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

ТРУДЫ НИИСИ РАН

ТОМ 8 № 4

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ СЛОЖНЫХ СИСТЕМ:

ТЕОРЕТИЧЕСКИЕ И ПРИКЛАДНЫЕ АСПЕКТЫ

МОСКВА
2018

Тематика номера:
Моделирование физических процессов, математическое моделирование
и визуализация, вопросы программирования.

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

The topic of the issue:
Modeling in physical processes. mathematical modeling
and visualization, problems in programming.

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

Заведующий редакцией: Ю.Н.Штейников

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

СОДЕРЖАНИЕ

I. МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ

II. МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ВИЗУАЛИЗАЦИЯ

III. ВОПРОСЫ ПРОГРАММИРОВАНИЯ

Гауссова аппроксимация спектральной плотности для 1D модели Изинга

Л.Б. Литинский[1], И.М. Каганова[2]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1] litin@mail.ru, [2] imkaganova@gmail.com

Аннотация: Изучена плотность состояний для 1D модели Изинга. Установлено, что гауссова аппроксимация спектральной плотности делает метод n-окрестностей непригодным для описания 1D модели Изинга. Найдено точное комбинаторное выражение для распределения энергий в n-окрестности. На этой основе получено распределение намагниченности для произвольной температуры.

Ключевые слова: плотность состояний, 1D модель Изинга, метод n-окрестностей.

1. Введение

Метод n-окрестностей может стать универсальным методом вычисления свободной энергии термодинамической системы. Основная идея метода состоит в замене неизвестной нам точной спектральной плотности спиновой системы гауссовым распределением с известными средним значением и дисперсией. Для модели Изинга на кубических решетках больших размерностей - $d \geq 3$ - метод n-окрестностей позволил получить аналитическое выражение для критической температуры, которым хорошо описываются результаты компьютерного эксперимента [1] (точных результатов для решеток таких размерностей нет, поэтому мы ориентировались на общепризнанные результаты компьютерных экспериментов). Вычисленные значения критических показателей также находятся в согласии с компьютерным экспериментом [2].

В то же время, для модели Изинга на кубических решетках малых размерностей $d = 1$ и $d = 2$, хорошо изученных теоретически [3], метод n-окрестностей оказывается либо вовсе неприменим ($d = 1$), либо – как для $d = 2$ - неверно предсказывает тип фазового перехода. Актуальными являются попытки усовершенствовать метод n-окрестностей и расширить область его применимости.

В настоящей работе анализируется одномерная модель Изинга ($d = 1$), для которой удалось получить точное комбинаторное выражение для спектральной плотности состояний из n-окрестности. Получено аналитическое выражение для частоты каждого значения энергии (кратность вырождения энергии) - см. раздел 2. Это позволило непосредственно сравнить точную спектральную плотность и ее аппроксимацию гауссовым распределением. Оказалось, что точная спектральная плотность хорошо аппроксимируется в районе точки максимума, и гораздо хуже - на границах интервала. Последнее обстоятельство делает невозможным использование метода n-окрестностей в его нынешней «гауссовой» формулировке для описания 1D модели Изинга (раздел 3). Кроме того, с помощью точного выражения для спектральной плотности получено распределение намагниченности при различных значениях обратной температуры β - см. раздел 4. Оказалось, что с ростом β распределение намагниченности выпола-живается – стремится к равновероятному распределению. Этот неожиданный результат интерпретирован в содержательных физических терминах. Технические вопросы вынесены в Приложения.

2. Точная спектральная плотность

1. Суть метода n-окрестностей вычисления свободной энергии заключается в следующем [1,4,5]. Зафиксируем начальную конфигурацию $\mathbf{s}_0 \in \mathbf{R}^N$, а остальные конфигурации распределим по ее n-окрестностям Ω_n, объединяя в Ω_n конфигурации, отличающиеся от $\mathbf{s}_0$ противоположными значениями n спинов:

$$\Omega_n = \{\mathbf{s} : (\mathbf{s}, \mathbf{s}_0) = N - 2n\}, 0 \leq n \leq N .$$

Обозначим истинное распределение энергий состояний из Ω_n через $D_n(E)$. Как правило $D_n(E)$ неизвестно, однако можно получить точные выражения для среднего значения E_n и дисперсии σ_n^2 этого распределения, выразив их через характеристики матрицы связи и начальной конфигурации $\mathbf{s}_0$ [5].

Основная идея метода n-окрестностей состоит в замене неизвестного распределения $D_n(E)$ гауссовым распределением со средним значением E_n и дисперсией σ_n^2:

$$G_n(E) \sim \exp\left(-\frac{1}{2}\left(\frac{E-E_n}{\sigma_n}\right)^2\right).$$

Обоснование метода n-окрестностей дано в [5], границы его применимости для модели Изинга исследованы в [1].

2. Для одномерной цепочки спинов с циклическими граничными условиями энергия состояния имеет вид [3]:

$$\begin{aligned} E(\mathbf{s},H) = & -\frac{(\mathbf{Js},\mathbf{s})}{2N} - H\frac{\sum_{i=1}^{N} s_i}{N} = \\ & = E(\mathbf{s}) - H\cdot\left(1-2\frac{n}{N}\right), \end{aligned} \qquad (1)$$

где H - величина внешнего магнитного поля, $\mathbf{J}$ - матрица связи одномерной модели Изинга (А1).

В качестве начальной конфигурации возьмем основное состояние спиновой системы:

$$\mathbf{s}_0 = (1,1,\ldots 1) \in R^N \Rightarrow E(\mathbf{s}_0) = E_0 = -1.$$

В Приложении 1 показано (см. (А5), (А6)), что для состояний из n-окрестности энергия $E(\mathbf{s})$ принимает в точности n различных значений $E(k)$, кратность вырождения которых $D(n,k)$ имеет вид:

$$\begin{aligned} E(k) &= -\left(1-4\frac{k}{N}\right), k = 1,2,..,n \\ D(n,k) &= \frac{N\cdot k}{(N-n)n} C_{N-n}^{k} C_{n}^{k}. \end{aligned}, \qquad (2)$$

Номер окрестности n пробегает значения от 1 до $N/2$.

Для одномерной модели Изинга выражения (2) полностью определяют спектральную плотность для состояний из n-окрестности.

Нам потребуется непрерывный аналог этих выражений в пределе $N \to \infty$. Введем два числовых параметра: $x = n/N$ и $y = k/N$, и преобразуем формулы (2) к непрерывному виду:

$$\begin{aligned} E(y) &= -(1-4y), \\ D(x,y) &= \frac{\mathrm{e}^{-N\cdot\left(xS\left(\frac{y}{x}\right)+(1-x)S\left(\frac{y}{1-x}\right)\right)}}{2\pi N\sqrt{x(1-x)(x-y)(1-x-y)}}, \\ y &= \frac{k}{N} \in [0,x],\ x = \frac{n}{N} \in [0,1/2]. \end{aligned} \qquad (3)$$

Здесь $S(a) = a\ln a + (1-a)\ln(1-a)$ - функция Шеннона.

Параметр y индексирует различные значения энергии, его значения принадлежат интервалу $y \in [0,x]$.

Параметр x индексирует различные значения намагниченности $m = 1-2x$, и меняется в интервале $x \in [0,1/2]$.

Нам потребуются асимптотические выражения для среднего (по n-окрестности) значения энергии и дисперсии [1]:

$$\begin{aligned} E_x &= \lim_{N\to\infty} E_n = -(1-2x)^2, \\ \sigma_x^2 &= \lim_{N\to\infty} \sigma_n^2 = 16x^2(1-x)^2/N \end{aligned}.$$

Тогда асимптотическое выражение для гауссовой аппроксимации истинной спектральной плотности принимает вид:

$$\begin{aligned} G(x,y) = & \frac{\mathrm{e}^{-\frac{1}{2}\left(\frac{E-E_x}{\sigma_x}\right)^2}}{\sqrt{2\pi\sigma_x}} = \\ & = \frac{\mathrm{e}^{-N\left(S(x)+\frac{1}{2}\left(1-\frac{y}{x(1-x)}\right)^2\right)}}{2\pi N\left(x(1-x)\right)^{3/2}} \end{aligned}. \qquad (4)$$

Нормировки в (3), (4) выбраны так, что $\iint D(x,y)dxdy = \iint G(x,y)dxdy = 2^N$.

3. Сравнение $D(x,y)$ и $G(x,y)$

Сделаем два замечания. Мы будем сравнивать распределения $D(x,y)$ и $G(x,y)$ при различных значениях параметра x. Переменная y индексирует значения энергии: $y=(1+E)/4$ - см. (3). Иными словами, мы будем сравнивать две плотности распределения энергий – точную и ее гауссову аппроксимацию, однако удобнее при этом пользоваться переменной y, а не E. Кроме того, сравнивать будем не сами функции $D(x,y)$ и $G(x,y)$, а их показатели экспоненты, деленные на N:

$$d(x,y)=-xS\left(\frac{y}{x}\right)-(1-x)S\left(\frac{y}{1-x}\right),\qquad g(x,y)=-S(x)-\frac{1}{2}\left(1-\frac{y}{x(1-x)}\right)^2, \tag{5}$$

где $x\in[0,1/2]$, $y\in[0,x]$. Влиянием предэкспонент можно пренебречь.

1. Графики функций $d(x,y)$ при различных значениях параметра x показаны на верхней панели рис.1.

Каждая кривая достигает максимума в точке $y_M=x(1-x)$, после чего монотонно убывает до значения $d(x,x)=-(1-x)S\left(\frac{x}{1-x}\right)$ в граничной точке $y=x$. Функция $d(x,y)$, отвечающая значению параметра $x=0.5$, есть функция Шеннона, взятая с обратным знаком: $d(0.5,y)=-S(y)$. Она симметрична относительно точки максимума $y_M=0.25$.

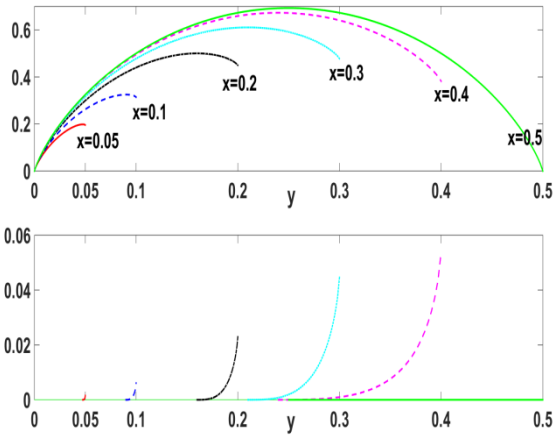
Можно было бы ожидать, что и любая функция $d(x,y)$ будет симметричной относительно своей точки максимума y_M. Иначе говоря, что разность значений функции $d(x,y)$ в точках, симметрично расположенных относительно y_M, будет тождественно равна нулю:

$$\delta(x,y)=d(x,y)-d(x,2y_M-y)\overset{?}{\equiv}0, \tag{6}$$

когда $y\in[y_M,x]$

Это, однако, не так – на нижней панели рис.1 для тех же значений параметра x показаны графики разностей $\delta(x,y)$ (6). Все функции $d(x,y)$ - несимметричны, кроме той, что отвечает значению $x=0.5$. Отсутствие симметрии у функций обусловлено наличием множителей x и $1-x$, на которые в выражении для $d(x,y)$ умножаются симметричные функции $S(y/x)$ и $S(y/1-x)$.

Рис.1. Верхняя панель - графики функций



$d(x,y)$ (5) для различных значений x, нижняя панель – графики разностей $\delta(x,y)$ (6).

Наконец рассмотрим как ведут себя функции $d(x,y)$ на краях интервала $y\in[0,x]$. Нетрудно получить следующую таблицу значений:

$$\begin{aligned} &d(x,0)=0, \quad d(x,x)=-(1-x)\times \\ &\qquad\qquad \times S\left(x/(1-x)\right), \\ &d'_y(x,0)=\infty, \quad d'_y(x,x)=-\infty, \\ &d''_{yy}(x,0)=-\infty, \; d''_{yy}(x,x)=-\infty. \end{aligned} \tag{7}$$

Любая функция $d(x,y)$ начинается на левом конце интервала из 0, ее первая производная стремится здесь к $+\infty$, а вторая производная – к $-\infty$. На правом конце интервала значение функции $d(x,y)$ явным образом зависит от x, а значения ее производных от x не зависят. Такое поведение кардинально отличается от поведения аппроксимирующей $d(x,y)$ функции $g(x,y)$, к рассмотрению которой мы переходим.

2. На рис.2 для шести значений параметра $x\in[0.05,0.5]$ показаны графики функций $d(x,y)$ и $g(x,y)$. Видно – это легко усмотреть и из формул (5), - что при

любом значении x функция $g(x,y)$ достигает максимума в той же точке $y_M = x(1-x)$, что и $d(x,y)$, а их значения здесь совпадают: $d(x,y_M) = g(x,y_M) = -S(x)$. Это является следствием нашего подхода к аппроксимации спектральной плотности в n-окрестности, когда горб реального распределения аппроксимируется гауссовым колоколом. В небольшой окрестности точки максимума y_M гауссова кривая хорошо приближает истинную кривую $d(x,y)$.

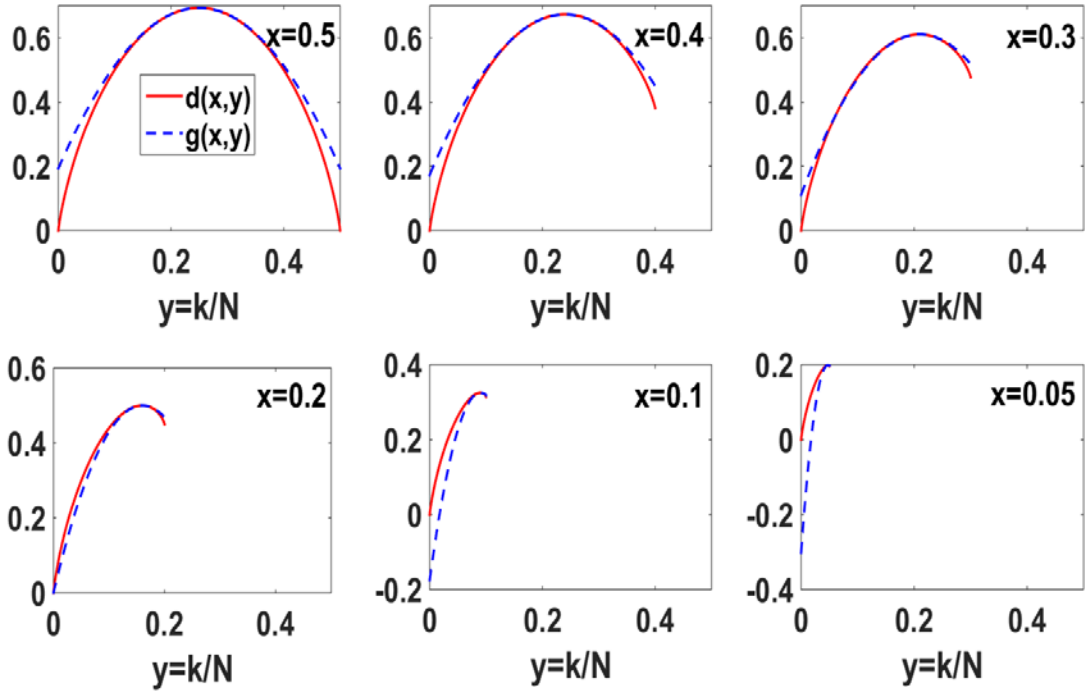


Рис. 2. Графики точных функций $d(x,y)$ (сплошная линия) и их гауссовой аппроксимации $g(x,y)$ (штриховая линия) (5) для значений параметра $x = 0.5, 0.4, 0.3, 0.2, 0.1, 0.05$.

Из общих соображений понятно, что $g(x,y)$ как функция y симметрична относительно точки максимума y_M - иначе говоря, разность значений функции $g(x,y)$ в точках, симметричных относительно точки максимума y_M, будет тождественно равна нулю:

$$g(x, y_M + \eta) - g(x, y_M - \eta) \equiv 0 \quad \forall \eta \in [0, x^2].$$

Это – одно из заметных отличий $g(x,y)$ от $d(x,y)$.

Существенно различается также поведение функций $g(x,y)$ и $d(x,y)$ на краях интервала изменения y, в районе точек $y = 0$ и $y = x$. Это легко усмотреть как из графиков рис.2, так и из таблицы граничных значений функции $g(x,y)$, которая аналогична таблице (7):

$$g(x,0) = -S(x) - 0.5, \qquad g(x,x) = -S(x) - 0.5 \cdot \left(x/(1-x)\right)^2,$$

$$g'_y(x,0) = \frac{1}{x(1-x)}, \qquad g'_y(x,x) = -\frac{1}{(1-x)^2},$$

$$g''_{yy}(x,0) = -\frac{1}{(x(1-x))^2}, g''_{yy}(x,x) = -\frac{1}{(x(1-x))^2}.$$

Значения функции $g(x,y)$ и ее производных по y на краях интервала явным образом зависят от x, и для $x \neq 0$ ни одна производная не обращается на краях интервала в бесконечность - ср. с (7). Поясним, почему такое значение придается поведению функций $g(x,y)$ и $d(x,y)$ в граничных точках интервала изменения y.

3. Функцию $D(x,y)$ (3) правильно было бы называть *частичной спектральной плотностью* (чсп), поскольку она описывает распределение энергий только для части всех состояний, а именно: для состояний, принадлежащих n-окрестности начальной конфигурации $\mathbf{s}_0$. (Возможно, правильнее было бы говорить об x-окрестности, но мы будем пользоваться исходным термином – n-окрестность). Просуммировав $D(x,y)$ по всем n-окрестностям, получим *полную спектральную плотность* (псп): $D(y)=\int D(x,y)dx$, которая описывает распределение энергий для всего множества 2^N конфигураций. Запишем псп $D(y)$ в виде: $D(y)=\exp(N\cdot d(y))$. То же самое проделаем для гауссовой аппроксимации $G(x,y)$, и аналогично представим гауссову псп в виде $G(y)=\int G(x,y)dx=\exp(N\cdot g(y))$. На языке псп наглядно видно, почему для 1D модели Изинга аппроксимация чсп $D(x,y)$ гауссовым распределением $G(x,y)$ не работает.

Наметим основные моменты получения псп $D(y)$ интегрированием $D(x,y)$ по x (подробности см. в Приложении 2):

$$D(y)=\int_{y}^{1-y} D(x,y)dx \sim \int \mathrm{e}^{N\cdot d(x,y)}\,dx \sim \sim \mathrm{e}^{N\cdot \max\limits_{x} d(x,y)} = \mathrm{e}^{N\cdot d(0.5,y)} \quad (8)$$

Последнее равенство в (8) обусловлено тем, что при любом значении y максимум функции $d(x,y)$ по x достигается при $x=0.5$ (см. Приложение 2). Иными словами, показатель экспоненты псп $D(y)$ совпадает с показателем экспоненты чсп $D(0.5,y)$:

$$d(y)=d(0.5,y)=-S\left(2y\right). \quad (9)$$

Аналогично показывается, что

$$G(y)=\int_{y}^{1-y} G(x,y)dx \sim \ldots = \exp\left(N\cdot g(0.5,y)\right)$$ -

то есть, что экспоненциальная часть псп $G(y)$ совпадает с экспоненциальной частью чсп $G(0.5,y)$:

$$g(y)=g(0.5,y)=\ln 2-\frac{(1-4y)^2}{2}. \quad (10)$$

В работе [6] получены следующие необходимые условия того, чтобы функция могла служить показателем экспоненты в экспоненциальном представлении полной спектральной плотности: пусть E_0 - энергия основного состояния, и псп представлена в виде $F(E)=\exp\left(N\cdot\Psi(E)\right)$, где $E\in[E_0,|E_0|]$. Тогда для функции $\Psi(E)$ из показателя экспоненты должно выполняться:

- $\lim\limits_{E\to E_0}\Psi(E)=0$ - в противном случае основное состояние будет многократно вырожденным (чего для модели Изинга быть не может);
- $\lim\limits_{E\to E_0}\Psi'(E)=\infty$ - в противном случае система окажется в основном состоянии при конечной температуре (что противоречит основным физическим принципам).

Для 1D модели Изинга имеем: $E_0=-1$, и $y=(1+E)/4\to 0$ когда $E\to E_0$. В терминах параметра y вышесказанное означает: необходимым условием того, что функция $\psi(y)$ есть показатель экспоненты псп, является выполнение условий: $\lim\limits_{y\to 0}\psi(y)=0$, $\lim\limits_{y\to 0}\psi'(y)=\infty$. Естественно, функция $d(y)$ (9), являющаяся показателем экспоненты точной псп, этим условиям удовлетворяет – в этом легко убедиться непосредственным вычислением. Напротив, функция $g(y)$ (10) не удовлетворяет ни одному из них – в этом также легко убедиться непосредственно.

Для 1D модели Изинга именно использование гауссовой плотности распределения для аппроксимации в n-окрестности истинной чсп $D(x,y)$ делает метод n-окрестностей непригодным к использованию.

4. Намагниченность $m=1-2x$

1. Точная спектральная плотность $D(x,y)$ (3) позволяет вычислить статистическую сумму $Z(\beta,H)$ и другие физические характеристики системы.

$$Z_N(\beta,H)=\sum_{n=0}\sum_{\mathbf{s}\in\Omega_n} D(n,k)e^{-\beta NE(\mathbf{s},H)}\to$$
$$\frac{N}{2\pi}2\int_0^{1/2}\frac{dx}{\sqrt{x(1-x)}}\int_0^{x}\frac{dy\cdot \mathrm{e}^{-N\cdot\varphi(x,y,\beta,H)}}{\sqrt{(x-y)(1-x-y)}}\sim \quad (11)$$
$$\sim \exp\left(-N\cdot\varphi(x_c,y_c,\beta,H)\right),$$

где функция в показателе экспоненты имеет вид - см. (1)-(3):

$$\varphi(x,y,\beta,H)= \quad x\cdot S\left(\frac{y}{x}\right)+(1-x)\cdot S\left(\frac{y}{1-x}\right)+$$
$$+\beta\cdot\left(4y-1+H\cdot(2x-1)\right),$$

а y_c и x_c в (11) отвечают глобальному минимуму функции $\varphi(x,y)$. Для лучшей обозримости формул будем опускать зависимость функций от глобальных параметров β и H, и писать $\varphi(x,y)$ вместо $\varphi(x,y,\beta,H)$ и т.п.

В выражении (11) y_c является решением уравнения $\partial\varphi(x,y)/\partial y=0$. Легко видеть, что

$$y_c(x)=\frac{2x(1-x)}{1+q(x)}=\frac{q(x)-1}{2b},\ b=e^{4\beta}-1,$$
$$q(x)\ \equiv\ q(x,\beta)\ =\sqrt{1+4x(1-x)\cdot b}.$$

Заметим, что перевальная «точка» $y_c(x)$ не зависит от магнитного поля H.

Для отыскания x_c необходимо решить уравнение $\partial\varphi(x,y_c(x))/\partial x=0$, которое после преобразований приводится к виду:

$$\varphi'_x(x,y_c(x))\ =\ 2\beta H+$$
$$+\ln\left(1-\frac{y_c(x)}{x}\right)-\ln\left(1-\frac{y_c(x)}{1-x}\right)=0. \quad (12)$$

Положим, что магнитное поле отсутствует: $H=0$. Тогда решением уравнения (12) будет

$$x_c\equiv x_0=\frac{1}{2}\ ,$$

и нетрудно вычислить: $y_c(x_c)=\dfrac{1}{2(1+\mathrm{e}^{2\beta})}$,

$\varphi(x_c,y_c)=S\left(\dfrac{1}{1+\mathrm{e}^{2\beta}}\right)-\beta\dfrac{\mathrm{e}^{2\beta}-1}{\mathrm{e}^{2\beta}+1}$. Это дает выражение для свободной энергии:

$$f(\beta)=-\frac{1}{\beta}\lim_{N\to\infty}\frac{\ln Z_N}{N}=\frac{\varphi(x_c,y_c)}{\beta}=$$
$$=\frac{1}{\beta}S\left(\frac{1}{1+\mathrm{e}^{2\beta}}\right)-\frac{\mathrm{e}^{2\beta}-1}{\mathrm{e}^{2\beta}+1}\equiv-\frac{1+\ln(1+\mathrm{e}^{-2\beta})}{\beta}.$$

Последнее тождество в этой цепочке придает выражению для свободной энергии вид, традиционно принятый в литературе [3].

2. Переменная x напрямую связана с намагниченностью $m=1-2x$, которая является важной характеристикой состояния системы. Ограничившись в (11) интегрированием по y, получим характеристику, пропорциональную числу состояний с намагниченностью $m=1-2x$ при данной температуре β:

$$P_N(x,\beta)=\ \sqrt{\frac{N}{2\pi}}\int_0^{x}\frac{\exp\left(-N\cdot\varphi(x,y)\right)\cdot dy}{\sqrt{(x-y)(1-x-y)}}=$$
$$=\frac{\exp\left(-N\cdot\varphi(x,y_c)\right)}{\sqrt{2\pi Nx(1-x)}\sqrt{q(x,\beta)}}.$$

Поделив последнее выражение на статистическую сумму $Z(\beta)$ получим плотность распределения намагниченности при данной температуре β:

$$p_N(x,\beta)=\frac{P_N(x,\beta)}{Z_N(\beta)}=\frac{\mathrm{e}^{-N\left[\varphi(x,y_c(x))+\beta+\ln\left(1+e^{-2\beta}\right)\right]}}{\sqrt{2\pi Nx(1-x)}\sqrt{q(x,\beta)}}\ ,$$

$$x\in[0,1/2],\ \beta\in[0,\infty)\ . \quad (13)$$

Вообще говоря, x не может сколь угодно близко приближаться к 0, поскольку $n=xN$ должно быть достаточно велико, чтобы можно было применять формулу Стирлинга. Скажем, $n_{\min}\sim 10^2-10^3$, и должно выполняться $x_{\min}=n_{\min}/N>0$. Это ограничение предохраняет нас от сингулярности при оценке значения $p_N(x,\beta)$, когда $x\to 0$: предэкспонента при этом стремится к $1/\sqrt{2\pi n_{\min}}$, а показатель экспоненты стремится к $-N\cdot\ln\left(1+e^{-2\beta}\right)$. Таким образом,

$$\lim_{x\to 0}p_N(x,\beta)\sim\frac{\exp\left(-N(1+e^{-2\beta})\right)}{\sqrt{2\pi n_{\min}}}\ . \quad (14)$$

Это выражение нам потребуется.

На рис.3 для различных значений β показаны графики плотности состояний $p_N(x,\beta)$ (13) (левая панель) и соответствующих показателей экспоненты $\ln\left(p_N(x,\beta)\right)/N$ (правая панель); число спинов здесь $N=1000$. Когда β находится около 0 - $\beta\sim 0 \Rightarrow T=1/\beta\sim\infty$ - намагниченность почти всех состояний принадлежит узкой области около нуля – см. пик плотности распределения в районе значения $x_0=0.5 \Leftrightarrow m_0=0$. С ростом β - то есть, с понижением температуры T - высота пика падает, он все больше и больше расплывается, и в конце концов выполаживается. Это довольно неожиданно.

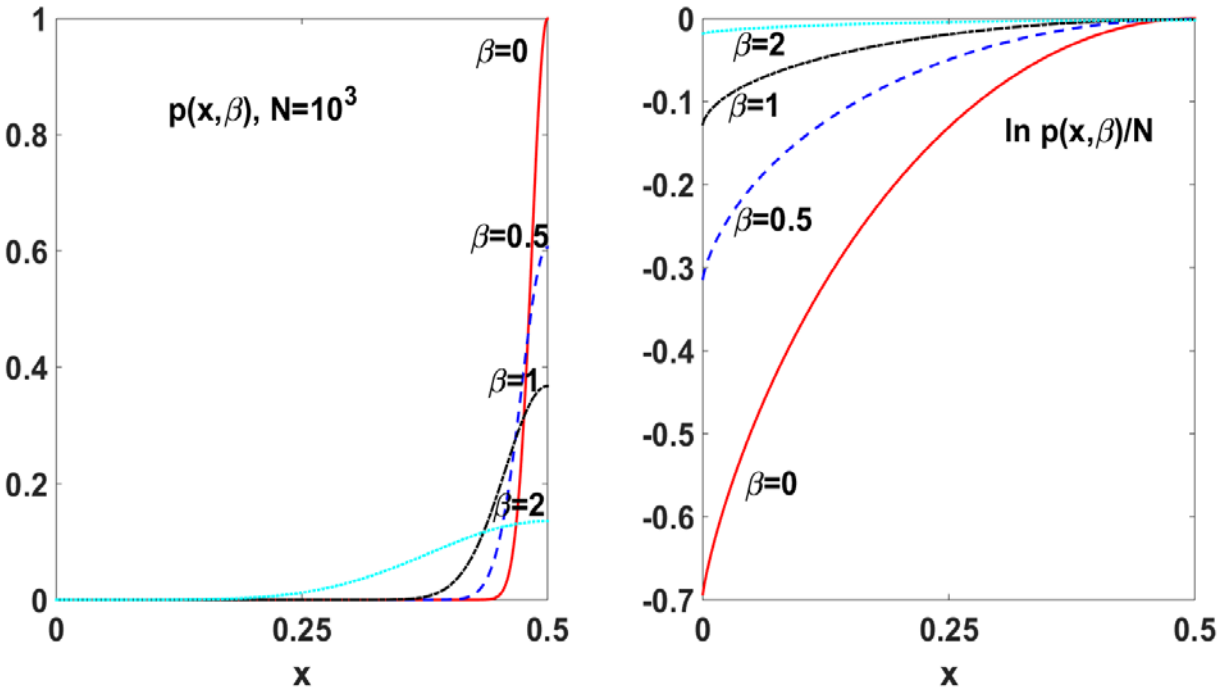


Рис. 3. Плотность распределения намагниченности $p_N(x,\beta)$ для $\beta=0, 0.5, 1, 2$ (левая панель) и $\ln\left(p_N(x,\beta)\right)/N$ (правая панель).

В самом деле, по мере понижения температуры T система стремится в основное состояние $\mathbf{s}_0$, намагниченность которого равна 1. Казалось бы, с ростом β на левой границе интервала - в районе $x_1=0 \Leftrightarrow m_1=1$ - должен формироваться пик плотности распределения (13). Однако никакого пика на левом конце не образуется. С ростом β кривая $\ln\left(p_N(x,\beta)\right)/N$ на правой панели рис.3 все сильнее выполаживается: ее левый конец поднимается вверх, оставаясь при этом ниже правого конца.

Отсутствие пика распределения на левом конце интервала можно обосновать не только графически. Оценим величину $p_N(x,\beta)$ на правом конце интервала, когда $x\to 1/2$: показатель экспоненты в (13) стремится здесь к 0, величина $q(x)$ стремится к $e^{2\beta}$, и имеем:

$$\lim_{x\to 1/2} p_N(x,\beta)=\sqrt{\frac{2}{\pi N}}e^{-\beta}. \qquad (15)$$

Сравнивая это выражение с (14) видим, что для каждого значения β, при $N\to\infty$, значение показателя экспоненты на левом конце интервала $-N(1+e^{-2\beta})$ много меньше значения показателя $-\beta$ на правом конце. Левый конец кривой $p_N(x,\beta)$ всегда ниже ее правого конца, и при $\beta\to\infty$ распределение стремится к равномерному.

