2(6), . pp. 421-427

10. Z.Ren, R.Venugopal, S.Goasguen. NanoMOS 2.5: a MOSFETs // *IEEE Trans Electron Devices.* 2003, № 6(50), pp. 1914-1920

11. F.Djeffal, T.Bendib, M.Abdi. A two-dimensional semianalytical analysis of the subthreshold-swing behavior including free carriers and interfacial traps effects for nanoscale double gate MOSFETs // *Microelectron J*. 2011, №. 8 (42), pp. 1391-1399

12. X. Liang, Y.Taur. A 2-D analytical solution for SCEs in DG MOSFETs // *IEEE Trans. Electron Devices*, 2004, № 51(8), pp.1385–1391.

13. C.S.Hou, C.Y.Wu. 2-D analytical model for the threshold voltage of fully depleted short gate-length Si-SOI MOSFETs // *IEEE Trans Electron Devices*, 1995, v. 42, № 2156

14. T.K.Chiang, Y.H.Wang, M.P.Houng. Modeling of threshold voltage and subthreshold swing of short channel SOI MEFET's // *Solid-State Electron*, 1999, v. 43, № 123

15. V.P.Trivedi, J.G.Fossum. Quantum-mechanicaleffects on the threshold voltage of undoped double-gate MOSFETs // *IEEE Electron Device Lett.*, 2005, № 8(26), pp. 579–582

16. Y.Taur, X.Liang, W.Wang. Continuous, analytic drain current model for DG MOSFETs // *IEEE Electron Device Lett.*, 2004, № 1(25), pp. 107-112

17. D.Segev, A.Janotti, C.G.van de Walle. Self-consistent band-gap corrections in density functional theory using modified pseudopotentials // *Physical Rev. B.*, 2007, v. 75, pp. 035201 _1 - 035201 _9

18. D.Esseni, M.Mastrapasqua, G.Celler, C.Fiegna, L.Selmi, E.Sangiorgi. An experimental study of mobility enhancement in ultrathin SOI transistors operated in double-gate mode // *IEEE Trans. Electron Devices*.? 2003? № 3(50), pp. 802-810

19. Q.Chen, E.Harrell, J.Meindl. A physical short-channel threshold voltage model for undoped symmetric double-gate MOSFETs // *IEEE Trans. Electron Devices*.? 2003, № 7(50), pp. 1631–1637

20. В.А.Горячев. Некоторые аспекты совершенствования германиевых технологий КМОП ИС // *Труды НИИСИ РАН,* 2014, т. 4, №1, с. 86 – 91

21. Н.В.Масальский. Моделирование характеристик транзисторных структур «германий на изоляторе» // *Труды НИИСИ*, 2014, т. 4, № 2 (в этом номере)

22. A.Walsh, J.Silva, S.Wei. Origins of band-gap renormalization in degenerately doped semiconductors // *Physical Rev. B.*, 2008, v. 78, pp. 075211 _1-075211 5

23. R.Rao, A.Srivastava, D.Blaauw, D.Sylvester. Statistical analysis of subthreshold leakage current for VLSI circuits // *IEEE Trans. VLSI*, 2004, v. 12, № 2, pp. 131–139

О НЕКОТОРЫХ ЭКСТРЕМАЛЬНЫХ ЗАДАЧАХ В КОНЕЧНОМЕРНЫХ ЧЕБЫШЁВСКИХ ПРОСТРАНСТВАХ

В.Б.Демидович[1], А.С.Кочуров[1]

1 - кандидат физико-математических наук

Аннотация: В конечномерном чебышёвском пространстве рассматриваются вопросы о явном построении обобщённого полинома наименьшего уклонения от нуля (в метриках C, L_1 и L_2), а также о неравенствах для производных в этих метриках. Указанные полиномы играют важную роль в вычислительной математике.

Ключевые слова: чебышёвское пространство, неравенство для производных, выпуклость, субдифференцирование

ИСХОДНЫЕ ПОНЯТИЯ

Приведём сначала исходные определения теории чебышёвских пространств, основы которой можно найти в [9]-[12], причём при изложении этих определений мы будем придерживаться работы [1].

Подпространство L_{n+1} $(n \ge 0)$ пространства $C([a,b])$ размерности (n+1) называется **чебышёвским пространством** (или **Т-пространством** от слова **Tchebycheff**) **n-го порядка**, если любая нетривиальная функция из L_{n+1} имеет на [a,b] не более **n различных нулей**. Любой **базис** в таком Т-пространстве называется ***Т-системой*** порядка **n** на [a, b].

Назовём (n+1)-мерное подпространство L_{n+1} пространства $C^n([a,b])$ **обобщённым чебышёвским пространством** (или **ET-пространством** от слова **extended**) **n-го** порядка, если любая его нетривиальная функция имеет на [a, b] не более **n нулей с учётом их кратностей**.

Базис $\{e_k(\cdot)\}_{k=0}^{n}$ в **Т-пространстве** L_{n+1} называется **полной чебышёвской системой** (или **CT-системой** от слова **complete**) порядка **n** на [a, b], если все подпространства $L_m = \operatorname{span}\{e_k(\cdot)\}_{k=0}^{m}$ $(0 \le m \le n)$ являются чебышёвскими пространствами. Подпространство L_{n+1} с таким базисом называется **полным чебышёвским пространством** (или **CT-пространством**) **n-го** порядка.

Базис $\{e_k(\cdot)\}_{k=0}^{n}$ в **ET-пространстве** L_{n+1} называется **обобщённой полной чебышёвской системой** (или **ECT-системой**) порядка **n** на [a, b], если все подпространства $L_m = \operatorname{span}\{e_k(\cdot)\}_{k=0}^{m}$ $(0 \le m \le n)$ являются **обобщёнными чебышёвскими пространствами**. Подпространство L_{n+1} с таким базисом называется **обобщённым полным чебышёвским пространством (**или **ECT-пространством**) **n-го** порядка.

Элементы введенных пространств будем называть просто **обобщёнными полиномами**.

Важнейшим **примером** ECT-пространства **n-го** порядка в $C([a,b])$ служит подпространство P_n классических алгебраических многочленов степени $\le n$. Богатый материал по их свойствам систематизирован в [13]-[14].

Другие примеры **ECT-систем** порядка n на **[a,b]**, порождающих в $C([a,b])$ соответствующие **ECT –пространства n-го** порядка, в частности, таковы:

экспоненты:

$$\{1, e^{\alpha_1 t}, \ldots, e^{\alpha_n t}\},$$
$$0 < \alpha_1 < \ldots < \alpha_n;$$

степени $(b > a > 0)$:

$$\{1, t^{\alpha_1}, \ldots, t^{\alpha_n}\},$$
$$0 < \alpha_1 < \ldots < \alpha_n;$$

рациональные дроби $(b > a > 0)$:

$$\{1, \frac{1}{t+\alpha_1}, \ldots, \frac{1}{t+\alpha_n}\},$$
$$0 < \alpha_1 < \ldots < \alpha_n;$$

логарифмы $(b > a > 0)$:

$$\{1, \ln(t+\alpha_1), \ldots, \ln(t+\alpha_n)\},$$
$$0 < \alpha_1 < \ldots < \alpha_n;$$

гауссовы функции $(b > a > 0)$:

$$\{1, e^{-\frac{(t+\alpha_1)^2}{\beta}}, \ldots, e^{-\frac{(t+\alpha_n)^2}{\beta}}\},$$
$$0 < \alpha_1 < \ldots < \alpha_n, \quad \beta > 0;$$

экспонентно-степенные функции $(b > a > 0)$:

$$\{1, \ldots, t^{\beta_0 - 1}, e^{\alpha_1 t}, \ldots, t^{\beta_1 - 1} e^{\alpha_1 t}, \ldots, e^{\alpha_j t}, \ldots, t^{\beta_j - 1} e^{\alpha_j t}\},$$
$$0 < \alpha_1 < \ldots < \alpha_j, \quad \beta_i \in \mathbb{N},$$
$$0 \le i \le j, \quad \sum_{i=0}^{j} \beta_i = n+1.$$

В теории приближения рассматривались некоторые экстремальные задачи, связанные с подпространством P_n пространства $C([a,b])$, то есть с пространством классических алгебраических многочленов степени $\le n$ (см., например, [4]-[5]). Среди них - задача **построения в P_n многочлена наименьшего уклонения от нуля** в пространстве $C([a,b])$ для его естественной метрики C, а также для метрик пространств L_1 или L_2. Во всех этих случаях подробно изучены подобные многочлены. Приведём (для *канонического* отрезка [-1, 1]), **применяя представление Г.Г. Лоренца для многочленов** (см. [13, гл. 1, § 1.2, п. 1.2.5]), *выражения для* используемых в этих случаях *классических многочленов.*

Итак, **на каноническом отрезке [-1, 1], в случае метрики** C, для записи **многочлена наименьшего уклонения от нуля с единичным старшим коэффициентом** используются многочлены $T_n(\cdot)$ (при n > 0) вида:

$$T_n(t) = \sum_{k=0}^{n} \vartheta_k \cdot (t+1)^{n-k} (t-1)^k,$$
$$\vartheta_k = \frac{(2n)!}{2^{2n-1}(2k)!(2n-2k)!}. \qquad (1)$$

Первое явное представление для такого многочлена было найдено П.Л.Чебышёвым, и в честь него $T_n(\cdot)$ стал называться **многочленом Чебышёва 1-го рода**. Выражение (1) для $T_n(\cdot)$ непосредственно следует из формулы Стефана Пашковского (см. [2, гл. I, § 1, формулу (26)]).

В свою очередь, **на каноническом отрезке [-1, 1], в случае метрики** L_1, для записи **многочлена наименьшего уклонения от нуля с единичным старшим коэффициентом** используются многочлены $U_n(\cdot)$ (при n > 0) вида:

$$U_n(t) = \sum_{k=0}^{n} \upsilon_k \cdot (t+1)^{n-k} (t-1)^k,$$
$$\upsilon_k = \frac{(2n+2)!}{2^{2n+1}(2k+1)!(2n-2k+1)!}. \qquad (2)$$

Первое явное представление для такого многочлена было найдено учениками П.Л.Чебышёва - А.Н.Коркиным и Е.И.Золотаревым, и, вслед за ними, $U_n(\cdot)$ стали называть **многочленом Чебышёва 2-го рода**. Выражение (2) для $U_n(\cdot)$ также непосредственно следует из другой формулы Стефана Пашковского (см. [2, гл. I, § 1, формулу (27)]).

Наконец, **на каноническом отрезке [-1, 1], в случае метрики** L_2, для записи **многочлена наименьшего уклонения от нуля с единичным старшим коэффициентом** используются многочлены $G_n(\cdot)$ (при n > 0) вида:

$$G_n(t) = \sum_{k=0}^{n} \gamma_k \cdot (t+1)^{n-k} (t-1)^k,$$
$$\gamma_k = \frac{(n!)^4}{(2n)![k!(n-k)!]^2}. \qquad (3)$$

Искомое представление для такого многочлена вытекает из известной формулы Олинда Родриго (см, например, [3, гл. IV, § 1, формулу (4)]), показавшего, что

$$G_n(t) = \frac{n!}{(2n)!}[(t+1)^n (t-1)^n]^{(n)}. \qquad (4)$$

Действительно, на основании **правила** Готфрида Лейбница для **n-кратного** дифференцирования

$$[u(t) \cdot v(t)]^{(n)} = \sum_{k=0}^{n} C_n^k \cdot u^{(k)}(t)\, v^{(n-k)}(t), \qquad (5)$$

последовательно получаем

$$[(t+1)^n (t-1)^n]^{(n)} =$$
$$= \sum_{k=0}^{n} \{C_n^k \cdot [n(n-1)\cdots(n-k+1)]\,(t+1)^{n-k} \times$$
$$[n(n-1)\cdots(k+1)]\,(t-1)^k\} =$$

$$= \sum_{k=0}^{n} \{C_n^k \cdot \frac{n(n-1)\cdots(n-k+1)(n-k)!}{(n-k)!}\,(t+1)^{n-k} \times$$
$$\frac{n(n-1)\cdots(k+1)k!}{k!}\,(t-1)^k\} =$$

$$= \sum_{k=0}^{n} C_n^k \cdot \frac{n!}{(n-k)!}(t+1)^{n-k} \cdot \frac{n!}{k!}(t-1)^k =$$

$$= \sum_{k=0}^{n} \frac{n!}{k!(n-k)!} \cdot \frac{n!}{(n-k)!} \cdot \frac{n!}{k!} \cdot (t+1)^{n-k}(t-1)^k =$$

$$= (n!)^3 \cdot \sum_{k=0}^{n} \frac{1}{[k!(n-k)!]^2} \cdot (t+1)^{n-k}(t-1)^k. \quad (6)$$

Подставляя в формулу (4) найденное в (6) выражение для $[(t+1)^n(t-1)^n]^{(n)}$, мы и приходим к представлению (3).

Первое явное представление для такого многочлена было получено Адриеном Лежандром, и потому $G_n(\cdot)$ стали называть **многочленом Лежандра**. Его свойства подробно изучены в классической теории ортогональных многочленов (см., в частности, [2]).

Представляется интересным получить *аналогичные формулы для обобщённых полиномов*, распространяющие справедливые в пространстве алгебраических многочленов P_n выражения (1)-(3) на случай **произвольного ECT-пространства** соответствующего порядка.

ПОСТАНОВКА ЗАДАЧИ

Пусть (n+1)-мерное подпространство L_{n+1} пространства $C^n([a,b])$ является ECT-пространством порядка n, и пусть

$$\{e_k(\cdot)\}_{k=0}^{n} \in C^n([a,b]) \quad (7)$$

- заданный **базис** пространства L_{n+1}, представляющий собой ECT-систему порядка n на [a, b]. Обозначая через Φ **совокупность обобщённых полиномов n-го порядка** (из рассматриваемого ECT-пространства L_{n+1}) **с единичным старшим коэффициентом** и произвольными остальными коэффициентами $(x_k)_{k=0}^{n-1}$:

$$\Phi = \{\phi = \phi(t) : \phi(t) = e_n(\cdot) + \sum_{k=0}^{n-1} x_k e_k(\cdot)\}, \quad (8)$$

и вводя функцию

$$f(t,\phi) := |\phi(t)| = |e_n(t) + \sum_{k=0}^{n-1} x_k e_k(t)|, \quad (9)$$

приведем формализации интересующих нас **обобщений** задач (1)-(3).

1. Задача об обобщённом полиноме n-го порядка наименьшего уклонения от нуля в метрике C.

Полагая

$$f_1(\phi) := \max_{t\in[a,b]} f(t,\phi), \quad (10)$$

данная задача формализуется в виде

$$f_1(\phi) \to \min, \qquad \phi \in \Phi. \qquad (P_1)$$

2. Задача об обобщённом полиноме n-го порядка наименьшего уклонения от нуля в метрике L_1.

Полагая

$$f_2(\phi) := \int_a^b f(t,\phi)dt, \quad (11)$$

эта задача формализуется в виде

$$f_2(\phi) \to \min, \qquad \phi \in \Phi. \qquad (P_2)$$

3. Задача об обобщённом полиноме n-го порядка наименьшего уклонения от нуля в метрике L_2.