Такое поведение распределения намагниченности можно объяснить. Когда $\beta\to\infty$ и система стремится к основному состоянию $\mathbf{s}_0$, энергия принимает значения,

близкие к $E_0 = -1$. Иными словами, значения энергий локализуются вблизи значения E_0. Из этого вовсе не следует, что значения намагниченности должны аналогично группироваться вблизи 1. В самом деле, если бы энергии, близкие к E_0, достигались *только* на состояниях из небольшой окрестности $\mathbf{s}_0$, тогда значения намагниченности при $\beta \to \infty$ должны были бы группироваться около 1. Однако легко видеть, что в *каждой* n-окрестности $\mathbf{s}_0$ имеются N конфигураций, энергия которых отличается от E_0 на бесконечно малую величину и равна $E(1) = -(1 - 4/N)$. Чтобы в этом убедиться, подставим в выражение (2) для $D(n,k)$ значение $k = 1$, отвечающее энергии $E(1)$ - получим, что $D(n,1) = N$. Таким образом, конфигурации, почти не отличающиеся от E_0 по энергии, вовсе не локализуются в районе основного состояния $\mathbf{s}_0$, но сравнительно равномерно распределены по всему конфигурационному пространству. Этим и объясняется тот факт, что с ростом β кривые $p_N(x,\beta)$ выполаживаются и стремятся к равномерному распределению.

4. Заключение и выводы

Спектральной плотностью определяется статистическая сумма и многие макро-характеристики модели – свободная энергия, намагниченность, внутренняя энергия, энтропия и пр. Однако получение точного выражения для спектральной плотности является трудной задачей, которая решена только в нескольких исключительных случаях. Для 1D модели Изинга нами получено точное комбинаторное выражение для спектральной плотности в n-окрестности основного состояния. Это позволило сравнить точную спектральную плотность с ее гауссовой аппроксимацией, обычно используемой в методе n-окрестностей. Оказалось, что центральная часть распределения аппроксимируется гауссовой плотностью хорошо, но различия в поведении функций на краях интервала приводят к тому, для 1D модели Изинга метод n-окрестностей не работает – по крайней мере, в его нынешней «гауссовой» формулировке.

Предстоит выяснить, во-первых, каким образом гауссова аппроксимация позволяет получить хорошие результаты для решеток больших размерностей $d \geq 3$? Для таких размерностей методом n-окрестностей получено простое аналитическое выражение для критической температуры, которым удовлетворительно описываются общепризнанные результаты компьютерного эксперимента для гиперкубических решеток размерностью $3 \leq d \leq 7$ [1,2]. Подобное совпадение трудно считать случайным.

И второй вопрос: удастся ли адаптировать метод n-окрестностей для адекватного описания решеток малых размерностей $d < 3$? В первую очередь нас интересует планарная модель Изинга ($d = 2$), и включение в нее внешнего магнитного поля – эта задача ждет своего решения больше полувека.

Отметим, что даже для хорошо изученной 1D модели Изинга найденное точное выражение для спектральной плотности позволило получить новый содержательный результат – оказалось, что при стремлении температуры к 0 распределение намагниченности стремится к равномерному распределению. Можно надеяться, что на планарную модель удастся обобщить точное выражение для спектральной плотности, полученное для 1D модели Изинга.

Работа выполнялась в рамках программы 0065-2018-0001.

ПРИЛОЖЕНИЕ 1

1. Представим матрицу связи одномерной модели Изинга $\mathbf{J}$ в виде суммы матрицы одношагового сдвига $\mathbf{T}$ и матрицы $\mathbf{T}^+$, транспонированной к $\mathbf{T}$:

$$\mathbf{J} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & \ddots & 0 \\ 0 & \ddots & \ddots & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} = \mathbf{T} + \mathbf{T}^+, \qquad \text{(A1)}$$

где $\mathbf{T} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & \ddots & 0 \\ 0 & \ddots & \ddots & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$.

Константа взаимодействия между спинами равна 1: $J_{ii+1} = 1$.

Вектор $\mathbf{e}_i$ будет означать i-й орт в пространстве $\mathbf{R}^{\mathbf{N}}$: $\left(\mathbf{e}_i,\mathbf{e}_j\right)=\delta_{ij}$, где δ_{ij} - символ Кронекера, $i=1,...,N$. Матрица $\mathbf{T}$ переводит i-й орт в $(i+1)$-й : $\mathbf{T}\mathbf{e}_i=\mathbf{e}_{i+1}$, а поскольку граничные условия являются периодическими, выполняется соотношение: $\mathbf{T}\mathbf{e}_N=\mathbf{e}_1$. Обозначим через $F(\mathbf{s})$ значение квадратичной формы для матрицы $\mathbf{T}$: $F(\mathbf{s})=\left(\mathbf{T}\mathbf{s},\mathbf{s}\right)$. Тогда значение энергии состояния $\mathbf{s}$ равно:

$$E(\mathbf{s})=-\frac{\left(\mathbf{J}\mathbf{s},\mathbf{s}\right)}{2N}=-\frac{F(\mathbf{s})}{N}. \qquad \text{(A2)}$$

С функцией $F(\mathbf{s})$ работать удобнее, чем с $E(\mathbf{s})$.

В качестве начальной конфигурации возьмем основное состояние $\mathbf{s}_0=(1,1,...,1)=\sum_{i=1}^{N}\mathbf{e}_i$. Конфигурацию из n-окрестности $\mathbf{s}_0$ обозначим $\mathbf{s}_{i_1i_2..i_n}$, подразумевая, что «-1» стоят на позициях i_1, $i_2,..,i_n$:

$$\mathbf{s}_{i_1i_2..i_n}=\mathbf{s}_0-2\left(\mathbf{e}_{i_1}+\mathbf{e}_{i_2}...+\mathbf{e}_{i_n}\right).$$

Легко видеть, что

$$F(\mathbf{s}_{i_1i_2..i_n})=N-4\cdot n+ \\ +4\left(\mathbf{e}_{i_1+1}+...+\mathbf{e}_{i_n+1},\mathbf{e}_{i_1}+...+\mathbf{e}_{i_n}\right). \qquad \text{(A3)}$$

Обозначим скалярное произведение в выражении (A3) специальным символом:

$$\Delta_{i_1i_2..i_n}=\left(\mathbf{e}_{i_1+1}+...+\mathbf{e}_{i_n+1},\mathbf{e}_{i_1}+...+\mathbf{e}_{i_n}\right). \qquad \text{(A4)}$$

Спектр энергий для n-окрестности определяется совокупностью различных значений $\Delta_{i_1i_2..i_n}$.

Несложно разобраться в том, как эти величины устроены. Пусть для начала все индексы идут подряд: $i_2=i_1+1,\ i_3=i_2+1,..,i_n=i_{n-1}+1$. Легко понять, что соответствующее значение характеристики $\Delta_{i_1i_2..i_n}$ (A4) равно $n-1$. Далее, пусть совокупность индексов распадается на две изолированные подгруппы, внутри которых индексы опять-таки идут подряд: первая группа индексов имеет вид $i,i+1,..,i+a$, вторая - $k,k+1,..,k+b$, причем выполняется равенство $a+b+2=n$, но между значениями индексов $i+a$ и k есть пробел: $k>i+a+1$. Легко видеть, что, независимо от состава обеих групп индексов, значение $\Delta_{i_1i_2..i_n}$ (A4) равно $n-2$.

Аналогично, когда набор индексов $i_1,i_2,..,i_n$ распадается на три изолированные подгруппы - внутри которых индексы идут подряд - независимо от состава групп значение $\Delta_{i_1i_2..i_n}$ равно $n-3$.

И так далее – когда набор индексов $i_1,i_2,..,i_n$ распадается на k изолированных подгрупп (внутри которых индексы идут подряд), значение $\Delta_{i_1i_2..i_n}$ - независимо от состава и размера групп - равно $n-k$.

Для $n\le N/2$ максимально возможное число изолированных подгрупп индексов есть $k_{\max}=n$.

Используя (A2), (A3) и (A4) можем записать: энергии состояний из n-окрестности $\mathbf{s}_0$ принимают в точности n значений:

$$E(k)=-\left(1-4k/N\right),\ k=1,2,...n, \qquad \text{(A5)}$$

где $n=1,2,..,N/2$.

2. Разберемся теперь, какова кратность вырождения энергии $E(k)$ для состояний из n-окрестности – дело в том, что кратность вырождения одного и того же значения энергии $E(k)$ в разных n-окрестностях, как правило, различна.

Число состояний n-окрестности, энергия которых равна $E(k)$, будем обозначать $D(n,k)$.

0-окрестность состоит из самой конфигурации $\mathbf{s}_0$, $F(\mathbf{s}_0)=\left(\mathbf{s}_0,\mathbf{s}_0\right)=N$, и можем записать: $E_0=-1,\ D(0,0)=1$.

1-окрестность $\mathbf{s}_0$ состоит из N конфигураций, отличающихся от $\mathbf{s}_0$ противоположным значением всего одного спина – следовательно, все N конфигураций из 1-окрестности характеризуются одной и той же энергией $E(1)$, а $D(1,1)=N$.

Легко видеть, что C_N^2 конфигураций *2-окрестности* $\mathbf{s}_0$ распадаются на две группы:

во-первых, это N конфигураций, у которых две «-1» стоят подряд – они характеризуются энергией $E(1)$;

во-вторых, все остальные конфигурации из 2-окрестности – они

характеризуются энергией $E(2)$. Следовательно,

$$D(2,1) = N,\ D(2,2) = \frac{N(N-3)}{2}.$$

Несложно показать, что в *3-окрестности* ровно N конфигураций характеризуются значением энергии $E(1)$, $N(N-4)$ конфигураций характеризуются значением $E(2)$, а все остальные конфигурации – значением $E(3)$. Тогда получаем:

$$D(3,1) = N,\ D(3,2) = N(N-4),$$
$$D(3,3) = \frac{N(N-4)(N-5)}{3!}.$$

Начиная с этого момента рассмотрение усложняется, однако с помощью рутинных вычислений не составляет проблемы получить для *4-окрестности* и *5-окрестности* следующие выражения:

$$D(4,1) = N,\ D(4,2) = \frac{3}{2}N(N-5),$$
$$D(4,3) = \frac{N(N-5)(N-6)}{2},$$
$$D(4,4) = \frac{N(N-5)(N-6)(N-7)}{4!},$$

и, соответственно:

$$D(5,1) = N,\ \ D(5,2) = 2N(N-6),$$
$$D(5,3) = N(N-6)(N-7),$$
$$D(5,4) = \frac{N(N-6)(N-7)(N-8)}{3!},$$
$$D(5,5) = \frac{N(N-6)(N-7)(N-8)(N-9)}{5!}.$$

Анализ выражений для частот энергии $E(k)$, полученных для пяти n-окрестностей, позволил выписать комбинаторную формулу, которая затем была проверена в компьютерных экспериментах для различных значений N, n и k:

$$D(n,k) = \frac{N\cdot k}{(N-n)\cdot n} C_{N-n}^{k}\cdot C_{n}^{k}, \qquad (A6)$$
$$n = 1,2,\ldots N/2;\ \ k = 1,2,\ldots n.$$

ПРИЛОЖЕНИЕ 2

Получим выражение для полной спектральной плотности $G(y)$, проинтегрировав по x частичную спектральную плотность $G(x,y)$ (4):

$$G(x,y) = \frac{e^{-N\cdot\varphi(x,y)}}{2\pi N\left(x(1-x)\right)^{3/2}},$$

где $\varphi(x,y) = S(x) + \frac{1}{2}\left(1 - \frac{y}{x(1-x)}\right)^2$.

Используя метод Лапласа вычисления интеграла от функции, у которой в показателе экспоненты стоит большой множитель N, получаем цепочку равенств:

$$G(y) = N\int_{y}^{1-y} G(x,y)dx = \frac{2}{2\pi}\int_{y}^{1/2}\frac{e^{-N\cdot\varphi(x,y)}dx}{\left(x(1-x)\right)^{3/2}} = \frac{\left(2\pi N\right)^{-1/2}\cdot e^{-N\cdot\varphi(x_{\min},y)}}{\left(x_{\min}(1-x_{\min})\right)^{3/2}\cdot\sqrt{\varphi''_{xx}(x_{\min},y)}}, \qquad (A7)$$

где $x_{\min}$ - точка глобального минимума функции $\varphi(x,y)$ по переменной x. Вообще говоря, $x_{\min}$ зависит от значения переменной y, однако, как мы увидим, не в данном случае.

В самом деле, уравнение для отыскания $x_{\min}$ имеет вид:

$$\ln\frac{x}{1-x} + \frac{y(1-2x)}{\left(x(1-x)\right)^2}\left(1 - \frac{y}{x(1-x)}\right) = 0. \qquad (A8)$$

Очевидно, что точка $x_0 = 1/2$ является решением уравнения (A8). Легко видеть, что вторая производная $\varphi''_{xx}(x_0,y) = 4\left(1 - 8y + 32y^2\right)$ положительна при всех значениях y - следовательно x_0 всегда является точкой минимума функции $\varphi(x,y)$. Нетрудно видеть, что пока $y > 0.053$, x_0 является единственным минимумом $\varphi(x,y)$. Когда же $y < 0.053$, у уравнения (A8) появляются еще два решения: точка минимума x_m и точка максимума x_M, такие, что $0 < x_m < x_M < x_0$. Непосредственное исследование показывает, что минимум $\varphi(x_0,y)$ всегда глубже расположенного на левом краю минимума $\varphi(x_m,y)$, и таким образом, при любом значении y глобальный минимум функции $\varphi(x,y)$ по переменной x всегда находится в

точке $x_0 = 1/2$. Подставляя в (A7) x_0 в качестве $x_{\min}$, окончательно получаем:

$$G(y) = \frac{4 \cdot \mathrm{e}^{N\cdot\left(\ln 2 - \frac{(1-4y)^2}{2}\right)}}{\sqrt{2\pi N}\cdot\sqrt{1-8y+32y^2}}, \quad \text{(A9)}$$

где $y \in [0,1/2]$.

Проделывая аналогичные вычисления для частичной спектральной плотности $D(x,y)$ (3), убеждаемся в том, что $x_0 = 1/2$ всегда является единственной точкой минимума, и получаем:

$$D(y) = \frac{2\cdot\exp\left(-N\cdot S(2y)\right)}{\sqrt{2\pi N}\cdot\sqrt{2y(1-2y)}} = 2C_N^{2y\cdot N}. \quad \text{(A10)}$$

Последнее выражение совпадает с тем, что было получено в конце 30-х годов совсем из других соображений [7,8].

Normal approximation of a spectral density for 1D Ising Model

Leonid Litinskii and Inna Kaganowa

Abstract: We investigated a spectral density of a 1D Ising Model. It was shown that when we used the normal approximation of the spectral density the n-vicinity method was not appropriate in description of the behavior of the 1D Ising model. We found an exact combinatorial expression for the energy distribution in the n-vicinity. Basing on this result, we obtained a magnetization distribution for an arbitrary temperature.

Keywords: 1D Ising model, spectral density, n-vicinity method.

Литература

[1] B. Kryzhanovsky and L. Litinskii. Applicability of n-vicinity method for calculation of free energy of Ising Model. “Physica A”, v. 468 (2017), pp. 493–507.

[2] P.H. Lundow, K. Markstrom The discontinuity of the specific heat for the 5D Ising model. “Nuclear Physics B”, v.895 (2015), pp.305-318.

[3] Р. Бэкстер. Точно решаемые модели статистической физики, Москва: Мир, 1985.

[4] Б.В. Крыжановский, Л.Б. Литинский. Обобщенное уравнение Брэгга-Вильямса для систем с произвольным дальнодействием. “ДАН”, т.459 (2014), №6, с. 1-5.

[5] Boris Kryzhanovsky and Leonid Litinskii. Generalized approach to description of energy distribution of spin system. “Opt. Mem. & Neu. Netw”, v.24 (2015). No.3, pp.165-185.

[6] Leonid Litinskii and Boris Kryzhanovsky. Spectral density and calculation of free energy. “Physica A” (2018), в печати.

[7] R. Becker, W. Doring. Ferromagnetism, Berlin: Springer Verlag, 1939.

[8] С. В. Вонсовский и Я. С. Шур. Ферромагнетизм, М.-Л.: ГИТТЛ, 1948.

Классификация полей проницаемости карбонатных трещиноватых коллекторов специальными методами ГИС и ГДИ

А.А. Колеватов, Ю.М. Штейнберг, Ю.Б. Чен-лен-сон, А.Г. Дяченко

ФГУ ФНЦ НИИСИ РАН, Москва, Россия , *E-mail:* akolevatov@niisi.ras.ru

Аннотация: В статье рассматриваются особенности карбонатных трещиноватых нефтенасыщенных коллекторов связанные с различиями их фильтрационно-емкостных свойств, на основе которых возможно разделение вскрытых скважинами пород-коллекторов на петрофизические классы с идентификацией особенностей выделенных классов.

Ключевые слова: карбонатные трещиноватые коллектора, геофизические исследования скважин (ГИС), акустический каротаж, гидродинамические исследования скважин (ГДИ)

Введение

В процессе разведки месторождений нефти и освоении скважин проводятся как исследования в открытом стволе, так и отработка на режимах с целью оценки потенциала и определения типа коллектора. Карбонатные трещиноватые коллектора в отличии от терригенных имеют гораздо большую степень неоднородности в части структуры пустотного пространства в зависимости в том числе от вторичных процессов. Для таких коллекторов стандартный комплекс геофизических исследований скважин (ГИС) (без применения специальных методов), позволяет с некоторой долей погрешности определить наличие или отсутствие нефтенасыщения отдельных толщин. Но установление нефтенасыщенных толщин не означает возможности получения притока из тех же интервалов. Идентификация структуры порового пространства для карбонатных трещиноватых коллекторов возможна только при задействовании специальных методов исследования, таких как широкополосный акустический каротаж с регистрацией волны Стоунли, а также посредством гидродинамических исследований скважин с регистрацией кривой восстановления давления (КВД).

Далее будут приведены особенности идентификации петрофизических классов карбонатных трещиноватых нефтенасыщенных коллекторов посредством применения ГДИ и специальных методов ГИС открытого ствола.

Описание объекта исследования

В карбонатных трещиноватых коллекторах приток может быть получен не только из мощностей имеющих пористость до 15%, но и из мощностей с пористостью 2-3%, имеющих систему трещин разного порядка.

Одним из основных способов учета наличия трещиноватости в исследуемом пласте является комплексирование данных ГИС с данными сейсморазведки с целью построения так называемых кубов проницаемости и кубов трещиноватости.

Кубы трещиноватости в некоторой степени позволяют получить представление о геометрии трещин в исследуемом объекте.

Но такие кубы не дают информации о проницаемости или залеченности трещин. Кроме того, кубы не дают информации о трещинах, размеры которых не позволяют получить достаточно сильный сигнал на фоне помех, имеющих место при сейсморазведке.

На рисунке 1 представлен общий вид трещиновато-пористого коллектора.

Такой коллектор, по сути, представляет из себя систему матричных блоков, разделенных между собой трещинами разного порядка, которые могут быть проницаемы или залечены.

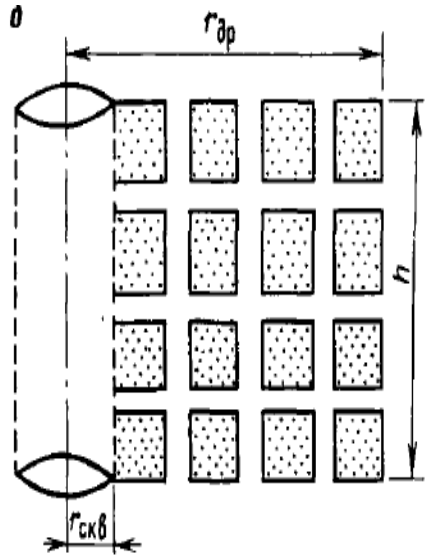


Рисунок 1. Модель трещиновато-порового коллектора Уоррена-Рута. Наличие проницаемости матрицы и трещин.

Другими словами, притоки в скважину в таких коллекторах могут идти как из матрицы, так и из трещин, либо только из матрицы, либо только из трещин. Эта особенность карбонатного трещиноватого коллектора в зависимости от зоны в которую попала скважина, по разному отражается на результатах ГИС открытого ствола и на результатах ГДИ.

Выделение трещиноватости с помощью скважинных геофизических методов является весьма сложной задачей до сих пор не имеющей универсального решения. Известная фирма “Schlumberger” в конце 80 годов создала электрический микроимиджер “FMI” позволяющий выделить и посчитать количество трещин на стенке скважины. К сожалению, создание “FMI” не решило всех проблем по оценке трещиноватости, зачастую мы не можем разделить открытые и «залеченные» трещины, хотя понятно, что последние не являются проводящими и через них не может продвигаться флюид. Запись “FMI” возможна только в скважинах пробуренных с помощью раствора на водной основе, что исключает исследование целого ряда скважин пробуренных растворами на нефтяной основе (РНО). Также запись микроимиджера “FMI” довольно дорогостоящая, в силу того, что производится только самой компанией “Schlumberger”и поэтому большого распространения в России не получили.

1. Описание методов изучения структуры пустотного пространства.

1.1. Специальные методы скважинной геофизики для идентификации структуры пустотного пространства.

Одним из методов, показания которого наиболее плотно связаны со структурой горной породы и входящий в состав обязательного комплекса ГИС является акустический каротаж (АК). Но традиционно, из пакета волн используется только продольная волна для определения коэффициента пористости, изредка пытаются использовать поперечные волны, и практически не используют волны Лэмба и Стоунли также входящие в полный волновой пакет. Применение цифровой регистрации данных акустического каротажа (АК) несколько подняло интерес к волнам Лэмба и Стоунли, параметры которых применяют для решения различных геологических и технических задач. Среди них выделение коллекторов с различной структурой порового пространства, прогнозирование и определение интервалов развития трещин гидроразрыва, оценка качества цементирования обсадных колонн и др.

Акустический каротаж (АК) основан на изучении характеристик упругих волн ультразвукового и звукового диапазона в горных породах. При АК в скважине возбуждаются упругие колебания, которые распространяются в ней и в окружающих породах и воспринимаются приемниками, расположенными в той же скважине.

Нормальные волны распространяются в телах ограниченных размеров (волноводах) без изменения формы. Во всех волноводах на сколь угодно низких частотах распространяется с постоянной скоростью нормальная волна нулевого порядка. Для волн более высоких порядков (мод) существуют критические частоты, ниже которых они не распространяются. Для цилиндрической скважины диаметром 0,2 м нормальные волны высших (ненулевого) порядков не могут распространяться в газовой среде на частотах ниже 1,65 кГц, а при заполнении водой - 7,5- 8 кГц.

Рисунок 2. Схема смещения частиц среды при распространении продольных (а) и поперечных (б) волн.

Волны Лэмба распространяются в твердой пластине (слое) или стержне со свободными границами. При малой толщине пластины в ней возможно распространение только двух волн нулевого порядка: продольной и изгибной. Продольная волна (далее будет обозначена литерой L) похожа на продольную волну Р в неограниченном пространстве. В ней преобладает продольное смещение частиц. Кроме плоских пластин из однородного изотропного материала, волны Лэмба распространяются в искривленных пластинах и в пластинах с неоднородными механическими свойствами, что характерно для обсадных колонн.

Волна Стоунли - один из типов поверхностных волн, распространяющихся вдоль границы твердого тела с другими средами и затухающие при удалении от границы раздела. Если второй средой служит сильно разреженная газовая среда, то это будет волна Рэлея. Ее энергия сосредоточена в поверхностном слое твердого вещества. Фазовая скорость волны Рэлея (υR) примерно равна 0,9 υS, где υS, - фазовая скорость плоской поперечной волны. Если второй средой служит жидкость, то вдоль границы распространяется волна Стоунли (или Стонли, Stoneley). Фазовая скорость υST волны Стоунли меньше значений продольной и поперечной волн в обеих средах, то есть $\upsilon ST < \upsilon P, uS, u_{ж}$, где $u_{ж}$ - скорость волны в жидкости.

В обеих средах волна Стоунли распространяется с затуханием, обычным для объемных волн. Энергия волны убывает экспоненциально от границы. В твердом теле она распространяется в слое толщиной 0,5 λж/π (где λж - длина этой волны в жидкости), а в жидкости - в слое толщиной, значительно превосходящей λж. Скорость распространения и затухание волны Стоунли в пористых средах зависят также от проницаемости.

Существование в обсадной колонне независимой (необобщенной) волны Лэмба доказано построенными без вмешательства оператора годографами волн, распространяющихся в скважине и в околоскважинном пространстве (рис. 3) [3]. Их получили с помощью скважинного прибора с изменяющимися длинами измерительных зондов [6]. В приборе, полностью воспроизводящем геометрию АК, при остановке скважинного зонда инициируется движение (в винипластовой трубе) приемника упругих колебаний, возбуждаемых двумя излучателями, разнесенными на 0,5 м. Регистрация и отображение данных производятся на дневной поверхности синхронно с движением приемника средствами каротажной лаборатории.

Волновые пакеты и годографы содержат информацию о четырех типах волн, начиная с момента их образования. Три из них распространяются в разных средах: в скважинной жидкости (Рж), обсадной колонне (Lк) и в горной породе (продольная головная волна Рп). Колебания поперечной волны (Ps) отсутствуют, так как соотношение $\upsilon S/\upsilon_{ж}$ неблагоприятно для ее образования. Самой медленной волной, интенсивные колебания которой прослеживаются в волновых пакетах и на годографах, является низкочастотная поверхностная волна Стоунли (St). Ее свойства будут описаны ниже.

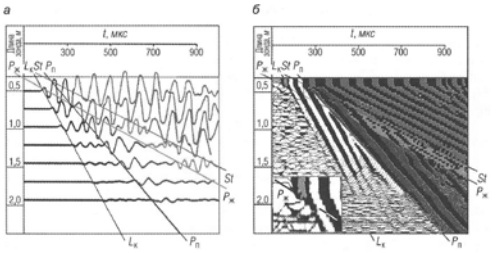


Рисунок 3. Волновые пакеты (а) и годографы (б) упругих волн, полученные в обсаженной скважине прибором с изменяющимися длинами измерительных зондов [10]. Литерами обозначены волны: Рж - в скважинной жидкости; L. - Лэмба в колонне; St - Стоунли; Рп - продольная в породе.

Волна Стоунли

Две сугубо практические задачи связаны с идентификацией волны Стоунли в волновом пакете и измерениями ее характеристик - скорости υ_{st} (интервального времени Δt_{st}), амплитуд A_{st} и затухания α_{st}. Первая обязана возможности расчета скорости распространения поперечной волны в породах,

в которых $\upsilon_s < \upsilon_ж$, либо измерения υ_s, превышающие $\upsilon_ж$, затруднены сильным затуханием поперечной волны. Решением этой задачи заинтересованы специалисты сейсморазведки и добывающих организаций, выполняющих расчет параметров гидроразрывов пластов на малых глубинах. Вторая задача традиционно принадлежит ГИС - это выделение проницаемых пород, являющихся потенциальными коллекторами нефти и газа.

Взаимосвязь между скоростями распространения волны Стоунли Us1 в цилиндрической скважине и поперечной волны с вертикальной поляризацией (SV) в массиве пород впервые установлена в работах [5, 7, 9]. Позже к ним добавились другие авторы [2, 7]. Решение получено для источника волны сжатия, расположенного в скважинной жидкости. Условия распространения упругих волн в ограниченной среде заключались в непрерывности на границе жидкости и твердого тела нормальных к границе напряжений и смещений, равенства нулю касательных напряжений. Для частот, стремящихся к нулю, найдено

$$\upsilon_{St} = \frac{\upsilon_ж}{\sqrt{1+\frac{\sigma_ж \upsilon_ж^2}{\sigma \upsilon_S^2}}}, \qquad (1)$$

где υ_{st} и $\upsilon_ж$ - скорости волн Стоунли и в жидкости, заполняющей скважину; υ_s - скорость поперечной волны в твердом теле; $\sigma_ж$ и σ - плотности жидкости и твердого тела.

Уравнение (1) обладает свойствами некоторой периодической функции, что определяется применением для его вывода функций Бесселя. Фазовая и групповая скорости волны Стоунли являются функциями частоты колебаний, соотношений плотностей жидкости и твердого тела, радиуса скважины и длины волны.

Различают два случая: $\upsilon_s > \upsilon_ж$ и $\upsilon_s < \upsilon_ж$ [8, 9]. В первом случае при достаточно высоких частотах, когда длина волны намного меньше радиуса скважины, поверхностная волна распространяется вдоль стенки скважины со скоростью, близкой к скорости волны Стоунли вдоль плоской границы.

Применение параметров волны Стоунли (Δt_{st}, A_{st}, α_{st}) для выделения проницаемых пород, практически не признаваемое производственными организациями РФ, не менее обоснованно, чем применение для этих целей данных ЯМК или керна.

Покажем это на примере скважин, вскрывших карбонатный и терригенный разрезы. Одна из скважин в Восточной Сибири исследована полным комплексом ГИС, в ней выполнен практически полный отбор керна (рис. 4). Карбонатные коллекторы, суммарные толщины которых достигают полусотни метров, сложены порово-кавернозно-трещиноватыми доломитами. Доломиты сопровождаются отложениями галитов и ангидритов в паровом пространстве. Признаками и количественными критериями коллекторов служат значения эффективной пористости ($k_{п.эф}$) и проницаемости ($k_{пр}$) пород, установленные по материалам ядерно-магнитного каротажа (ЯМК) в сильном магнитном поле и данным АК. Коллекторы оттенены более интенсивной закраской на кривых ГИС.

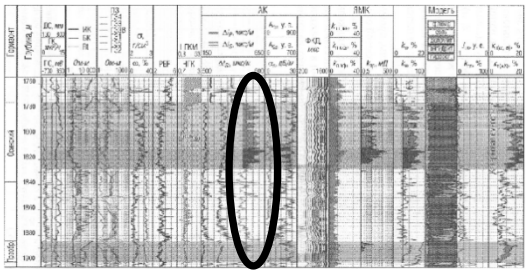


Рисунок 4. Выделение и количественная оценка коллекторов по материалам методов, отражающих пористость и литологию пород. Месторождение в Восточной Сибири. Коллекторы обладают эффективной пористостью (кп.эф) и проницаемостью по данным ЯМК в сильном магнитном поле, им присуще увеличение (Δts), а в трещиноватых интервалах - уменьшение амплитуд (As) поперечной волны; кавернозная пористость определена как разность значений, рассчитанных по данным ГГК-П-НК и АК. Нефть (коричневый цвет) и водонасыщенность (голубой цвет) определены по данным электрометрических видов ГИС.

Результаты этого примера свидетельствуют, что определения значений kп.эф, kпр по материалам ЯМК и тех же величин с использованием интервальных времен Δt st волны Стоунли базируются на одной и той же физической основе (рис. 5). Это могут быть вынужденные колебания флюида в суммарном поровом пространстве (рис. 5, а), то есть динамическая пористость, либо движения того же флюида в каналах, обеспечивающих проницаемость пород. Бесспорное сходство графиков на рис. 5, а и б не позволяет сделать выбор в пользу одной из версий. В англоязычной литературе предпочтение отдается зависимости Δtst от kпр, а вторая версия не рассматривается.[1.]

Второй пример также выполнен по материалам скважины, пробуренной в Восточ-

ной Сибири. Скважина вскрыла весь подсолевой осадочный комплекс. Единичные толщины карбонатных и терригенных коллекторов находятся в пределах десяти метров. Важно, что керн из этой скважины исследован в лаборатории. Прямые сопоставления проницаемости пород на образцах керна с измеренными в скважине интервальными временами Δt st волны Стоунли показывают наличие выраженных взаимосвязей между этими величинами для основных типов коллекторов: карбонатных, песчаников и алевролитов (рис. 6). Наличие трех зависимостей определяется плотностью пород (а она существенно различна для названных литологических разностей) входит в формулу (1) определения Δtst. К тому же, карбонатные коллекторы, разбурены на высокоминерализованной промывочной жидкости, а терригенные - на опресненной. Плотность жидкости также входит в выражение (1).