Полагая

$$f_3(\phi) := \int_a^b f^2(t,\phi)dt, \quad (12)$$

данная задача формализуется в виде

$$f_3(\phi) \to \min, \qquad \phi \in \Phi. \qquad (P_3)$$

Прежде всего заметим, что все эти три задачи являются **выпуклыми,** причём относящимися к **конечномерным экстремальным задачам без ограничений.** Тем самым все эти экстремальные задачи можно переформулировать в едином виде:

$$f(x) \to \min, \ x \in R^n \ (x := (x_k)_{k=0}^{n-1}). \qquad (P)$$

ОСНОВНЫЕ УТВЕРЖДЕНИЯ О СТРУКТУРНЫХ ФОРМУЛАХ ДЛЯ ОБОБЩЁННЫХ ПОЛИНОМОВ НАИМЕНЬШЕГО УКЛОНЕНИЯ ОТ НУЛЯ

Приведём теперь некоторые **гипотезы о структурных формулах** для записи ответов в задачах $(P_1)-(P_3)$.

В задаче (P_1) для записи **обобщённого полинома n-го порядка наименьшего уклонения от нуля, в метрике C на отрезке [a, b], с единичным старшим коэффициентом (**аналогичного многочлену $T_n(\cdot)$) можно использовать выражение $T_n(\cdot)$ (здесь, и далее, для обозначения *определителя* мы ставим, с каждой его стороны, *две* вертикальные черты) структурного вида (13):

$$T_n(t)=\{\Theta_0\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a)\\ e'_0(a) & \cdots & e'_n(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_n^{(n-2)}(a)\\ e_0^{(n-1)}(a) & \cdots & e_n^{(n-1)}(a)\\ e_0(t) & \cdots & e_n(t)\end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a)\\ e'_0(a) & \cdots & e'_{n-1}(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_{n-1}^{(n-2)}(a)\\ e_0^{(n-1)}(a) & \cdots & e_{n-1}^{(n-1)}(a)\end{Vmatrix}}+\Theta_1\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a)\\ e'_0(a) & \cdots & e'_n(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_n^{(n-2)}(a)\\ e_0(b) & \cdots & e_n(b)\\ e_0(t) & \cdots & e_n(t)\end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a)\\ e'_0(a) & \cdots & e'_{n-1}(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_{n-1}^{(n-2)}(a)\\ e_0(b) & \cdots & e_{n-1}(b)\end{Vmatrix}}+$$

$$+\Theta_2\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a)\\ e'_0(a) & \cdots & e'_n(a)\\ \cdots & \cdots & \cdots\\ e_0(b) & \cdots & e_n(b)\\ e'_0(b) & \cdots & e'_n(b)\\ e_0(t) & \cdots & e_n(t)\end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a)\\ e'_0(a) & \cdots & e'_{n-1}(a)\\ \cdots & \cdots & \cdots\\ e_0(b) & \cdots & e_{n-1}(b)\\ e'_0(b) & \cdots & e'_{n-1}(b)\end{Vmatrix}}+\cdots\cdots+\Theta_n\frac{\begin{Vmatrix} e_0(b) & \cdots & e_n(b)\\ e'_0(b) & \cdots & e'_n(b)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(b) & \cdots & e_n^{(n-2)}(b)\\ e_0^{(n-1)}(b) & \cdots & e_n^{(n-1)}(b)\\ e_0(t) & \cdots & e_n(t)\end{Vmatrix}}{\begin{Vmatrix} e_0(b) & \cdots & e_{n-1}(b)\\ e'_0(b) & \cdots & e'_{n-1}(b)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(b) & \cdots & e_{n-1}^{(n-2)}(b)\\ e_0^{(n-1)}(b) & \cdots & e_{n-1}^{(n-1)}(b)\end{Vmatrix}}\},\qquad(13)$$

где $\{\Theta_k\}_{k=0}^n$ - некоторые константы. Вопрос о том, как *конкретизировать* эти константы, остаётся открытым.

В свою очередь, в задаче (P_2) для записи **обобщённого полинома n-го порядка наименьшего уклонения от нуля, в метрике L_1 на отрезке [a, b], с единичным старшим коэффициентом** (аналогичного многочлену $U_n(\cdot)$), можно использовать выражение $U_n(\cdot)$ структурного вида (14):

$$U_n(t)=\{\Upsilon_0\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a)\\ e'_0(a) & \cdots & e'_n(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_n^{(n-2)}(a)\\ e_0^{(n-1)}(a) & \cdots & e_n^{(n-1)}(a)\\ e_0(t) & \cdots & e_n(t)\end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a)\\ e'_0(a) & \cdots & e'_{n-1}(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_{n-1}^{(n-2)}(a)\\ e_0^{(n-1)}(a) & \cdots & e_{n-1}^{(n-1)}(a)\end{Vmatrix}}+\Upsilon_1\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a)\\ e'_0(a) & \cdots & e'_n(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_n^{(n-2)}(a)\\ e_0(b) & \cdots & e_n(b)\\ e_0(t) & \cdots & e_n(t)\end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a)\\ e'_0(a) & \cdots & e'_{n-1}(a)\\ \cdots & \cdots & \cdots\\ e_0^{(n-2)}(a) & \cdots & e_{n-1}^{(n-2)}(a)\\ e_0(b) & \cdots & e_{n-1}(b)\end{Vmatrix}}+$$

$$+\Upsilon_2\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a) \\ e'_0(a) & \cdots & e'_n(a) \\ \cdots & \cdots & \cdots \\ e_0(b) & \cdots & e_n(b) \\ e'_0(b) & \cdots & e'_n(b) \\ e_0(t) & \cdots & e_n(t) \end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a) \\ e'_0(a) & \cdots & e'_{n-1}(a) \\ \cdots & \cdots & \cdots \\ e_0(b) & \cdots & e_{n-1}(b) \\ e'_0(b) & \cdots & e'_{n-1}(b) \end{Vmatrix}}+\cdots\cdots+\Upsilon_n\frac{\begin{Vmatrix} e_0(b) & \cdots & e_n(b) \\ e'_0(b) & \cdots & e'_n(b) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(b) & \cdots & e_n^{(n-2)}(b) \\ e_0^{(n-1)}(b) & \cdots & e_n^{(n-1)}(b) \\ e_0(t) & \cdots & e_n(t) \end{Vmatrix}}{\begin{Vmatrix} e_0(b) & \cdots & e_{n-1}(b) \\ e'_0(b) & \cdots & e'_{n-1}(b) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(b) & \cdots & e_{n-1}^{(n-2)}(b) \\ e_0^{(n-1)}(b) & \cdots & e_{n-1}^{(n-1)}(b) \end{Vmatrix}}\}, \qquad (14)$$

где $\{\Upsilon_k\}_{k=0}^n$ - некоторые константы. И, снова, вопрос о том, как конкретизировать константы $\{\Upsilon_k\}_{k=0}^n$, остаётся открытым.

Наконец, в задаче (P_3) для записи **обобщенного полинома n-го порядка наименьшего уклонения от нуля, в метрике L_2** ***на отрезке*** **на отрезке [a, b,** ***с единичным старшимс коэффициентом*** (аналогичного многочлену $G_n(\cdot)$**),** можно использовать выражение $G_n(\cdot)$ структурного вида (15):

$$G_n(t)=\{\Gamma_0\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a) \\ e'_0(a) & \cdots & e'_n(a) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(a) & \cdots & e_n^{(n-2)}(a) \\ e_0^{(n-1)}(a) & \cdots & e_n^{(n-1)}(a) \\ e_0(t) & \cdots & e_n(t) \end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a) \\ e'_0(a) & \cdots & e'_{n-1}(a) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(a) & \cdots & e_{n-1}^{(n-2)}(a) \\ e_0^{(n-1)}(a) & \cdots & e_{n-1}^{(n-1)}(a) \end{Vmatrix}}+\Gamma_1\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a) \\ e'_0(a) & \cdots & e'_n(a) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(a) & \cdots & e_n^{(n-2)}(a) \\ e_0(b) & \cdots & e_n(b) \\ e_0(t) & \cdots & e_n(t) \end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a) \\ e'_0(a) & \cdots & e'_{n-1}(a) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(a) & \cdots & e_{n-1}^{(n-2)}(a) \\ e_0(b) & \cdots & e_{n-1}(b) \end{Vmatrix}}+$$

$$+\Gamma_2\frac{\begin{Vmatrix} e_0(a) & \cdots & e_n(a) \\ e'_0(a) & \cdots & e'_n(a) \\ \cdots & \cdots & \cdots \\ e_0(b) & \cdots & e_n(b) \\ e'_0(b) & \cdots & e'_n(b) \\ e_0(t) & \cdots & e_n(t) \end{Vmatrix}}{\begin{Vmatrix} e_0(a) & \cdots & e_{n-1}(a) \\ e_{0'}(a) & \cdots & e'_{n-1}(a) \\ \cdots & \cdots & \cdots \\ e_0(b) & \cdots & e_{n-1}(b) \\ e'_0(b) & \cdots & e'_{n-1}(b) \end{Vmatrix}}+\cdots\cdots+\Gamma_n\frac{\begin{Vmatrix} e_0(b) & \cdots & e_n(b) \\ e'_0(b) & \cdots & e'_n(b) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(b) & \cdots & e_n^{(n-2)}(b) \\ e_0^{(n-1)}(b) & \cdots & e_n^{(n-1)}(b) \\ e_0(t) & \cdots & e_n(t) \end{Vmatrix}}{\begin{Vmatrix} e_0(b) & \cdots & e_{n-1}(b) \\ e'_0(b) & \cdots & e'_{n-1}(b) \\ \cdots & \cdots & \cdots \\ e_0^{(n-2)}(b) & \cdots & e_{n-1}^{(n-2)}(b) \\ e_0^{(n-1)}(b) & \cdots & e_{n-1}^{(n-1)}(b) \end{Vmatrix}}\}, \qquad (15)$$

где $\{\Gamma_k\}_{k=0}^n$ - некоторые константы.

Вопрос о том, как конкретизировать константы $\{\Gamma_k\}_{k=0}^n$, здесь может быть решён следующим образом.

Всякая чебышёвская система линейно независима. Но для линейно независимых систем на отрезке [a, b] *известна* справедливость представления Йоргена Грама (вытекающего из его исследований по ортогональным системам) полинома наименее уклоняющегося от нуля $G_n(\cdot)$ вида (16):

$$G_n(t)=\frac{\begin{Vmatrix} \int_a^b e_0^2(t)dt & \int_a^b e_0(t)e_1(t)dt & \cdots & \int_a^b e_0(t)e_{n-1}(t)dt & \int_a^b e_0(t)e_n(t)dt \\ \int_a^b e_1(t)e_0(t)dt & \int_a^b e_1^2(t)dt & \cdots & \int_a^b e_1(t)e_{n-1}(t)dt & \int_a^b e_1(t)e_n(t)dt \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \int_a^b e_{n-1}(t)e_0(t)dt & \int_a^b e_{n-1}(t)e_1(t)dt & \cdots & \int_a^b e_{n-1}^2(t)dt & \int_a^b e_{n-1}(t)e_n(t)dt \\ e_0(t) & e_1(t) & \cdots & e_{n-1}(t) & e_n(t) \end{Vmatrix}}{\begin{Vmatrix} \int_a^b e_0^2(t)dt & \int_a^b e_0(t)e_1(t)dt & \cdots & \int_a^b e_0(t)e_{n-1}(t)dt \\ \int_a^b e_1(t)e_0(t)dt & \int_a^b e_1^2(t)dt & \cdots & \int_a^b e_1(t)e_{n-1}(t)dt \\ \cdots & \cdots & \cdots & \cdots \\ \int_a^b e_{n-1}(t)e_0(t)dt & \int_a^b e_{n-1}(t)e_1(t)dt & \cdots & \int_a^b e_{n-1}^2(t)dt \end{Vmatrix}}, \qquad (16)$$

А из (15)-(16), приравнивая коэффициенты при $e_k(\cdot)$, k=0,1,…, n, получится *невырожденная система линейных уравнений* относительно неизвестных $\{\Gamma_k\}_{k=0}^n$, однозначно разрешимая.

ДОПОЛНИТЕЛЬНЫЕ УТВЕРЖДЕНИЯ И ГИПОТЕЗЫ

В $(n+1)$-мерном подпростанстве L_{n+1} пространства $C^n([a,b])$, являющемся *ECT-пространством порядка* n на $[a,b]$, при заданном *базисе* для L_{n+1}

$$\{e_k(\cdot)\}_{k=0}^n \in C^n([a,b]), \qquad (17)$$

представляющим собой *ECT-систему* порядка n на $[a,b]$, можно ввести оператор ***обобщённого*** дифференцирования k-го порядка ($0 \le k \le n$):

$$D^k : C^n([a,b]) \to C^{n-k}([a,b]), \qquad (18)$$

обобщающий оператор k-кратного *классического* дифференцирования, полагая, что значение соответствующей ***обобщённой*** производной k-го порядка от функции $x(\cdot) \in C^n([a,b])$ определяется формулой

$$D^k x(\cdot)=\frac{W(e_1(\cdot),\ldots,e_k(\cdot),x(\cdot))}{W(e_1(\cdot),\ldots,e_k(\cdot))}, \qquad (19)$$

$$1 \le k \le n, \qquad D^0 x(\cdot)=x(\cdot),$$

где $W(\ldots)$ - ***вронскиан*** от указанных аргументов.

В работе [1] была рассмотрена ***задача о максимизации на ECT-пространстве значения обобщённой производной в точке τ при наличии ограничения на функцию*** **в метрике** C ***на заданном отрезке***, формализуемая в виде

$$D^k x(\tau) \to \max, \quad \tau \in [a,b], \quad \| x(\cdot) \|_{C([a',b'])} \le 1,$$
$$[a',b'] \subseteq [a,b], \quad x(\cdot) \in L_{n+1}. \qquad (P_1')$$

В случае

$$[a',b'] \subset [a,b], \qquad \tau \in [a,b] \setminus [a',b'] \qquad (20)$$

там было показано, что решением такой ***экстраполяционной*** экстремальной задачи служит *значение k-ой обобщённой производной в точке τ обобщённого полинома n-го порядка наименьшего уклонения от нуля в метрике C, построенного для внутреннего отрезка $[a',b']$ и взятого с соответствующим нормирующим множителем.*

Углубляя эти исследования, представляется справедливой гипотеза об *обобщённом неравестве для производной в*

метрике C, согласно которой должно иметь место неравенство

$$\| D^k \phi_n(\cdot) \|_{C([a,b])} \le \frac{\| D^k T_n(\cdot) \|_{C([a,b])}}{\| T_n(\cdot) \|_{C([a,b])}} \cdot \| \phi_n(\cdot) \|_{C([a,b])}$$

$$(\forall \phi_n(\cdot) \in L_{n+1}), \qquad (21)$$

если только $T_n(\cdot)$ - обобщённый полином n-го порядка наименьшего уклонения от нуля в метрике C на отрезке $[a,b]$.

Замечание 1.

В классическом пространстве многочленов P_n *соответствующая оценка имеет вид неравенства (22)*

$$\| p_n^{(k)}(\cdot) \|_{C([a,b])} \le \frac{2^k}{(b-a)^k} \times$$

$$\frac{(n^2)(n^2-1^2)(n^2-2^2)\cdots(n^2-(k-1)^2)}{(2k-1)\cdot(2k-3)\cdots 5\cdot 3\cdot 1} \times \quad (22)$$

$$\| p_n(\cdot) \|_{C([a,b])} \quad (\forall p_n(\cdot) \in P_n),$$

найденного В.А.Марковым (см., например, [14, гл. 15, формулу (15.7.16]).

Аналогично можно поставить ***задачу о максимизации на*** ECT ***-пространстве значения обобщённой производной в точке*** τ ***при наличии ограничения функции*** **в метрике** L_1 ***на заданном отрезке,*** формализуя её в виде

$$D^k x(\tau) \to \max, \quad \tau \in [a,b], \quad [a',b'] \subseteq [a,b],$$

$$\| x(\cdot) \|_{L_1([a',b'])} \le 1, \quad x(\cdot) \in L_{n+1}. \qquad (P_2^{'})$$

Полагаем, что решение этой экстремальной задачи затем позволит разобраться со справедливостью нашей гипотезы об *обобщённом неравестве для производной в метрике* L_1, согласно которой

$$\| D^k \phi_n(\cdot) \|_{L_1([a,b])} \le$$

$$\frac{\| D^k U_n(\cdot) \|_{L_1([a,b])}}{\| U_n(\cdot) \|_{L_1([a,b])}} \cdot \| \phi_n(\cdot) \|_{L_1([a,b])}$$

$$(\forall \phi_n(\cdot) \in L_{n+1}), \qquad (23)$$

если только $U_n(\cdot)$ - обобщённый полином n-го порядка наименьшего уклонения от нуля в метрике L_1 на отрезке $[a,b]$.

Замечание 2.

Отметим, что, насколько нам известно, даже в классическом пространстве многочленов P_n *справедливость неравенства такого вида не доказана. Тем не менее, добавим, что некоторые «близкие» неравенства, связывающие норму производной многочлена с нормой самого многочлена, установлены (см., в частности, в [14, гл. 15, формулу (15.6.5)] и, вообще, литературу в [13] – [14]).*

Наконец, можно поставить также ***задачу о максимизации на*** ECT ***-пространстве значения обобщённой производной в точке*** τ ***при наличии ограничения функции*** **в метрике** L_2 ***на заданном отрезке***, формулизуя её в виде

$$D^k x(\tau) \to \max, \quad \tau \in [a,b], \quad [a',b'] \subseteq [a,b],$$

$$\| x(\cdot) \|_{L_2([a',b'])} \le 1, \quad x(\cdot) \in L_{n+1}. \qquad (P_3^{'})$$

И здесь полагаем, что решение данной экстремальной задачи позволит разобраться со справедливостью нашей гипотезы об *обощённом неравестве для производной в метрике* L_2, согласно которой

$$\| D^k \phi_n(\cdot) \|_{L_2([a,b])} \le$$

$$\frac{\| D^k G_n(\cdot) \|_{L_2([a,b])}}{\| G_n(\cdot) \|_{L_2([a,b])}} \cdot \| \phi_n(\cdot) \|_{L_2([a,b])}$$

$$(\forall \phi_n(\cdot) \in L_{n+1}), \qquad (24)$$

если только $G_n(\cdot)$ - обобщённый полином n-го порядка наименьшего уклонения от нуля в метрике L_2 на отрезке $[a,b]$.

Замечание 3.

И в этом случае нам представляется, что даже в классическом пространстве многочленов P_n *справедливость неравенства такого вида ещё не доказана.*

НЕКОТОРЫЕ ОБЩИЕ СВЕДЕНИЯ ИЗ ВЫПУКЛОГО АНАЛИЗА

Все сформулированные задачи можно изучать средствами *выпуклого анализа*. Перечислим, используя материал работ [6}–[8], некоторые известные факты из этой теории, представляющиеся полезными для подобных исследований

Теорема 1 (В.Л.Левина об очистке).

Пусть функция $F : [a,b] \times R^n \to R$ такова, что

a) функция $x \to F(t,x)$ выпукла на R^n для любого $t \in [a,b]$;

b) функция $t \to F(t,x)$ непрерывна на $[a,b]$ для любого $x \in R^n$;

c) $M = \inf_{x \in R^n} \max_{t \in [a,b]} F(t,x) > -\infty$.

Тогда найдутся натуральное $m \le n+1$ и точки $\tau_i \in [a,b]$, $1 \le i \le m$, такие, что $M = \inf_{x \in R^n} \max_{1 \le i \le m} F(\tau_i, x)$.

Определение субдифференциала.

Для выпуклой функции $f : R^n \to R$ *и точки* $\hat{x} \in R^n$ *множество*

$$\partial f(\hat{x}) = \{z \in R^n \mid f(x) - f(\hat{x}) \ge z \cdot (x - \hat{x}), \forall x \in R^n\} \quad (25)$$

(здесь $z \cdot (x - \hat{x}) = \sum_{i=1}^{n} z_i (x_i - \hat{x}_i)$, $z = (z_1, \ldots, z_n)$, $x = (x_1, \ldots, x_n)$ и $\hat{x} = (\hat{x}_1, \ldots, \hat{x}_n)$) называется *субдифференциалом* функции f в точке $\hat{x}$.

Замечание 4.
Если функция f, при этом, дифференцируема в точке $\hat{x}$, *то* $\partial f(\hat{x})$ *состоит из одного элемента* $f'(\hat{x})$ *- производной от f в этой точке.*

Терема 2 (обобщение теоремы Пьера Ферма для выпуклых функций в субдифференциальной форме).

Пусть $f : R^n \to R$ - выпуклая функция. Точка $\hat{x} \in R^n$ является *точкой минимума* для *f,* тогда и только тогда, когда

$$0 \in \partial f(\hat{x}). \quad (26)$$

Теорема 3 (А.Я.Дубовицкого-А.А.Милютина о субдифференциале максимума выпуклых функций).

Пусть $f_i : R^n \to R$, $1 \le i \le m$ - выпуклые функции, непрерывные в точке $\hat{x}$, и пусть

$$f_1(\hat{x}) = \ldots = f_m(\hat{x}).$$

Тогда

$$\partial \max(f_1, \ldots, f_m)(\hat{x}) = \mathrm{conv}(\cup_{i=1}^{m} \partial f_i(\hat{x})), \quad (27)$$

где "conv (M)" означает выпуклую оболочку множества M.

Теорема 4 (В.М.Тихомирова - Г.Г.Магарила-Ильяева о точке минимума для максимума функции, фигурирующей в теореме об очистке).

Пусть функция $F : [a,b] \times R^n \to R$ удовлетворяет условиям (a) и (b) теоремы об очистке. Тогда $\hat{x} \in R^n$ является *точкой минимума* для функции

$$x \to f(x) = \max_{t \in [a,b]} F(t,x) \quad (28)$$

в том и только том случае, когда найдутся натуральное число $m \le n+1$, точки $\tau_i \in [a,b]$, векторы $y_i \in \partial F_x(\tau_i, \hat{x})$ (здесь $\partial F_x(\tau_i, \hat{x})$ - субдифференциал функции $x \to F(\tau_i, x)$ в точке $\hat{x}$) и числа

$\alpha_i > 0 \quad (i = 1, \ldots, m)$, $\sum_{i=1}^{m} \alpha_i = 1$ такие, что

$$\sum_{i=1}^{m} \alpha_i y_i = 0, \quad (29)$$

причём

$$f(\hat{x}) = F(\tau_i, \hat{x}) \qquad (i = 1, \ldots, m). \quad (30)$$

КРИТЕРИИ НАИМЕНЬШЕГО УКЛОНЕНИЯ ОТ НУЛЯ ДЛЯ ОБОБЩЁННЫХ ПОЛИНОМОВ ИЗ ЧЕБЫШЁВСКИХ ПРОСТРАНСТВ

Пусть L_{n+1} - подпространство в $C^n([a,b])$, являющееся **ECT-пространством** порядка *n* с базисом $\{e_k(\cdot)\}_{k=0}^{n} \in C^n([a,b])$, представляющим собой ECT-систему порядка *n* на [a, b].

Напомним, что в L_{n+1} мы ввели **совокупность обобщенных полиномов n-го порядка с единичным старшим коэффициентом** и произвольными остальными коэффициентами $(x_k)_{k=0}^{n-1}$ вида $\Phi = \{\phi_n(\cdot) = e_n(\cdot) + \sum_{k=0}^{n-1} x_k e_k(\cdot)\}$. Используя функции (10)-(12), мы, затем, произвели формализации задач **об обобщённых полиномах n-го порядка наименьшего уклонения от нуля** в метриках C, L_1 и L_2 в виде выпуклых экстремальных задач (P_1) - (P_3), **существование и единственность решения** которых, в условиях *гладкости*

исходного пространства и *конечномерности рассматриваемого ECT-пространства*, хорошо известны в общей теории аппроксимации. В отношении **критериев наименьшего уклонения от нуля** для обобщённых полиномов в указанных метриках можно сформулировать конкретные утверждения. Но перед их формулировкой приведём, сначала, одно важное утверждение:

Теорема 5 (о нулях обобщённых полиномов наименьшего уклонения от нуля).

Для каждой из метрик C, L_1, L_2 *все нули соответствующего обобщённого полинома* ***n-го порядка*** *наименьшего уклонения от нуля на отрезке [a, b] действительны, различны и расположены в интервале (a, b).*

Используя эту теорему, на базе вышеизложенных общих сведений из выпуклого анализа, для каждой из экстремальных задач (P_1) - (P_3) можно сформулировать уже условия, при которых

$$\hat{\phi}_n(\cdot) = e_n(\cdot) + \sum_{k=0}^{n-1} \hat{x}_k e_k(\cdot) \in \Phi \subset L_{N+1} \qquad (31)$$

является соответствующим обобщённым полиномом **n-го порядка** наименьшего уклонения от нуля.

Именно, справедлива

Теорема 6 (критерий решения задачи (P_1)).

Для того чтобы обобщённый полином

$$\hat{\phi}_n(\cdot) = e_n(\cdot) + \sum_{k=0}^{n-1} \hat{x}_k e_k(\cdot) \in \Phi \qquad (32)$$

был бы решением задачи (P_1) *необходимо и достаточно, чтобы нашлись точки*

$$\tau_1 < \cdots < \tau_n < \tau_{n+1} \in [a,b] \qquad (33)$$

и числа

$$\{\alpha_i\}_{i=1}^{n+1} > 0 \qquad (\sum_{i=1}^{n+1} \alpha_i = 1) \qquad (34)$$

такие, что выполняются равенства

$$\sum_{i=1}^{n+1} \left(\alpha_i \operatorname{sgn} \hat{\phi}_n(\tau_i)\right) e_j(\tau_i) = 0 \quad (j = 0,1,\cdots,n-1), \qquad (35)$$

причём

$$|\hat{\phi}_n(\tau_i)| = \|\hat{\phi}_n(\cdot)\|_{C([a,b])} \quad (j = 0,1,\cdots,n-1). \qquad (36)$$

Замечание 5.

Совокупность равенств (35) эквивалентна соотношению

$$\sum_{i=1}^{n+1} \left(\alpha_i \operatorname{sgn} \hat{\phi}_n(\tau_i)\right) \phi(\tau_i) = 0 \quad (\forall \phi(\cdot) = \sum_{k=0}^{n-1} x_k e_k(\cdot) \in L_{n-1}), \qquad (37)$$

которое принято называть ***основным тождеством для задачи*** (P_1). *Учитывая это, Теорему 6 можно переформулировать в виде следующего* ***обобщённого критерия Чебышёва****:*

среди всех обобщённых полиномов ***n –го порядка с единичным старшим коэффициентом*** *обобщённый полином (32) является* ***наименее уклоняющимся*** *(в метрике* C *на отрезке [a, b])* ***от нуля*** *в том, и только в том случае, когда он достигает на этом отрезке максимума своего модуля в* ***(n+1)-ой различной точке*** *с последовательной переменой в них знака (указанные в этом критерии точки принято называть точками* ***чебышёвского альтернанса****).*

В свою очередь, справедлива

Теорема 7 (критерий решения задачи (P_2)).

Для того чтобы обобщённый полином

$$\hat{\phi}_n(\cdot) = e_n(\cdot) + \sum_{k=0}^{n-1} \hat{x}_k e_k(\cdot) \in \Phi \qquad (38)$$

был бы решением задачи (P_2) *необходимо и достаточно, чтобы выполнялись равенства*

$$\int_a^b \operatorname{sgn} \hat{\phi}_n(t) e_j(t) dt = 0 \quad (j = 0,1,\cdots,n-1). \qquad (39)$$

Замечание 6.

Согласно Теореме 5 все нули обобщённого полинома (38) действительны, различны и лежат в интервале (a, b). Если эти нули $\{\xi_j\}_{j=1}^{n}$ *упорядочены:*

$$a < \xi_1 < \xi_2 < \cdots < \xi_n < b, \qquad (40)$$

то, полагая

$$a =: \xi_0, \qquad b =: \xi_{n+1}, \qquad (41)$$

соотношения (39) можно свести к равенствам

$$\sum_{i=0}^{n} (-1)^{n-i} \int_{\xi_i}^{\xi_{i+1}} e_j(t) dt = 0 \quad (j = 0,\cdots,n-1). \qquad (42)$$

Совокупность равенств (42) эквивалентна соотношению:

$$\sum_{i=0}^{n} (-1)^{n-i} \int_{\xi_i}^{\xi_{i+1}} \phi(t) dt = 0 \quad (\forall \phi(\cdot) = \sum_{k=0}^{n-1} x_k e_k(\cdot) \in L_{n-1}), \qquad (43)$$

которое принято называть ***основным тождеством для задачи*** (P_2).

Наконец, справедлива

Теорема 8 (критерий решения задачи (P_3)).

Для того чтобы обобщённый полином

$$\hat{\phi}_n(\cdot) = e_n(\cdot) + \sum_{k=0}^{n-1} \hat{x}_k e_k(\cdot) \in \Phi \qquad (44)$$

был бы решением задачи (P_3) *необходимо и достаточно, чтобы выполнялись равенства*

$$\int_a^b \hat{\phi}_n(t) e_j(t) dt = 0 \quad (j = 0,1,\cdots,n-1). \qquad (45)$$

Замечание 7.

Совокупность равенств (45) эквивалентна соотношению:

$$\int_a^b \hat{\phi}_n(t)\phi(t)dt = 0 \qquad (\forall \phi(\cdot) = \sum_{k=0}^{n-1} x_k e_k(\cdot) \in L_{n-1}), \qquad (46)$$

которое принято называть ***основным тождеством для задачи*** (P_3).

ON SOME EXTREMAL PROBLEMS IN FINITE DIMENSIONAL TCHEBYCHEFF SPACES

V.B.Demidovich, A.S.Kochurov

Abstract: In a finite-dimensional Tchebycheff space considered questions of explicit construction of generalized polynomial of least deviation from zero (in the metrics C, L_1 and L_2) as well as of inequalities for derivatives in these metrics. These polynomials play an important role in Computational Mathematics.

Keywords: Tchebycheff space, inequality for derivatives, convexity, subdifferentiation.

Литература

1. В.Б.Демидович, Г.Г.Магарил-Ильяев, В.М.Тихомиров. Экстремальные задачи для линейных функционалов на чебышёвских пространствах. Фундаментальная и прикладная математика, т. 11 (2005), № 2, 87-100

2. Ст.Пашковский. Вычислительные применения многочленов и рядов Чебышёва. М., "Наука", 1983

3. П.К.Суетин. Классические ортогональные многочлены. М., "Наука", 1979

4. А.Ф.Тиман. Теория приближения функций действительного переменного. М., "Физматлит", 1960

5. В.М.Тихомиров. Некоторые вопросы теории приближений. М., "Изд -во Моск. ун -та", 1976

6. Г.Г.Магарил-Ильяев, В.М.Тихомиров «Выпуклый анализ и его приложения». М., "Эдиториал УРСС", 2002

7. А.Д.Иоффе, В.М.Тихомиров. Теория экстремальных задач. М., "Наука", 1974

8. В.Б.Демидович, А.В.Рождественский. Практическая теория экстремума: гладкие и выпуклые экстремальные задачи. М., "НИИСИ РАН", 2011

9. С.Карлин, В.Стадден. Чебышёвские системы и их применение в анализе и статистике». М., "Наука", 1976

10. М.Г.Крейн, А.А.Нудельман. Проблемы моментов Маркова и экстремальные задачи. М., "Наука", 1973

11. В.Б.Демидович. Приближённые вычисления с помощью обобщённых полиномов из чебышёвских пространств: чебышёвские обобщённые полиномы. М., "Изд -во Моск. ун -та", 1990

12. В.Б.Демидович. Приближённые вычисления с помощью обобщённых полиномов из чебышёвских пространств: простое интерполирование, кратное интерполирование, формулы тейлоровского типа». М., "Изд -во Моск. ун -та", 1994

13. G.V.Milovanović, D.S.Mitrinović, Th.M.Rassias. Topics in Polynomials: Extremal Problems, Inequalities, Zeros. Singapore, "World Scientific Publ.", 1994

14. Q.Rahman, G.Schmeisser. Analytic Theory of Polynomials. Oxford New York, "Clarendon Press", 2002

Численное моделирование двухфазной фильтрации нефти и воды

Р.М. Кац[1], Е.Р. Волгин, И.В. Афанаскин[1]

1 – кандидат технических наук.