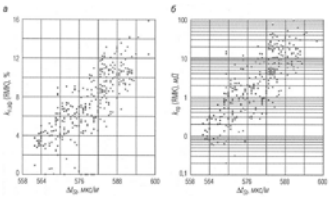


Рисунок 5. Графики сопоставления интервальных времен Δt st со значениями эффективной пористости kп.эф (а) и проницаемости kпр (б), рассчитанными по материалам ЯМК в сильном магнитном поле (прибор ЯМТК-145 разработки ООО "Нефтегазгеофизика")

1.2. Гидродинамические методы идентификации типа исследуемого коллектора.

Гидродинамические методы исследования скважин позволяют получить информацию о некоторых фильтрационно-емкостных свойствах в процессе отработки скважин на режимах (анализ забойного давления и дебита), а также в процессе регистрации изменений забойного давления при остановке скважин: проницаемость пласта, пластовое давление в зоне дренирования скважины за исследуемый период, степень совершенства связи скважины с пластом (скин-фактор) и механизм дренирования пласта, дающего приток в исследуемую скважину.

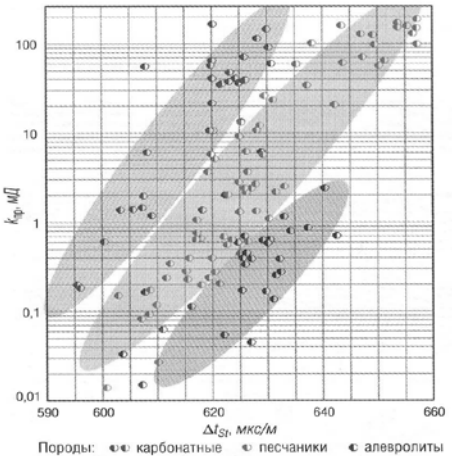


Рисунок 6. Сопоставление измеренных в скважине значений Δtst с абсолютной проницаемостью пород по керну (описание и измерения на образцах керна проницаемости выполнены в лаборатории ТО "СургутНИПИнефть")

1.2.1. Теоретические основы интерпретации данных исследований добывающих скважин методами восстановления (падения) давления

В общем виде вычисление параметров пласта и прискважинной зоны осуществляется посредством совмещения фактической кривой давления с типовыми кривыми. Впервые типовые кривые появились в литературе в 70-х годах [12]. Получение наилучшего совмещения позволяет вычислить уравнение пьезопроводности, в котором параметры пласта и прискважинной зоны выведены с использованием безразмерных переменных: Pd(безразмерное давление), td (безразмерное время) и Cd (характеризующий состояние призабойной зоны пласта).

Типовые кривые — графическое представление давления как функции от времени для определенных конфигураций “скважина-пласт-(граница)”. Они вычисляются на основе существующих аналитических моделей и выражаются в безразмерных переменных.

Существует несколько видов типовых кривых, которые используются для анализа данных ГДИС в случае бесконечного гомогенного пласта. Среди них:

- Типовые кривые Agarwal;
- Типовые кривые McKinley;
- Типовые кривые Earlougher и Kersch;

• Типовые кривые Gringarten;

Типовые кривые Gringarten рис. 7 наиболее совершенны и удобны для применения. Также они наиболее широко применяются в нефтяной индустрии.

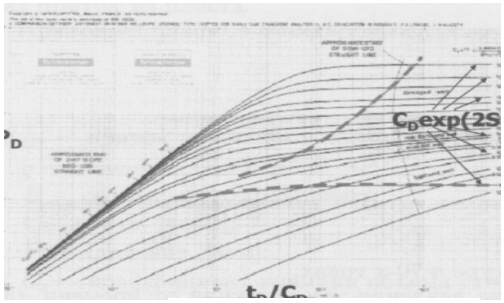


Рисунок 7. Типовые кривые Грингартена**.**

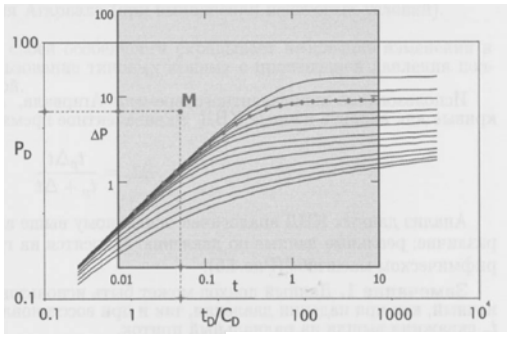


Рисунок 8. Пример определения параметров пласта и прискважинной зоны с применением типовых кривых посредством совмещения с реальными данными**.**

Совмещение по оси давления позволяет определить произведение проницаемости на мощность (kh), а совмещение по оси времени позволяет определить коэффициент влияния ствола скважины (BCC, Cs). Параметр Cd позволяет определить скин-фактор.

Приведенная на рис. 8 иллюстрация определения параметров соответствует интерпретационной модели «однородный пласт». Это означает, что даже при наличии отдельных блоков матрицы, имеющиеся во вскрытом объеме пород трещины не являются каналом притока и «передачи» изменений давления от скважины в пласт. Приток преимущественно осуществляется по поровому объему (межзерновая пористость) либо по системе микротрещин с сечением не превышающим сечение межзерновых сообщающихся пор.

Примеры интерпретационной модели однородного пласта по скважинам, вскрывшим карбонатный нефтенасыщенный коллектор приведены на рисунке 9.

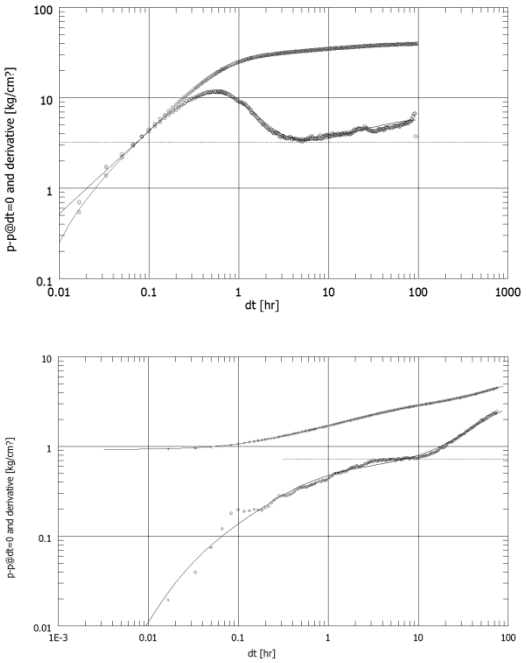


Рисунок 9. Диагностические графики в билогарифмических координатах интепретационной модели «однородный пласт» с одной (верх) и несколькими (низ) непроницаемыми границами, полученные по результатам ГДИ скважин, вскрывших карбонатный коллектор.

Приведенная на рисунке 10 иллюстрация структуры коллекторов в современном представлении получила название «двойной проницаемости».

Это означает, что условно разобщенные, либо слабо сообщающиеся по системе трещин проницаемые прослои могут математически описываться как коллектор, состоящий из двух продуктивных пластов.

При проведении гидродинамических исследований такие коллектора математически описываются [12, 13] уравнениями, включающими два пласта с разной проводимостью и имеющими характерный вид диагностического графика.

В модели двойной проницаемости наиболее проницаемый пласт характеризуется величинами k_1 и h_1.

Наименее проницаемый пласт характеризуется величинами k_2 и h_2**.**

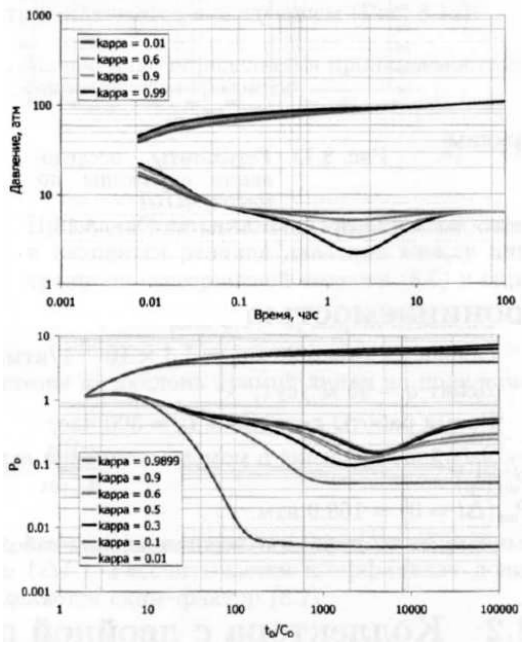


Рисунок 10. Диагностический график модели двойной проницаемости. Скважина вскрывает один (верх) или оба (низ) продуктивных пропластка.

Систему описывают 3 дополнительных параметра:

• доля наиболее проницаемого пласта к емкости системы:

$$\omega = \frac{(\phi c_t h)_1}{(\phi c_t h)_1 + (\phi c_t h)_2}$$

• удельный коэффициент проводимости между пластами:

$$\lambda = \frac{k_2 h_2}{k_1 h_1 + k_2 h_2} \alpha r_w^2$$

• отношение kh наиболее проницаемого пласта к общей kh системы:

$$\kappa = \frac{k_1 h_1}{k_1 h_1 + k_2 h_2}$$

Первый случай - скважина вскрывает оба продуктивных пропластка. В начальный момент времени система действует как два однородных пласта и её можно охарактеризовать как однородный пласт с суммарной kh системы.

Когда наиболее проницаемый пласт работает более активно, начинает проявляться перепад давления между пластами и, соответственно, переток между ними. Переходный процесс характеризуется отклонением производной вниз на диагностическом билогарифмическом графике (рис. 10, слева).

Когда *к* равно 1, то проницаемость менее проницаемого пласта равна «0», и поведение системы соответствует двойной пористости. Менее проницаемый слой представляет матрицу, приток из которой возможен при вскрытии наиболее проницаемого пласта (трещина гидроразрыва или естественная трещиноватость).

Также в карбонатных коллекторах с низкой пористостью матрицы и вследствии доломитизации имеет место вторичное растрескивание пород. По результатам ГДИ в зоне такого пласта определяется интерпретационная модель «пласт с трещиной» (рис. 11). Трещина, как правило, имеет в этом случае бесконечную проводимость.

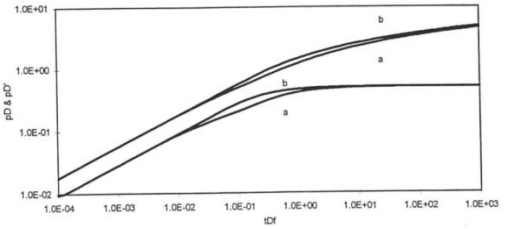


Рисунок 11. Диагностический график модели пласта с трещиной бесконечной проводимости в билогарифмических координатах.

Заключение.

1. Волны Лэмба и Стоунли, регистрируемые в скважине широкополосными приборами акустического каротажа, принадлежат, к разным типам волн. В первом случае это нормальная волна, распространяющаяся в стальной трубе как в волноводе, во втором - это поверхностная волна, которая распространяется по границе жидкости и породы, затрагивая в каждой среде объем, примерно равный половине длины волны. Частоты колебаний и скорости распространения этих волн различаются, по крайней мере, в 4 раза, что не позволяет объединять их в одну волну Лэмба-Стоунли.

На частотах акустического каротажа в скважинной жидкости распространяется только волна нулевого порядка с постоянной скоростью, равной скорости в массиве жидкости. Она наиболее высокочастотная и не может служить компонентом волны Лэмба-Стоунли.

2. Затухание волны Лэмба, которую в акустической цементометрии именуют "волной по колонне", определяется сохранением

волноводных свойств обсадной колонны и служит мерилом качества ее цементирования.

3. Волна Стоунли регистрируется в открытом и в обсаженном стволах скважин. Т.к. волна Стоунли является поверхностной волной, то Δtst в каждой точке записи определяется разницей в длине пути. При неизменном растоянии между излучателем и приёмником, более длинный путь возникает при наличии открытых трещин. Направление распостранения трещиноватости для волны Стоунли большого значения не имеет, в отличии от поперечной волны которая уверенно регистрирует только наличие горизонтальных и субгоризонтальных трещин. Таким образом, волна Стоунли является индикатором открытой (незалеченной) трещиноватости, и также можем оценивать динамическую пористость и проницаемость.

4. Приведенные выше данные о петрофизических свойствах коллекторов при исследовании методами широкополосной акустики показали возможность регистрации параметра трещиноватости, который будет изменяться в зависимости от исследуемого типа карбонатных нефтенасыщенных коллекторов: низкопористая матрица с преимущественно трещинным типом проводимости (в основном доломитизированные породы); матрица с высокой пористостью и среднеразвитой трещиноватостью (пористый известняк со значительной проницаемостью связанных пор, имеющий в том числе развитую систему трещин); чисто поровая матрица, в которой трещиноватость в основном представлена на микроуровне, и проницаемость трещин не превосходит проницаемости связанных пор.

5. Аналогично выявлению особенностей пустотного пространства карбонатных нефтенасыщенных коллекторов методами широкополосной акустики можно говорить о возможности выделения как минимум трех типов карбонатного нефтенасыщенного коллектора с точки зрения гидродинамических исследований скважин: пласт с преимущественным притоком через трещины бесконечной проводимости (чаще всего доломитизированный низко-пористый коллектор); пласт со смешанным притоком как через связанные поры, так и через систему трещин (как правило известняк, не подвергшийся вторичным процессам и доломитизации); поровый карбонатный коллектор не имеющий развитой системы трещин (проницаемость трещин как правило, меньше проницаемости связанных пор и относится к микротрещиноватости).

«Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта №18-07-00673 А».

Permeability area units classification of carbonate fractured reservoirs through specialized open-hole logging and well-testing

A.A. Kolevatov, Y.M. Steinberg, Y.B. Chen-len-son, A.G. Dyachenko

Abstract: Article overviews carbonate fractured reservoirs features related to filtration-capacitive properties, which are applicable for sub-division on several petro-physical classes.

Keywords: carbonate fractured reservoirs, open-hole logging, acoustic logging, well testing,

Литература

1. В. Ф. Козяр, Н. В. Козяр. Волны Лэмба и Стоунли в скважине и решаемые с их помощью задачи промысловой геофизики// НТВ "Каротажник". Тверь: Изд. АИС. 2013. Вып. 46. С. 100- 120.
2. А.А.Кауфман, А,Л.Левшин. Введение в теорию геофизических методов. М.: ООО "Недра-Бизнесцентр" , 2006. Ч. 5. 663 с.
3.В.Ф. Козяр, Д.В.Белоконь, Н.В.Козяр. Успехи и недостатки применения акустического каротажа. Направления развития теории и практики на ближайшее время: Сб. трудов XI сессии Российского акустического общества. Секция "Геоакустика". М.: ГЕОС, 2001. С. 155- 158.
4. В.Н.Косков, В.Б.Косков. Геофизические исследования скважин и интерпретация данных ГИС : Пермский университет 2007. С109-113
5. П.В.Крауклис, Л.А.Крауклис. Волновое поле точечного источника в скважине: Сб. "Вопросы динамической теории распространения сейсмических волн". Л.: Наука, 1976. Вып. 16. С. 41-53.
6. В.Г.Рафиков, Д.В.Белоконь, В.Ф.Козяр. Аппаратура акустического каротажа с изменяющейся длиной зонда// Геофизическая аппаратура. 1974. Вып. 56. С. 84-89.
7. Н.А.Смирнов. Обоснование параметров и разработка основных узлов аппаратуры акустического каротажа для раздельного возбуждения и регистрации продольной, поперечной и Лэмба-Стоунли волн: Автореф. дис. канд. техн. наук. Тверь: АООТ НПП 'ТЕРС", 1996. 25 с.
8. H.C.Cheng, M.N.Toksoz. Generation, Propagation and Analysis Tube Waves // Trans. SPWLA 23th Annual Logging Symposium. 1982. Paper P.
9. C.H. Cheng, M.N.Toksoz. Elastic Wave Propagation in a Fluid-filled Borehole and Synthetic Acoustic Logs // Geophysics. 1981. V. 46. P. 1042- 1053.
10. O.Liu O, У. Stonely Wave-derived Jlt Shear Log // Trans. SPWLA 25th Annual Logging Symposium. 1984. Paper ZZ.
11. X.Tang, M.Altunbay, D.Storey. Joint Interpretation of Formation Permeability from Wireline Acoustic, NMR and Image Log Date // Trans. SPWLA 39th Annual Logging Symposium. 1998. Paper K.K.
12. Гидродинамические исследования скважин: анализ и интерпретация данных / Т.А.Деева, М.Р.Камартдинов, Т.Е.Кулагина, П.В.Мангазеев – Томск 2009.
13. Д.Бурде. Интерпретация результатов исследований скважин // Материалы лекций. Petroleum Engineering and Related Management Traning Gubkin Academy. Москва, 1994. 109 с.

Разработка методики определения типа карбонатного нефтенасыщенного коллектора

А.А. Колеватов[1], Ю.М. Штейнберг[2], Ю.Б. Чен-лен-сон[3], А.А. Егоров[4], А.М. Гиацинтов[5], А.Г. Дяченко[6]

[1,2,3,5,6] ФГУ ФНЦ НИИСИ РАН, Москва, Россия, [4] ООО «КБ АССА», Москва, Россия,

E-mail: akolevatov@niisi.ras.ru

Аннотация: В статье представлен подход к определению структуры пустотного пространства карбонатного нефтенасыщенного коллектора. Рассматриваются вопросы классификации карбонатного нефтенасыщенного коллектора. Описывается разработка методики определения типа коллектора посредством выявления закономерностей в данных гидродинамических исследований, геофизических исследований скважин в открытом стволе и промыслово-геофизических исследований.

Ключевые слова*:* карбонатные нефтенасыщенные коллектора, геофизические исследования скважин (ГИС), гидродинамические исследования скважин (ГДИ), промыслово-геофизические исследования (ПГИ), методика определения типа карбонатного нефтенасыщенного коллектора

Введение

В процессе проектирования разработки месторождений нефти анализируются данные по освоению скважин, данные исследований ГИС в открытом стволе, данные по исследованиям керна нефтенасыщенных коллекторов с целью определения коллекторских свойств. Производится определение типа пласта коллектора и прогноз распределения коллектора в межскважинном пространстве. На основе результатов анализа проектировщики строят различные петрофизические зависимости (керн-ГИС), необходимые для оценки типа коллектора и коллекторских свойств вновь пробуренных скважин без отбора керна. Такой подход широко используется для месторождений нефти и газа в терригенных коллекторах. Это позволяет получить геологическую и гидродинамическую модели месторождений, которые не требуют значительных корректировок в части пространственного распределения фильтрационно-емкостных свойств (ФЕС). При проектировании разработки и моделировании карбонатных нефтенасыщенных коллекторов описанный выше подход срабатывает лишь частично для коллекторов, не подвергшихся вторичным процессам (доломитизация и растрескивание). Карбонатные коллектора, в отличие от терригенных, имеют гораздо большую степень неоднородности в части структуры пустотного пространства. Для таких коллекторов петрофизические зависимости керн-ГИС непредставительны. Это связано с ограниченным объемом выноса керна из-за разрушения в процессе бурения. Для построения приближенных к реальности петрофизических зависимостей карбонатных трещиноватых коллекторов и разработки методики определения типа коллектора необходимо комплексирование ГДИ, ПГИ, и ГИС открытого ствола с задействованием специальных методов.

1. Описание объекта исследования

Результатом анализа керновых данных и данных ГИС при построении петрофизической и геологической моделей как правило является зависимость керн-ГИС, описывающая взаимосвязь пористости и проницаемости вскрытых скважинами нефтенасыщенных коллекторов (рис. 1).

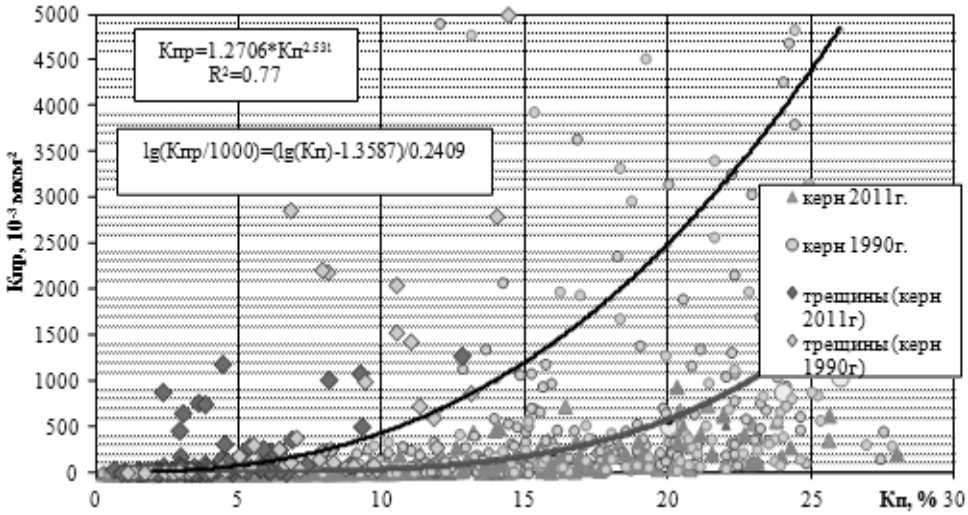


Рисунок 1. Пример взаимосвязи пористости и проницаемости для одного из месторождений нефти в карбонатных коллекторах позднего девона.

Представленная на рисунке 1 петрофизическая зависимость строилась по результатам исследований месторождения нефти в Тимано-Печорской нефтегазоносной области (ТП НГО) для двух типов коллектора – пористая матрица (зеленые точки) и трещиноватый коллектор (синие точки). Данные зависимости имеют погрешности связанные прежде всего с тем, что стандартные методы ГИС открытого ствола дают объективную информацию о ФЕС только для терригенного коллектора, либо карбонатного пористого коллектора, не подвергшегося вторичным процессам (доломитизация и растрескивание). Однако, даже в этом случае зависимости на рисунке 1 могут содержать значительные неточности, связанные с тем, что в силу естественных причин структура порового пространства коллекторов с пористостью первые проценты и с пористостью более 20% будет кардинально отличаться и методы ГИС открытого ствола не будут отражать в полной мере даже таких различий в части корректной оценки ФЕС. При изучении карбонатных коллекторов, в том числе методами ГИС, необходима дополнительная информация, учитывающая изменения структуры пустотного пространства, в том числе при наличии трещин разного порядка [1]. Еще одним ограничением петрофизической зависимости на рис. 1 является использование керновых данных в качестве источника информации о фильтрационных свойствах пласта. Керн даже при идеальном отборе никоим образом не отражает процесс дренирования из объема породы, который условно относят к зоне дренирования скважины. При наличии высокопроницаемых трещин (рис. 2) различия в ФЕС по керну (который, априори, представляет собой матричную часть коллектора) и по пласту вцелом могут достигать нескольких порядков. В этом случае необходима информация о механизме фильтрации в условной зоне дренирования скважины, как имеющей наибольшее значение для разработки месторождения. Таким источником являются гидродинамические исследования скважин (ГДИ). Для более точной идентификации типа коллектора, вскрытого скважиной, необходима взаимная увязка данных ГИС открытого ствола и ГДИ.

2. Анализ соответствия петрофизических зависимостей «пористость-проницаемость»

График петрофизической зависимости проницаемости от пористости (рис. 1) был построен в процессе создания проектного документа для месторождения нефти в карбонатных коллекторах поздне-девонского возраста. График имеет две группы точек условно разделенных по наличию/отсутствию трещиноватости. Для более корректной увязки и создания адекватной гидродинамической модели такие зависимости должны быть верифицированы посредством данных ГДИ для тех же продуктивных интервалов. На рисунке 3 представлен график описывающий взаимную увязку проницаемостей и пористостей коллекторов по керновым данным с данными ГДИ. Условно две группы точек с рисунка 1 были аппроксимированы двумя полиномами зеленого цвета. Синим цветом обозначены точки, полученные по результатам ГДИ тех же продуктивных интервалов. Некоторое совпадение можно отметить только по нижней зеленой линии (преимущественно

матричный поровый коллектор). Верхняя зеленая линия была аппроксимирована по керновым данным, отнесенным к поро-во-трещинному или трещинному типу коллектора. Как можно отметить, о каком либо совпадении значений проницаемости, полученных по ГДИ с зависимостью проницаемости от пористости по керновым данным говорить не приходится. Можно отнести к случайным, нежели закономерным.

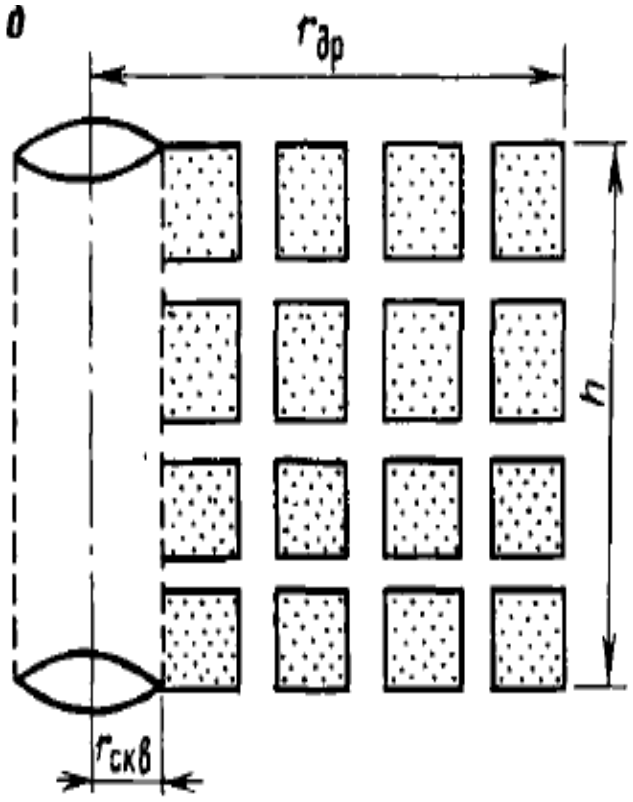


Рисунок 2. Модель трещиновато-порового коллектора Уоррена-Рута.
Наличие проницаемости матрицы и трещин.

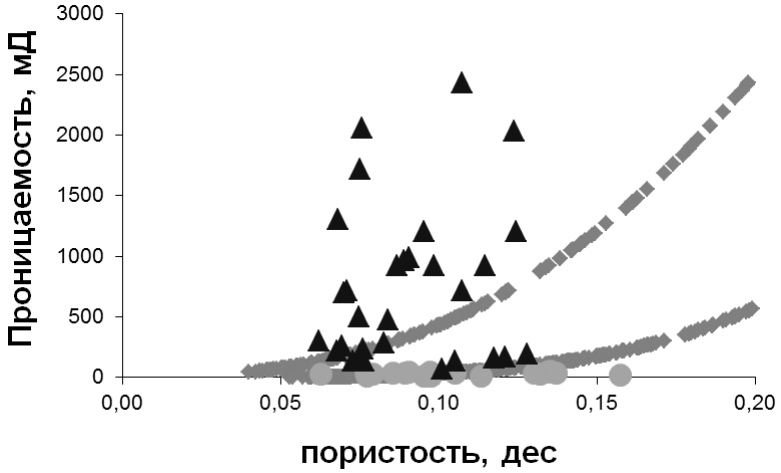


Рисунок 3. Сопоставленние данных ГДИ и керн-ГИС для пористого и трещиноватого типа коллекторов.
Месторождение нефти в карбонатных коллекторах поздне-девонского возраста.

При изучения материалов по петрофизическим зависимостям проницаемости от пористости рассматривались работы авторов, которые так или иначе классифицировали

карбонатные коллектора разных типов (рис. 4 и 5). Первый, наиболее очевидный, вывод по графикам на рисунках 4 и 5 – зависимости проницаемости от пористости имеют преимущественно субвертикальный (наклонный) характер. Зависимости для разных типов карбонатных коллекторов локализованы в группах по пористости. Вероятнее всего это связано с различиями в структуре пустотного пространства для разных объемов пористости, имеющей непосредственную связь с генезисом изучаемых нефтенасыщенных коллекторов.

Таким образом, петрофизическую зависимость керн-ГИС (рис. 3) и данные по проницаемости, полученные по ГДИ, необходимо проанализировать с целью выявления особенностей, позволяющих сгруппировать данные по пористости и проницаемости месторождения нефти в карбонатных коллекторах, по данным которого строился рис. №3. Особенность данных по проницаемости от пористости, рассчитанных по ГДИ скважин (вынесены на рисунок №3 синими точками) в том, что при первичной интерпретации ГДИ не принимались в полной мере особенности фильтрации нефти в скважину. Как уже упоминалось, в карбонатном коллекторе в зависимости от особенностей процесса формирования и влияния вторичных процессов, нефть в пласте и при попадании в скважину может фильтроваться не только через связанный объем пор, но и через систему трещин разного порядка (рис. №2 и №6). В пользу этого аргумента свидетельствуют результаты интерпретации промыслово-геофизических исследований скважин (рис. 7 и 8), по которым строились петрофизические зависимости (рис. 1 и 3). Главная особенность результатов ПГИ на разных режимах в том, что в зависимости от режима работы происходят значительные изменения в регистрируемых параметрах, а именно в локализации «работающих толщин». Рисунки №7 и №8 являются достаточным основанием полагать наличие преимущественной фильтрации в пласте через систему трещин разного порядка. В статье [1] данный механизм фильтрации нефти к скважине (интерпретационная модель) назывался «двойная проницаемость». С целью получения группировки значений проницаемости от пористости было принято решение переинтерпретировать результаты ГДИ с преимущественным предположением о наличии механизма фильтрации «двойная проницаемость». Переинтерпретация ГДИ с преимущественным предположением о наличии механизма фильтрации «двойная проницаемость» дала положительный результат в плане группировки точек на петрофизической зависимости проницаемости от пористости (рис. 9).