Аннотация: Рассматривается математическая модель двухфазной несмешивающейся фильтрации двух слабосжимаемых жидкостей (нефти и воды). Описывается концепция суммарной скорости фильтрации. Приводится система конечно-разностных уравнений для указанной модели. Описывается программная реализация этой модели и результаты тестирования программы.

Ключевые слова: численное моделирование, двухфазная фильтрация, слабосжимаемые жидкости

Введение

Запасы большинства нефтяных месторождений России в значительной мере выработаны, в основном с применением заводнения. Достигнутая при этом нефтеотдача редко превышает 30-40% начальных запасов нефти. В некоторых случаях нефтеотдача не достигает и 25 %. В итоге накопление остаточных запасов нефти в обводненных пластах во многих нефтедобывающих регионах страны идет весьма быстрыми темпами.

Накопленный опыт применения и опробования известных методов повышения нефтеотдачи пластов в ряде случаев свидетельствует об их недостаточной технологической и (или) экономической эффективности из-за значительной выработанности (низкой текущей нефтенасыщенности) объектов, истощения пластовой энергии, высокой обводненности пластов, существенной неоднородности коллекторов, наличия линз и блоков, в различной мере изолированных тектонически, стратиграфически или литологически, нехватки и дороговизны материалов, реагентов, оборудования и квалифицированных специалистов, высокой себестоимости добываемой нефти и пр.

В этой связи проблема извлечения из обводненных пластов остаточной нефти требует изучения состояния длительно разрабатываемых залежей, установления характера распределения текущей насыщенности горных пород нефтью, водой и газом, определения основных факторов, влияющих на эффективность извлечения нефти, совершенствования методов воздействия на выработанные пласты для повышения их нефтеотдачи. Крайне важным становится вопрос определения местоположения целиков остаточной нефти в обводненных пластах. Поиск этих невыработанных зон должен осуществляться с помощью комплексирования различных методов исследования пласта и скважин (сейсморазведка, геофизические, промыслово-геофизические, гидродинамические исследования пластов и скважин, трассерные исследования, исследования керна и пр.) и численного моделирования процессов разработки нефтяного пласта для воспроизведения истории разработки с целью построения карт текущей плотности запасов нефти.

Поэтому актуальной является проблема численного моделирования процессов разработки нефтяных месторождений. Для такого моделирования существует специальное программное обеспечение – гидродинамические симуляторы.

Наиболее распространенными в России гидродинамическими симуляторами являются Eclipse компании Schlumberger; VIP компании Landmark; Tempest-MORE компании Roxar; комплекс программ Imex, Gem, Stars компании Computer Modelling Group Ltd; «Техсхема» тюменского отделения «СургутНИПИнефть» ОАО «Сургутнефтегаз»; t-Navigator компании Rock Flow Dynamics; МКТ компании TimeZYX.

Кроме того, существует множество других симуляторов (коммерческих и свободно распространяемых), находящихся на разных стадиях развития (как активно совершенствуемых и расширяемых, так и заброшенных). Например: DuMu^x немецкого университета Universitat Stuttgart; MATLAB Reservoir Simulation Toolbox компании SINTEF Applied Matemathics; The Open Porous Media Initiative создаваемый в союзе SINTEF Applied Math, IRIS Energy, Uni CIPR, Bergen, University of Bergen, University of Stuttgart, Statoil, Total E&P, Ceetron AS, University of Heidelberg, HPC-Simulation-Software & Services; OpenFOAM от Lehrstuhl für Modellierung und Simulation; BS Eagle компании GitHub; 3DSL компании Streamsim Technologies; SimMatch от ИПНГ РАН; NGT BOS от ОАО «УфаНИПИнефть»; РН-КИМ компании ОАО «НК «Роснефть»; НИМФА от ФГУП РФЯЦ-ВНИИЭФ; LAURA от ОАО «ВНИИнефть» и т.д.

Стоимость одной лицензии коммерческих зарубежных гидродинамических симуляторов составляет несколько сотен тысяч долларов США, отечественных – несколько десятков. Некоммерческие симуляторы также имеют существенные недостатки (которых в значительной мере не лишены и коммерческие симуляторы): низкая степень доступности для сторонних организаций и физических лиц, нарекания на

точность расчетов и адекватность математических моделей, невозможность для пользователя вносить коррективы и изменения в программу и пр.

Поэтому для проведения исследований в области разработки нефтяных месторождений актуальным является создание собственного программного обеспечения для численного математического моделирования процессов разработки нефтяных месторождений. Как уже указывалось выше, большинство месторождений России разрабатывается с применением заводнения, поэтому на первом этапе актуальным является разработка программы для моделирования фильтрации нефти и воды.

1. Математическая модель двухфазной фильтрации нефти и воды

Система уравнений, описывающая упругую двухфазную фильтрацию с учетом капиллярных и гравитационных сил, состоит из двух уравнений сохранения массы (для нефти и воды) и обобщенного закона Дарси [1, 2]:
уравнение сохранения массы

$$\frac{\partial}{\partial t}\left(\frac{m\,S_\phi}{B_\phi}\right)+\nabla\frac{\mathbf{w}_\phi}{B_\phi}= =-\sum_j q_{\phi_j}\,\delta(x-x_j,y-y_j,z-z_j) \tag{1.1}$$

обобщенный закон Дарси

$$\boldsymbol{w}_\phi=-k\frac{f_\phi(S_\phi)}{\mu_\phi}\left(\nabla P_\phi-\gamma_\phi\nabla D\right), \tag{1.2}$$

$$\sum_\phi S_\phi=1, \tag{1.3}$$

$$P_н=P_в+P_c, \tag{1.4}$$

капиллярное давление

$$P_c=\sigma\cos\theta\sqrt{\frac{m}{k}}\,J(S), \tag{1.5}$$

дельта-функция

$$\delta(x-x_j,y-y_j,z-z_j)= =\begin{cases}\dfrac{1}{dxdydz}-\text{в точках }(x_j,y_j,z_j)\\ \qquad\text{размещения источника или стока } j\\ 0-\text{в остальной области}\end{cases} \tag{1.6}$$

где q_{ϕ_j} - интенсивность стока (положительна) или источника (отрицательна) по фазе ϕ в поверхностных условиях, ϕ = *н, в*.

2. Концепция суммарной скорости

Уравнения (1.1)-(1.2) являются нелинейными по давлению, при их численном решении приходится производить итерации по нелинейности. В то же время, систему нефть-вода-пористая среда можно в первом приближении считать слабосжимаемой. Такое предположение представляется вполне удовлетворительным для практических целей, с другой стороны, оно позволяет существенно упростить уравнения (1.1)-(1.2), произведя линеаризацию по давлению. Уравнение (2.1) производит линеаризацию пористости по давлению:

$$m=m_0\left(1+\beta_c\left(P-P_0\right)\right). \tag{2.1}$$

Уравнение (2.2) производит линеаризацию по давлению объемных коэффициентов:

$$B_\phi=B_{0\phi}\left(1-\beta_\phi\left(P-P_0\right)\right), \tag{2.2}$$

где P_0 – опорное давление, m_0, $B_{0\phi}$ – пористость и объемный коэффициент фазы ϕ при давлении P_0. Подстановка линеаризованных коэффициентов в (1.1) дает следующее приближение:

$$\begin{aligned}&\frac{\partial}{\partial t}\frac{m\,S_\phi}{B_\Phi}=\frac{m}{B_\Phi}\frac{\partial S_\phi}{\partial t}+S_\phi\frac{\partial}{\partial t}\frac{m}{B_\Phi}\approx\\ &\approx\frac{m_0\left(1+\beta_c\left(P-P_0\right)\right)}{B_{0\phi}\left(1-\beta_\phi\left(P-P_0\right)\right)}\frac{\partial S_\phi}{\partial t}+\\ &+S_\phi\frac{\partial}{\partial t}\frac{m_0\left(1+\beta_c\left(P-P_0\right)\right)}{B_{0\phi}\left(1-\beta_\phi\left(P-P_0\right)\right)}\approx\\ &\approx\frac{m_0}{B_{0\phi}}\frac{\partial S_\phi}{\partial t}+\left(\frac{m_0\,\beta_c}{B_{0\phi}}-m_0\,\frac{-B_{0\phi}\beta_\phi}{B_{0\phi}^{\,2}}\right)S_\phi\frac{\partial P}{\partial t}=\\ &=\frac{m_0}{B_{0\phi}}\frac{\partial S_\phi}{\partial t}+m_0\,\frac{\beta_c+\beta_\phi}{B_{0\phi}}S_\phi\frac{\partial P}{\partial t}=\\ &=\frac{m_0}{B_{0\phi}}\left(\frac{\partial S_\phi}{\partial t}+\left(\beta_c+\beta_\phi\right)S_\phi\frac{\partial P}{\partial t}\right).\end{aligned} \tag{2.3}$$

Подстановка (2.3) в (1.1) приводит к следующему уравнению:

$$\frac{m_0}{B_{0\phi}}\left(\frac{\partial S_\phi}{\partial t}+\left(\beta_c+\beta_в\right)S_\phi\frac{\partial P}{\partial t}\right)+\nabla\frac{\mathbf{w}_\phi}{B_{0\phi}}= =-\sum_j q_{\phi_j}\,\delta\left(x-x_j,y-y_j,z-z_j\right) \tag{2.4}$$

Складывая уравнения (2.4) для ϕ = в, н, получаем окончательную систему уравнений:

$$\begin{aligned}&m_0\left(\beta_c+\left(\beta_в-\beta_н\right)S_в+\beta_н\right)\frac{\partial P}{\partial t}+\nabla\mathbf{w}=\\ &=-\sum_j\sum_\phi B_{0\phi}q_{\phi_j}\,\delta\left(x-x_j,y-y_j,z-z_j\right)\end{aligned} \tag{2.5}$$

$$\begin{aligned}&m_0\left(\frac{\partial S_в}{\partial t}+\left(\beta_c+\beta_в\right)S_в\frac{\partial P}{\partial t}\right)+\nabla\mathbf{w}_в=\\ &=-\sum_j B_{0в}q_{в_j}\,\delta\left(x-x_j,y-y_j,z-z_j\right)\end{aligned} \tag{2.6}$$

Уравнение (2.5) используется для вычисления давления, а уравнение (2.6) – для насыщенности. В этой системе скорость жидкости представляет собой сумму

$$\mathbf{w}=\mathbf{w}_н+\mathbf{w}_в. \tag{2.7}$$

Выведем уравнения для скоростей жидкости и воды, учитывающие гравитационные и капиллярные силы. Для этого выпишем закон Дарси для воды и для нефти:

$$\begin{cases} \mathbf{w}_в = -kc_в\left(\nabla P_в - \gamma_в \nabla D\right) \\ \mathbf{w}_н = -kc_н\left(\nabla P_н - \gamma_н \nabla D\right) \\ P_н = P_в + P_c \end{cases} \quad (2.8)$$

Пусть P – давление в нефтяной фазе. Исключим давление в водной фазе из уравнения для скорости воды:

$$\begin{cases} \mathbf{w}_в = -kc_в\left(\nabla P - \nabla P_c - \gamma_в \nabla D\right) \\ \mathbf{w}_н = -kc_н\left(\nabla P - \gamma_н \nabla D\right) \\ P_в = P - P_c \end{cases} \quad (2.9)$$

Сложим первое и второе уравнение системы (2.9)

$$\begin{cases} \mathbf{w} = -kc_н\left(\nabla P - \gamma_н\right) - kc_в\left(\nabla P - \nabla P_c - \gamma_в \nabla D\right) \\ \mathbf{w}_в = -kc_в\left(\nabla P - \nabla P_c - \gamma_в \nabla D\right) \end{cases} \quad (2.10)$$

Раскроем скобки в первом уравнении системы (2.10) и сгруппируем члены относительно ∇P

$$\begin{aligned} &\mathbf{w} = -kc_н\left(\nabla P - \gamma_н\right) - kc_в\left(\nabla P - \nabla P_c - \gamma_в \nabla D\right) = \\ &= -k\left(c_н + c_в\right)\nabla P + kc_в \nabla P_c + k\left(c_н \gamma_н + c_в \gamma_в\right)\nabla D = \\ &= -k\left(c_н + c_в\right)\left(\nabla P - \varphi_в \nabla P_c - \left(\varphi_н \gamma_н + \varphi_в \gamma_в\right)\nabla D\right) \end{aligned} \quad (2.11)$$

Выразим ∇P

$$\nabla P = -\frac{1}{k\left(c_н + c_в\right)}\mathbf{w} + \varphi_в \nabla P_c + \left(\varphi_н \gamma_н + \varphi_в \gamma_в\right)\nabla D\,. \quad (2.12)$$

Подставим ∇P в уравнение для скорости воды в (2.10):

$$\begin{aligned} &\mathbf{w}_в = -kc_в\left(\nabla P - \nabla P_c - \gamma_в \nabla D\right) = \\ &= -kc_в\left(-\frac{1}{k\left(c_н + c_в\right)}\mathbf{w} + \varphi_в \nabla P_c + \left(\varphi_н \gamma_н + \varphi_в \gamma_в\right)\nabla D\right) + \\ &+ kc_в \nabla P_c + kc_в \gamma_в \nabla D = \\ &= \frac{c_в}{c_н + c_в}\mathbf{w} - kc_в\left(\varphi_н \gamma_н + \varphi_в \gamma_в\right)\nabla D - c_в k \varphi_в \nabla P_c + \\ &+ kc_в \nabla P_c + kc_в \gamma_в \nabla D = \\ &= \varphi_в \mathbf{w} + kc_в\left(\gamma_в - \varphi_н \gamma_н - \varphi_в \gamma_в\right)\nabla D + \left(1 - \varphi_в\right)kc_в \nabla P_c = \\ &= \varphi_в \mathbf{w} + kc_в\left(\varphi_в \gamma_в + \varphi_н \gamma_в - \varphi_н \gamma_н - \varphi_в \gamma_в\right)\nabla D + \\ &+ kc_в \varphi_н \nabla P_c = \\ &= \varphi_в \mathbf{w} + kc_в \varphi_н\left(\gamma_в - \gamma_н\right)\nabla D + \varphi_в c_н k \nabla P_c = \\ &= \varphi_в \mathbf{w} + k\varphi_в c_н\left(\gamma_в - \gamma_н\right)\nabla D + \varphi_в c_н k \nabla P_c = \\ &= \varphi_в\left(\mathbf{w} + kc_н\left(\Delta\gamma_{в,н}\nabla D + \nabla P_c\right)\right) \end{aligned} \quad (2.13)$$

где: разность удельных весов

$$\Delta\gamma_{в,н} = \gamma_в - \gamma_н\,, \quad (2.14)$$

функция Баклея-Леверетта (доля фазы в потоке)

$$\varphi_ф = \frac{c_ф}{\sum_i c_i}\,, \quad (2.15)$$

подвижность фазы

$$c_ф = \frac{f_ф\left(S_ф\right)}{\mu_ф}\,. \quad (2.16)$$

Подводя итог проделанным выкладкам, получаем следующие уравнения для скоростей:

$$\mathbf{w} = -k\left(c_н + c_в\right)\left(\nabla P - \gamma \nabla D - \varphi_в \nabla P_c\right), \quad (2.17)$$

$$\mathbf{w}_в = \varphi_в\left(\mathbf{w} + kc_н\left(\Delta\gamma_{вн}\nabla D + \nabla P_c\right)\right). \quad (2.18)$$

В уравнении (2.17) удельный вес смеси, обозначенный γ, рассчитывается по следующей формуле:

$$\gamma = \varphi_в \gamma_в + \varphi_н \gamma_н\,. \quad (2.19)$$

3. Система разностных уравнений

Введем понятие «блок», или «ячейка», характеризующее область, ограниченную плоскостями, проведенными между данной и соседними точками. Для вычисления первых и вторых производных по пространственной координате и по времени в одномерном случае используем шеститочечный шаблон, рис. 1. Шаблон, построенный подобным образом для вычисления производных в трехмерной задаче, будет включать 14 точек.

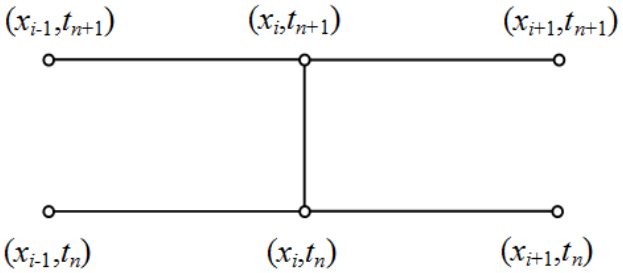


Рис. 1. Шеститочечный шаблон, применяемый для решения одномерной задачи

Проинтегрируем систему уравнений (2.5)-(2.6) по объему ячейки и запишем явную для насыщенности и неявную для давления конечно-разностную систему уравнений (IMPES):

$$\begin{aligned} &m_0 V_A\left(\beta_c + \left(\beta_в - \beta_н\right)S_A^n + \beta_н\right)\frac{P_A^{n+1} - P_A^n}{\tau_{n+1}} + \sum_{i=1}^{N_c}\tilde{W}_{A,B_i}^{n+1} = \\ &= -\sum_j^{N_w}\sum_{ф=в,н} B_{0ф} Q_{ф_j}^n \end{aligned} \quad (3.1)$$

$$\frac{m_0 V_A}{\tau_{n+1}}\left(S_A^{n+1} - S_A^n + \left(\beta_c + \beta_в\right)S_A^n\left(P_A^{n+1} - P_A^n\right)\right) + \sum_{i=1}^{N_c} W_{в_{A,B_i}}^{n+1} = -\sum_{j=1}^{N_w} B_{0в} Q_{в_j}^n \quad (3.2)$$

$$\tilde{W}_{A,B_i}^{n+1} = -\tilde{W}_{B_i,A}^{n+1} = g_{A,B_i}^n\left(P_{B_i}^{n+1} - P_A^{n+1}\right), \quad (3.3)$$

$$\tilde{W}_{в_{A,B_i}}^{n+1} = -\tilde{W}_{в_{B_i,A}}^{n+1} = \varphi_{в_{A,B_i}}^n \tilde{W}_{A,B_i}^{n+1}\,, \quad (3.4)$$

где скорость $W_{A,B}$ перетока жидкости между ячейками A и B определяется по формуле

$$W_{A,B} = W_{B,A} = \begin{cases} g_{A,B}\left(P_B - P_A\right), P_A < P_B \\ g_{A,B}\left(P_A - P_B\right), P_A > P_B \end{cases} \quad (3.5)$$

$$g_{B,A} = g_{A,B} = -\frac{\left(c_{н_A} + c_{в_A} + c_{н_B} + c_{в_B}\right)}{2}T_{A,B} \quad (3.6)$$

$$\tilde{W}_{A,B} = -\tilde{W}_{B,A} = \begin{cases} W_{A,B}, P_A < P_B \\ -W_{B,A}, P_A > P_B \end{cases} \qquad (3.7)$$

$T_{A,B}$ - проводимость между ячейками A и B, $Q^n_{\phi_j}$ – отбор или закачка флюида ϕ в единицу времени по скважине j на момент t_n, n – номер временного шага, для которого решение уже получено, n+1 – номер временного шага, для которого ищется решение, τ_{n+1} - интервал времени между t_n и t_{n+1}, P^n и S^n – давление и насыщенность в момент времени t_n.

В случае учета капиллярных и гравитационных сил уравнения (3.3-3.4) принимают вид:

$$\tilde{W}^{n+1}_{A,B_i} = -\tilde{W}^{n+1}_{B_i,A} = g^n_{A,B_i}\left[P^{n+1}_{B_i} - P^{n+1}_A - \gamma^n_{A,B_i}\Delta D_{A,B_i} - \right.$$
$$\left. - \frac{1}{2}\left(\varphi^n_{в_A} + \varphi^n_{в_{B_i}}\right)\left(P^n_{c_{B_i}} - P^n_{c_A}\right)\right] \qquad (3.8)$$

$$\tilde{W}^{n+1}_{в_{A,B_i}} = -\tilde{W}^{n+1}_{в_{B_i,A}} = \varphi^n_{в_{A,B_i}} W^{n+1}_{A,B_i} +$$
$$+ T_{A,B_i}\left(d^n_{в_{A,B_i}}\Delta\gamma^n_{вн_{A,B_i}}\Delta D_{A,B_i} + \zeta^n_{в_{A,B_i}}\left(P^n_{c_{B_i}} - P^n_{c_A}\right)\right) \qquad (3.9)$$

где удельные веса:

$$\Delta\gamma^n_{вн_{A,B}} = \Delta\gamma^n_{вн_{B,A}} = \gamma_в - \gamma_н = const, \qquad (3.10)$$

$$\gamma^n_{A,B_i} = \gamma^n_{B_i,A} = \frac{\gamma^n_A + \gamma^n_{B_i}}{2}, \qquad (3.11)$$

$$\gamma^n_A = \varphi^n_{в_A}\gamma_в + \varphi^n_{н_A}\gamma_н, \qquad (3.12)$$

превышение:

$$\Delta D_{A,B} = -\Delta D_{B,A} = D_B - D_A, \qquad (3.13)$$

доля воды в потоке:

$$\varphi^n_{в_{A,B}} = \varphi^n_{в_{B,A}} = \begin{cases} \varphi^n_{в_B}, & W^{n+1}_{A,B} < 0 \\ \varphi^n_{в_A}, & W^{n+1}_{A,B} \ge 0 \end{cases} \qquad (3.14)$$

доля воды:

$$\varphi^n_{в_A} = \frac{c^n_{в_A}}{c^n_{в_A} + c^n_{н_A}}, \qquad (3.15)$$

дебит воды:

$$Q^n_{в_j} = \begin{cases} Q^n_j\varphi\left(S_{в_{res}}\right), & Q^n_j \le 0 \\ Q^n_j\varphi\left(S_{в_A}\right), & Q^n_j > 0 \end{cases} \qquad (3.16)$$

подвижность фазы:

$$c^n_{\phi_A} = \frac{f_\phi\left(S^n_A\right)}{\mu_\phi}, \qquad (3.17)$$

вспомогательные коэффициенты:

$$d^n_{в_{A,B}} = \begin{cases} \varphi^n_{в_B}c^n_{н_A}, & \Delta\gamma^n_{вн_{A,B}}\cdot\Delta D_{A,B} \ge 0 \\ \varphi^n_{в_A}c^n_{н_B}, & \Delta\gamma^n_{вн_{A,B}}\cdot\Delta D_{A,B} < 0 \end{cases} \qquad (3.18)$$

$$\zeta^n_{в_{A,B}} = \begin{cases} \varphi^n_{в_A}c^n_{н_B}, & P^n_{c_B} - P^n_{c_A} \ge 0 \\ \varphi^n_{в_B}c^n_{н_A}, & P^n_{c_B} - P^n_{c_A} < 0 \end{cases} \qquad (3.19)$$

Межблочные свойства ячеек, такие как $\Delta\gamma$, ΔD, W и др., приписываются ячейке с большим номером.

В уравнениях (3.1)-(3.2) граничные условия – дебит или приемистость Q скважины, расположенной в ячейке A, могут задаваться как явно, так и неявно. Неявный способ подразумевает задание давления в стволе скважины – забойного давления, связанного с дебитом по формуле Дюпюи:

$$Q_{пл} = \Omega_{прод}\left(P_{пл} - P_{заб}\right), \qquad (3.20)$$

где

$$\Omega_{прод} = \frac{2\pi k_A H_A}{ln\left(\frac{R_k}{r_c}\right)}\sum_\phi c_\phi$$ - коэффициент продуктивности, (3.21)

H_A – эффективная насыщенная толщина, r_c - радиус скважины, R_k - эффективный радиус контура питания, определяемый по формуле Писмана [3].

С учетом (3.3)-(3.21) перепишем уравнения (3.1)-(3.2) в следующем виде:

$$m_0 V_A\left(\beta_c + (\beta_в - \beta_н)S^n_A + \beta_н\right)\frac{P^{n+1}_A - P^n_A}{\tau_{n+1}} +$$
$$+ \sum_{i=1}^{N_c} g^n_{A,B_i}\left[P^{n+1}_{B_i} - P^{n+1}_A - \gamma^n_{A,B_i}\Delta D_{A,B_i} - \right.$$
$$\left. - \frac{1}{2}\left(\varphi^n_{в_A} + \varphi^n_{в_{B_i}}\right)\left(P^n_{c_{B_i}} - P^n_{c_A}\right)\right] =$$
$$= -\sum_j^{N_w}\begin{cases} \sum\limits_{\phi=в,н} B_{0\phi}Q^n_{\phi_j}, & \text{задан дебит} \\ \Omega^n_{ж_j}\left(P^{n+1}_A - P_{заб_j}\right), & \text{задано забойное давление} \end{cases} \qquad (3.22)$$

$$\frac{m_0 V_A}{\tau_{n+1}}\left(S^{n+1}_A - S^n_A + (\beta_c + \beta_в)S^n_A\left(P^{n+1}_A - P^n_A\right)\right) + \sum_{i=1}^{N_c}\tilde{W}^{n+1}_{в_{A,B_i}} =$$
$$= -\sum_j^{N_w}\begin{cases} B_{0в}Q^n_{в_j}, & \text{задан дебит} \\ \Omega^n_{в_j}\left(P^{n+1}_A - P_{заб_j}\right), & \text{задано забойное давление} \end{cases} \qquad (3.23)$$

Систему линейных уравнений для вычисления давления можно получить раскрытием скобок и приведением подобных членов в (3.22):

$$\left(E^n_A - \sum_{i=1}^{Nbr} g^n_{A,B_i} + \sum_{j=1}^{N_w}\begin{bmatrix} 0, & \text{задан дебит} \\ \Omega^n_{ж_j}, & \text{задано забойное давление} \end{bmatrix}\right)P^{n+1}_A +$$
$$+ \sum_{i=1}^{Nbr} g^n_{A,B_i}P^{n+1}_{B_i} =$$
$$= F^n_{A,B_i} + E^n_A P^n_A + \sum_{j=1}^{N_w}\begin{bmatrix} -\sum\limits_{\phi=в,н} B_{0\phi}Q^n_{ж_j}, & \text{задан дебит} \\ \Omega^n_{ж_j}P_{заб_j}, & \text{задано забойное давление} \end{bmatrix} \qquad (3.24)$$

где

$$E^n_A = \left(\beta_c + (\beta_в - \beta_н)S^n_A + \beta_н\right)\frac{m_{0_A}V_A}{\tau_{n+1}}, \qquad (3.25)$$

$$F^n_{A,B_i} = -F^n_{B_i,A} = \sum_{i=1}^{Nbr} g^n_{A,B_i}\left(\gamma^n_{A,B_i}\Delta D_{A,B_i} + \frac{1}{2}\left(\varphi^n_{в_A} + \varphi^n_{в_{B_i}}\right)\left(P^n_{c_{B_i}} - P^n_{c_A}\right)\right) \qquad (3.26)$$

Поле водонасыщенности на момент t_{n+1} рассчитывается явно по формуле:

$$S_A^{n+1} = \left(1 - \left(\beta_c + \beta_в\right)\left(P_A^{n+1} - P_A^n\right)\right)S_A^n +$$
$$+ \frac{\tau_{n+1}}{m_{0_A} V_A}\left(- \sum_{j=1}^{N_w}\left[\begin{array}{c} B_{0в} Q_{в_j}^n, \text{ задан дебит} \\ \Omega_{в_j}^n \left(P_A^{n+1} - P_{заб_j}\right), \text{ задано забойное давление} \end{array}\right] \right.$$
$$\left. - \sum_{i=1}^{N_c} \tilde{W}_{в_{A,B_i}}^{n+1} \right) \qquad (3.27)$$

где $\tilde{W}_{в_{A,B_i}}^{n+1}$ рассчитывается по формуле (3.4).

4. Программа для математического моделирования двухфазной фильтрации нефти и воды

Программа написана на языке C++.

Проводимость между узлами может быть вычислена одним из методов Oldtran, Newtran, Harint. В программе реализован расчет для блочно-центрированной разностной сетки или для сетки в геометрии «угловой точки».

Для решения системы линейных уравнений реализованы следующие методы: метод вложенных сечений, блочный метод сопряженных градиентов, метод сопряженных градиентов с неполным разложением матрицы ILU, метод сопряженных градиентов с неполным разложением матрицы и естественным упорядочением.

Добывающие скважины могут управляться дебитом жидкости в поверхностных или в пластовых условиях, либо дебитом нефти или воды в поверхностных условиях. Нагнетательные скважины могут управляться расходом по жидкости в поверхностных или пластовых условиях, либо расходом по воде в поверхностных или пластовых условиях. В качестве технологического ограничения на добывающих и нагнетательных скважинах можно задавать забойное давление. Так же скважины могут управляться забойным давлением и ограничиваться дебитом (расходом). В качестве экономического ограничения на добывающих скважинах можно использовать обводненность.

Программа для математического моделирования несмешивающейся двухфазной фильтрации нефти и воды с учетом гравитационных и капиллярных сил получила название Dz10. Авторы программы: Р.М. Кац, Е.Р. Волгин.

5. Тестирование программы

Для тестирования программы Dz10 была построена численная модель элемента пятиточечной системы разработки нефтяного месторождения, рис. 2-3.

Количество ячеек модели по осям X, Y, Z – 20x20x20 шт. Размеры ячеек 25x25x1 м. Горизонтальная проницаемость пласта 100 мД, вертикальная проницаемость пласта 10 мД. Пористость 20 %.

Свойства воды: объемный коэффициент 1,0022 $м^3/м^3$, вязкость в пластовых условиях 0,33 сПз, сжимаемость $1{,}0*10^{-5}$ 1/атм, плотность в поверхностных условиях 1018 $кг/м^3$. Свойства нефти: объемный коэффициент 1,22 $м^3/м^3$, вязкость в пластовых условиях 0,82 сПз, сжимаемость $7{,}0*10^{-5}$ 1/атм, плотность в поверхностных условиях 833 $кг/м^3$, растворимость газа в нефти 50 $м^3/м^3$, давление насыщения нефти газом 120 атм. Сжимаемость породы $2{,}6*10^{-4}$ 1/атм. Опорное давление 250 атм.