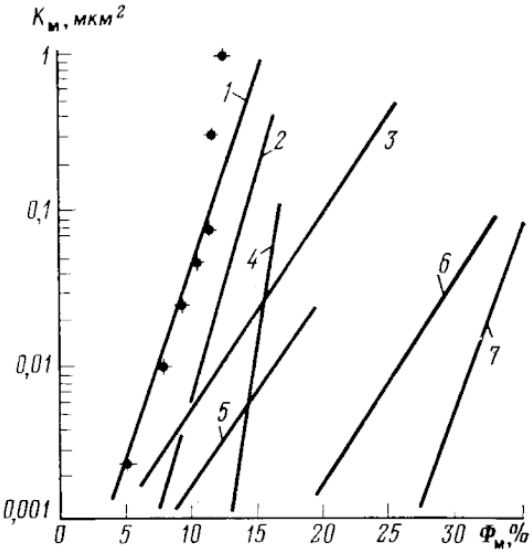


Рисунок 4. График зависимости Км—Фм для различных типов горных пород.
1 — рифовые известняки; 2 — оолитовые известняки; 3 — сахаровидные доломиты; 4 — хорошо сцементированные песчаники; 5 — известняки и доломиты с межгранулярной пустотностью; 6 — мелоподобные известняки; 7 — тонкозернистые песчаники [3]

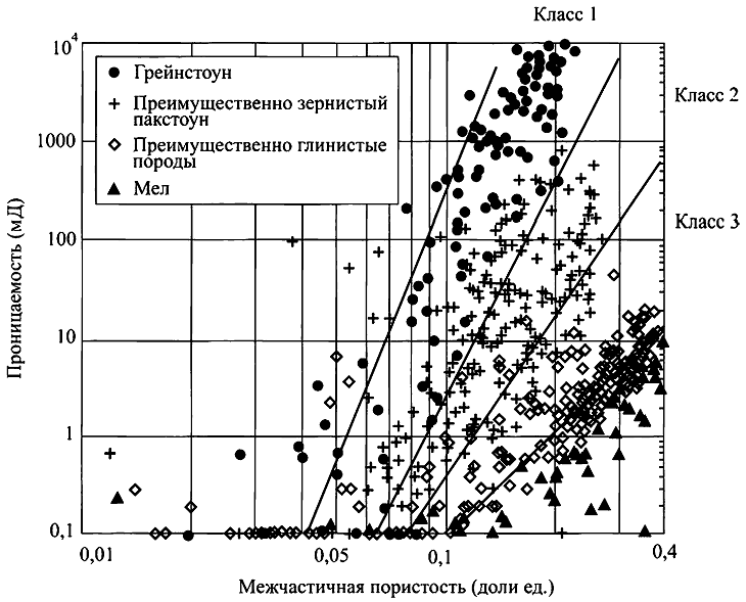


Рисунок 5. Зависимости между пористостью и проницаемостью для кавернозных известняков на примере одного из месторождений [4]

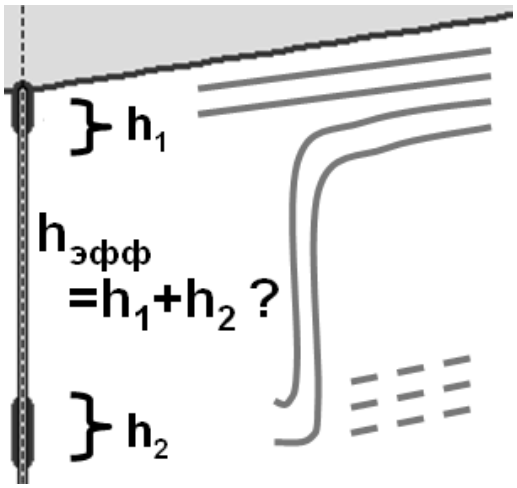


Рисунок 6. Схематичное описание вероятных путей поступления нефти в скважину по системе трещин из удаленных зон пласта

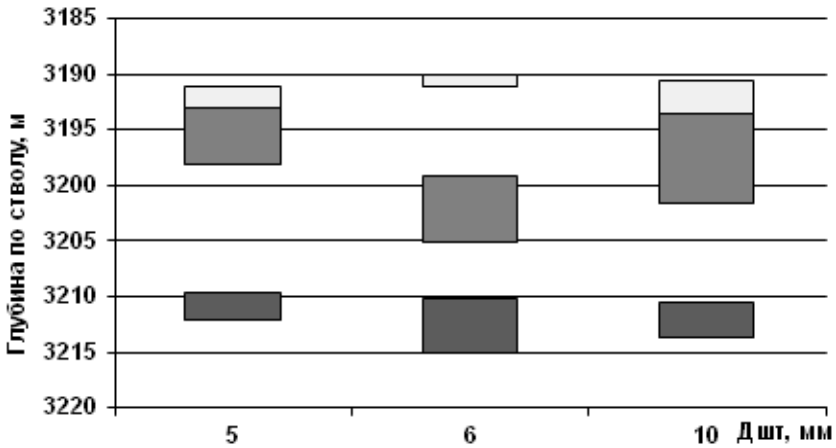


Рисунок 7. Локализация работающих толщин в зависимости от режима работы скважины, вскрывшей карбонатные нефтенасыщенные коллектора поздне-девонского возраста.

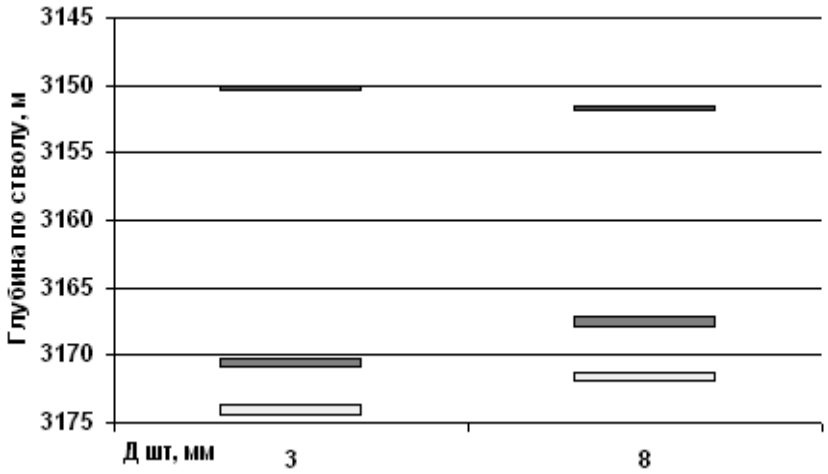


Рисунок 8. Локализация работающих толщин в зависимости от режима работы скважины, вскрывшей карбонатные нефтенасыщенные коллектора поздне-девонского возраста.

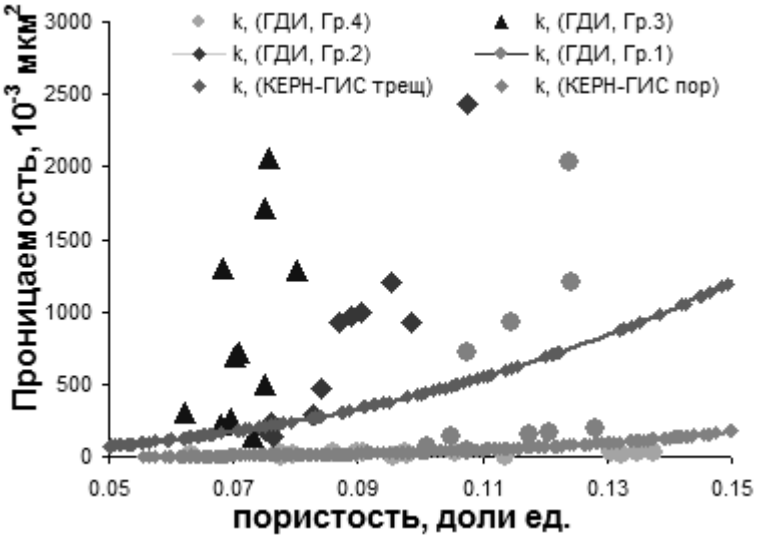


Рисунок 9. Распределение проницаемости от средневзвешенной по мощности (эффективной нефтенасыщенной толщине) пористости по данным ГДИ.

Как можно отметить, значения проницаемости от пористости на графике петрофизической зависимости локализовались в несколько групп (рис. 9). Субгоризонтальная группа оранжевого цвета фактически подтвердила линию петрофизической зависимости по данным КЕРН-ГИС для чисто порового типа коллектора. Следовательно, значение трещиноватости для этого типа коллектора минимально. Точки синего, красного и серого цветов фактически локализовались в 3 группы, имеющие разные пределы по пористости. Вероятнее всего для этих трех групп трещиноватость имеет преобладающее значение в части проницаемости. А петрофизическая зависимость для «трещиноватого» коллектора на основе корреляции данных керн-ГИС далеко не в полной мере отражает взаимозависимость структуры пустотного пространства, минералогического состава и значения проницаемости. Локализация значений проницаемости от пористости для точек синего, красного и серого цветов преимущественно соответствует классификациям карбонатных нефтенасыщенных коллекторов, опубликованных Т. Голф-Рахтом [3] и Дж. Лусиа [4].

Дальнейший анализ коллекторов, вскрытых скважинами, исследованными в том числе посредством ГДИ, будет проводиться преимущественно в направлении анализа данных каротажных диаграмм стандартного

каротажа в открытом стволе, а также в направлении анализа специальных методов каротажа (широкополосная акустика с регистрацией волны Стоунли). Главная цель дальнейшего анализа каротажа - подтверждение выделенных на петрофизической зависимости групп коллекторов.

3. Анализ каротажных диаграмм с целью идентификации отличительных признаков разных типов нефтенасыщенных карбонатных коллекторов.

3.1. Анализ каротажных диаграмм зарегистрированных в открытом стволе стандартным комплексом ГИС.

При изучении коллекторов позднедевонского возраста, вскрытых скважинами в Тимано-Печорском и других регионах были обнаружены особенности каротажных диаграмм, которые могут быть использованы как надежные идентификационные признаки.

Так при анализе разработки скважин, эксплуатирующих отложения позднего девона, были идентифицированы особенности отношения значений пористости, рассчитанной для таких коллекторов при интерпретации данных акустического и нейтрон-нейтронного каротажа [2].

В частности, производился расчет материального баланса [5] с целью выявления режима дренирования нефтенасыщенных карбонатных коллекторов. Одним из выводов был факт разной реакции нефтенасыщенных колекторов на приток подстилающей воды из законтурной области. Разный наклон графиков на рисунке 10 объясняется разными петрофизическими и литологическими характеристиками пласта коллектора.

Каротажные диаграммы на рисунках 11 и 12 иллюстрируют различия в литологических и петрофизических свойствах карбонатных коллекторов с преимущественно трещинным характером проницаемости (доломитизированный коллектор) и наличием проницаемости пор (известняк). Следующим идентификационным признаком, позволяющим определить тип карбонатного нефтенасыщенного коллектора, является различие в данных нейтронного гамма каротажа для нефтенасыщенных интервалов. С целью выявления диагностических признаков по данным стандартного каротажа в открытом стволе по всем методам были построены гистограммы для выявления статистических закономерностей регистрируемых значений по каждому виду каротажа. По всем основным методам ГИС, за исключением нейтронного гамма каротажа (НГК), гистограммы не показали значительных различий в накопленных значениях параметров для выделенных групп коллекторов согласно рисунка №9. Гистрограмма накопленных частот, построенная по данным НГК по выделенным группам коллекторов показала значительные отличия для группы значений, соответствующих поровому типу коллектора (рис. 13, группа 4).

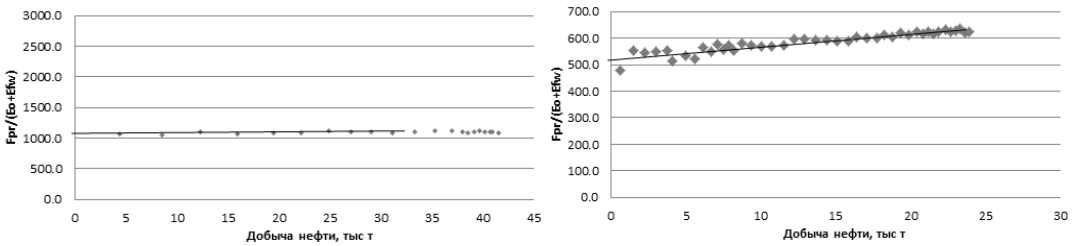


Рисунок 10. Диагностический график режима дренирования пласта в районе скважин № XX9 и № XX6.

Для пласта коллектора 4-й группы (чисто поровый коллектор) частота встречаемости значений параметра находится преимущественно в пределах от 2 до 6 условных единиц. На фоне распределения частот для остальных групп коллекторов такое распределение может быть использовано как устойчивый идентификационный признак.

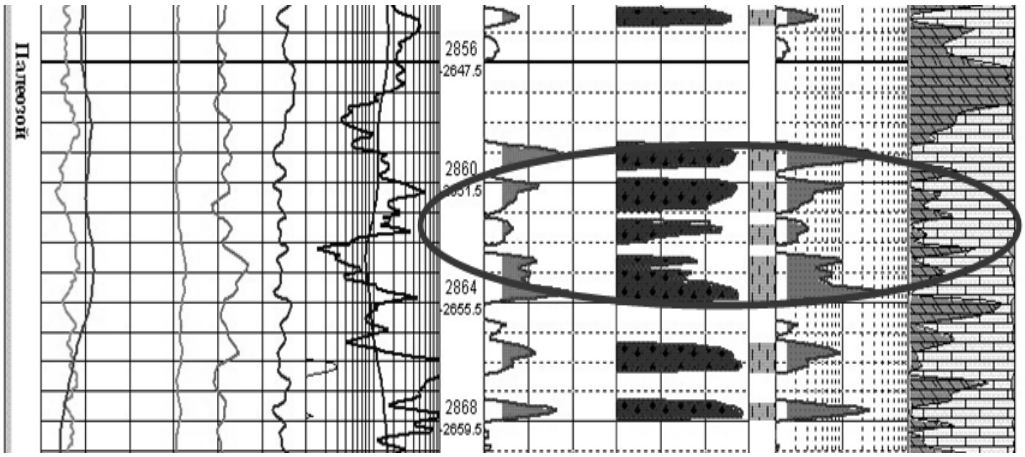


Рисунок 11. Выкопировка из планшета по комлексным ГИС скв № ХХ6.
Преимущественно нефтенасыщенные известняки (разница пористости по АК и ННК менее 6%)

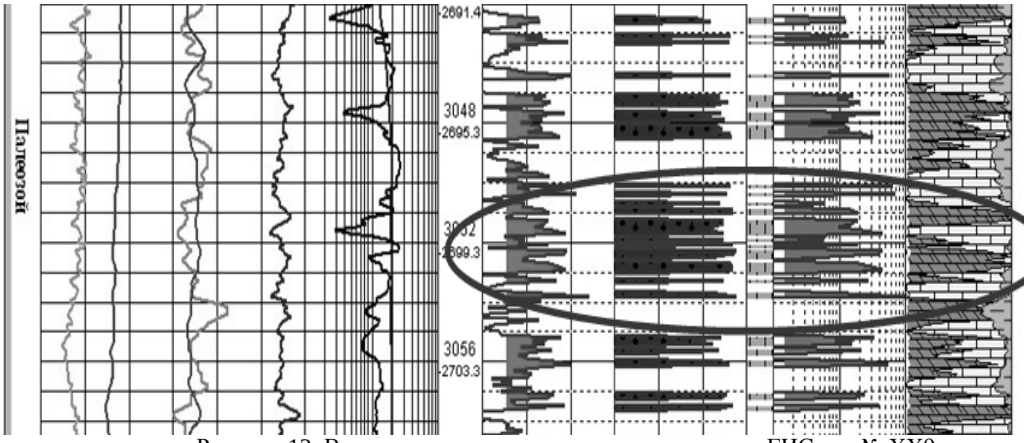


Рисунок 12. Выкопировка из планшета по комлексным ГИС скв № ХХ9.
Преимущественно нефтенасыщенные доломиты (разница пористости по АК и ННК более 6%)

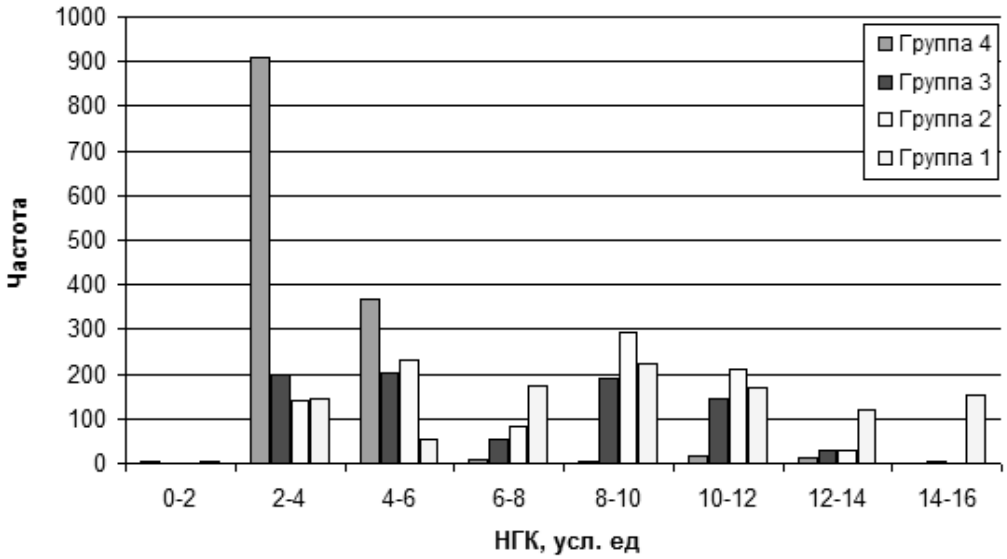


Рисунок 13. Гистограммы распределения частоты встречаемости параметров

3.2. Анализ каротажных диаграмм зарегистрированных в открытом стволе специальными методами ГИС.

В рамках анализа каротажных диаграмм специальных методов ГИС (широкополосная акустика, регистрация времени пробега волны Стоунли, [1]) для коллекторов с преимущественно трещинным типом проницаемости (группы выделенные синим, красным и серыми цветами на рисунке 9) были обнаружены особенности, позволяющие относить к тому или иному типу выделенные нефтенасыщенные коллектора.

По группе 1 (синий цвет на рис. 9) кривая DTL (волна Стоунли) (на фоне роста времени пробега продольной и поперечной волн) имеет участки резких обратных «рывков» (рис. 14). По группе 2 (красный цвет на рис. 9) кривая DTL (волна Стоунли) (на фоне роста времени пробега продольной и поперечной волн) имеет сопоставимый со стандартным акустическим каротажом рост и сходный характер изменений (рис. 15). По группе 3 (серый цвет на рис. 9) кривая DTL (волна Лэмба-Стоунли) (на фоне роста времени пробега продольной и поперечной волн) имеет участки резких обратных «рывков», но эти «рывки» имеют большую амплитуду, чем для точек 1-й группы (рис. 14).

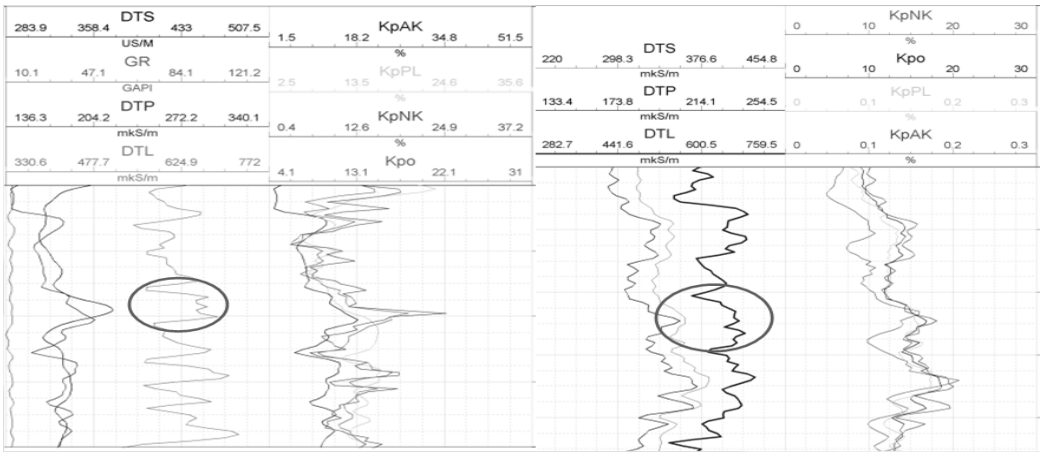


Рисунок 14. Характер изменения волны Лэмба-Стоунли для пластов, проницаемости по которым попали в группу 1

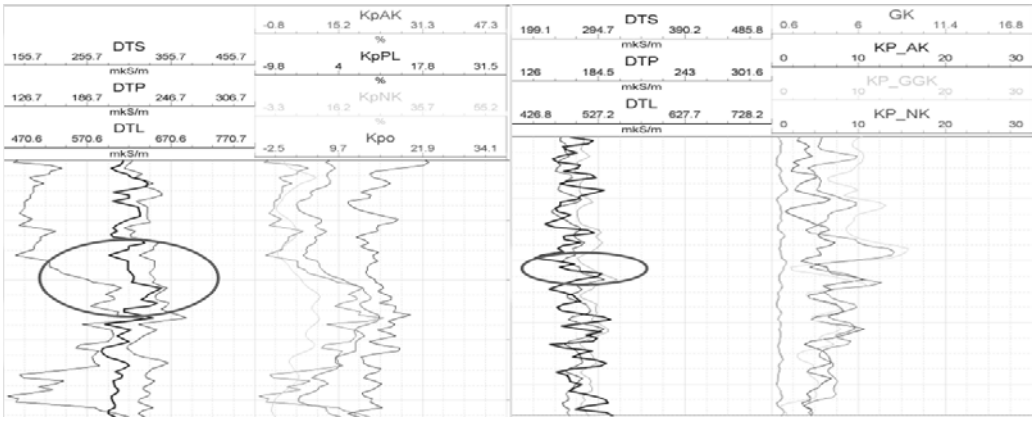


Рисунок 15. Характер изменения волны Лэмба-Стоунли для пластов, проницаемости по которым попали в группу 2

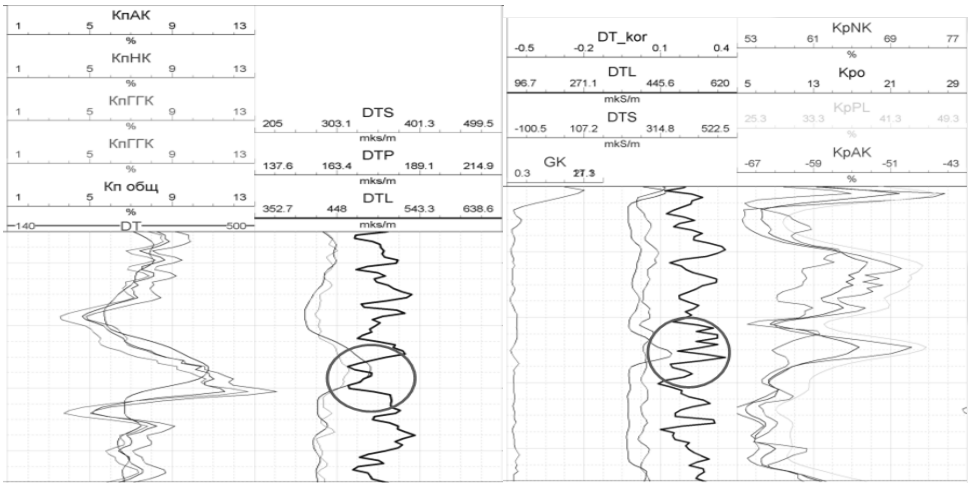


Рисунок 16. Характер изменения волны Лэмба-Стоунли для пластов, проницаемости по которым попали в группу 3

Заключение

1. Карбонатные коллектора разного литологического состава и петрофизических свойств имеют значительные отличия в значениях пористости по результатам интерпретации акустического каротажа и нейтронного гамма каротажа.

2. Карбонатные нефтенасыщенные коллектора с представленным выше разделением на группы имеют значительные различия по нейтронному гамма каротажу.

3. Волна Стоунли имеет значительные качественные отличия по форме диаграмм для разных типов карбонатных нефтенасыщенных коллекторов.

4. Перечисленные выше качественные признаки позволяют с высокой долей вероятности разделять карбонатные нефтенасыщенные коллектора посредством анализа данных ГИС открытого ствола.

.

Выделенные на рисунке 9 группы коллекторов с однозначным разделением по данным ГИС позволяют говорить о корректности предлагаемой петрофизической зависимости проницаемости от пористости.

5. Предложенная петрофизическая зависимость проницаемости от пористости соответствует концепции классификации коллекторов других авторов [3] и [4]. Таким образом, выделяемые типы коллекторов можно разделять в том числе по параметру пористости, т.к. каждая из групп локализована в своих рамках.

6. Анализ карбонатных нефтенасыщенных коллекторов по каротажным диаграммам согласно перечисленных выше идентификационных признаков с высокой вероятностью позволит производить разделение исследуемых коллекторов.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта №18-07-00673 А

Carbonate oil saturated reservoir type classification methodology development

A.A. Kolevatov, Y.M. Steinberg, Y.B. Chen-len-son,
A.A. Egorov, A.M. Giatsintov, A.G. Dyachenko

Abstract: Article overviews approach to porous media structure identification of carbonate oil saturated reservoir. Carbonate oil saturated reservoir classification aspects also envisaged. Article overviews reservoir type identification methodology development through regularities identification based on well test data, open-hole logs data and cased-hole production logging data.

Keywords: carbonate oil saturated reservoirs, open-hole logging, well testing, cased-hole logging, carbonate oil saturated reservoir type identification methodology.

Литература

1. А.А. Колеватов, Ю.М. Штейнберг, Ю.Б. Чен-лен-сон, А.Г. Дяченко. Классификация полей проницаемости карбонатных трещиноватых коллекторов посредством специальных методов ГИС и ГДИ. ТРУДЫ НИИСИ РАН, т. 8, № 3 (2018).
2. А.А.Колеватов, Ю.М.Штейнберг, А.Г.Дяченко. Определение режима дренирования и уточнение литологии сложнопостроенного трещиноватого коллектора на основе анализа промысловых данных. // Труды конференции «Трофимуковские чтения - 2017», стр. 210-213. Новосибирск, ИНГГ СО РАН, ОИТ, 630090, Новосибирск, просп. Академика Коптюга, 3.
3. Т.Д.Голф-Рахт. Основы нефтепромысловой геологии и разработки трещиноватых коллекторов. М.: Недра, 1986.
4. Ф.Дж.Лусиа. Построение геолого-гидродинамической модели карбонатного коллектора: интегрированный подход. — М.-Ижевск: НИЦ «Регулярная и хаотическая динамика», Ижевский институт компьютерных исследований, 2010. — 384 с.
5. L.P.Dake. Fundamentals of reservoir engineering / L.P. Dake – NY.: ELSEVIER SCIENCE PUBLISHING COMPANY B.V., 1978. – 453 c.

Оценка чувствительности математических моделей разработки нефтяных месторождений к исходным данным

И.В. Афанаскин[1], Н.П. Ефимова[2], Д.В. Солопов[3], П.В. Ялов[4]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия

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

Аннотация. В работе подробно описывается математическая модель трехфазной фильтрации нефти, газа и воды. Для этой модели приводится полностью неявная численная схема на сетке Вороного. Анализируются исходные данные для моделирования, их достоверность и пути получения. Рассматриваются вопросы адаптации моделей. На примере реальной модели одного из месторождений Западной Сибири с помощью коммерческого симулятора Rubis (Kappa Engineering) анализируется чувствительность математической модели трехфазной фильтрации к исходным данным. Даются рекомендации по адаптации моделей на основе анализа влияния точности параметров модели на результаты расчетов.

Ключевые слова: теория фильтрации, математическое моделирование, разработка нефтяных месторождений, анализ чувствительности.

Введение

Разработка нефтяных месторождений требует высоких капитальных и операционных затрат при достаточно больших сроках окупаемости (десятки лет). Стоимость бурения одной скважины на глубину 2500 м. составляет порядка 1,6 млн. долларов (в зависимости от региона). Стоимость проведения одной операции по интенсификации добычи нефти с помощью гидроразрыва пласта составляет порядка 100 тыс. долларов. Поэтому любы затратные мероприятия, проводимые на скважине, как то: бурение бокового ствола, проведение гидроразрыва пласта (не говоря уже о бурении самой скважины) сопровождаются численным гидродинамическим моделированием. Кроме того, моделирование используется при финансовом планировании, планировании регулирования разработки, для контроля разработки, при создании проектных документов и пр. В том числе, в составе комплекса различных видов исследований, математическое моделирование используется и при выявлении невыработанных зон на нефтяных месторождениях и подсчёта остаточных запасов нефти. При этом используемые математические модели могут быть очень сложными, учитывать множество различных физико-химических процессов. Такие модели требуют большое количество разнообразных исходных данных. Поэтому актуальным является вопрос оценки чувствительности математических моделей разработки нефтяных месторождений к исходным данным, которому посвящена эта работа.

1. Математическая модель трехфазной фильтрации нефти, газа и воды

Для моделирования разработки нефтяных месторождений с помощью заводнения при забойном (и пластовом) давлении ниже давления насыщения нефти газом используется модель Маскета-Мереса. Она предполагает трехфазную изотермическую фильтрацию нефти, газа и воды, нефть и вода являются несмешивающимися жидкостями, газ растворим в нефти и нерастворим в воде, нефть испаряться в газ не может.

Уравнения сохранения объемов фаз в поверхностных условиях в рамках модели Маскета-Мереса имеют вид [1, 4, 5, 8]:

$$\frac{\partial}{\partial t}\left(\frac{mS_\alpha}{B_\alpha}\right)+div\left(\frac{\vec{w}_\alpha}{B_\alpha}\right)=-\bar{q}_\alpha,\ \alpha=o,w, \quad (1)$$

$$\frac{\partial}{\partial t}\left[m\left(\frac{S_g}{B_g}+\frac{R_s S_o}{B_o}\right)\right]+div\left(\frac{\vec{w}_g}{B_g}+\frac{\vec{w}_o R_s}{B_o}\right)= \quad (2)$$
$$=-\bar{q}_{fg}-\bar{q}_o R_s$$

где m - пористость пласта, S_α - насыщенность пласта фазой $\alpha=o,w,g$, B_α - объемный коэффициент фазы α, $\vec{w}_\alpha$ - скорость фильтрации фазы α, $\bar{q}_\alpha$ - плотность источника (стока) фазы α, R_s - растворимость газа в

нефти, $\bar{q}_{fg}$ - плотность источника (стока) свободного (нерастворенного в нефти) газа.

В качестве уравнений сохранения количества движения запишем обобщенный закон Дарси [1, 4, 5, 8]:

$$\vec{w}_\alpha = -\frac{kk_{r\alpha}}{\mu_\alpha}\left[grad\left(P_\alpha\right) - g\rho_\alpha grad\left(z\right)\right], \quad \alpha = o, w, g, \tag{3}$$

где k - абсолютная проницаемость пласта, $k_{r\alpha}$ - относительная проницаемость фазы α, μ_α - динамическая вязкость фазы α, P_α - давление в фазе α, g - ускорение свободного падения, ρ_α - плотность фазы α, z - вертикальная координата от какой-либо фиксированной горизонтальной плоскости.

Для удобства примем в качестве пластового давления давление в нефтяной фазе $P \equiv P_o$.

Система уравнений (1)-(3) на самом деле содержит шесть уравнений (по одному уравнению сохранения объема и сохранения количества движения для каждой из трех фаз). Подстановка уравнений для скорости фильтрации (3) в уравнения сохранения объемов (1) и (2) сокращает количество уравнений до трех:

$$\frac{\partial}{\partial t}\left(\frac{mS_\alpha}{B_\alpha}\right) - div\left(\frac{kk_{r\alpha}}{\mu_\alpha B_\alpha}\left[grad\left(P_\alpha\right) - g\rho_\alpha grad\left(z\right)\right]\right) = -\bar{q}_\alpha \quad \alpha = o, w, \tag{4}$$

$$\frac{\partial}{\partial t}\left[m\left(\frac{S_g}{B_g} + \frac{R_s S_o}{B_o}\right)\right] - div\left\{\frac{kk_{rg}}{\mu_g B_g}\left[grad\left(P_g\right) - g\rho_g grad\left(z\right)\right] + \frac{R_s kk_{ro}}{\mu_o B_o}\left[grad\left(P_o\right) - g\rho_o grad\left(z\right)\right]\right\} = -\bar{q}_{fg} - \bar{q}_o R_s \tag{5}$$

В результате мы имеем недоопределенную систему (4), (5) - три уравнения и шесть неизвестных: три давления и три насыщенности. Замыкающими соотношениями для системы (4), (5) являются:

$$P_o - P_w = P_{cow}\left(S_w\right), \tag{6}$$

$$P_g - P_o = P_{cgo}\left(S_g\right), \tag{7}$$

$$S_o + S_w + S_g = 1, \tag{8}$$

где P_{cow} - капиллярное давление в системе нефть-вода, P_{cgo} - капиллярное давление в системе газ-нефть.