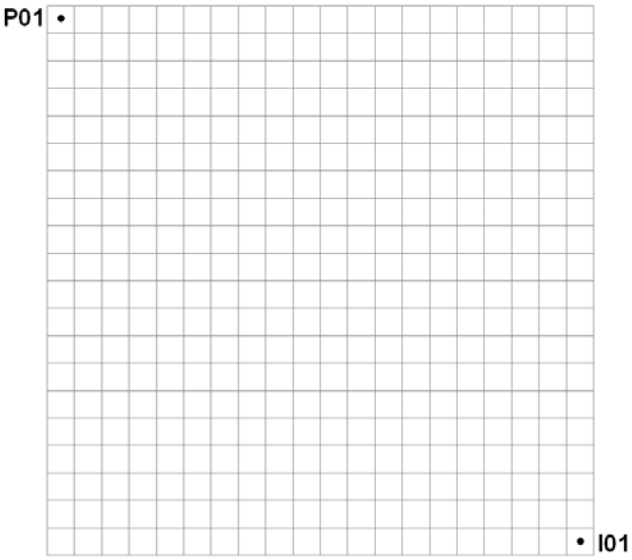


Рис. 2. Элемент пятиточечной системы разработки, P01 - добывающая скважина, I01 - нагнетательная скважина

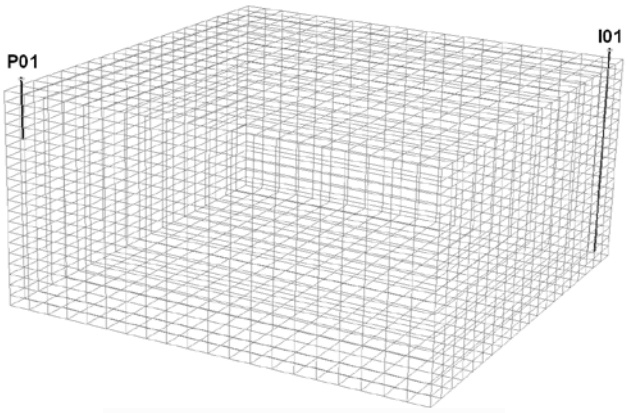


Рис. 3. Расчетная сетка со скважинами, P01 - добывающая скважина, I01 - нагнетательная скважина

Относительные фазовые проницаемости и капиллярное давление приведены на рис. 4. Равновесное начальное распределение давления и насыщенности по слоям с учетом капиллярных и гравитационных сил представлено на рис. 5-6.

Добывающая скважина вскрывает первые 5 слоев, а нагнетательная – последние 5 слоев. Добывающая скважина работает с постоянным дебитом жидкости 95 $м^3$/сут, ограничением по минимальному забойному давлению 150 атм. и ограничением по максимальной обводненности 95 %. Нагнетательная скважина работает при постоянной закачке воды с расходом 100 $м^3$/сут и ограничением по максимальному забойному давлению 400 атм.

Поле нефтенасыщенности через 1 месяц после начала работы скважин представлено на рис. 6.

Результаты расчетов по описанной модели с помощью программы Dz10 сравнивались с результатами расчетов с помощью программы Eclipse компании Schlumberger.

В программе ECLIPSE моделирование проводилось с помощью полностью неявной (как для давления, так и для насыщенности) расчетной схемы. Для решения нелинейных уравнений используется метод Ньютона. При этом матрица Якоби полностью разложена по всем переменным, что обеспечивает квадратичную скорость сходимости. Система линейных уравнений на каждой ньютоновской итерации решается методом гнездовой факторизации с ускорением за счет применения метода Ортомина [4].

На рис. 8-14 представлены результаты моделирования с применением Dz10 и Eclipse. Первый год выдача результатов расчетов производилась помесячно, затем один раз в год.

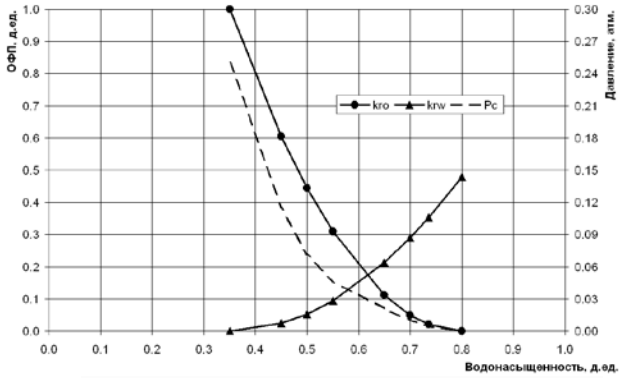


Рис. 4. Относительные фазовые проницаемости (ОФП) по нефти (kro) и воде (krw) и капиллярное давление (Pc)

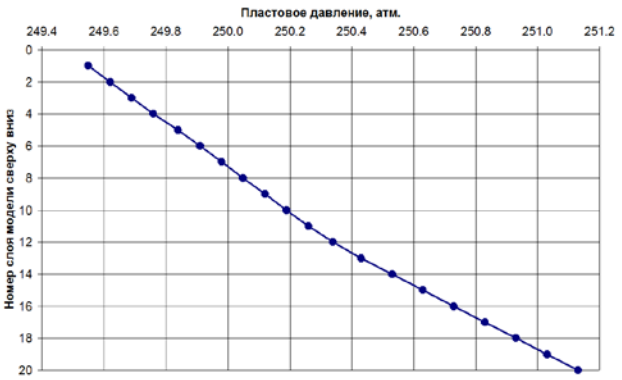


Рис. 5. Начальное распределение давления по слоям

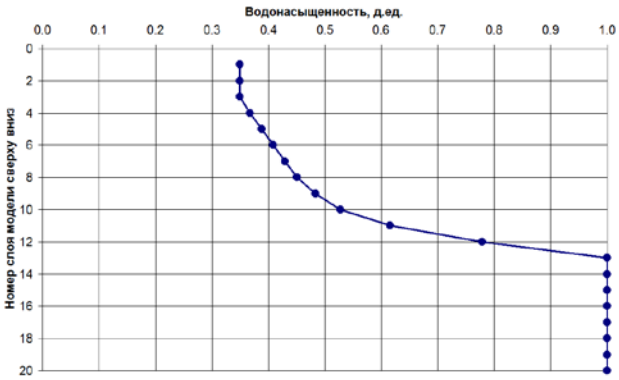


Рис. 6. Начальное распределение насыщенности по слоям

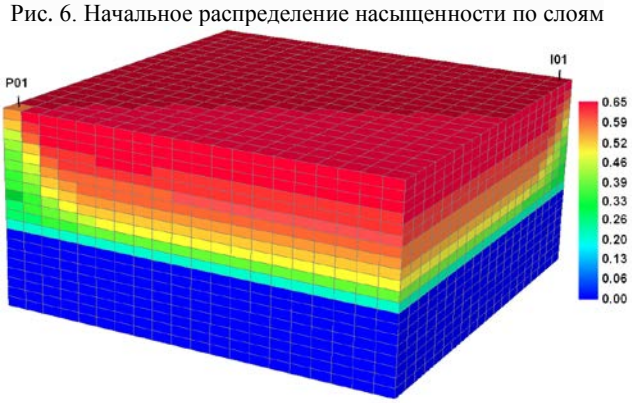


Рис. 7. Поле нефтенасыщенности через один месяц после начала работы скважин, P01 - добывающая скважина, I01 - нагнетательная скважина

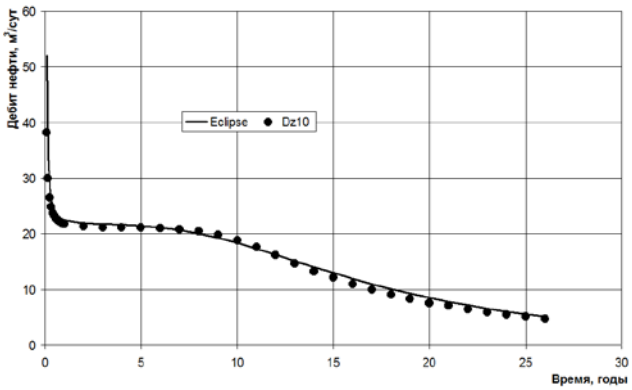


Рис. 8. Дебит нефти

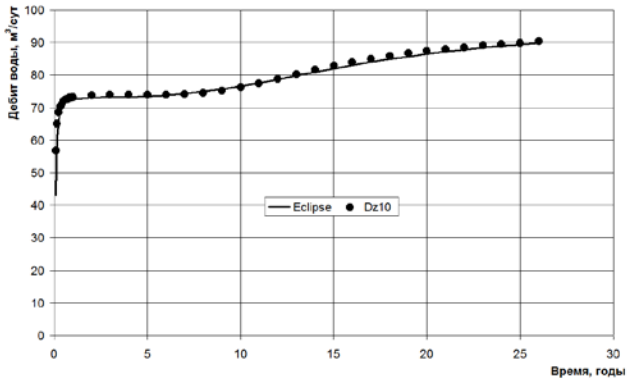


Рис. 9. Дебит воды

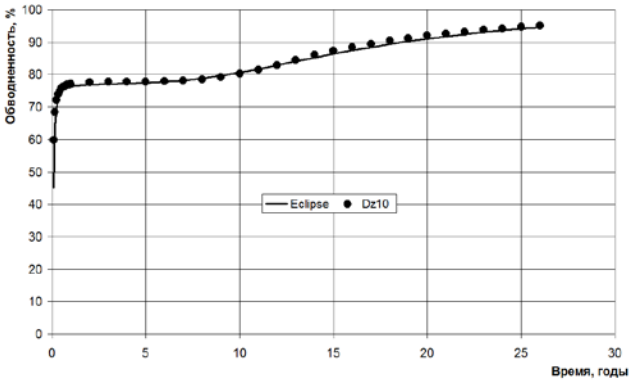


Рис. 10. Обводненность продукции

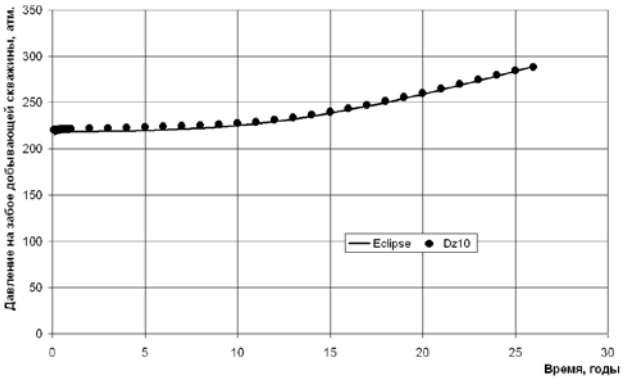


Рис. 11. Давление на забое добывающей скважины

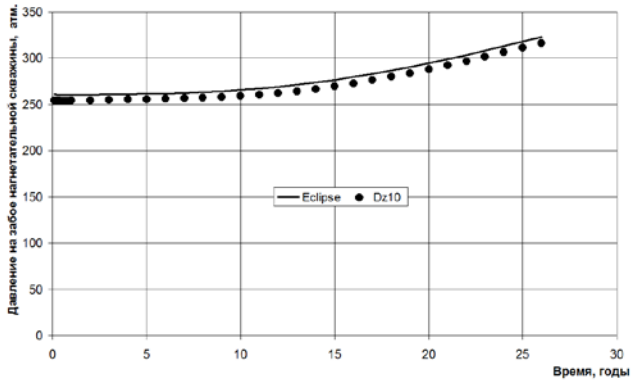


Рис. 12. Давление на забое нагнетательной скважины

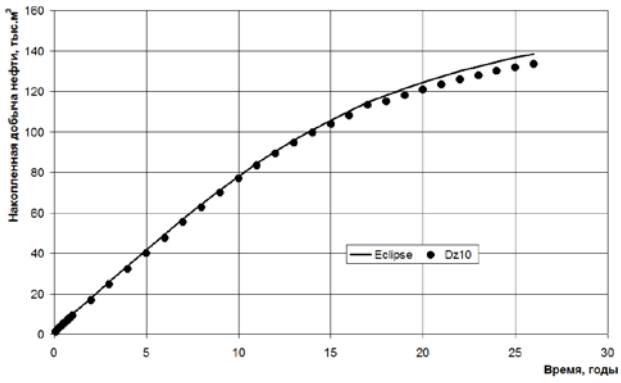


Рис. 13. Накопленная добыча нефти

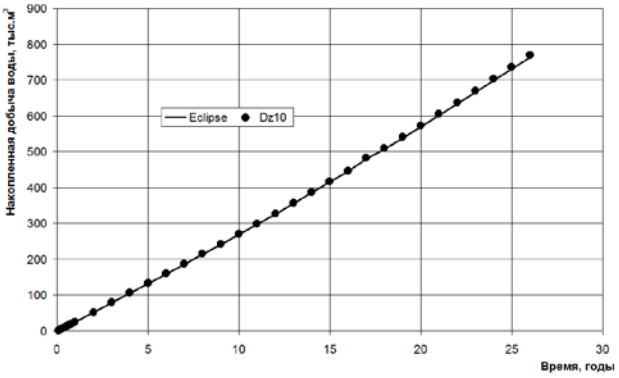


Рис. 14. Накопленная добыча воды

Следует отметить хорошее совпадение результатов расчетов с применением указанных программ. Однако есть и некоторые отличия. Основное отличие – это разные дебиты нефти и воды добывающей скважины за первые 1-2 месяца расчета, рис. 8-9.

Это различие связано с постановкой задачи и использованием различных разностных схем (явно-неявной в Dz10 и полностью неявной в Eclipse). Дело в том, что для тестирования математических моделей многофазной фильтрации, учитывающих гравитационные и капиллярные эффекты, удобно использовать задачи с образованием конуса подошвенной воды, как рассмотренный выше пример. Но при этом для решения небольших задач о конусообразовании, в которых большое количество жидкости, выраженное в единицах порового объема ячеек, может проходить через ячейки вблизи ствола скважины за один шаг по времени, лучше подходят полностью неявные методы. Использование явно-неявных методов для решения таких задач может приводить к уменьшению шага по времени до неприемлемо малых значений. Загрублением минимального шага по времени вызвано различие в дебитах нефти и воды добывающей скважины за первые 1-2 месяца расчета, рис. 8-9.

Заключение

Разработана математическая модель несмешивающейся двухфазной фильтрации слабосжимаемых жидкостей (нефти и воды). Программа, в которой реализована описанная модель, получила название Dz10, авторы Р.М. Кац и Е.В. Волгин.

Программа позволяет решать задачи моделирования разработки нефтяных месторождений на упругом режиме, естественном водонапорном режиме и при закачке воды.

Сравнительные расчеты на тестовой модели показали хорошее совпадение результатов расчетов с помощью программы Dz10 с результатами расчетов с помощью программы Eclipse Schlumberger.

Numerical simulation of two-phase flow of oil and water

R.M. Katz, E.R. Volgin, I.V. Afanaskin

Abstract: A mathematical model of two-phase immiscible filtration of slightly compressible fluids (oil and water). Describes the concept of the total flow rate. Provides a system of finite-difference equations for the model. The software realization of this model and the results of the testing program described.

Keywords: numerical simulation, two-phase flow, slightly compressible fluids.

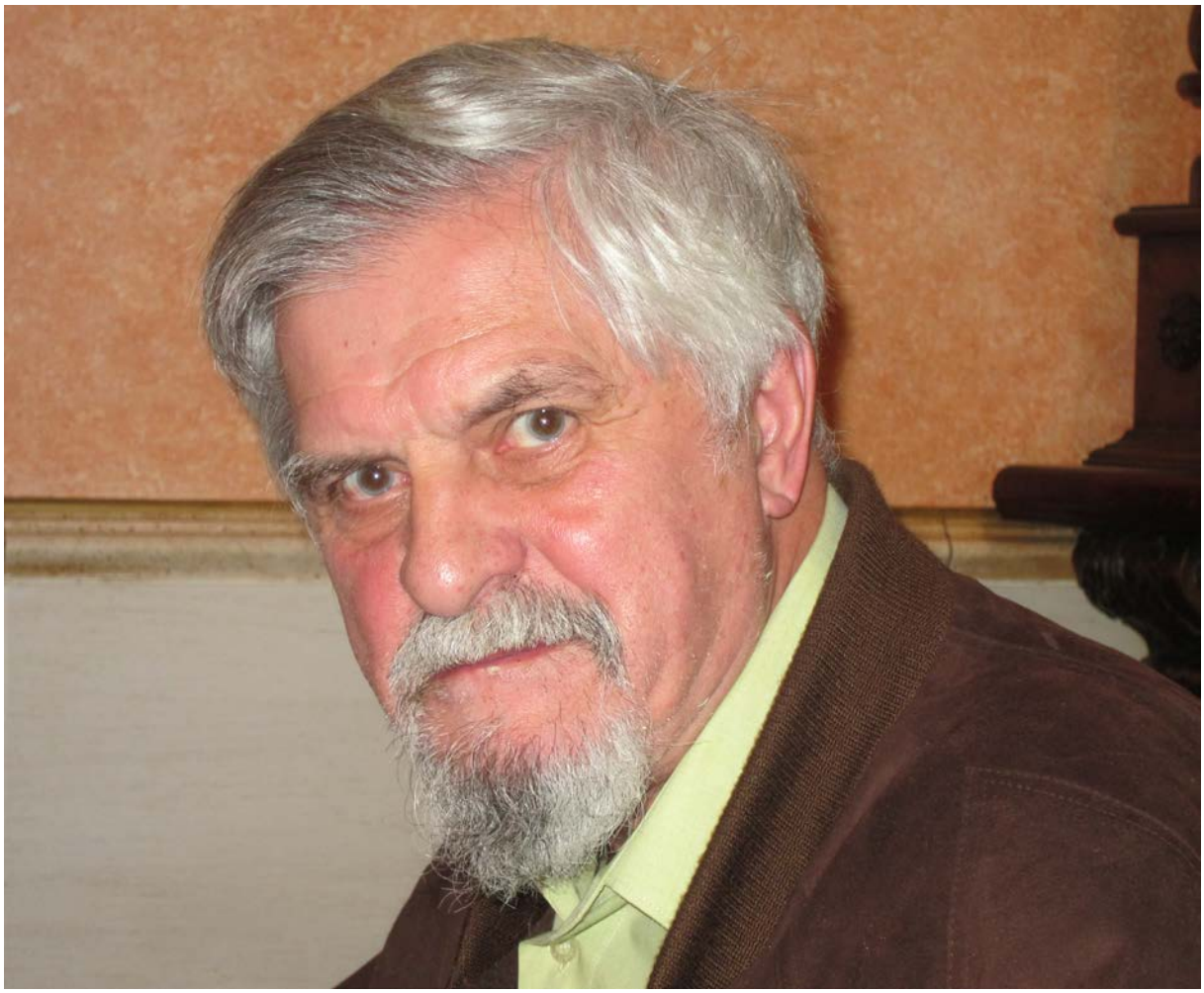
Литература

1. Х.Азиз, Э.Сеттари. Математическое моделирование пластовых систем. М.- Ижевск: Институт компьютерных исследований, НИЦ «Регулярная и хаотическая динамика», 2004.
2. П.В.Индельман, Р.М.Кац. Математическое моделирование процесса разработки нефтяного месторождения с помощью модели двухфазной фильтрации слабосжимаемых жидкостей // Сб. тр. ВНИИ, 1982, № 81, с. 57-62.
3. D.W.Peaceman. Interpretation of well-block pressures in numerical reservoir simulation with nonsquare grid blocks and anisotropic permeability. // SPE Journal, 1983, v. 23, № 3. pp. 531-543.
4. Eclipse Reservoir Simulation Software. Version 2011.2 Technical Description. Schlumberger. 2011, p. 1816.

К теоретическим основам вычислительной математики: о творчестве Владимира Михайловича Тихомирова

В.Б.Бетелин[1], Г.Г.Магарил-Ильяев[2], А.Г.Кушниренко[3] , В.Б.Демидович[3]

1 – академик РАН, 2 – доктор физико-математических наук, профессор, 3 – кандидат физико-математических наук



Аннотация: Статья посвящена творчеству замечательного учёного, специалиста по теории приближений, теории экстремальных задач, выпуклому анализа - областям, формирующим теоретическую основу современной вычислительной математики - профессора Московского государственного университета Владимира Михайловича Тихомирова по случаю его восьмидесятилетия.

22 ноября 2014-го года исполнилось 80 лет профессору механико-математического факультета Московского государственного университета имени М.В.Ломоносова Владимиру Михайловичу Тихомирову. Вся его творческая жизнь связана с математикой, которой он беззаветно предан на протяжении вот уже более 60-ти лет. Блестящий учёный и замечательный педагог, он и поныне неутомим в этой благородной деятельности.

I. Немного из биографии В.М.Тихомирова

Владимир Михайлович родился в Москве. Его родители - отец Михаил Никандрович Тихомиров (1906-1995) и мать Людмила Юльевна /урожд. Гурвиц/ (1910-1993) - были врачами, окончившими *санитарный* факультет Московского медицинского института. Брак распался в начале Великой Отечественной войны, и воспитанием мальчика занимались родители матери – Юлий Осипович Гурвиц (1882-1953) и Елизавета Фёдоровна /урожд. Банова/ (1882-1954).

Дедушка Владимира Михайловича - Юлий Осипович - был известным московским учителем математики. Его книга по преподаванию геометрии *Р.В.Гангнус, Ю.О.Гурвиц «Геометрия. Методическое пособие для высших педагогических учебных заведений и преподавателей средней школы: Часть 1 (Планеметрия), Часть 2 (Стереометрия)». Москва, Учпедгиз, 1934-1935*, написанная в соавторстве с другим математиком и педагогом-новатором - Рудольфом Вильгельмовичем Гангнусом (1883-1949), широко издавалась в СССР. А бабушка Владимира Михайловича - Елизавета Фёдоровна - образования не имела.

Школу Владимир Михайлович окончил в 1952 году с золотой медалью. На семейном совете было решено, что ему следует попытаться поступить на механико-математический факультет Московского государственного университета.Полагавшееся для медалистов собеседование Владимир Михайлович преодолел успешно, и он стал студентом Мехмата МГУ.

С первого курса Владимир Михайлович стал посещать спецсеминар Евгения Борисовича Дынкина (1924-2014), которого и поныне причисляет к своим учителям. Под его руководством на втором курсе Владимир Михайлович написал свою курсовую работу по спинорной алгебре. Но формулируемые Евгением Борисовичем задачи, в основном относящиеся к теории представлений групп, Владимира Михайловича не увлекли, и на четвёртом курсе он уже писал курсовую работу, связанную с теорией вероятностей, под руководством Юрия Михайловича Прохорова (1929-2013). А в конце 4-го курса, в апреле 1956-го года, в свои ученики Владимира Михайловича пригласил сам Андрей Николаевич Колмогоров (1903-1987). На беседе с Андреем Николаевичем, состоявшейся на даче Колмогорова в подмосковной *Комаровке*, Владимиру Михайловичу были предложены несколько задач, с которыми он довольно быстро справился. Так Владимир Михайлович навсегда влился в плеяду *колмогоровских учеников*.

В 1957-ом году Владимир Михайлович успешно окончил кафедру теории вероятностей Мехмата МГУ, причём его дипломная работа, объёмом свыше 100 страниц машинописного текста, была отмечена похвальной грамотой. В том же году он поступил в аспирантуру Мехмата МГУ.

Уже на первом году аспирантуры Владимир Михайлович опубликовал заметку в ДАН СССР. А чуть позже он стал готовить (совместно Андреем Николаевичем) большую статью по "эпсилон-энтропии" для Успехов математических наук, опубликованную на третьем году его обучения в аспирантуре.

В октябре 1960-го года Владимир Михайлович успешно защитил в Отделе прикладной математики МИ АН СССР имени В.А.Стеклова (впоследствии ставшем Институтом прикладной математики АН СССР, а ныне - Институтом прикладной математики имени М.В.Келдыша РАН) свою кандидатскую диссертацию "Поперечники множеств в функциональных пространствах". Оппонентами по диссертации были Израиль Моисеевич Гельфанд (1913-2009) и Константин Иванович Бабенко (1919-1987), о её содержании тепло отозвался Сергей Михайлович Никольский (1905-2012). В том же 1960-ом году Владимир Михайлович был

зачислен на работу на Мехмат МГУ - в Лабораторию кафедры теории вероятностей.

В 1961-ом году Владимир Михайлович уехал на работу в Воронежский государственный университет. Но в 1962-ом году он вернулся на Мехмат МГУ, где вскоре стал доцентом кафедры теории функций и функционального анализа (ТФФА). В 1963-ем году он открыл на Мехмате МГУ свой спецсеминар, посвятив его общим вопросам теории приближений. Довольно быстро этот спецсеминар "оброс" молодёжью. Среди его первых участников он упоминает Алексея Львовича Левина (р. 1944), Александра Давидовича Иоффе (р. 1938), Михаила Ароновича Ольшанецкого (р. 1938), Вольдемар-Берен-карда Константиновича Рогова (р. 1938).

В 1969-ом году, под влиянием Сергея Васильевича Фомина (1917-1975), а также по совету Владимира Михайловича Алексеева (1932-1980), Владимир Михайлович Тихомиров перешёл с кафедры ТФФА на кафедру общих проблем управления (ОПУ), созданную на Мехмате МГУ в апреле 1966-го года во главе с академиком АН СССР Вадимом Александровичем Трапезниковым (1905-1994), и реально руководимую в те годы С.В.Фоминым. С этим было связано и расширение тематики спецсеминара Владимира Михайловича, в круг интересов которого включились вопросы теории оптимального управления.

В 1971-ом году, на Мехмате МГУ, Владимир Михайлович успешно защитил свою докторскую диссертацию "Некоторые вопросы теории приближений", оппонентами по которой были Андрей Николаевич Колмогоров, Константин Иванович Бабенко и Сергей Борисович Стечкин (1920-1995). Через два года Владимир Михайлович стал профессором кафедры ОПУ, а с 1989-го по 2011-ый годы он заведовал этой кафедрой.

В целом творческая деятельность Владимира Михайловича развернулась необычайно широко. Им опубликовано около двухсот научных работ, в том числе два десятка книг, популярных не только в России, но и далеко за её пределами. Под его руководством воспитана целая плеяда талантливых учеников, из которых около пятидесяти защитили кандидатские диссертации, а свыше десятка - докторские диссертации. Педагогическое мастерство его служит ярким примером подготовки специалистов высочайшей квалификации. При этом особо подчеркнём доброжелательное стремление Владимира Михайловича поддержать собеседника (скажем, докладчика на спецсеминаре) за малейший замеченный его успех - ведь, особенно для молодых, так важно, чтобы его вовремя "окрылили" на дальнейшую деятельность.

Владимир Михайлович ведёт и большую научно-организационную работу. Он является почётным профессором МГУ, членом редколлегий ряда российских и международных математических журналов, членом правления Московского Математического общества. Долгое время он был руководителем секции математики Московского Дома Учёных. И всюду он трудится с полной отдачей, не жалея ни сил, ни времени.

Нельзя не отметить и про историко-математическую деятельность Владимира Михайловича. Подробные публикации про развитие различных областей математики (функционального анализа, теории экстремума и др.) в российских и зарубежных журналах, десятки "статей-персоналий" о математиках, регулярные выступления с популярными математическими докладами в Московском Доме Учёных - ко всему этому Владимир Михайлович относится исключительно ответственно и безотказно.

В последние годы Владимир Михайлович увлечён идеей создания "синтетического курса математики", пропагандирующего возможность взглянуть в процессе обучения на развитие магистральных математических понятий, "от детского сада до университета" в их естественном единстве. «Когда я поступил на Мехмат МГУ - рассказывал Владимир Михайлович, - то на первой же лекции нам, первокурсникам, было сказано: "Забудьте о

том, что вы изучали по математике в школе". Но потом я осознал, что этот призыв не правильный – нет ни "школьной математики", ни "университетской математики", а есть "единая" математика, которая лишь требует своего доступного изложения в зависимости от того, на каком возрастном уровне находится обучающийся человек».

Пожелаем Владимиру Михайловичу крепкого здоровья и успехов в реализации всех его задуманных планов.

II. Подробнее о научном творчестве В.М.Тихомирова.

Как уже отмечалось, основные научные интересы Владимира Михайловича Тихомирова связаны с теорией приближений, теорией экстремальных задач и выпуклым анализом. Его влияние на становление и/или развитие этих областей математики было неоспоримым. Монографии и учебники Владимира Михайловича по данным дисциплинам переведены на многие языки мира и давно стали настольными книгами для специалистов, работающих в этих направлениях. Скажем несколько слов о каждой из этих областей математики.

Теория приближений.

Принято считать, что теория приближений началась с классических работ Пафнутия Львовича Чебышёва (1821-1894) о наилучшей аппроксимации *индивидуальной функции* конечномерными подпространствами. Затем стали изучать задачи о наилучшем приближении уже *класса функций* теми или иными конечномерными подпространствами. В 1936 году А.Н.Колмогоров ввёл понятие *поперечника* - величину, которая характеризует наилучшее приближение данного класса функций всеми подпространствами *фиксированной размерности.* Тем не менее, вплоть до начала 1950-ых годов практически не было публикаций на эту тему. И в 1956-ом году А.Н.Колмогоров предложил Владимиру Михайловичу вернуться к тематике, связанной с поперечниками, а именно заняться исследованием ε*-энтропии* функциональных классов.

Первые итоги деятельности А.Н.Колмогорова и его учеников по энтропии функциональных классов подведены в статье в «Успехах математических Наук» (1959 г.), написанной Владимиром Михайловичем совместно с А.Н.Колмогоровым. Эта работа оказала значительное влияние на всю теорию аппроксимации в целом. В эти же годы Владимир Михайлович начинает активную деятельность, связанную с развитием новой темы в теории приближений - нахождение наилучших методов аппроксимации классов гладких и аналитических функций. Он вводит ряд величин, которые, наряду с *поперечником по Колмогорову*, характеризуют аппроксимативные возможности данного множества (*проекционный поперечник*, *линейный поперечник, поперечник по Гельфанду* и др.), разрабатывает новые методы исследования этих величин, которые впоследствии многократно использовались и обобщались в десятках работ, получает множество конкретных результатов *по точным* и *асимптотически точным* значениям различных поперечников и т.п. Определенный итог этого этапа своей деятельности Владимир Михайлович подвёл в монографии *В.М.Тихомиров «Некоторые вопросы теории приближений». М., Изд-во МГУ, 1976.*

В теории приближений, начиная с работ Сергея Натановича Бернштейна (1880-1958), подробно изучаются наилучшие аппроксимации отдельных функций и классов функций на прямой. С.Н.Бернштейн ввёл для этого некий аналог тригонометрических полиномов, а именно, так называемое пространство $B_\sigma(\square)$, $\sigma > 0$, представляющее собой сужение на прямую $\square$ целых функций, которые удовлетворяют определённым условиям роста. В последующие годы появились другие средства приближения классов функций на прямой, например, пространства *сплайнов* и

вейвлетов. Эти пространства, как и пространство $B_\sigma(\square)$, бесконечномерны, а стандартные классы функций на прямой (скажем, введенные Сергеем Львовичем Соболевым (1908-1989) так называемые *соболевские классы*), для приближения которых они используются - некомпактны. Как ставить вопрос об аппроксимативных характеристиках таких классов и как сравнивать различные средства их приближения? Впервые этот вопрос был поставлен американским инженером Клодом Шенноном (1916-2001), в связи с чем он дал определение «энтропии на единицу времени» случайного сигнала на прямой. А Андрей Николаевич Колмогоров модифицировал это определение для обычных (не случайных) функций.