Вместо уравнений состояния используются задаваемые таблично или в виде функций с известными коэффициентами зависимости:

$$B_\alpha = B_\alpha\left(P\right), \quad \mu_\alpha = \mu_\alpha\left(P\right), \quad \alpha = o, g, \quad R_s = R_s\left(P\right), \tag{9}$$

кроме того

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

где B_{w0} - объемный коэффициент воды при начальном давлении P_0, C_w - сжимаемость воды,

$$\rho_\alpha\left(P\right) = \rho_{\alpha STC} / B_\alpha\left(P\right), \ \alpha = o, w, g, \tag{11}$$

где $\rho_{\alpha STC}$ - плотность фазы $\alpha = o, w, g$ в стандартных условиях.

Для пористости используется соотношение:

$$m\left(P\right) = m_0\left[1 + C_r\left(P - P_0\right)\right], \tag{12}$$

где m_0 - пористость при начальном давлении P_0, C_r - сжимаемость горной породы.

Относительные фазовые проницаемости задаются в виде зависимостей для двухфазных систем:

$$k_{rw} = k_{rw}\left(S_w\right), \ k_{row} = k_{row}\left(S_w\right), \tag{13}$$

$$k_{rg} = k_{rg}\left(S_g\right), \ k_{rog} = k_{rog}\left(S_g\right). \tag{14}$$

Относительная фазовая проницаемость (ОФП) для нефти при трехфазной фильтрации является функций двух насыщенностей и задается по какой-нибудь из известных моделей. В качестве примера рассмотрим вторую модель Стоуна [1, 4]:

$$k_{ro}\left(S_w, S_g\right) = k_{rocw}\left\{\left[\frac{k_{row}\left(S_w\right)}{k_{rocw}} + k_{rw}\left(S_w\right)\right] \times \left[\frac{k_{rog}\left(S_g\right)}{k_{rocw}} + k_{rg}\left(S_g\right)\right] - k_{rw}\left(S_w\right) - k_{rg}\left(S_g\right)\right\} \tag{15}$$

где $k_{rocw} = k_{row}\left(S_{wcr}\right) = k_{rog}\left(S_g = 0\right)$ - ОФП по нефти при насыщенности связанной водой S_{wcr} в отсутствии газа.

Для решения системы (4)-(5) необходимо задать начальные условия:

$$P = P\left(x, y, z, t = 0\right), \quad S_o = S_o\left(x, y, z, t = 0\right), \quad S_w = S_w\left(x, y, z, t = 0\right). \tag{16}$$

В качестве граничных условий обычно используются условия непротекания на внешних границах либо условия постоянного давления. Внутренние граничные условия (скважины) моделируются с помощью источниковых слагаемых. Приток жидкости из законтурной области также часто моделируются с помощью источниковых слагаемых [1].

Работа l-ого участка ствола скважины моделируются с помощью следующего выражения [1, 4, 5, 8]:

$$q_{\alpha l} = WI_l \lambda_{\alpha l}\left[P_l - P_w - \bar{\gamma}\left(D_l - D_{ref}\right)\right],$$
$$q_\alpha = \sum_l q_{\alpha l}, \qquad (17)$$

где WI_l - индекс скважины, зависящий от фильтрационно-емкостных свойств и геометрических параметров расчетной сетки, $\lambda_{\alpha l}$ - подвижность фазы α, P_l - пластовое давление в зоне l-ого участка ствола скважины, P_w - забойное давление в скважине на опорной глубине D_{ref}, $\bar{\gamma}$ - средний вес жидкости в стволе скважины, D_l - глубина l-ого участка ствола скважины.

Построим сетку Вороного в горизонтальной плоскости (плоскости пласта), рис. 1. Для получения трехмерной сетки - сетку, построенную в плане, копируем для разных слоев, изменяя при этом глубину залегания кровли ячеек и их толщину. Пронумеруем все ячейки. Обозначим i-номер ячейки трехмерной сетки, а n - номер шага по времени.

Аппроксимируем систему уравнений (4), (5) на сетке Вороного с помощью полностью неявной численной схемы:

$$\frac{V_i}{\Delta t^{n+1}}\left[\left(\frac{mS_o}{B_o}\right)_i^{n+1} - \left(\frac{mS_o}{B_o}\right)_i^{n}\right] +$$
$$+ \sum_{j=1}^{N_i}\left[k_{ij}\left(\frac{k_{ro}}{\mu_o B_o}\right)_{ij}^{n+1} F_{ij} \times\right. \qquad (18)$$
$$\left.\times \frac{P_i^{n+1} - P_j^{n+1} - \rho_{oij}^{n+1} g\left(z_i - z_j\right)}{L_{ij}}\right] = -q_{oi}^{n+1}$$

$$\frac{V_i}{\Delta t^{n+1}}\left[\left(\frac{mS_w}{B_w}\right)_i^{n+1} - \left(\frac{mS_w}{B_w}\right)_i^{n}\right] +$$
$$+ \sum_{j=1}^{N_i}\left[k_{ij}\left(\frac{k_{rw}}{\mu_w B_w}\right)_{ij}^{n+1} F_{ij} \times\right.$$
$$\left.\times \frac{P_i^{n+1} - P_j^{n+1} - P_{cowi}^{n+1} + P_{cowj}^{n+1} - \rho_{wij}^{n+1} g\left(z_i - z_j\right)}{L_{ij}}\right] =$$
$$= -q_{wi}^{n+1} \qquad (19)$$

$$\frac{V_i}{\Delta t^{n+1}}\left\{\left[m\left(\frac{S_g}{B_g} + \frac{R_s S_o}{B_o}\right)\right]_i^{n+1} - \left[m\left(\frac{S_g}{B_g} + \frac{R_s S_o}{B_o}\right)\right]_i^{n}\right\} +$$
$$+ \sum_{j=1}^{N_i}\left[k_{ij}\left(\frac{k_{rg}}{\mu_g B_g}\right)_{ij}^{n+1} F_{ij} \times\right.$$
$$\times \frac{P_i^{n+1} - P_j^{n+1} + P_{cgoi}^{n+1} - P_{cgoj}^{n+1} - \rho_{gij}^{n+1} g\left(z_i - z_j\right)}{L_{ij}} +$$
$$+ k_{ij}\left(\frac{R_s k_{ro}}{\mu_o B_o}\right)_{ij}^{n+1} F_{ij} \times$$
$$\left.\times \frac{P_i^{n+1} - P_j^{n+1} - P_{cowi}^{n+1} + P_{cowj}^{n+1} - \rho_{oij}^{n+1} g\left(z_i - z_j\right)}{L_{ij}}\right] =$$
$$= -q_{fgi}^{n+1} - \left(q_o R_s\right) \qquad (20)$$

где V_i - объем i-ой ячейки сетки, Δt^{n+1} - переменный шаг по времени [1, 4], N_i - количество соседних ячеек для i-ой ячейки сетки, j - номер соседней ячейки сетки, L_{ij} - расстояние между центрами соседних ячеек, F_{ij} - площадь общей грани соседних ячеек i и j.

Средняя проницаемость k_{ij} между соседними ячейками i и j обычно вычисляется как среднее геометрическое, ОФП $\left(k_{r\alpha}\right)_{ij}^{n+1}$ взвешивается вверх по потоку, вязкость $\left(\mu_\alpha\right)_{ij}^{n+1}$, объемный коэффициент $\left(B_\alpha\right)_{ij}^{n+1}$ и плотность $\left(\rho_\alpha\right)_{ij}^{n+1}$ вычисляются как среднее арифметическое.

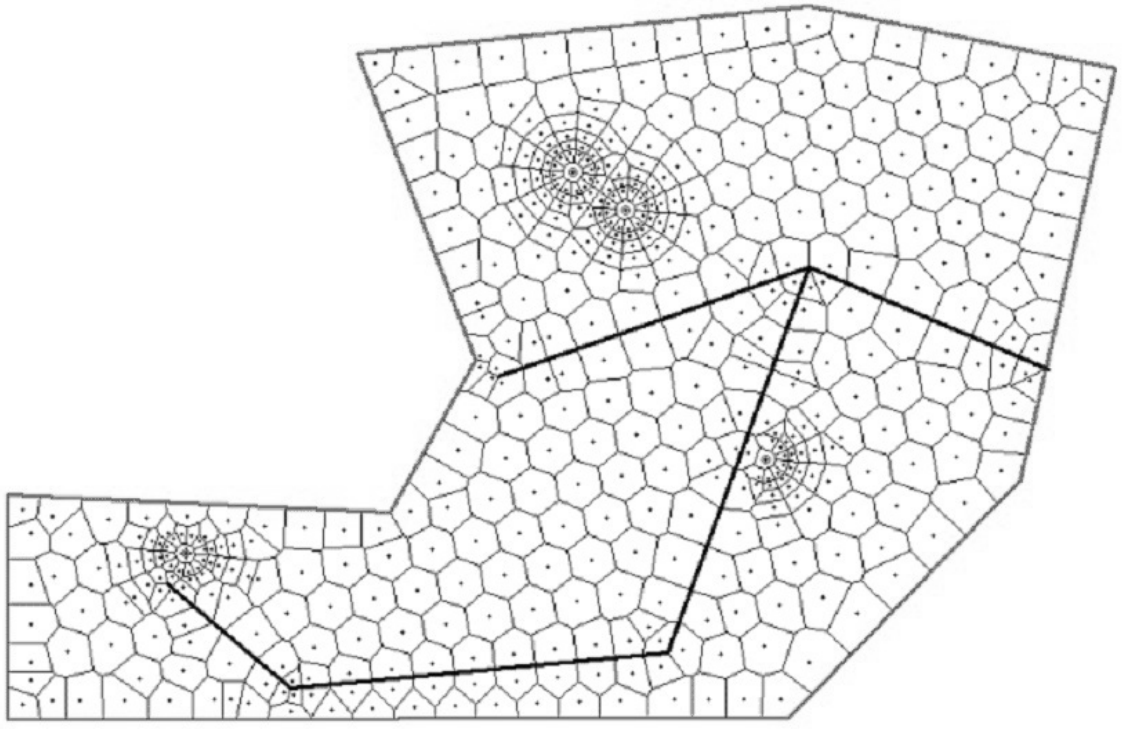


Рис. 1. Пример сетки Вороного [10]

Система (18)-(20) является системой нелинейных алгебраических уравнений, где на каждом временном шаге 3*N* уравнений (18)-(20) и 4*N* неизвестных (P_i^{n+1}, S_{oi}^{n+1}, S_{wi}^{n+1}, S_{gi}^{n+1}), *N* - количество ячеек. Для замыкания системы используется чсиленная аппроксимация выражения (8):

$$S_{oi}^{n+1} + S_{wi}^{n+1} + S_{gi}^{n+1} = 1. \qquad (21)$$

Средняя проницаемость k_{ij} между соседними ячейками i и j обычно вычисляется как среднее геометрическое, ОФП $\left(k_{r\alpha}\right)_{ij}^{n+1}$ взвешивается вверх по потоку, вязкость $\left(\mu_\alpha\right)_{ij}^{n+1}$, объемный коэффициент $\left(B_\alpha\right)_{ij}^{n+1}$ и плотность $\left(\rho_\alpha\right)_{ij}^{n+1}$ вычисляются как среднее арифметическое.

Система (18)-(20) является системой нелинейных алгебраических уравнений, где на каждом временном шаге 3*N* уравнений (18)-(20) и 4*N* неизвестных (P_i^{n+1}, S_{oi}^{n+1}, S_{wi}^{n+1}, S_{gi}^{n+1}), *N* - количество ячеек. Для замыкания системы используется чсиленная аппроксимация выражения (8):

$$S_{oi}^{n+1} + S_{wi}^{n+1} + S_{gi}^{n+1} = 1. \qquad (22)$$

Нелинейные слагаемые в системе (18)-(20) линеаризуются методом Ньютона. Например, источниковый член $q_{\alpha i}^{n+1}$, зависящий от давления и насыщенностей, линеаризуется следующим образом:

$$q_{\alpha i}^{n+1} \approx q_{\alpha i}^{n} + \left(\frac{\partial q_\alpha}{\partial P}\right)_i^n \left(P_i^{n+1} - P_i^n\right) + \left(\frac{\partial q_\alpha}{\partial S_o}\right)_i^n \left(S_{oi}^{n+1} - S_{oi}^n\right) + \left(\frac{\partial q_\alpha}{\partial S_w}\right)_i^n \left(S_{wi}^{n+1} - S_{wi}^n\right) \qquad (23)$$

2. Исходные данные для моделей, их достоверность и адаптация моделей

Анализируя систему уравнений (18)-(20) получим следующий список исходных данных для моделирования:

1. Данные о вычислительной сетке - количество блоков (ячеек), их форма, размеры, взаимное расположение и пр.
2. Фильтрационно-емкостные свойства пласта для каждого блока сетки: абсолютная проницаемость (возможно – зависящая от направления потока), доля коллектора, пористость при начальном давлении.
3. PVT-параметры флюидов: вязкость и объемный коэффициент нефти и газа в зависимости от давления; растворимость газа в

нефти в зависимости от давления; максимальное давление насыщения нефти газом; сжимаемость, объемный коэффициент и вязкость воды; сжимаемость породы пласта; плотность нефти, газа и воды при стандартных условиях.

4. ОФП и капиллярные давления как функции соответствующих насыщенностей.
5. Начальное распределение давления и насыщенности.
6. Условия на внешних границах; для залежей, связанных с активной водоносной областью: объем законтурной водоносной области, площадь и зона контакта, продуктивность законтурной области, давление в ней, ее сжимаемость.
7. Расположение скважин, их траектория и интервалы перфорации, продуктивность скважин во времени, история работы скважин.

Для получения информации о пласте и насыщающих его флюидах применяют следующие основные методы исследований:

1. Геофизические исследования скважин (ГИС) – выделение интервалов коллектора во всей толщине пласта, определение эффективной толщины (доли коллектора), насыщенности, пористости и пр.
2. Гидродинамические исследования скважин (ГДИ) – определение проницаемости пласта, пластового давления и температуры, состояния околоскважинной зоны, структуры порового пространства, расположения разломов (внутренних непроницаемых границ), коэффициентов продуктивности и приемистости скважин, степени взаимовлияния скважин, контроль за разработкой и пр.
3. Промыслово-геофизические исследования скважин (ПГИ) – определение работающих интервалов пласта в скважинах, контроль целостности эксплуатационной колонный и цементного камня, определение интервалов обводнения скважин пр.
4. Исследования керна – определение открытой пористости, абсолютной проницаемости, ОФП, капиллярного давления, минерального состава пород, гранулометрического состава пород, построение корреляции между пористостью и проницаемостью и пр.
5. Трассерные исследования – определение неоднородности пласта по проницаемости, взаимовлияния скважин, степени охвата пласта заводнением и пр.
6. Сейсмические исследования – определение глубины кровли и подошвы пласта в виде карты, поиск крупных разломов.
7. PVT-исследования флюидов – определение вязкости, плотности, объемного коэффициента и химического состава флюидов, давления насыщения нефти газом, растворимости газа в нефти и пр.

Исходя из промыслового опыта авторов, по некоторым техническим и технологическим причинам наименее достоверными параметрами являются:

1. Абсолютная проницаемость.
2. Объем фаз в пласте, т.е. произведение пористости на долю коллектора и на насыщенность. Геометрический объем породы при этом мы опускаем.
3. Начальное пластовое давление.
4. Функции ОФП и капиллярного давления.
5. Параметры водоносной области.

В 999 случаях из 1000 данные по размерам и активности водоносной области отсутствуют и подбираются при адаптации модели.

Кроме того, по промысловым причинам часто бывает недостоверной история работы скважин, но это связано с человеческим фактором и рассматриваться далее не будет.

Кроме того, различные источники информации о пласте дублируют друг друга, но они дают информацию о параметрах в разных пространственных масштабах (от 10^{-3} м для шлифов или керна до 10 см - 1 м для ГИС, 100-500 м. для ГДИ и нескольких километров для сейсмических данных), что осложняет их согласование, порой – до полной невозможности.

Кроме того, большинство методов изучения пласта дают или информацию о параметрах вблизи стенки скважины или средние параметры в большом объеме пласта, существенно превышающем одну расчетную ячейку.

Поэтому при построении трехмерной модели свойств пласта в начальный момент времени (так называемой геологической модели) после межскважинной корреляции (прослеживания и сопоставления идентичных пропластков в соседних скважинах) активно используются различные методы интерполяции [2, 3, 6, 7] для получения трехмерного распределения свойств.

Геологическую модель с включенными в нее PVT-свойствами, кривыми ОФП, историей работы скважин и рассчитанными динамическими показателями (давление, насыщенность и дебиты фаз, изменяющиеся во времени) называют гидродинамической или фильтрационной моделью.

Учитывая сложившиеся обстоятельства - в нефтепромысловой практике обычно используется следующая этапность математического моделирования разработки месторождений:

1. Создание геологической модели.
2. Масштабирование геологическое модели (апскейлинг) – по необходимости, при невозможности проведения фильтрационных расчетов на мелкой геологической сетке по причине нехватки вычислительных мощностей или несовершенства имеющегося в наличии программного обеспечения.
3. Создание гидродинамической (фильтрационной модели).
4. Адаптация (подгонка) гидродинамической модели – совмещение расчетных и фактических показателей разработки на имеющемся историческом периоде.
5. Прогнозные расчеты показателей разработки объекта по гидродинамической модели для принятого набора вариантов.
6. Уточнение модели на основании сопоставления прогнозных и фактических показателей разработки (по истечению некоторого времени после реализации п. 5, при накоплении новых данных), новых результатов исследований старых и вновь пробуренных скважин. Иногда в результате получения новых данных приходится возвращаться к п. 1 и заново создавать геологическую модель.

В литературе уделяется особое внимание адаптации модели – подгонке расчетных и фактических показателей разработки на историческом отрезке времени путем модификации параметров модели, что вызвано низкой точностью исходных данных. Существуют некоторые рекомендации и схемы этапов адаптации, рис. 2, и алгоритмов адаптации, рис. 3.

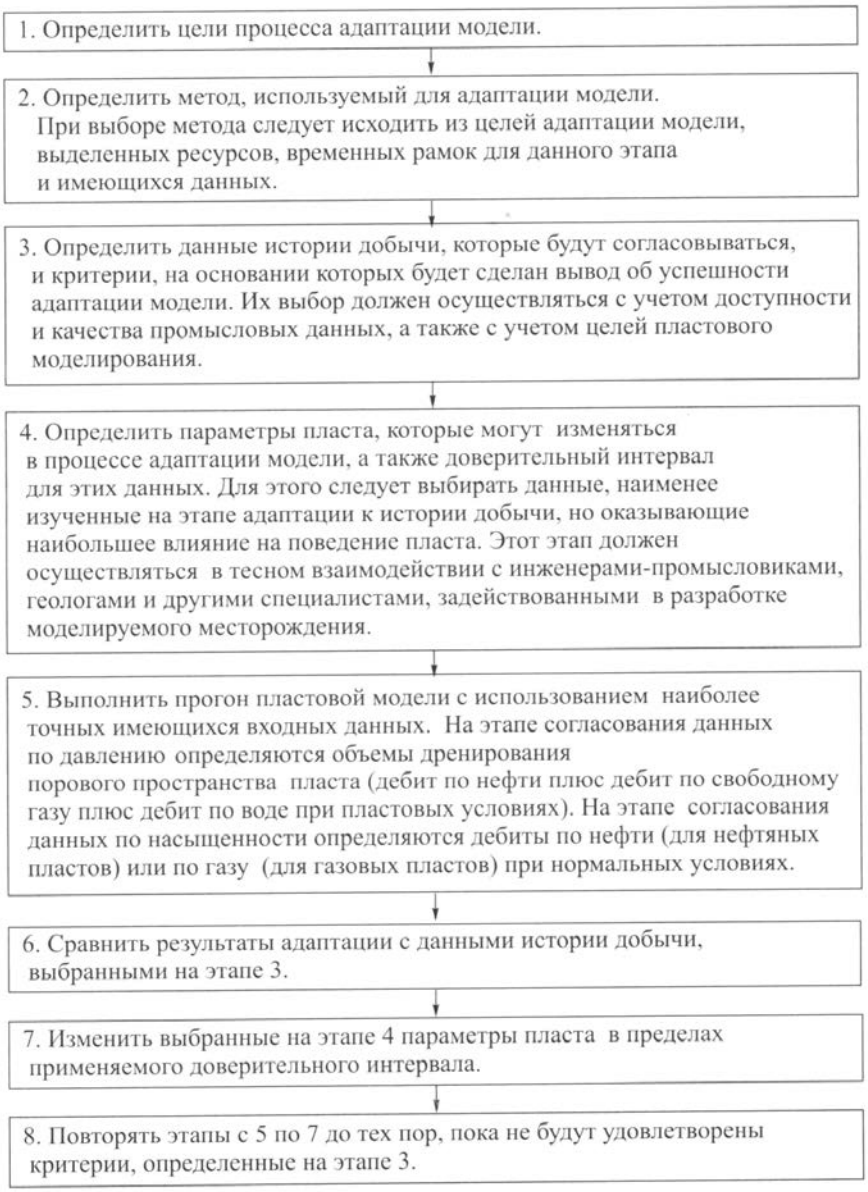


Рис. 2. Этапы адаптации гидродинамической модели [8]

Рис. 3. Алгоритм ручной адаптации модели [8]

3. Анализ чувствительности математической модели трехфазной фильтрации на примере реальных данных

Оценим чувствительность одной из наиболее распространенных математических моделей разработки нефтяных месторождений, модели трехфазной фильтрации нефти, газа и воды, описанной в разделе 1, к основным исходным данным на примере полномасштабной модели одного из пластов реального месторождения Западной Сибири. Номера скважин и некоторые параметры изменены для соблюдения конфиденциальности данных недропользователя.

Залежь нефти имеет пластовое строение с активной законтурной водоносной областью, коллектор терригенный. Средняя толщина пласта 7,2 м.

Абсолютная проницаемость: минимальная – 30,2 мД; максимальная - 564,9 мД; средняя – 190,5 мД; стандартное отклонение – 74,0 мД.

Пористость: минимальная – 0,121 д.ед.; максимальная – 0,236 д.ед.; средняя – 0,210 д.ед.; стандартное отклонение – 0,013 д.ед.

Доля коллектора: минимальная – 0,1 д.ед.; максимальная – 1,0 д.ед.; средняя – 0,71 д.ед.; стандартное отклонение – 0,25 д.ед.

PVT-параметры флюидов: давление насыщения нефти газом – 199 бар; растворимость газа в нефти – 22,1 $м^3/м^3$; объемный коэффициент нефти 1,1 $м^3/м^3$; вязкость нефти в пластовых условиях – 4,3 сПз; объемный коэффициент воды – 1,001 $м^3/м^3$; вязкость воды в пластовых условиях – 0,5 сПз; сжимаемость воды – $3{,}8\cdot10^{-5}$ 1/бар; сжимаемость породы – $4{,}1\cdot10^{-5}$ 1/бар; плотность нефти в стандартных условиях – 875 $кг/м^3$; плотность воды в стандартных условиях – 1054 $кг/м^3$.

ОФП в системах вода-нефть и нефть-газ приведены на рис. 4 и 5 соответственно, капиллярным давлением пренебрегается.

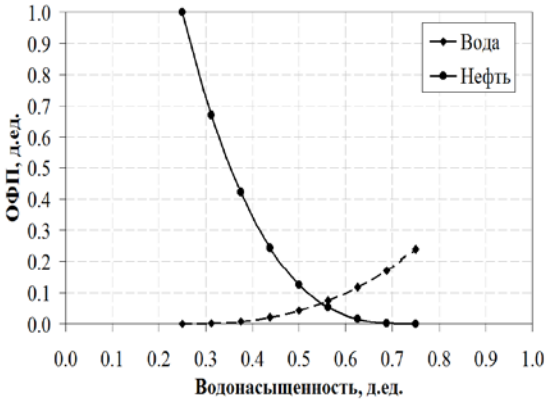


Рис. 4. Относительные фазовые проницаемости (ОФП) в системе вода-нефть

Начальное пластовое давление на глубине водонефтяного контакта – 241 бар. Средняя начальная нефтенасыщенность в чистонефтяной зоне – 0,73 д.ед. Свободного газа в начальный момент времени нет.

В эксплуатации перебывало 82 скважины, из них 67 добывающих и 15 нагнетательных, расстановка скважин приведена на рис. 6. На 4 скважинах проведен гидроразрыв пласта. Добывающие скважины управляются дебитом жидкости, нагнетательные – закачкой воды и ограничением по максимальному забойному давлению 450 бар (давление автогидроразрыва). История разработки – 20 лет.

На конец истории разработки коэффициент извлечения нефти - 0,316 д.ед.; обводненность продукции - 91 %; газовый фактор – 21,5 $м^3/м^3$; добыча нефти - 71 $м^3$/сут, добыча жидкости - 712 $м^3$/сут; закачка воды – 691 $м^3$/сут.

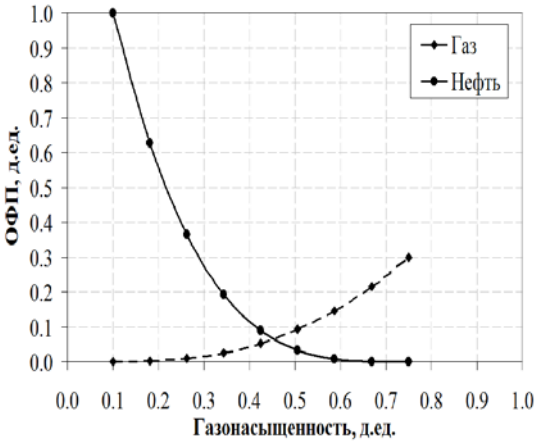


Рис. 5. Относительные фазовые проницаемости (ОФП) в системе нефть-газ

Расчеты проводись в программе Rubis (Kappa Engineering) [9] по трехфазной модели фильтрации на сетке Вороного, рис. 7.

С помощью симулятора анализировалась чувствительность модели к абсолютной проницаемости, пористости, доле коллектора, начальной нефтенасыщенности, начальному пластовому давлению, насыщенности связанной водой и остаточной нефтенасыщенности при вытеснении нефти водой, рис. 8-14 соответственно.

Исходные данные варьировались в пределах +/- 10 % для давления и +/- 30 % для остальных параметров.

При этом получены ошибки в накопленных показателях разработки вплоть до 80 %.

Оценивалось влияние исходных данных на накопленную добычу нефти, воды и газа, а также наколенную закачку воды.

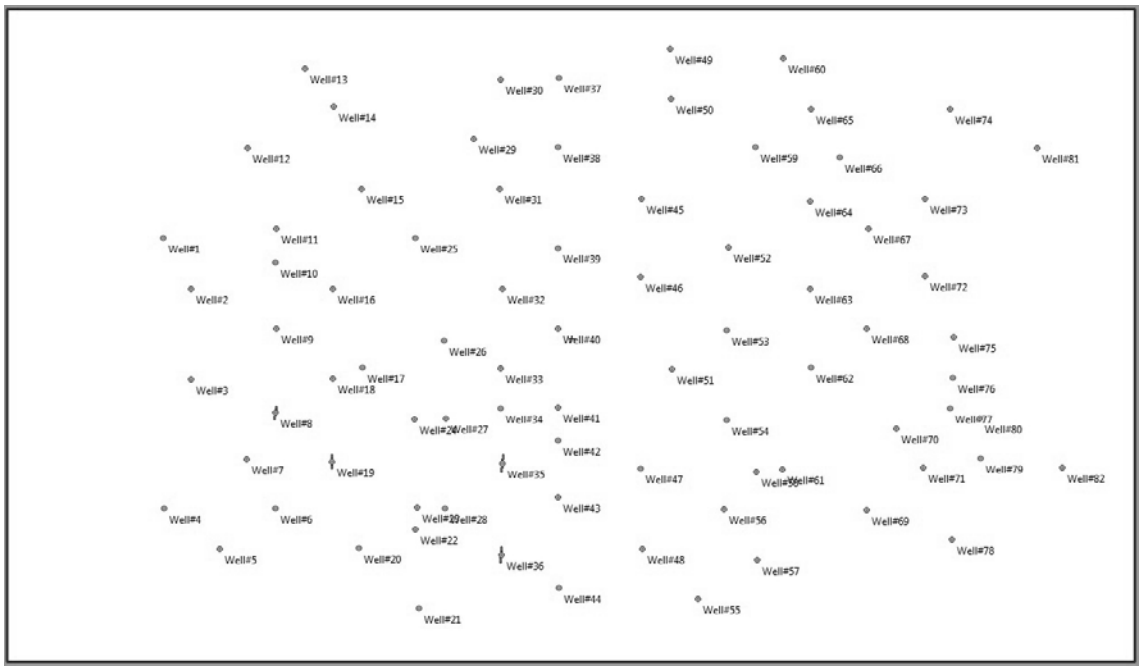


Рис. 6. Расстановка скважин в плане

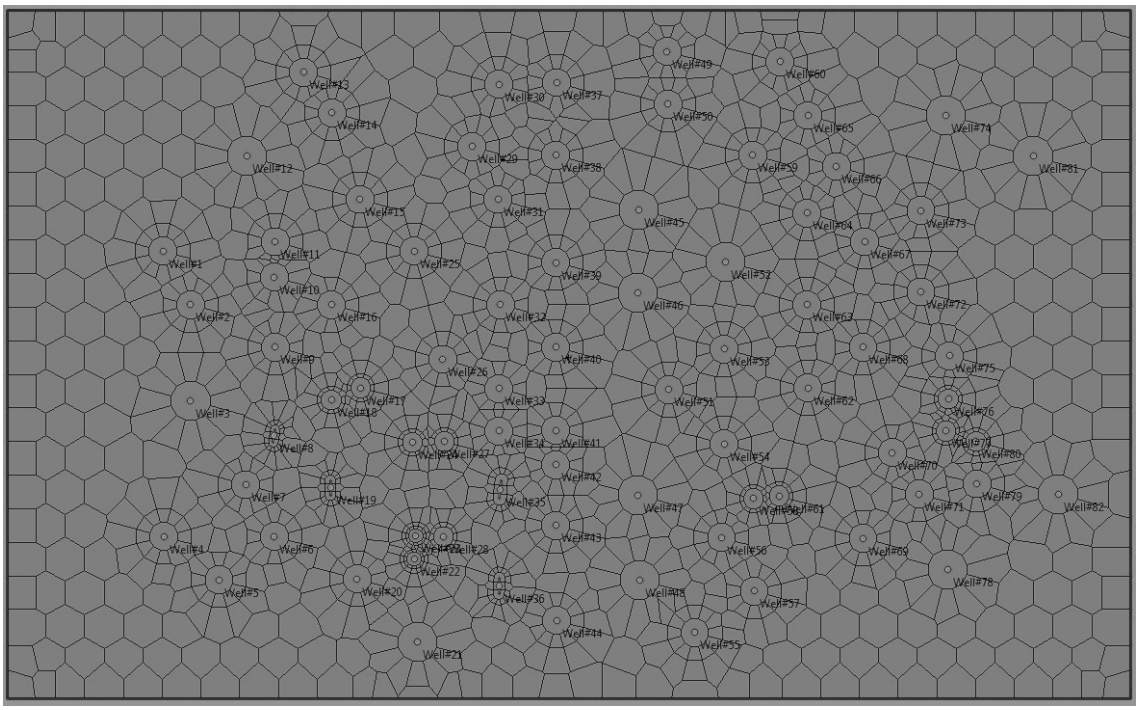


Рис. 7. Расчетная сетка для моделирования в программе Rubis (Kappa Engineering)