Первый результат в этом направлении – энтропия на единицу времени ограниченных функций из $B_\sigma(\square)$ - был получен Владимиром Михайловичем в работе *В.М.Тихомиров «Об ε-энтропии некоторых классов аналитических функций». ДАН СССР, т. 117 (1957), № 2, 191-194.* Впоследствии Владимир Михайлович ввёл аппроксимационную характеристику класса, аналогичную колмогоровской, но отправляясь не от энтропии, а от поперечника по Колмогорову, которую он назвал *средней ε-размерностью*. Это послужило началом для многих исследований, связанных с вычислением средней ε-размерности различных некомпактных функциональных классов. В частности, отчетливые очертания эта тематика получила в работах Георгия Георгиевича Магарил-Ильяева (р. 1944), который ввёл понятия *средней размерности пространства*, а также *усредненных поперечников*, и нашёл их точные и асимптотически точные значения для ряда функциональных классов.

В шестидесятые годы прошлого века Владимиром Михайловичем был построен класс специальных функций, связанных с собственными функциями нелинейных *уравнений типа Штурма-Лиувилля* (введенных ещё в XIX столетии французскими математиками Жаком Шарлем Франсуа Штурмом (1803-1855) и Жозефом Лиувиллем (1809-1882)), позволившими найти оптимальные методы аппроксимации классов функций, задаваемых ядрами, не повышающими осцилляцию. Тем самым была решена проблема, которая занимала многих математиков и решения которой были получены лишь в отдельных частных случаях.

В те же шестидесятые годы была поставлена задача *об оптимальном восстановлении* значений линейного функционала на классе элементов по приближенной информации о самих элементах. Постановка этой задачи идеологически близка к работе Колмогорова о поперечниках классов функций, упомянутой выше. Владимир Михайлович, совместно с Георгием Георгиевичем Магарил-Ильяевым и Константином Юрьевичем Осипенко (р. 1950), начал исследование этой задачи с позиций теории экстремума. Выяснилось, что задачи оптимального восстановления естественным образом встраиваются в классическую теорию приближений. Точнее говоря, практически каждой задаче теории аппроксимации можно сопоставить задачу оптимального восстановления, которая придаёт исходной задаче вполне отчётливый информационный смысл. Последующее развитие этой тематики дало возможность решить ряд задач, имеющих явную прикладную направленность - оптимальное восстановление сигналов по неточной информации об их спектре и решений дифференциальных (а также разностных) уравнений по неточным исходным данным.

Теория экстремума.

Шестидесятые годы прошлого века стали началом активного развития общей теории *экстремальных задач* (основным стимулом для этого послужило получение Львом Семёновичем Понтрягиным (1908-1988) и его сотрудниками необходимых условий минимума в задаче оптимального

управления, которые были названы *принципом максимума Понтрягина*). Владимир Михайлович включился в эти исследования, принимая участие в работе семинара Алексея Алексеевича Милютина (1925-2001) по теории экстремума, что оказало на него значительное влияние. Вместе со свои учеником А.Д.Иоффе он начинает продумывать общие принципы теории экстремума, имея уже определённый опыт решения конкретных экстремальных задач теории приближений. Они приходят к выводу, что необходимые условия экстремума во всех экстремальных задачах, от *правила множителей Лагранжа* (сформулированному ещё в 1797-ом году французским математиком Жозефом Луи Лагранжем (1736-1813)) до *принципа максимума Понтрягина*, строятся по единому рецепту, который они назвали *принципом Лагранжа*. Понимание только этого принципа, без какой-либо апелляции к теории, достаточно для решения многих экстремальных задач (что согласуется с известным высказыванием французского философа Клода-Адриана Гельвеция (1715-1771): «Знание некоторых принципов вполне компенсирует незнание многих фактов»).

Систематическое изложение необходимых условий экстремума для различных экстремальных задач с точки зрения принципа Лагранжа приведено в двух монографиях: *А.Д.Иоффе, В.М.Тихомиров «Теория экстремальных задач». М., «Наука», 1974* и *В.М.Алексеев, В.М.Тихомиров, С.В.Фомин «Оптимальное управление». М.,»Наука», 1979.* В совместной работе Василия Борисовича Демидовича (р. 1943) с Владимиром Михайловичем Тихомировым и Георгием Георгиевичем Магарил-Ильяевым - *В.Б.Демидович, Г.Г.Магарил-Ильяев, В.М.Тихомиров «Экстремальные задачи для линейных функционалов на чебышёвских пространствах». Фундаментальная и прикладная математика, т. 11 (2005), № 2, 87-100* – в частности, исследованы некоторые экстремальные задачи для *обобщённых полиномов* (из, так называемых, *чебышёвских* пространств).

Выпуклый анализ.

Работы Владимира Михайловича по *выпуклому анализу* связаны как собственно с развитием этой дисциплины, так и с приложениями её к задачам теории приближений и теории экстремума. Но важно и то, что под влиянием этих работ сформировался определенный взгляд на предмет выпуклого анализа, который оказался весьма плодотворным, особенно для приложений. Суть его состоит в том, что основное содержание выпуклого анализа - это *соотношения двойственности* для некоторого набора операторов и порожденное ими *выпуклое исчисление*. Можно привести множество утверждений из анализа, теории приближений и теории экстремума которые получаются как следствия соотношений двойственности и формул выпуклого исчисления для тех или иных операторов, переводящих выпуклые объекты (выпуклые множества и выпуклые функции) в себя.

В монографиях:

В.М.Тихомиров «Выпуклый анализ». Сб. «Итоги науки и техники, серия «Современные проблемы математики: Фундаментальные направления»». М., ВИНИТИ, 1987;

G.G.Magaril-Il'yaev, V.M.Tikhomirov «Convex Analysis: Theory and Applications». AMS, Translations of Mathematical Monographs, vol. 222, 2003;

Г.Г.Магарил-Ильяев, В.М.Тихомиров «Выпуклый анализ и его приложения». М., УРСС, 2011 (3 изд.)

а также в статье Владимира Михайловича и его голландского коллеги Яна Бринхауса (р. 1952):

Я. Бринхаус, В.М.Тихомиров «Двойственность и исчисление выпуклых объектов (теория и приложения)». Математический сборник, т. 198 (2007), № 2, 29–66, отражены основные воззрения авторов на предмет выпуклого анализа и на его взаимосвязь с анализом, геометрией, теорией экстремума и теорией приближений.

III. Несколько слов о знакомстве авторов с В.М.Тихомировым.

(В.Б.Бетелин – А.Г.Кушниренко) Наше знакомство с Владимиром Михайловичем произошло в разные годы.

Я – Анатолий Георгиевич Кушниренко (р. 1944) - Владимиру Михайловичу был *представлен* Владимиром Игоревичем Арнольдом (1937-2010) во время летнего воскресного отдыха на Рублёвском водохранилище в 1962-ом году. Арнольд представил меня (тогда ещё первокурсника, формально не являвшегося его учеником) как уже своего студента, получившего от него задачу.

А я – Владимир Борисович Бетелин (р. 1946) – познакомился с Владимиром Михайловичем, лишь в середине 1970-ых годов, когда стал сотрудником существовавшей тогда при кафедре ОПУ Лаборатории по проблемам управления, возглавляемой в ту пору Виктором Яковлевичем Шкадовым (р. 1935). В связи с этим напомню, что в 1979-ом году этой Лабораторией стал руководить Александр Васильевич Михалёв (р. 1940), а с 1981-года она была преобразована в Лабораторию вычислительных методов при кафедре «Вычислительная математика», воссозданной на Мехмате МГУ во главе с Николаем Сергеевичем Бахваловым (1934-2005).

Владимир Михайлович сам не занимался ни вычислительной математикой, ни программированием, но, в отличие от многих математиков Мехмата МГУ, считал, что освоение основ программирования и численных методов является важным элементом общего математического образования. Поэтому достаточно быстро (помнится, уже в 1975-ом году) при активной его поддержке, на кафедре ОПУ мы организовали семинар «Системное программирование», где стали выполняться курсовые и дипломные работы, а потом и кандидатские диссертации. Вся эта наша деятельность сопровождалась *внедрением* учебных систем программирования на Мехмате МГУ (и в других ВУЗах страны), причём мы хотим подчеркнуть, что особенно эффективно это внедрение происходило после назначения заведующим Лабораторией Александра Васильевича Михалёва.

В заключение добавим следующее.

Я - А.Г.Кушниренко - являясь сотрудником кафедры ОПУ, принимал также непосредственное участие в реализации важнейшего замысла, осуществляемого Владимиром Михайловичем (совместно с Сергеем Васильевичем Фоминым и Владимиром Михайловичем Алексеевым) – постановке преподавания по обязательному курсу «Вариационное исчисление и оптимальное управление» на Мехмате МГУ. В частности, Владимир Михайлович привлёк меня (а также Эльфата Михайловича Галеева (р. 1952)) к совместному написанию первого задачника по этой дисциплине: *Э.М.Галеев, А.Г.Кушниренко, В.М.Тихомиров «Сборник задач по оптимальному управлению». М., «Изд –во Моск. ун-та», 1980.*

В свою очередь, я - В.Б.Бетелин - специализируясь на Мехмате МГУ «как прикладник», не посещал «чисто математические» доклады, в частности, не присутствовал и на докладах Владимира Михайловича. Но один доклад Владимира Михайловича «на математические темы» я, всё-таки, посетил, и он произвёл на меня глубокое впечатление. Это был его доклад, если я не ошибаюсь, 1983-го года на Московском математическом обществе под названием «Механико-математический факультет МГУ как явление русской культуры». Воспоминания о нём у меня сохранились до сих пор ...

(Г.Г.Магарил-Ильяев) С Владимиром Михайловичем я познакомился в 1969-ом году. В это время я работал в Центральном научно-исследовательском институте комплексной автоматизации (ЦНИИКА), по распределению, после окончания Московского института нефтехимической и

газовой промышленности имени И.М.Губкина (МИНХ и ГП).

Каким-то образом (не помню, откуда) возникла задача об оптимальном управлении некой дискретной линейной системой. Я нашёл, как мне казалось, её решение и не знал с кем это можно обсудить. Мой старший товарищ по работе в ЦНИИКА, Борис Семёнович Дарховский (р. 1938), окончивший к этому времени инженерный поток на Мехмате МГУ, посоветовал мне обратиться к Владимиру Михайловичу Тихомирову. Я нашёл Владимира Михайловича, объяснил ситуацию и отдал ему текст. Он похвалил меня, сказав, что это наиболее правильный способ общения, и обещал через некоторое время всё посмотреть.

В те же времена мои близкие друзья по МИНХ и ГП независимо познакомились с Владимиром Михайловичем и его женой Наталией Ильиничной, причиной чему было совместное гуляние их четырехлетних дочерей. Это дало возможность мне несколько ближе познакомиться с Владимиром Михайловичем. Я сказал ему, что хочу заниматься математикой под его руководством. Он дал мне некоторое количество задач, и после того, как я их решил, согласился со мной «*возиться*» (что касается моей рукописи, то он её потерял сразу). И продолжается это вот уже 45 лет.

(В.Б.Демидович) Моему знакомству с Владимиром Михайловичем исполнилось более полувека – в 1963/64 учебном году он вёл в 409-ой группе Мехмата МГУ, где я учился, семинарские занятия по ТФКП. Уже тогда мне запомнилась его внимательное отношение к студентам: сформулировав нам очередную задачу, он подходил буквального к каждому из нас, одобряя выбор *разумного* (не обязательно стандартного) пути её решения, или же выражая скепсис, если путь решения представлялся ему *сомнительным*.

В 1970-ом году я стал сотрудником кафедры ОПУ Мехмата МГУ. Штатными сотрудниками кафедры тогда были всего пять человек: Сергей Васильевич Фомин, Владимир Михайлович Алексеев, Владимир Михайлович Тихомиров, Анатолий Георгиевич Кушниренко и Герман Юрьевич Данков (1935-2012). Заведующий нашей кафедрой, Вадим Александрович Трапезников, являясь директором Института проблем управления АН СССР, был внештатным сотрудником кафедры ОПУ, и на кафедру практически не приезжал. В этих обстоятельствах реальное руководство кафедрой ОПУ осуществляли С.В.Фомин с Учёным секретарём кафедры Г.Ю.Данковым. С моим появлением на кафедре ОПУ Г.Ю.Данков охотно «скинул на меня» свои «учёно-секретарские» обязанности. Но я об этом не жалею: время работы с С.В.Фоминым «у руля кафедры» я всегда вспоминаю с благодарностью.

После смерти С.В.Фомина я попросил освободить меня от обязанностей кафедрального Учёного секретаря, и эта должность перешла к А.Г.Кушниренко. Потом эту должность занимали, «ротационным образом», практически все зачисляемые на кафедру ОПУ молодые сотрудники.

В декабре 1986-гогода кафедра ОПУ, совместно с кафедрой вычислительной математики, провели в подмосковном пансионате «Зимёнки» выездную межвузовскую школу-семинар «Теория приближений и задачи вычислительной математики». Во главе Оргкомитета школы-семинара были Константин Иванович Бабенко, Николай Сергеевич Бахвалов и Владимир Михайлович Тихомиров, а я был назначен Ответственным секретарём Оргкомитета. К тому времени я уже «отошёл» от разностных уравнений, всерьёз заинтересовавшись возможностями применения чебышёвских систем в численном анализе. Эта тематика была близка Владимиру Михайловичу, и, по возвращению в Москву, я попросился работать в его семинаре. Так с весны 1987-го года я встал «под крыло» Владимира Михайловича, и, тем самым, считаю его одним из своих Учителей.

В 1989-ом году, ещё при жизни

В.А.Трапезникова, Владимир Михайлович стал заведующим кафедрой ОПУ. К тому времени кафедра расширилась до полутора десятка штатных сотрудников и нескольких «полставочников-совместителей». В 1994-ом году он назначил меня заместителем заведующего кафедрой ОПУ, коим я являюсь до сих пор.

И ещё. В преддверии 250-летия Московского государственного университета (2005-ый год) каждой кафедре Мехмата МГУ было поручено написать статью *об её прошлом и настоящем*. На кафедре ОПУ этим занялись мы с Владимиром Михайловичем. И в результате нашей совместной работы я увлёкся историко-математической тематикой (отсюда потом возникла моя серия «Мехматяне вспоминают», в выпусках которой печатаются интервью со старейшими сотрудниками Мехмата МГУ). В этом плане я также считаю себя *учеником* Владимира Михайловича, который всегда поддерживал мою историко-математическую деятельность.

On the theoretical foundations of Computational Mathematics: about the work of Vladimir Mikhailovich Tikhomirov

V.B.Betelin, G.G.Magaril-Il'yaev, A.G.Kushnirenko, V.B. Demidovich

Abstract: This article is devoted to the work of Professor of the Moscow State University Vladimir Mikhailovich Tikhomirov on his eightieth birthday - of remarkable scientist, specialist on Approximation Theory, on Theory of Extremal Problems and on Convex Analysis that form the theoretical basis of modern Computational Mathematics.