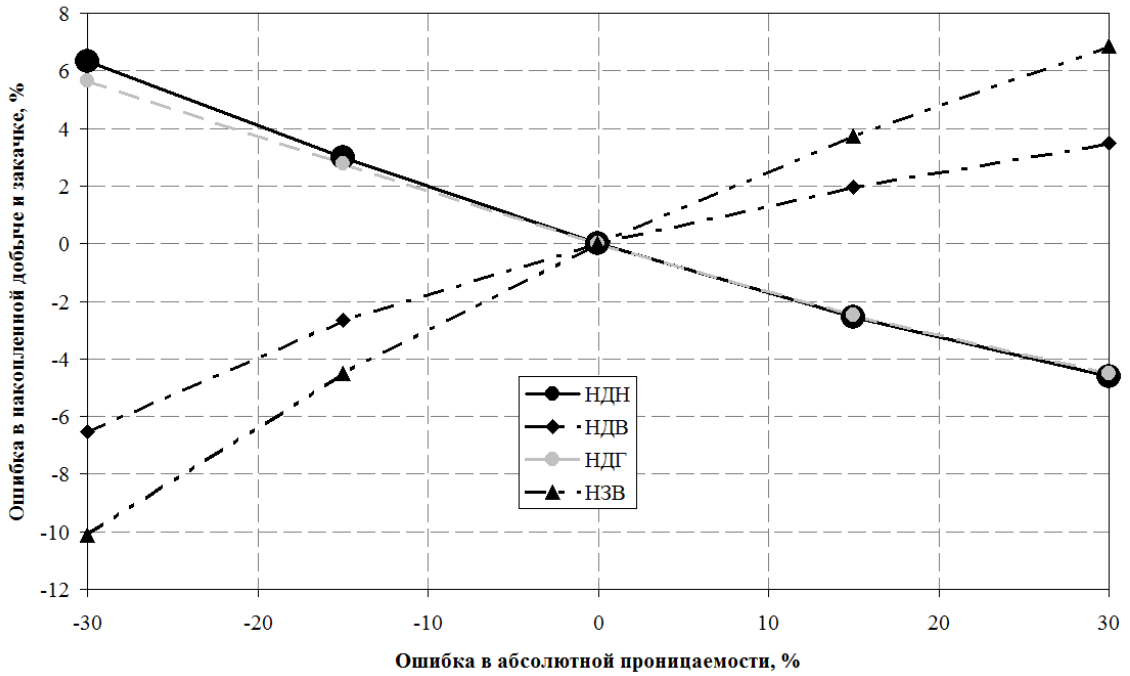


Рис. 8. Влияние абсолютной проницаемости на накопленную добычу и закачку
НДН, НДВ, НДГ и НЗВ – накопленная добыча нефти, воды, газа и накопленная закачка воды соответственно

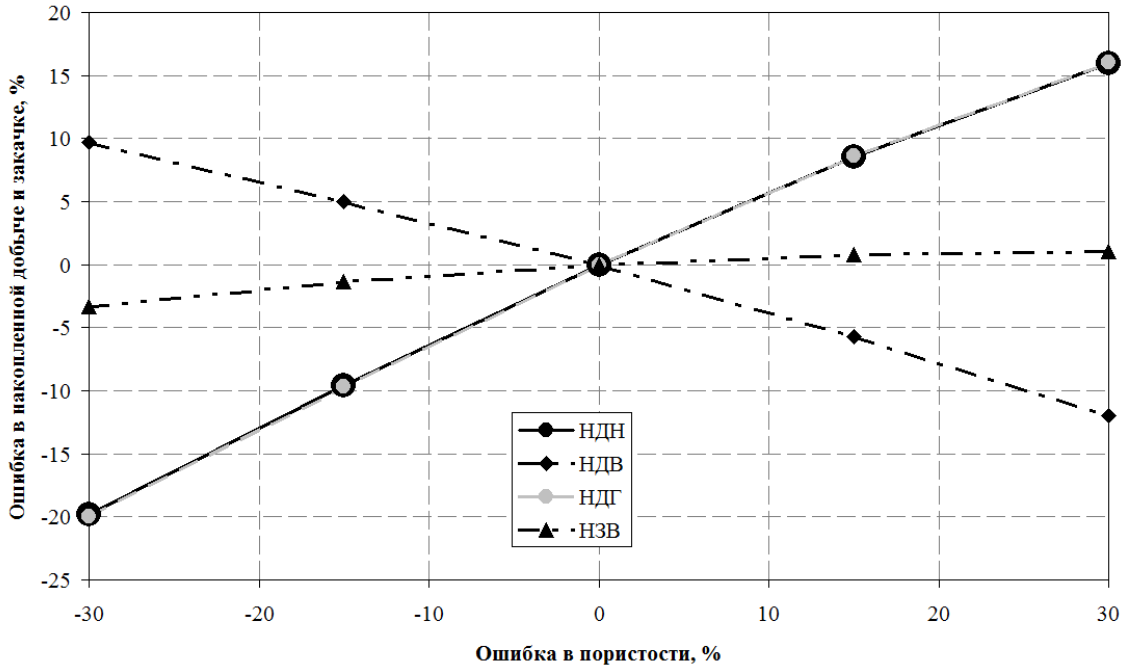


Рис. 9. Влияние пористости на накопленную добычу и закачку
НДН, НДВ, НДГ и НЗВ – накопленная добыча нефти, воды, газа и накопленная закачка воды соответственно

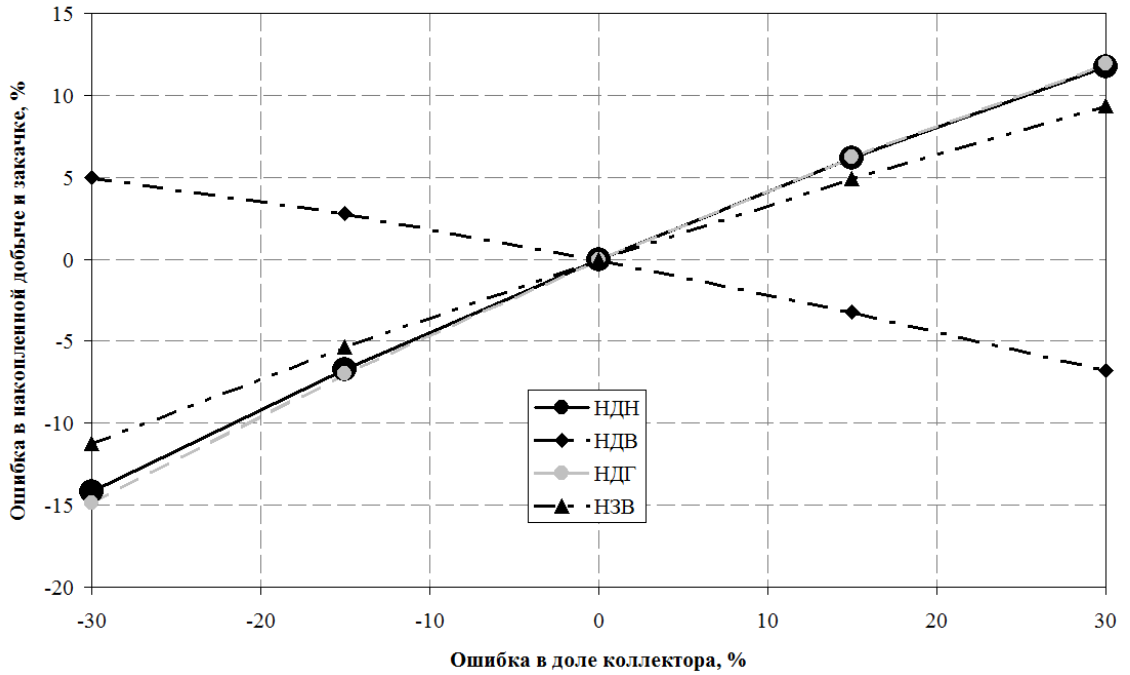


Рис. 10. Влияние доли коллектора на накопленную добычу и закачку
НДН, НДВ, НДГ и НЗВ – накопленная добыча нефти, воды, газа и накопленная закачка воды соответственно

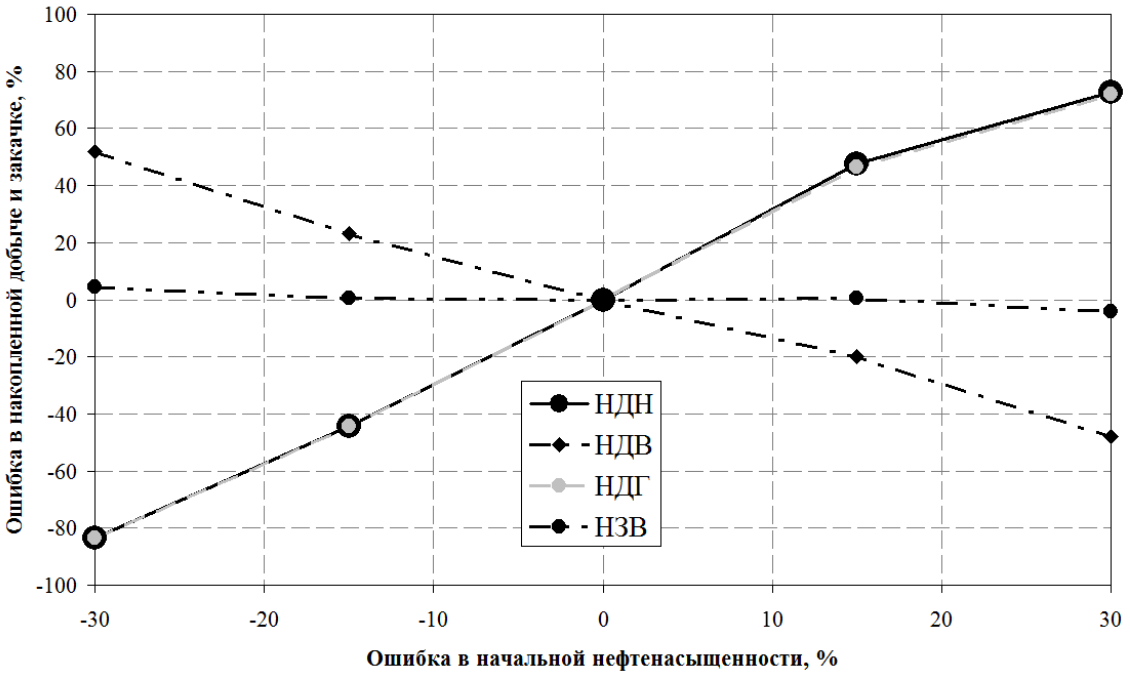


Рис. 11. Влияние начальной нефтенасыщенности на накопленную добычу и закачку
НДН, НДВ, НДГ и НЗВ – накопленная добыча нефти, воды, газа и накопленная закачка воды соответственно

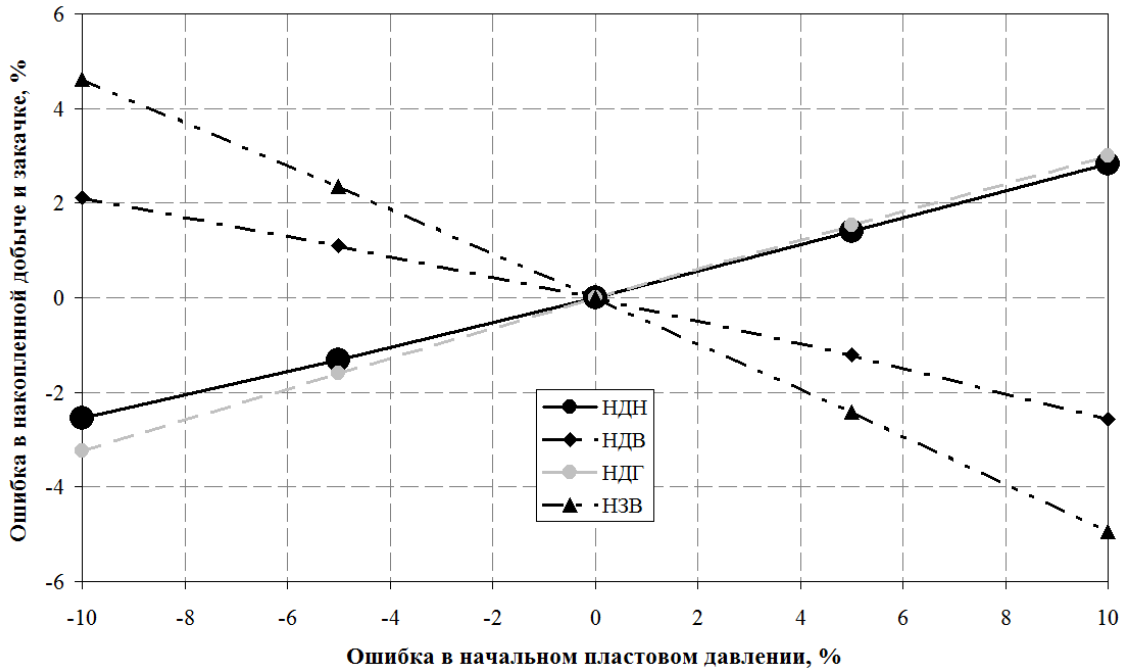


Рис. 12. Влияние начального пластового давления на накопленную добычу и закачку
НДН, НДВ, НДГ и НЗВ – накопленная добыча нефти, воды, газа и накопленная закачка воды соответственно

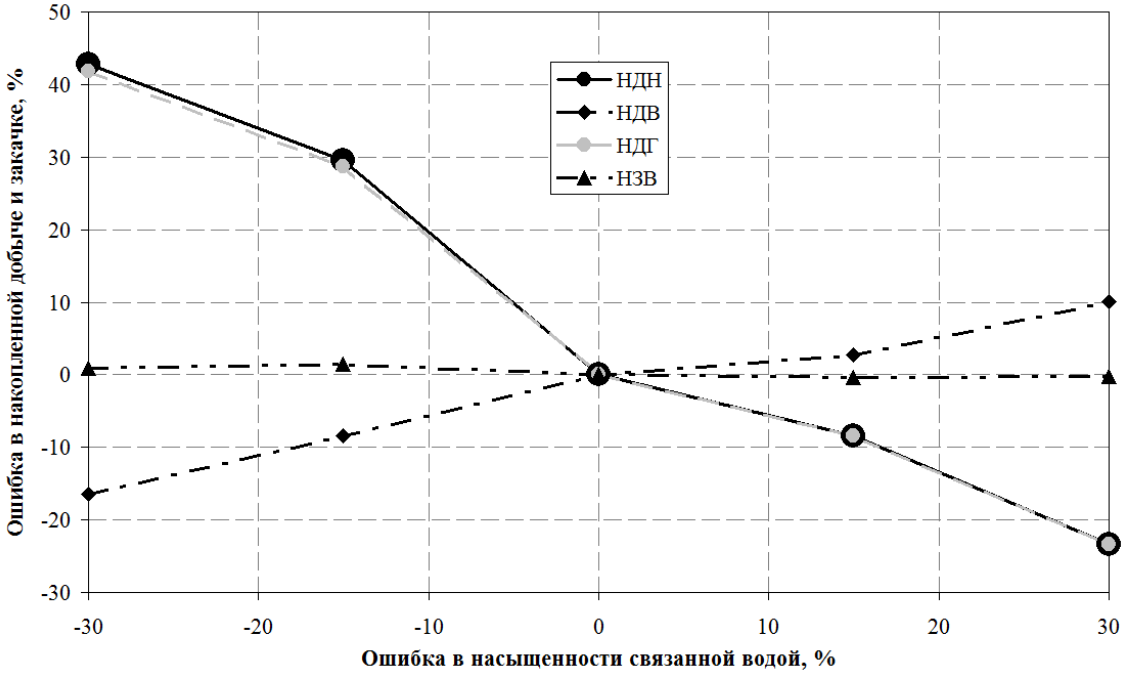


Рис. 13. Влияние насыщенности связанной водой на накопленную добычу и закачку
НДН, НДВ, НДГ и НЗВ – накопленная добыча нефти, воды, газа и накопленная закачка воды соответственно

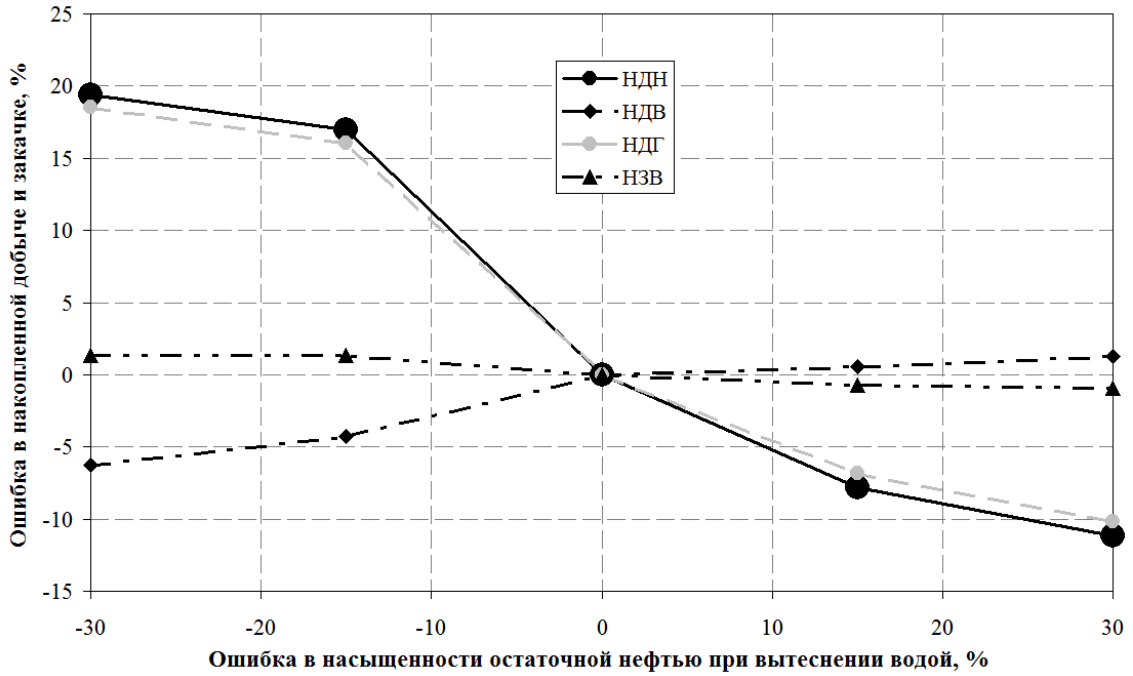


Рис. 14. Влияние насыщенности остаточной нефтью на накопленную добычу и закачку
НДН, НДВ, НДГ и НЗВ – накопленная добыча нефти, воды, газа и накопленная закачка воды соответственно

Заключение

На основании анализа литературных данных, а также вышеописанных собственных вычислительных экспериментов и опыта математического моделирования разработки нефтяных месторождений авторов работы можно выделить следующие факторы, оказывающие наибольшее влияние на основные показатели разработки:

1. Наибольшее влияние на пластовое давление имеют параметры законтурной водоносной области (ее объем, продуктивность и пр.), поровый объем пласта и проводимость пласта (произведение абсолютной проницаемости на эффективную толщину).
2. Наибольшее влияние на текущую и накопленную добычу фаз, а также на обводненность и газовый фактор имеют функции ОФП (которые могут зависеть от фильтрационно-емкостных свойств пласта, а также меняться от скважины к скважине, например для учета конусообразования), неоднородность пласта по проницаемости (в первую очередь – в вертикальном направлении, т.е. по толщине), поровый объем и наличие непроницаемых границ или суперпроницаемых коридоров.
3. Наибольшее влияние на закачку (с учетом ограничения по максимальному забойному давлению) имеет продуктивность скважин, проводимость пласта, пластовое давление и поровый объем.
4. Наибольшее влияние на забойное давление имеет продуктивность скважин, пластовое давление и поровый и объем.

Также всегда следует помнить о возможной погрешности в истории разработки по различным причинам, как технического и технологического характера, так (зачастую – в первую очередь) и под влиянием человеческого фактора. Необходимо критически относится к информации об исторических показателях работы скважин и сверять между собой карточки скважин, месячные эксплуатационные рапорты, технологические режимы и прочие документы, содержащие данные о дебитах и забойных давлениях скважин.

Существует множество параметров, имеющих существенное влияние на результаты математического моделирования разработки нефтяных месторождений с помощью модели трехфазной фильтрации нефти, газа и воды. После «сборки» гидродинамической модели (насыщения геологической модели данными об ОФП, капиллярном давлении, свойствах флюидов, параметров скважин и истории добычи) перед проведением адаптации желательно проводить анализ чувствительности модели, поскольку значимость отдельных исходных данных для показателей разработки, рассчитанных для различных моделей (моделей различных пластов и месторождений) при использовании

одной и той же системы уравнений для описания фильтрации, может существенно различаться.

Работа выполнена при поддержке Программы ФНИ государственных академий наук на 2013-2020 гг., проект № 0065-2018-0118.

Sensitivity to initial data changes evaluation for oil field development mathematic models

I.V. Afanaskin, N.P. Efimova, D.V. Solopov, P.V. Ialov

Abstract: Article amply describes 3-phase filtration model of oil, gas and water. Model calculated implicitly for Vorony grid. Initial data analyses for validity including sources. Model matching process overview. 3-phase filtration mathematic model sensitivity to initial data changes tested by commercial 3-d simulator Rubis (Kappa) using example of one West Siberian Field. Model matching recommendations based on modeled parameters accuracy influence on calculation results.

Keywords: filtration theory, mathematic modelling, oil fileds development, sensitivity analysis.

Литература

1. Х.Азиз, Э.Сеттари. Математическое моделирование пластовых систем. .-Ижевск: Институт компьютерных исследований, 2004. - 416 с.
2. В.А.Байков, Н.К.Бакиров, А.А.Яковлев. Математическая геология. Том 1. Введение в геостатистику. М.-Ижевск: Институт компьютерных исследований, 2012. - 228 с.
3. К.В.Дойч. Геостатистическое моделирование коллекторов. М.-Ижевск: Институт компьютерных исследований, 2011. - 400 с.
4. Р.Д.Каневская. Математическое моделирование гидродинамических процессов разработки месторождений углеводородов. М.-Ижевск: Институт компьютерных исследований, 2002. - 140 с.
5. Г.Б.Кричлоу. Современная разработка нефтяных месторождений - проблемы моделирования. М.: Недра, 1979. - 303 с.
6. В.И.Петерсилье, В.И.Пороскун, Г.Г.Яценко. Методические рекомендации по подсчету запасов нефти и газа объемным методом. Тверь: ВНИГНИ, НПЦ «Тверьгеофизика», 2003. – 259 с.
7. В.В.Стасенков, И.С.Гутман. Подсчет запасов нефти, газа, конденсата и содержащихся в них компонентов. М.: Недра, 1989. – 270 с.
8. Т.Эртекин, Дж.Абу-Кассем, Г.Кинг. Основы прикладного моделирования пластов. М.-Ижевск: Институт компьютерных исследований, 2012. - XXVIII+1060 с.
9. Kappa Engineering [Электронный ресурс] / Rubis – численное моделирование – Описание [электронный ресурс]. Режим доступа: https://www.kappaeng.com/software/rubis/overview (дата обращения 29.04.2018 г.)
10. Olivier Houze, Didier Viturat, Ole S. Fjaere. Dynamic Data Analysis. V 5.12. Kappa Engineering, 2017. – 743 P.

Нульмерная математическая модель для анализа разработки нефтяных месторождений методом ячеек заводнения

И.В. Афанаскин[1], А.В. Королев[2]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1]ivan@afanaskin.ru, [2]alexandre.korolev@mail.ru

Аннотация. Представлена нульмерная математическая модель для анализа разработки нефтяных месторождений методом ячеек заводнения. Кроме классических возможностей метода ячеек заводнения эта модель позволяет прогнозировать показатели разработки при режимах работы скважин, отличающихся от базовых. Модель справедлива при условии, что пластовое и забойное давление выше давления насыщения нефти газом. При этом скважины должны быть уже обводнены (параметры необводненных скважин принимаются по аналогии). Предлагается подход к автоматизации процесса адаптации модели ячеек заводнения. В приведенном примере использования модели получено хорошее совмещение фактических и расчетных показателей разработки не только на истории, но и при прогнозе. В примере дается анализ влияния шага по времени на точность расчета пластового давления и водонасыщенности.

Ключевые слова: ячейки заводнения, анализ разработки, двухфазная фильтрация.

Введение

Разработка нефтяных месторождений методом заводнения (который является доминирующим в России) предполагает длительную стадию одновременной добычи нефти и воды. При этом дебит нефти все время падает, а дебит воды растет. Добыча жидкости (смеси нефти и воды) компенсируется закачкой воды, а также внедрением воды водоносной области (находящейся за водонефтяным контуром) в нефтяную залежь в результате падения в ней давления. Для управления разработкой необходимо оценивать текущее состояние ячеек заводнения, определять: насыщенность, остаточные запасы нефти, пластовое давление, количество вторгшейся из законтурной области воды, продуктивность законтурной области и ее запас упругой энергии. Кроме того необходимо также делать прогнозные расчеты показателей разработки, как при сложившихся условиях работы скважин, так и при изменении режимов их работы. Для решения этих вопросов в данной статье предлагается нульмерная математическая модель для анализа разработки нефтяных месторождений методом ячеек заводнения

1. Математическая модель

Метод ячеек заводнения предполагает разбиения нефтяной залежи по площади на блоки [1-5], слабо взаимодействующие между собой или совсем не взаимодействующие. Блоки включают в себя добывающие и нагнетательные скважины, могут граничить с законтурной водоносной областью и получать из нее воду. В классическом варианте при использовании ячеек заводнения пластовое давление рассчитывается методом материального баланса [6-8], а добыча нефти определяется с помощью характеристик вытеснения [9]. Такой подход является эмпирическим (в силу применения характеристик вытеснения) и не позволяет прогнозировать работу скважин при изменении режимов (дебитов жидкости, закачки воды или забойных давлений).

Получим строгую математическую модель, позволяющую прогнозировать разработку нефтяных месторождений при заводнении путем прогнозирования изменения пластового давления и водонасыщенности пласта. Будем считать, что пластовое и забойное давление выше давления насыщения нефти газом. Фильтруются две слабосжимаемые несмешивающиеся жидкости (нефть и вода) в упругой пористой среде. По определению водонасыщенность S_w:

$$S_w = V_w / V_p , \qquad (1)$$

где

$$V_w = Fh\phi S_w , \; V_p = Fh\phi , \qquad (2)$$

V_w и V_p - объем воды и объем пор в контрольном объеме пласта соответственно; F, h и ϕ - площадь, толщина и пористость контрольного объема пласта соответственно

Продифференцируем выражение (1):

$$dS_w = d\left(\frac{V_w}{V_p}\right) = \frac{V_p dV_w - V_w dV_p}{V_p^2}. \quad (3)$$

Изменение объема воды в пласте происходит за счет работы скважин и законтурной области, а также за счет упругих сил:

$$dV_w = (q_{iw} + q_a - q_w) b_w dt - V_p S_w C_w dP, \quad (4)$$

где q_{iw}, q_w и q_a - расход нагнетаемой воды (суммарно по всем нагнетательным скважинам), дебит добываемой воды (суммарно по всем добывающим скважинам) и расход воды из законтурной области в поверхностных условиях соответственно; b_w - объемный коэффициент воды; dt - время, за которое произошло изменение объема воды dV_w; C_w - сжимаемость воды; dP - изменение пластового давления P за время dt.

Кроме того:

$$dV_p = Fh d\phi. \quad (5)$$

Подставляя (2), (4) и (5) в (3) и проводя элементарные преобразования получаем:

$$dS_w = \frac{dt}{Fh\phi}(q_{iw} + q_a - q_w) b_w - S_w C_w dP - S_w \frac{d\phi}{\phi} \quad (6)$$

Воспользуемся методом Эйлера и перейдем к дискретному аналогу выражения (6):

$$S_w(t+\Delta t) = S_w(t) - \frac{\Delta t}{Fh\phi(t)}\{[q_w(t) - q_{iw}(t)] b_w(t) - q_a(t) b_{wa}(t)\} - C_w S_w(t)[P(t+\Delta t) - P(t)] - S_w(t)\frac{\phi(t+\Delta t) - \phi(t)}{\phi(t)} \quad (7)$$

где Δt - шаг по времени, b_{wa} - объемный коэффициент воды в законтурной области.

Поскольку пласт упругий:

$$\phi(t) = \phi_0\{1 + C_r[P(t) - P_0]\}, \quad (8)$$

где ϕ_0 - пористость при начальном давлении P_0, C_r - сжимаемость породы.

Так как C_r <<1, то $1 + C_r[P(t) - P_0] \approx 1$. Тогда:

$$\frac{\phi(t+\Delta t) - \phi(t)}{\phi(t)} = C_r[P(t+\Delta t) - P(t)]. \quad (9)$$

Перепишем выражение для водонасыщенности (7) с учетом (9):

$$S_w(t+\Delta t) = S_w(t) - \frac{\Delta t}{Fh\phi(t)}\{[q_w(t) - q_{iw}(t)] b_w(t) - q_a(t) b_{wa}(t)\} - (C_r + C_w) S_w(t)[P(t+\Delta t) - P(t)] \quad (10)$$

Можно получить более сложные, чем (10), формулы для расчета водонасыщенности.

Получим формулу для водонасыщенности с прямым учетом потоков воды и нефти (опосредованно в формуле (10) приток нефти учитывается через изменение давления). Для этого по аналогии с (10) запишем выражение для нефтенасыщенности:

$$S_o(t+\Delta t) = S_o(t) - \frac{\Delta t}{Fh\phi(t)} q_o(t) b_o(t) - (C_r + C_o) S_o(t)[P(t+\Delta t) - P(t)] \quad (11)$$

Учитывая балансовое соотношение:

$$S_o + S_w = 1, \quad (12)$$

заменяя с учетом (12) нефтенасыщенность на водонасыщенность перепишем (11) в виде:

$$S_w(t+\Delta t) = S_w(t) + \frac{\Delta t}{Fh\phi(t)} q_o(t) b_o(t) + (C_r + C_o)[1 - S_w(t)][P(t+\Delta t) - P(t)] \quad (13)$$

Складывая два разных выражения для водонасыщенности (10) и (13), получим окончательное выражение для водонасыщенности:

$$S_w(t+\Delta t) = S_w(t) + \frac{1}{2}\left\{\frac{\Delta t}{Fh\phi(t)}[q_o(t) b_o(t) - [q_w(t) - q_{iw}(t)] b_w(t) + q_a(t) b_{wa}(t)] + [C_r - (C_o + C_w + 2C_r) S_w(t) + C_o] \times [P(t+\Delta t) - P(t)]\right\} \quad (14)$$

Складывая выражения для водо- и нефтенасыщенности (10) и (11) с учетом (12) получим формулу для пластового давления в ячейке заводнения:

$$P(t+\Delta t) = P(t) - \Delta t\{[q_w(t) - q_{iw}(t)] b_w(t) - q_a(t) b_{wa}(t) + q_o(t) b_o(t)\} / \{Fh\phi(t)[C_r + (C_w - C_o) S_w(t) + C_o]\} \quad (15)$$

По аналогии с (15) запишем выражение для давления в законтурной области:

$$P_a(t+\Delta t) = P_a(t) - \Delta t \frac{q_a(t) b_{wa}(t)}{V_a \phi_a(t)[C_{ra} + C_w]}, \quad (16)$$

где V_a, ϕ_a и C_{ra} - объем, пористость и сжимаемость породы в законтурной области.

С учетом выражения для давления (15) перепишем уравнение для водонасыщенности (10) следующим образом:

$$S_w(t+\Delta t)=S_w(t)+\Delta t\{(C_r+C_o)S_o(t)\times \times\{[q_w(t)-q_{iw}(t)]b_w(t)-q_a(t)b_{wa}(t)\}+ +(C_r+C_w)S_w(t)q_o(t)b_o(t)\}/ /\{Fh\phi(t)[C_r+C_wS_w(t)+C_oS_o(t)]\} \quad (17)$$

где нефтенасыщенность определяется из (12).

В результате мы получили три выражения для водонасыщенности (10), (14) и (17), которые при выборе правильного шага по времени должны давать одинаковый результат.

Так как мы приняли, что нефть и вода слабосжимаемые жидкости, можно записать следующие формулы для объемных коэффициентов:

$$B_\alpha(t)=B_{\alpha 0}\{1-C_\alpha[P(t)-P_0]\},\ \alpha=o,w, \quad (18)$$

где $B_{\alpha 0}$ - объемный коэффициент фазы α при давлении P_0.

Для осуществления контроля за точностью решения обыкновенных дифференциальных уравнений (ОДУ) методом Эйлера можно рассчитать водонасыщенность (14) пластовое давление в нефтяной залежи (15) и давление в законтурной нефтяной области (16) два раза - с шагом по времени Δt и $\Delta t/2$. Анализируя расположение друг относительно друга кривых насыщенности и давлений, рассчитанных с разным шагом, можно сделать вывод о сходимости численного решения ОДУ и выбрать правильный временной шаг.

2. Учет работы скважин

Дебиты n-ой добывающей скважины в поверхностных условиях по нефти $q_{o,n}$ и воде $q_{w,n}$, обводненность $WWCT_n$ и забойное давление $P_{w,n}$ определяются, как:

$$q_{o,n}(t)=q_{l,n}(t)[1-WWCT_n(t)], \quad (19)$$

$$q_{w,n}(t)=q_{l,n}(t)WWCT_n(t), \quad (20)$$

$$WWCT_n(t)=\frac{k_{rw,n}(t)}{k_{rw,n}(t)+k_{ro,n}(t)\dfrac{\mu_w b_w(t)}{\mu_o b_o(t)}}, \quad (21)$$

$$P_{w,n}(t)=P(t)-q_{l,n}(t)/\left\{2\pi k_n h_n\left[\frac{k_{ro,n}(t)}{\mu_o b_o(t)}+\frac{k_{rw,n}(t)}{\mu_w b_w(t)}\right]\Big/\left[ln\left(\frac{R_{c,n}}{r_{w,n}}\right)+Skin_n(t)\right]\right\} \quad (22)$$

где $q_{l,n}$, k_n, h_n, $R_{c,n}$, $r_{w,n}$ и $Skin_n$ - дебит жидкости, абсолютная проницаемость, эффективная толщина, радиус контура питания, радиус скважины и скин-фактор n-ой добывающей скважины соответственно; $k_{r\alpha,n}$, $\alpha=o,w$ - относительная фазовая проницаемость (ОФП) по фазе α n-ой добывающей скважины, определяемая, как:

$$k_{rw,n}(t)=A_n\left(\frac{S_w(t)-S_{wcr,n}}{1-S_{wcr,n}}\right)^{\alpha_n}, \quad (23)$$

$$k_{ro,n}(t)=B_n\left(\frac{1-S_{owcr,n}-S_w(t)}{1-S_{owcr,n}-S_{wcr,n}}\right)^{\beta_n}, \quad (24)$$

где $S_{wcr,n}$ и $S_{owcr,n}$ - насыщенность связанной водой и остаточной нефтью; A_n, B_n, α_n и β_n - параметры корреляций для функций ОФП.

Забойное давление m-ой нагнетательной скважины определяется, как:

$$P_{w,m}(t)=P(t)+q_{iw,m}(t)/\left\{2\pi k_m h_m\frac{k_{rw,m}^{max}}{\mu_w b_w(t)}\Big/\left[ln\left(\frac{R_{c,m}}{r_{w,m}}\right)+Skin_m(t)\right]\right\} \quad (25)$$

где $k_{rw,m}^{\max}$ - максимально возможное для m-ой нагнетательной скважины значение ОФП по воде.

Для случая горизонтальной скважины скин-фактор в формулах (22) и (25) может быть рассчитан, как [10]:

$$Skin_n=1-\frac{h_n}{L_n}\frac{1}{\sqrt{\alpha_{z,n}}}ln\left(\frac{4\pi^2 r_{w,n}z_{w,n}}{h_n^2}\sqrt{\alpha_{z,n}}\right)+ +ln\left(\frac{2{,}7r_{w,n}}{L_n}\right) \quad (26)$$

где L_n, $\alpha_{z,n}$ и $z_{w,n}$ - длина горизонтальной части ствола, коэффициент вертикальной анизотропии проницаемости и расстояние от подошвы пласта до горизонтальной скважины соответственно. Формула (26) справедлива при выполнении условия: $r_{w,n}\le z_{w,n}\le h_n/2$.

Для случая вертикальной скважины с трещиной гидроразрыва пласта (ГРП) скин-фактор в формулах (22) и (25) может быть рассчитан, как:

$$Skin_n=f_n-\ln\left(\frac{x_{f,n}}{r_{w,n}}\right), \quad (27)$$

где f_n - псевдоскин-функция, определяемая по аппроксимации Валко-Экономидеса:

$$f_n = \frac{1{,}65 - 0{,}328u_n + 0{,}116u_n^2}{1 + 0{,}18u_n + 0{,}064u_n^2 + 0{,}005u_n^3},$$

$$u_n = \ln\left(C_{fd,n}\right),\ C_{fd,n} = \frac{k_{f,n}\omega_n}{k_n x_{f,n}}, \qquad (28)$$

где $C_{fd,n}$, $k_{f,n}$, ω_n и $x_{f,n}$ - безразмерная проводимость, проницаемость, раскрытость и полудлина трещины ГРП. Аппроксимация (28) справедлива при $0{,}001 < C_{fd,n} < 100$.

Расход нагнетаемой воды (суммарно по всем нагнетательным скважинам) $q_{iw}(t)$, дебит добываемой воды $q_w(t)$ и нефти $q_o(t)$ (суммарно по всем добывающим скважинам) и расход воды из законтурной области $q_a(t)$ в формулах (14)-(17) вычисляются, как:

$$q_{iw}(t) = \sum_{m=1}^{N_m} q_{iw,m}(t),$$

$$q_w(t) = \sum_{n=1}^{N_n} q_{w,n}(t),\ q_o(t) = \sum_{n=1}^{N_n} q_{o,n}(t),$$

$$q_a(t) = PI_a\left[P_a(t) - P(t)\right], \qquad (29)$$

где N_m - количество нагнетательных скважин, N_n - количество добывающих скважин, PI_a - коэффициент продуктивности законтурной водоносной области.

3. Адаптация модели к истории разработки

Задача адаптации модели к истории разработки сводится к решению задачи минимизации невязки целевой функции $f(X)$, представляющей собой сумму квадратов разностей между фактическими и расчетными показателями разработки для всех точек по времени.

Будем решать эту задачу методом Ньютона. Пусть необходимо найти минимум функции многих переменных $f(X)$, где $X = (x_1, x_2, x_3, ..., x_n)$. Эта задача сводится к задаче нахождения значений X, при которых градиент функции $f(X)$ равен нулю:

$$grad\left(f(X)\right) = 0. \qquad (30)$$

Применим к (29) метод Ньютона:

$$grad\left[f\left(X^j\right)\right] + H\left(X^j\right)\left(X^{j+1} - X^j\right) = 0, \qquad (31)$$

где $j = 1,2,3,...,m$ - номер итерации, $H(X)$ - гессиан функции $f(X)$.

Напомним, что гессиан функции – это симметричная квадратичная форма, описывающая поведение функции во втором порядке:

$$H(X) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} x_i x_j, \qquad (32)$$

где $a_{ij} = \partial^2 f / \partial x_i \partial x_j$, функция $f(X)$ задана на n-мерном пространстве вещественных чисел.

В более удобном для вычислений виде формулу (31) можно представить в виде:

$$X^{j+1} = X^j - H^{-1}\left(X^j\right) grad\left[f\left(X^j\right)\right]. \qquad (33)$$

Адаптацию предлагается проводить в несколько этапов:

1. Сначала в качестве целевой функции выбирается пластовое давление в нефтяной залежи (15). При этом в качестве уточняемых переменных рекомендуется принимать продуктивность законтурной области, объем законтурной области, сжимаемость нефти, воды и породы пласта.
2. Затем в качестве целевой функции используется накопленная добыча нефти и воды по всем скважинам. При этом рассчитываются невязки отдельно по каждой скважине, а затем суммируются. В качестве уточняемых параметров рекомендуется принимать коэффициенты функций ОФП (23) и (24) $S_{wcr,n}$ - насыщенность связанной водой, A_n, B_n, α_n и β_n. Использовать насыщенность остаточной нефтью $S_{owcr,n}$ не рекомендуется, поскольку она определяет подвижные запасы и закреплена на балансе в органах государственного контроля за разработкой нефтяных месторождений.
3. После этого в качестве целевой функции используется забойное давление (22) и (25). В качестве уточняемых параметров можно рекомендовать произведение абсолютной проницаемости на эффективную толщину $k_n h_n$ и радиус контура питания $R_{c,n}$. Следует отметить, что хотя значение $R_{c,n}$ имеет большую неопределенность, но в широких пределах своего изменения оно мало влияет на забойное давление.
4. В случае необходимости этапы (1)-(3) могут быть несколько раз последовательно пройдены для улучшения качества адаптации. Так реализуется простейший метод многомерной оптимизации, называемый методом покоординатного спуска.

4. Пример использования нульмерной модели

Для иллюстрации работы данной модели были использованы данные по участку разработки одного из месторождений Западной Сибири, рис. 1.

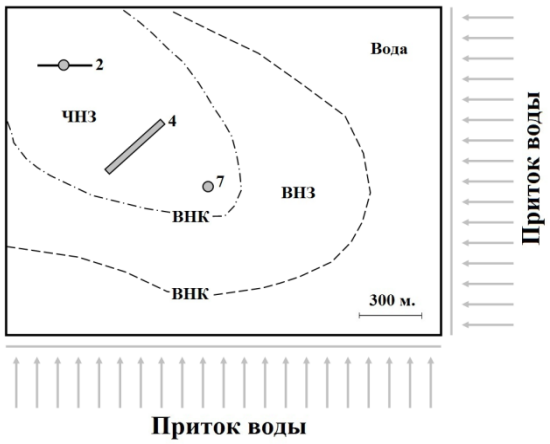


Рис. 1. Анализируемый участок. ВНК – водонефтяной контакт, ВНЗ – водонефтяная зона, ЧНЗ – чисто-нефтяная зона, 2 – скважина вертикальная с трещиной гидроразрыва пласта, 4 – горизонтальная скважина, 7 – вертикальная скважина

Породы терригенные, залежь нефтяная, пластового типа.

Анализируемый участок является экспериментальным (проводятся опытно-промышленные работы) и эксплуатируется тремя добывающими скважинами различной конструкции: 2 – скважина вертикальная с трещиной гидроразрыва пласта, 4 – горизонтальная скважина (пробурена в прикровельной зоне), 7 – вертикальная скважина.

Нефтяная залежь окружена активной водоносной областью.

Режим разработки близок к жесткому водонапорному. Пластовое и забойное давление выше давления насыщения нефти газом.

История разработки составляет 6 лет, однако скважины уже успели обводниться.

К сожалению, в нашем распоряжении были только годовые данные по добыче и замерам давления.

Для уверенного использования рассматриваемой модели ячейки заводнения желательно иметь более подробные данные. Поэтому для тестирования предлагаемой модели на основании годовых исторических данных была создана и адаптирована трехмерная численная гидродинамическая модель в симуляторе Rubis (Kappa Engineering). Качество адаптации этой модели хорошее. Поэтому в дальнейшем для тестирования модели были использованы расчетные данные, полученные с помощью упомянутого симулятора. Относительные фазовые проницаемости по нефти и воде, использованные в симуляторе и полученные в результате адаптации модели ячеек заводнения, приведены на рис. 2.

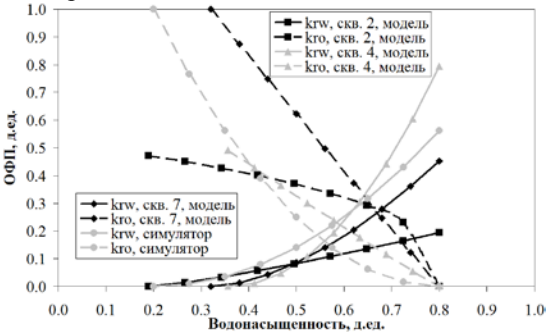


Рис. 2. Относительные фазовые проницаемости (ОФП) в системе нефть-вода, использованные при численном моделировании (симулятор) и полученные в результате адаптации модели ячейки заводнения (модель). Здесь kro – ОФП по нефти, krw – по воде

Фильтрационно-емкостные свойства пласта в численной гидродинамической модели (Rubis) описываются следующими статистическими характеристиками:

1. Абсолютная проницаемость.
   1.1. Минимальное значение – 0,7 мД.
   1.2. Максимальное значение – 101,7 мД.
   1.3. Математическое ожидание - 12,0 мД.
   1.4. Стандартное отклонение – 15,4 мД.
2. Открытая пористость.
   2.1. Минимальное значение – 0,101 д.ед.
   2.2. Максимальное значение – 0,197 д.ед.
   2.3. Математическое ожидание – 0,145 д.ед.
   2.4. Стандартное отклонение – 0,019 д.ед.
3. Доля коллектора.
   3.1. Минимальное значение – 0,110 д.ед.
   3.2. Максимальное значение – 1,000 д.ед.
   3.3. Математическое ожидание – 0,781 д.ед.
   3.4. Стандартное отклонение – 0,238 д.ед.

При расчете с помощью модели ячеек заводнения приняты следующие геолого-физические характеристики:

1. Площадь нефтеносности 614 тыс.м^2.
2. Средняя эффективная нефтенасыщенности толщина 8,69 м.
3. Пористость 0,113 д.ед.
4. Абсолютная проницаемость 12 мД.
5. глубина залегания кровли пласта 3355 м.
6. Начальное пластовое давление 335,3 бар.
7. Начальная пластовая температура 75 ºС.
8. Начальная водонасыщенность 0,220 д.ед.

9. Свойства воды - объемный коэффициент 1,02 $м^3/м^3$, сжимаемость $4{,}7 \cdot 10^{-5}$ 1/бар, вязкость 0,36 мПа·с.
10. Свойства нефти - объемный коэффициент 1,25 $м^3/м^3$, сжимаемость $1{,}01 \cdot 10^{-4}$ 1/бар, вязкость 0,40 мПа·с, растворимость газа в нефти 203,5 $м^3/м^3$, давление насыщения нефти газом 188 бар.
11. Сжимаемость породы $4{,}7 \cdot 10^{-5}$ 1/бар.
12. Радиус скважин 0,1 м., механический скин-фактор 0,0 безразм.

Все три скважины объединяются в одну ячейку заводнения, которая на каждый момент времени характеризуется одним значением давления и насыщенности.

На рис. 3-6 приведены результаты адаптации и прогнозных расчетов на три года. Видно хорошее качество адаптации и прогноза.

В процессе адаптации модели ячейки заводнения уточнялись функции ОФП (рис. 2) и сжимаемости нефти, воды и пласта, а также подбирались значения объема и продуктивности законтурной области. Объем законтурной области оценен в $8{,}0 \cdot 10^9$ $м^3$, что в 128 раз меньше, чем ранее оцененный для всего пласта целиком. Продуктивность законтурной области оценена в 3,0 $м^3/(сут \cdot бар)$, что в 84 раз меньше, чем ранее оцененная для всего пласта целиком.

Рассмотрено два прогнозных варианта:

1. Вариант 1. Базовый вариант – эксплуатация в сложившихся условиях при дебитах жидкости, аналогичных историческим значениям к моменту перехода на прогноз.
2. Вариант 2. Эксплуатация скважин при дебитах жидкости, максимальных за всю историю разработки.

На рис. 7 и 8 приведены результаты анализа чувствительности пластового давления и водонасыщенности к временному шагу.

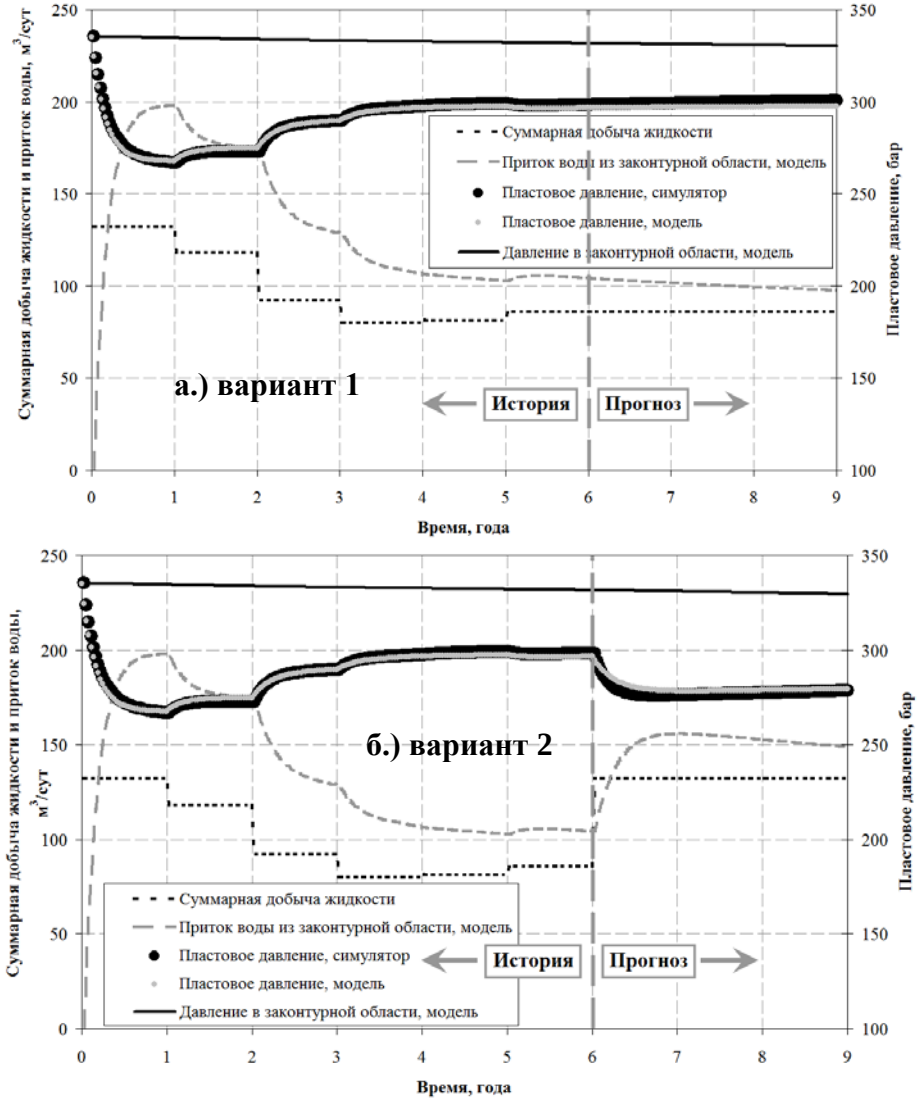


Рис. 3. Пластовое давление по вариантам

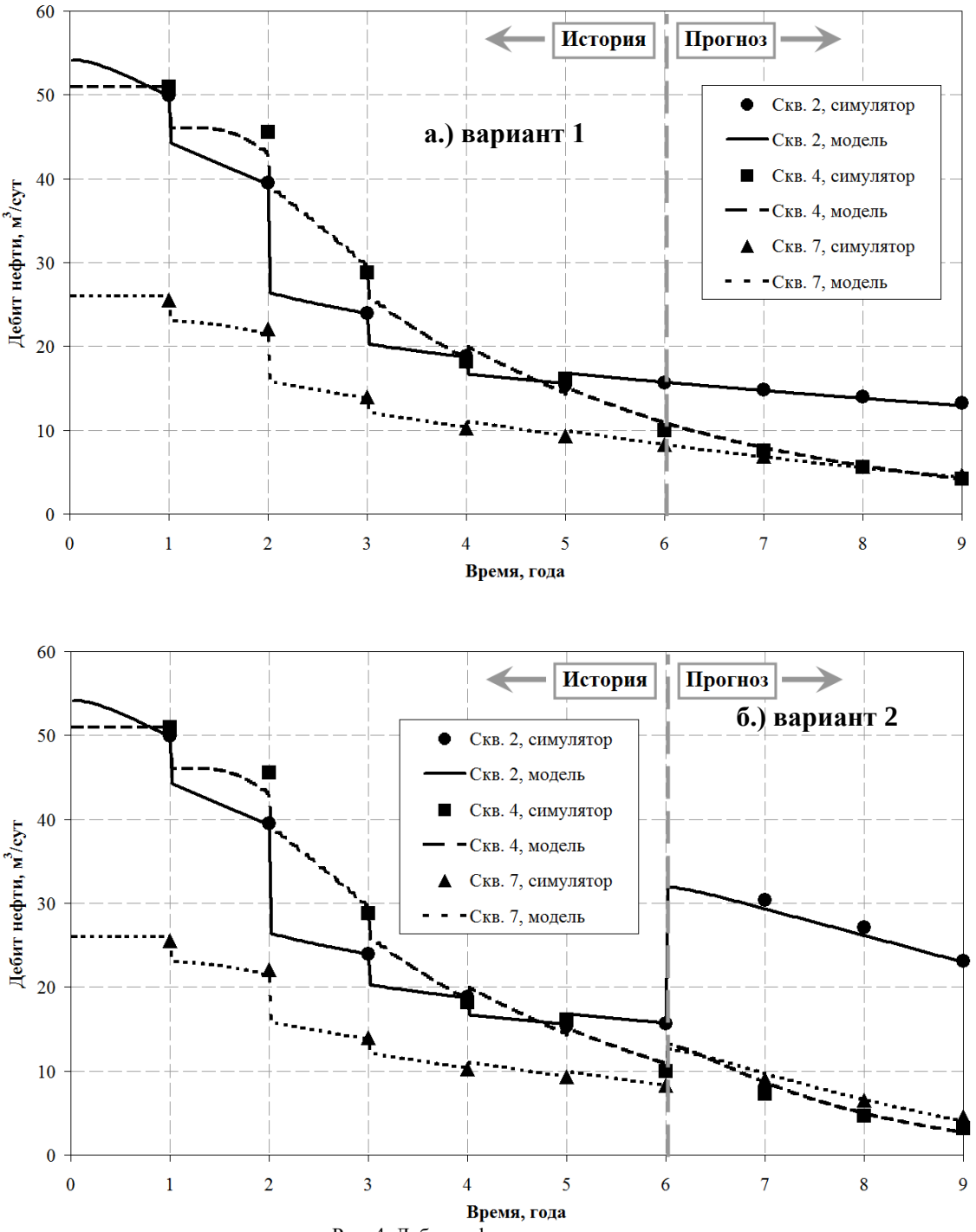
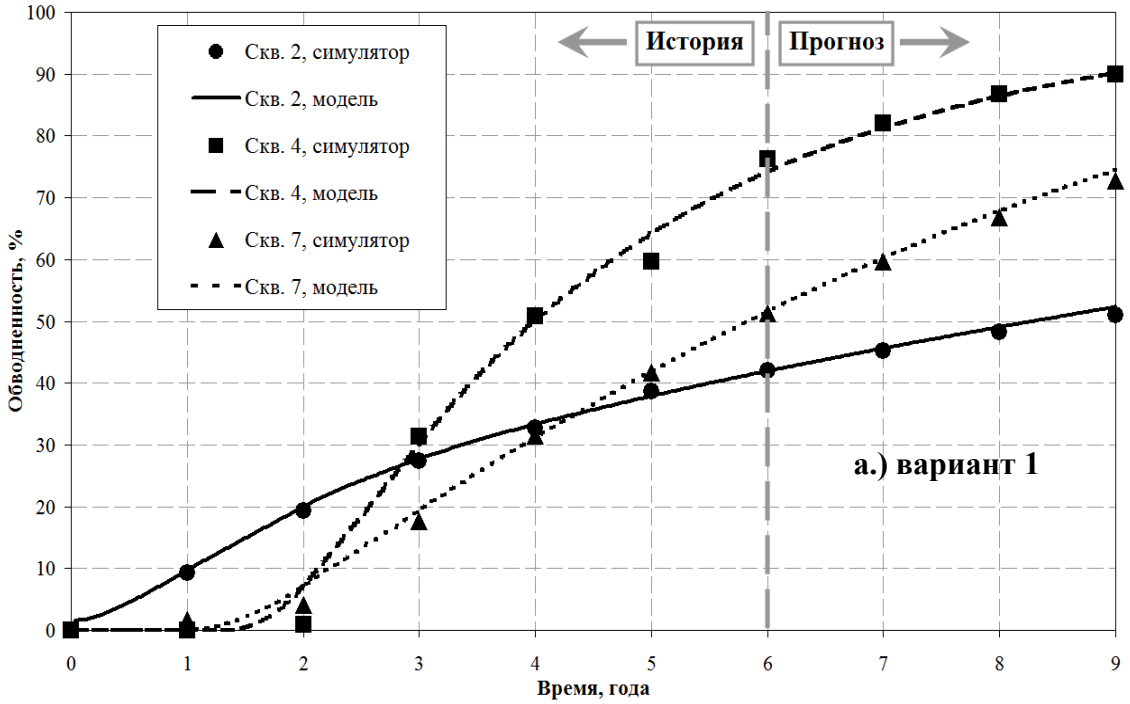


Рис. 4. Дебит нефти по вариантам



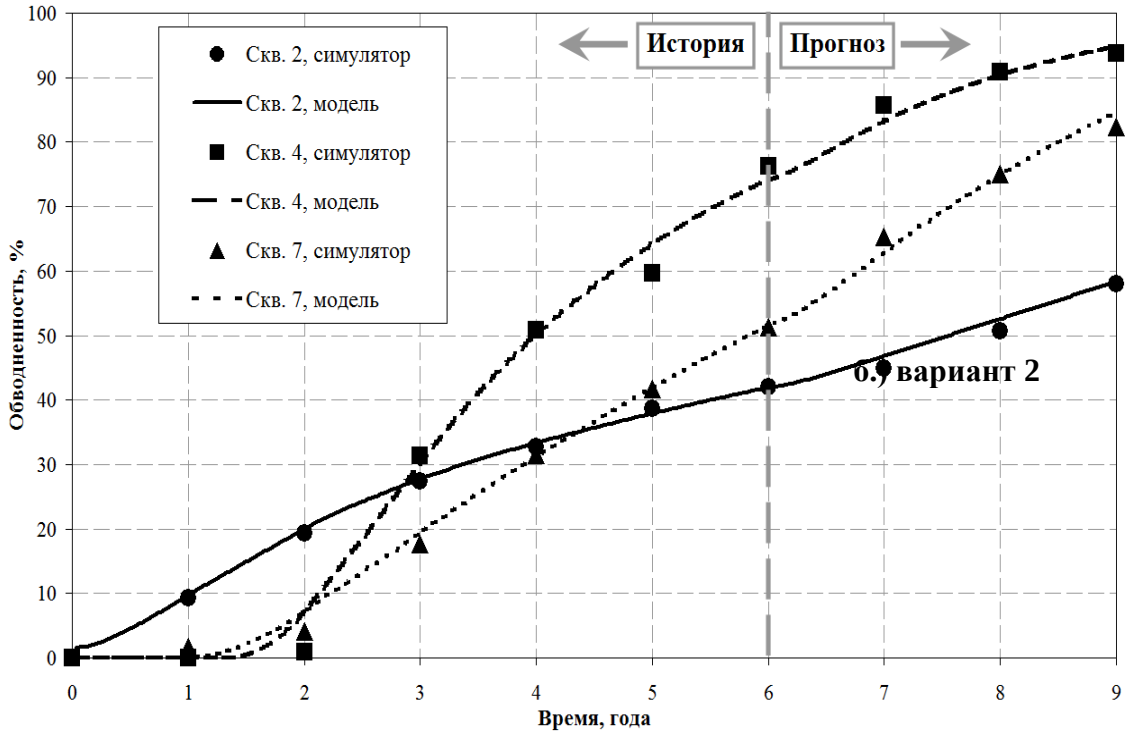


Рис. 5. Обводненность продукции скважин по вариантам

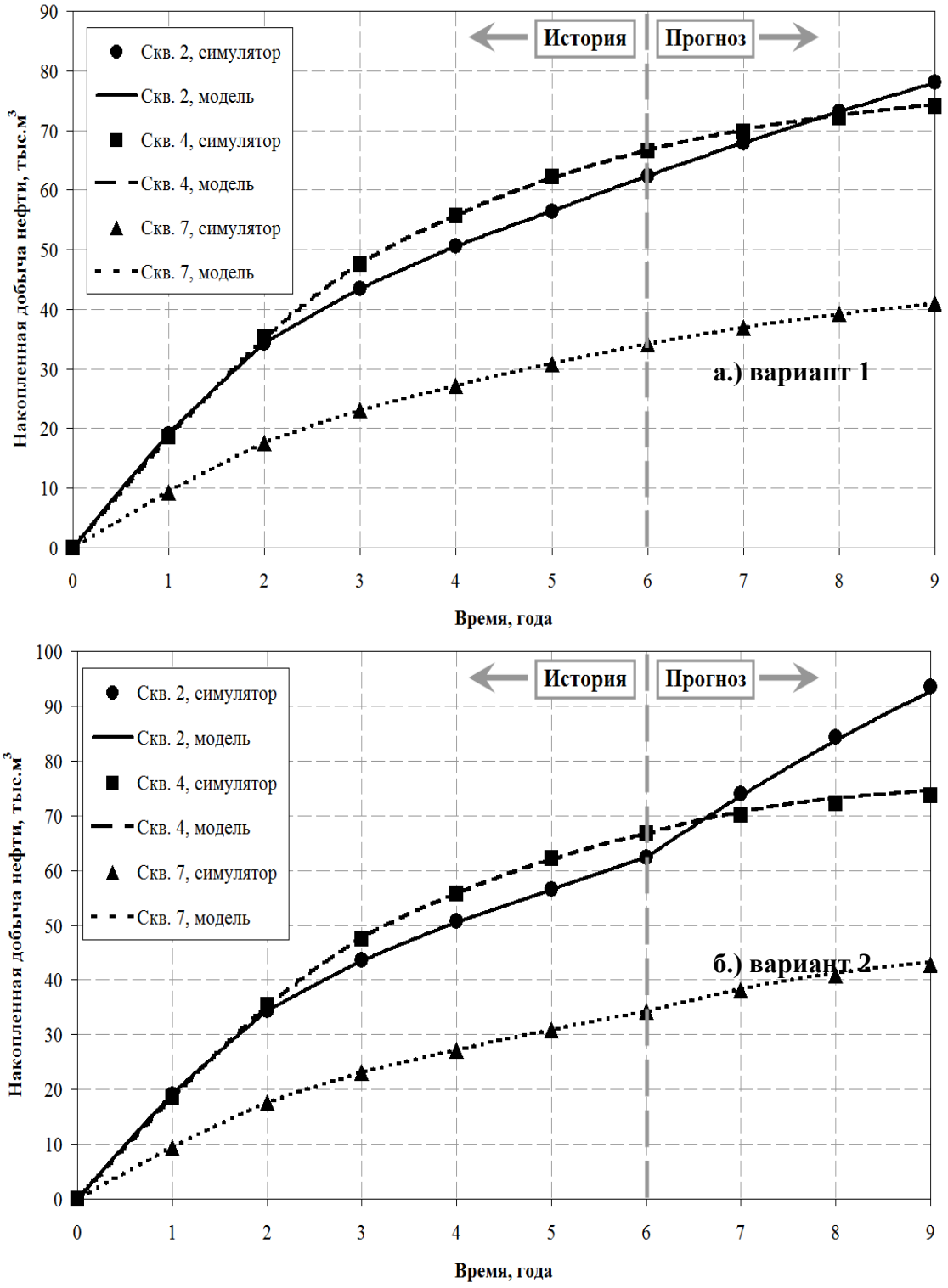


Рис. 6. Накопленная добыча нефти по вариантам

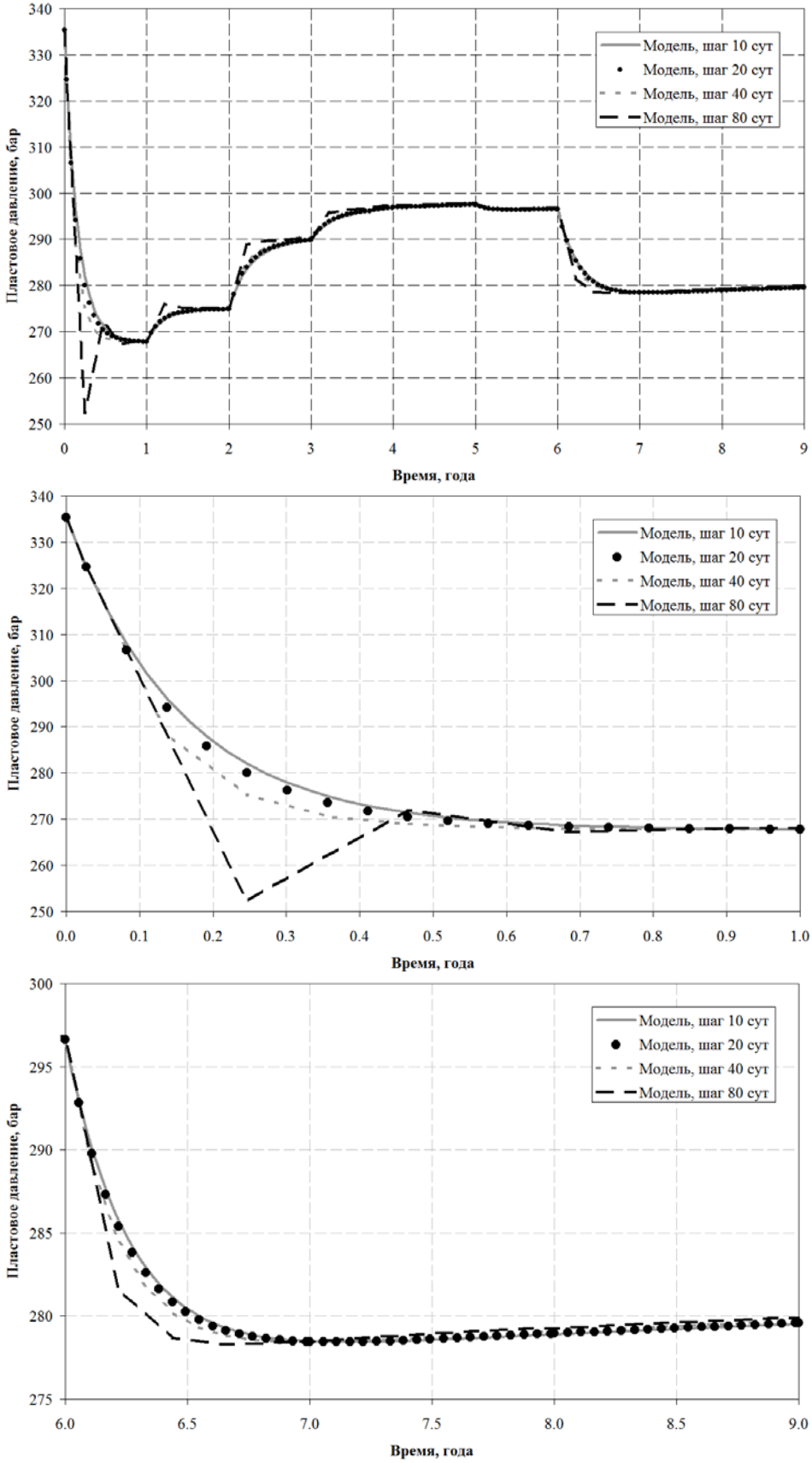


Рис. 7. Чувствительность пластового давления к шагу по времени (вариант 2)
с масштабированием участков 0-1 года и 6-9 лет

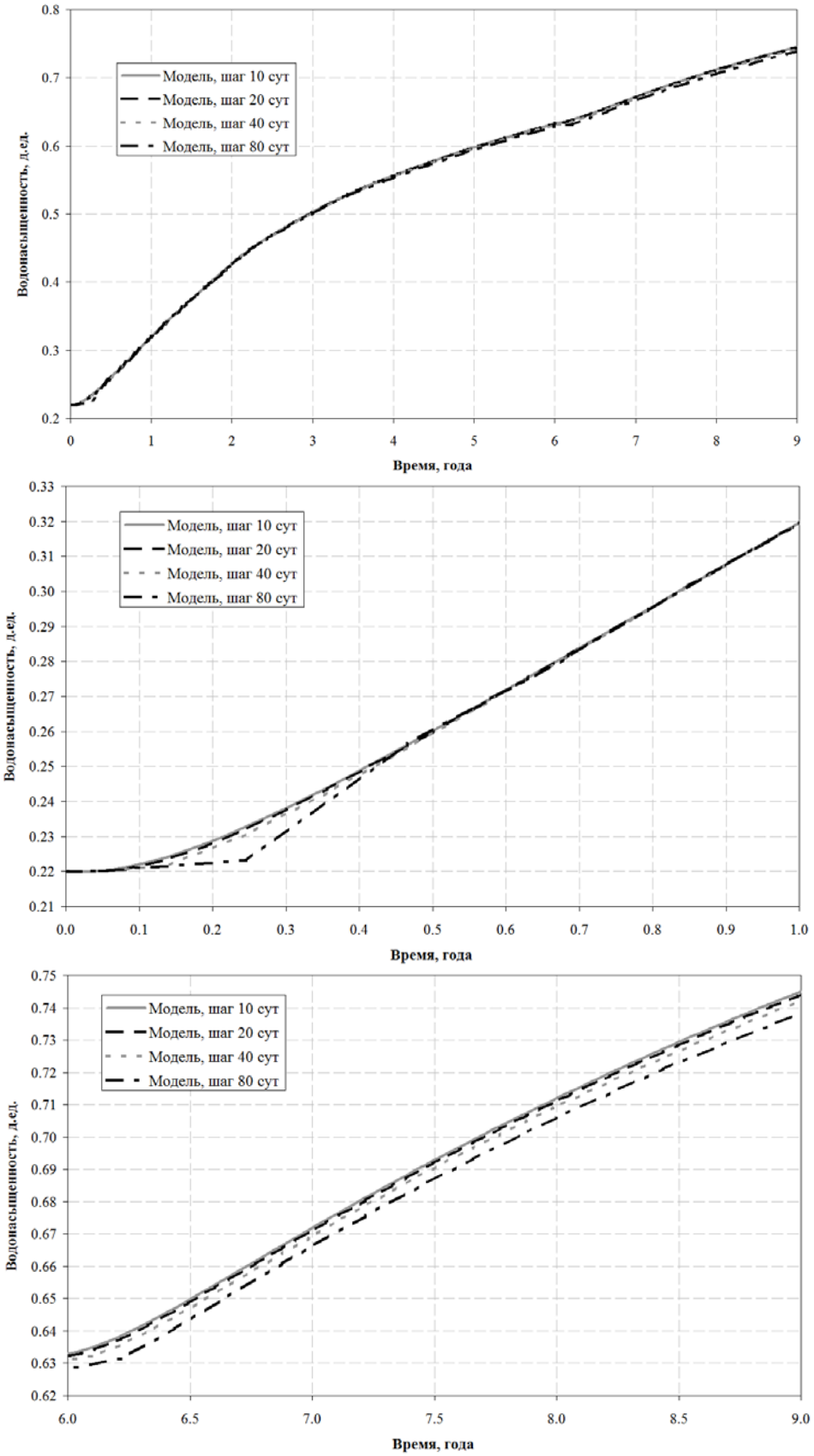


Рис. 8. Чувствительность водонасыщенности к шагу по времени (вариант 2)
с масштабированием участков 0-1 года и 6-9 лет

Заключение

Предлагается нульмерная математическая модель для анализа разработки нефтяных месторождений методом ячеек заводнения при пластовом и забойном давлении выше давления насыщения. Модель позволяет оценивать выработку запасов нефти, энергетический потенциал участка пласта, влияние законтурной области и количество вторгшейся из нее в нефтяную залежь воды. Кроме того, модель позволяет прогнозировать показатели разработки по скважинам. При этом скважины должны быть уже обводнены к моменту перехода на прогноз (параметры необводненных скважин принимаются по аналогии). В отличие от обычно применяемых для этих целей характеристик вытеснения, модель позволяет прогнозировать разработку не только в сложившихся условиях, но и при значительных изменениях уровней добычи жидкости и закачки воды по сравнению с историческими показателями.

Работа выполнена при поддержке гранта РФФИ 18-07-00677 А.

A null-dimensional mathematical model for the analysis of oil fields productions by the method of waterflood cells

I.V. Afanaskin, A.V. Korolev

Abstract: A null-dimensional mathematical model is derived for the oil field production analysis using the method of waterflooding cells. In addition to the classic capabilities of the waterflooding cell method, this model makes it possible to evaluate the development indices at well operating modes that differ from the basic ones. But the only limit in this case is that the reservoir and bottomhole pressures should be higher than the bubble point pressure. At the same time, the wells should already be flooded (the parameters of non-watered wells are taken by analogy). An approach is proposed for automatic history matching using the waterflood cell model. In the example given, good results for the model in question were obtained not only in history matching, but for the forecast too. The example gives an analysis of the time step effect on the reservoir pressure and water saturation calculation.

Keywords: flooding cells, production analysis, two-phase flow

Литература

1. Управление разработкой пласта. «TGL Reservoir Engineering Group», 2010.
2. И.Ф.Хатмуллин, Е.И.Хатмуллина, А.Т.Хамитов и др. Технология решения типовых задач по оптимизации заводнения. «Оптимизация заводнения на зрелых месторождениях»: Труды технической конференции SPE, 2013.
3. Acel. БАСПРО. [Электронный ресурс]. Режим доступа: http://www.baspro.ru/Acel
4. Атлас – Управление заводнением. ЗАО «ТИНГ». [Электронный ресурс]. Режим доступа: http://www.togi.ru/WaterFlooding
5. OFM Well and Reservoir Analysis Software. Schlumberger. [Электронный ресурс]. Режим доступа: http://www.software.slb.com/products/foundation/pages/ofm.aspx
6. Ш.К.Гиматудинов, Ю.П. Борисов, М.Д. Розенберг и др. Справочное руководство по проектированию разработки и эксплуатации нефтяных месторождений. Том 2. Проектирование разработки. М.: Недра, 1983. - 463 с.
7. Л.П.Дейк. Практический инжиниринг резервуаров. М.-Ижевск: Институт компьютерных исследований, НИЦ «Регулярная и хаотическая динамика», 2008. - 668 с.
8. М.Уолш, Л.Лейк. Первичные методы разработки месторождений углеводородов. М.-Ижевск: Институт компьютерных исследований, НИЦ «Регулярная и хаотическая динамика», 2008. - 672 с.
9. РД 153-39.1-004-96 Методическое руководство по оценке технологической эффективности применения методов увеличения нефтеотдачи. М.: Минтопэнерго РФ, НК Роснефть, РМНТК «Нефтеотдача», ВНИИнефть, 1996. – 88 с.
10. В.С.Евченко, Н.П.Захарченко, Я.М.Каган и др. Разработка нефтяных месторождений наклонно-направленными скважинами. М.: Недра, 1986. - 278 с.

Применение методики расчета двухфазной фильтрации жидкости в пористых средах на модели реального месторождения

О.И. Бутнев[1], В.Ю. Кузнецов[2], В.А. Пронин[3], М.Л. Сидоров[4], И.В. Афанаскин[5], А.В. Королев[6], П.В. Ялов[7]

[1,2,2,4] ФГУП «РФЯЦ - ВНИИЭФ», Саров, Россия, [5,6,7] ФГУ ФНЦ НИИСИ РАН, Москва, Россия

E-mail's: [1]boi@vniief.ru, [2]v.yu.kuznetsov@itmf.vniief.ru, [3]pronin@md08.vniief.ru, [4]M.L.Sidorov@itmf.vniief.ru
[6] ivan@afanaskin.ru, [6] alexandre.korolev@mail.ru, [7] petryalov@gmail.com

Аннотация: Ранее авторами была опубликована методика расчета двухфазной фильтрации жидкостей (на основе модели BlackOil) с использованием призматических структурированных и неструктурированных сеток, в которой для решения уравнений был применен полностью неявный конечно-объемный метод. Работоспособность методики была продемонстрирована на тестовой задаче, основанной на стандартном тесте SPE8. В данной статье приводятся результаты расчета гидродинамической модели реального месторождения по методике, изложенной ранее. Полученные результаты сравниваются с результатами коммерческого программного обеспечения.

Ключевые слова: двухфазная фильтрация, конечно-объемный метод, призматические сетки.

Введение

В работе [1] была представлена методика расчета двухфазной фильтрации и приведены результаты расчета тестовой задачи, основанной на тесте SPE8. В расчете использовались различные призматические структурированные и неструктурированные сетки. Результаты, полученные на различных сетках, имели хорошее качественное и количественное согласие с коммерческим программным продуктом Rubis [2].

Развитием представленной в [1] методики является проведение численного гидродинамического моделирования реального участка месторождения. Подобные модели являются сложными, т.к. в них учитываются как неоднородности геологического строения, так и скважинные данные, содержащие историю этапов разработки месторождения.

Приведем описание гидродинамической модели реального месторождения и сравним полученные результаты по упомянутой выше методике с результатами, полученными с помощью коммерческого программного обеспечения (КПО).

1. Описание гидродинамической модели

Моделировалось нефтяное месторождение, расположенное в Северо-Кавказском нефтегазоносном бассейне. Пласт представлен терригенным коллектором. Залежь нефти пластового типа с активной законтурной водоносной областью. Общий вид гидродинамической модели представлен на рис. 1. При дискретизации модели было использовано 47709 ячеек (57x31x27). Количество активных ячеек составляет порядка 33 тысяч. Месторождение запущено в разработку в 60-х годах. Оно эксплуатировалось в течение 22 лет. Затем в течение 3 лет постепенно останавливают свою работу все скважины. Месторождение стоит 6 лет. Затем в течение 6 лет скважины медленно начинают выводить из консервации. Всего история разработки насчитывает 37 лет. Затем следует прогнозный период.

Номера скважин и некоторые параметры изменены для сохранения конфиденциальности данных недропользователя.

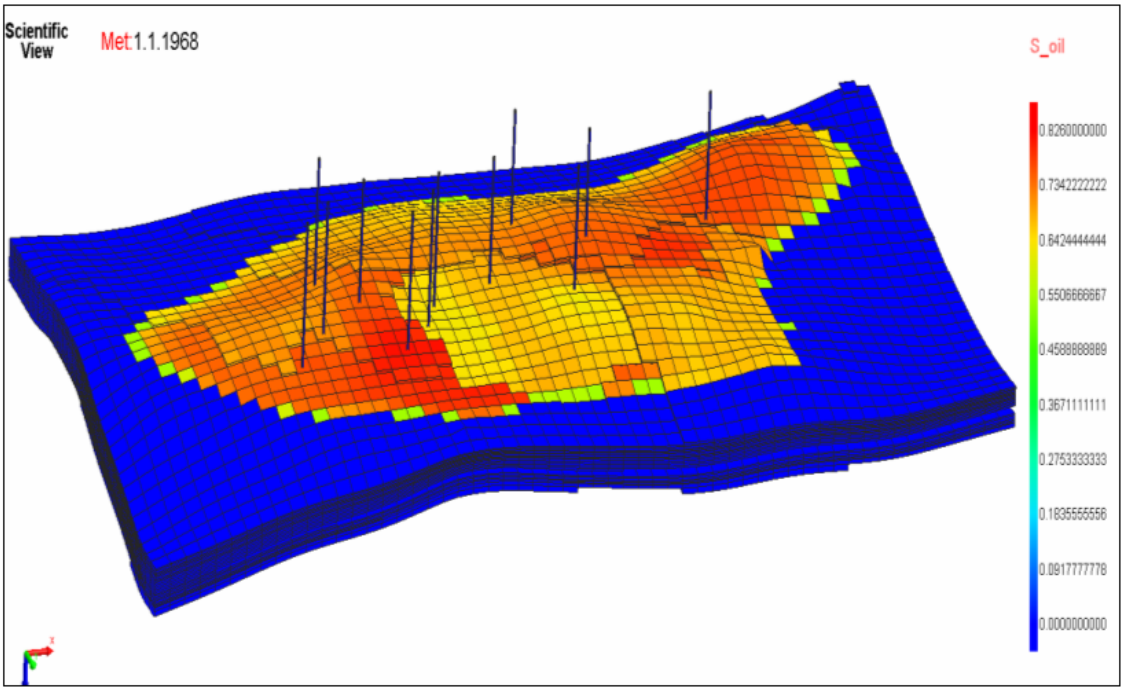


Рис. 1. Общий вид модели. Поле нефтенасыщенности на начальный момент времени, д.ед.

Месторождение разрабатывается при водонапорном режиме, с использованием 12 добывающих скважин, работающих в режиме заданного расхода жидкости (вода + нефть).

Процесс разработки моделируется с помощью модели двухфазной фильтрации, так как разработка месторождения осуществляется при пластовом давлении выше давления насыщения. PVT-свойства для воды и нефти указаны в табл. 1 и 2 соответственно.

Плотности жидкостей в стандартных условиях равны:

- для воды - 1056 кг/м^3,
- для нефти - 815 кг/м^3.

В табл. 3 представлены данные функций относительной фазовой проницаемости (ОФП). Капиллярным давлением пренебрегается по причине отсутствия достоверных данных.

Сжимаемость породы для всего месторождения составляет $4.7 \cdot 10^{-5}$ 1/бар.

Отбор извлеченной жидкости компенсируется притоком воды из водоносного пласта. Моделирование водоносного пласта осуществляется с помощью модели Фетковича [3].

Так как скважинные данные по истории разработки месторождения представляют достаточно большой объем даже для одной скважины, приведем историю разработки одной из скважин. В табл. 4 приведен режим эксплуатации скважины № 2. Графически история работы скважины № 2 по фазам представлена на рис. 2.

Таблица 1

PVT-свойства для воды

Давление, бар	B_w, **м^3/м^3**	**Сжимаемость, 1/бар**	**Вязкость, сПз**
339	1.02	$4.7 \cdot 10^{-5}$	0.36

Таблица 2

PVT-свойства для нефти

Давление, бар	B_o, **м^3/м^3**	**Сжимаемость, 1/бар**	**Вязкость, сПз**
339	1.55	$1.01 \cdot 10^{-4}$	0.397

Таблица 3

Функции относительной фазовой проницаемости (ОФП)

Водонасыщенность, д.ед., S_w	**ОФП по воде, д.ед., k_{rw}**	**ОФП по нефти, д.ед. k_{row}**
0	0	1
0.37	0	1
0.4046	0.013	0.242
0.4253	0.023	0.131
0.459	0.036	0.074
0.525	0.078	0.029
0.557	0.103	0.012
0.5839	0.143	0.01
0.6	0.191	0.008
0.63	0.502	0
1	1	0

Таблица 4

Режим работы скважины № 2 в период истории

Год	**Дебит нефти, $м^3$/сут**	**Дебит воды, $м^3$/сут**	**Дебит жидкости, $м^3$/сут**
1	108.702	2.232	110.934
2	97.061	1.275	98.337
3	65.244	1.081	66.326
4	50.035	5.769	55.804
5	44.753	5.263	50.016
6	48.590	6.361	54.951
7	69.330	8.273	77.603
8	68.217	9.691	77.909
9	78.770	14.908	93.678
10	69.430	28.652	98.083
11	5.643	12.767	18.410
12	16.690	5.494	22.184
13	13.423	3.630	17.052
14	5.905	3.753	9.658
15	3.762	4.611	8.372
16	2.461	3.998	6.459
17	2.966	5.022	7.988
18	3.856	6.222	10.077
19	3.681	5.875	9.556
20	3.731	6.289	10.020
21	3.212	6.125	9.337
22	3.343	5.159	8.503
23	2.971	3.246	6.217
24	0.0	0.0	0.0
25	0.0	0.0	0.0

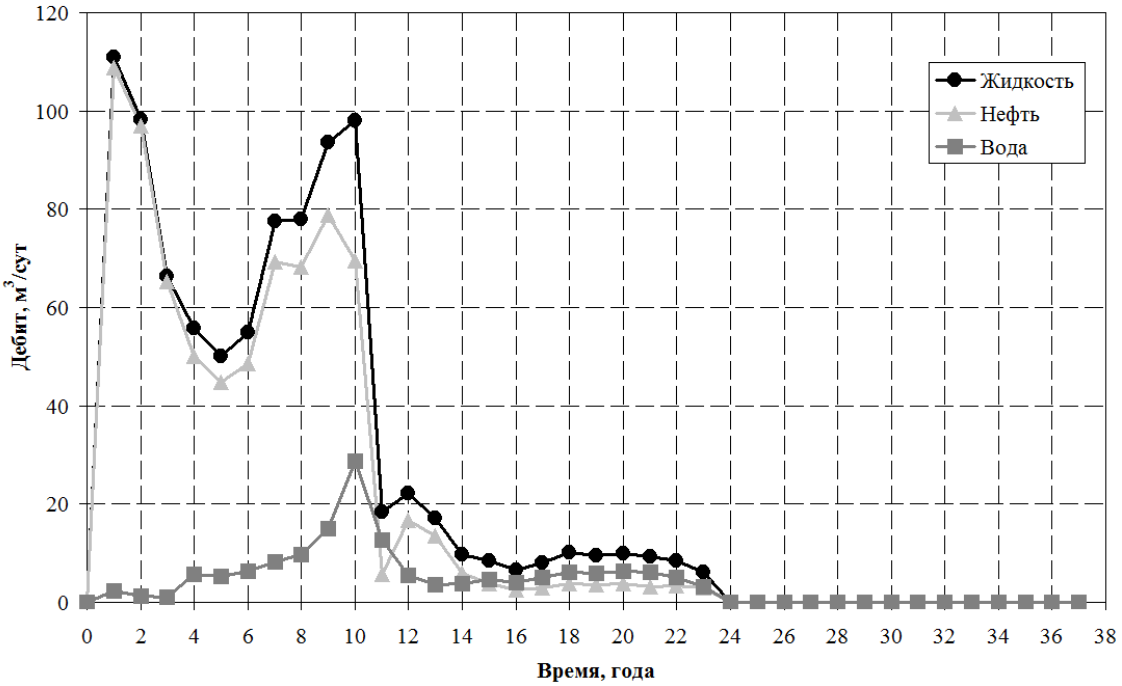


Рис. 2. Режим работы скважины № 2 в период истории

Из табл. 4 видно, что доля нефти в общем потоке постепенно снижается. Поскольку все скважины работают в режиме фонтанирования благодаря достаточно высокому газовому фактору (более 200 $м^3/м^3$), то в процессе обводнения вместе с падением доли нефти в потоке жидкости падает и дебит газа, что вызывает падение дебита жидкости (падает эффективность газлифта).

История разработки насчитывает 37 лет. Затем следует прогноз на 33 года. Прогнозные дебиты выбирались как максимальные стабильные дебиты за историю разработки в предположении перевода эксплуатации скважин с фонтанного режима работы на насосный (оборудование скважин электрическими центробежными насосами - ЭЦН), а так же производства ремонтно-изоляционных работ (РИР) и обработки призабойных зон (ОПЗ) с целью интенсификации добычи нефти и уменьшения добычи воды.

Таблица 5

Прогнозные дебиты скважин

Номер скважины	Дебит жидкости, $м^3$/сут
2	110.9
4	92.6
7	93.2
26	47.5
27	95.0
28	58.1
29	83.0
30	31.6
37	206.8
38	182.0
39	8.5
40	11.7

2. Модель водоносного пласта Фетковича

Модель водоносного пласта Фетковича использует упрощенный подход, основанный на псевдоустановившемся коэффициенте продуктивности и материальном балансе между давлением водоносного пласта и накопленным оттоком воды из него.

В модели предполагается, что отклик давления ощущается равномерно во всем водоносном пласте. Модель Фетковича наиболее подходит для небольших водоносных пластов, которые быстро приходят в псевдоустановившееся состояние.

Приток из водоносного пласта описывается уравнением:

$$Q_{ai} = \frac{d}{dt}\left(W_{ai}\right) = J\alpha_i\left[p_a - p_i + \rho g\left(d_i - d_a\right)\right] \quad (1)$$

где

Q_{ai} - интенсивность притока из водоносного пласта в присоединенный блок модели *i*;

W_{ai} - накопленный приток из водоносного пласта в присоединенный блок *i*;

J - заданный индекс продуктивности водоносного пласта;

α_i - часть области, соединяющаяся с блоком *i*;

p_a - давление в водоносном пласте в момент времени *t*;

p_i - давление в присоединенном блоке *i* в момент времени *t*;

ρ - плотность воды в водоносном пласте;

d_i - глубина блока сетки;

d_a - заданная глубина водоносного пласта.

Часть области для каждого соединения с блоком сетки задается следующим соотношением:

$$\alpha_i = \frac{m_i A_i}{\sum m_i A_i}, \qquad (2)$$

где

A_i - площадь грани блока, соединяющегося с водоносным пластом;

m_i - множитель коэффициента притока водоносного пласта.

Множитель притока m_i используют для облегчения процесса согласования истории добычи.

Отклик давления в водоносном пласте определяется уравнением материального баланса:

$$W_a = C_t V_{0w}\left(p_{a0} - p_a\right), \qquad (3)$$

где

W_a - накопленный приток из водоносного пласта;

C_t - полная (порода + вода) сжимаемость водоносного пласта;

V_{w0} - начальный объем воды в водоносном пласте;

p_{a0} - начальное давление воды в водоносном пласте.

Поведение водоносного пласта существенно зависит от двух параметров: постоянной времени водоносного пласта и индекса продуктивности. Постоянная времени водоносного пласта определяется выражением:

$$T_c = \frac{C_t V_{w0}}{J} \qquad (4)$$

и имеет размерность времени.

В предположении об однородности давления в залежи и в присоединенных блоках сетки, при интегрировании уравнений (1) и (3) средняя интенсивность притока на интервале времени выражается следующей формулой:

$$\overline{Q}_{ai} = J\alpha_i\left[p_a - p_i + \rho g\left(d_i - d_a\right)\right]\times \times \frac{1-\exp\left(\Delta t/T_c\right)}{\Delta t/T_c} \qquad (5)$$

где Δt - временной шаг.

В конце каждого временного шага полный накопленный приток из водоносного пласта увеличивается, а его давление пересчитывается по формуле (3).

Для представленной гидродинамической модели подход Фетковича можно изобразить в виде пласта (рис. 3), в котором цветом выделены ячейки «контактирующие» с водоносным пластом. В разностных уравнениях для этих сеточных блоков появляется дополнительное слагаемое-источник, интенсивность которого рассчитывается по формуле (5).

3. Результаты сравнения с коммерческим программным обеспечением

На рис. 4-6 представлены графики накопленной добычи нефти для скважин №№ 2, 7 и 37. Из этих графиков видно, что на этапе с заданной историей разработки по всем скважинам получено отличное качественное и количественное согласие расчетов по рассматриваемой методике и КПО.

На прогнозном этапе имеются небольшие отклонения, но в целом характер кривых схожий.

На рис. 7 представлен график накопленной добычи нефти по пласту. На нем сохранилась тенденция скважинных графиков. Отклонение результатов на момент окончания расчета составляет 0.66%, что находится в границах допустимого (стандарт отрасли – не более 5%).

По накопленный притоку воды из аквифера (водоносного пласта), представленного на рис. 8, так же получены хорошие результаты совмещения методики и КПО.

В целом, расчет представленной гидродинамической модели реального месторождения показал, что на данном типе задач разрабатываемая методика может давать результаты, согласующиеся с результатами КПО.

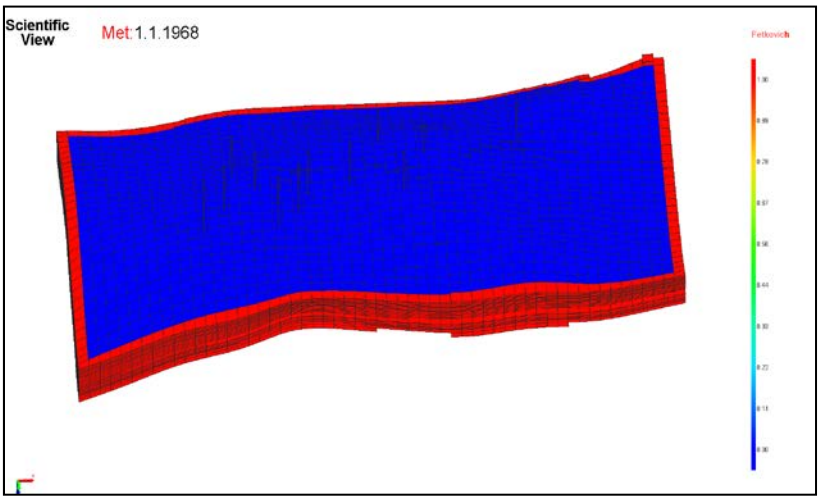


Рис. 3. Пласт с выделенными ячейками, контактирующими с водоносной областью

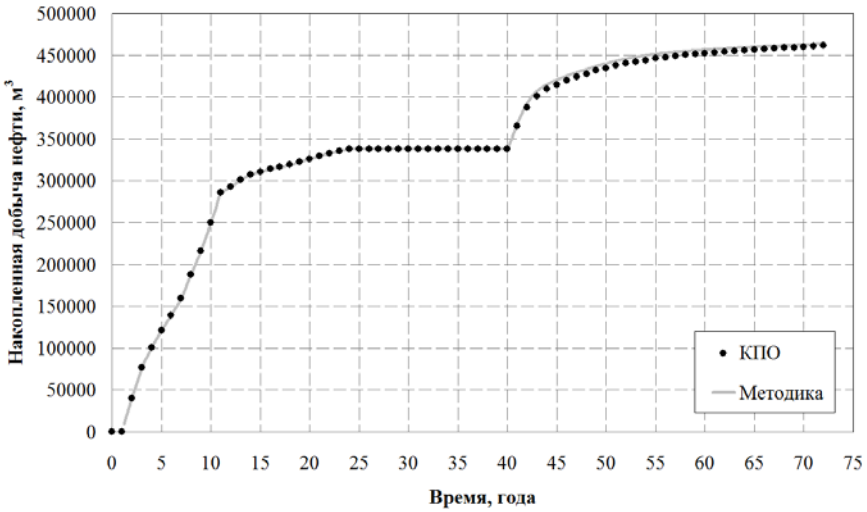


Рис. 4. Накопленная добыча нефти по скважине № 2. Методика и КПО

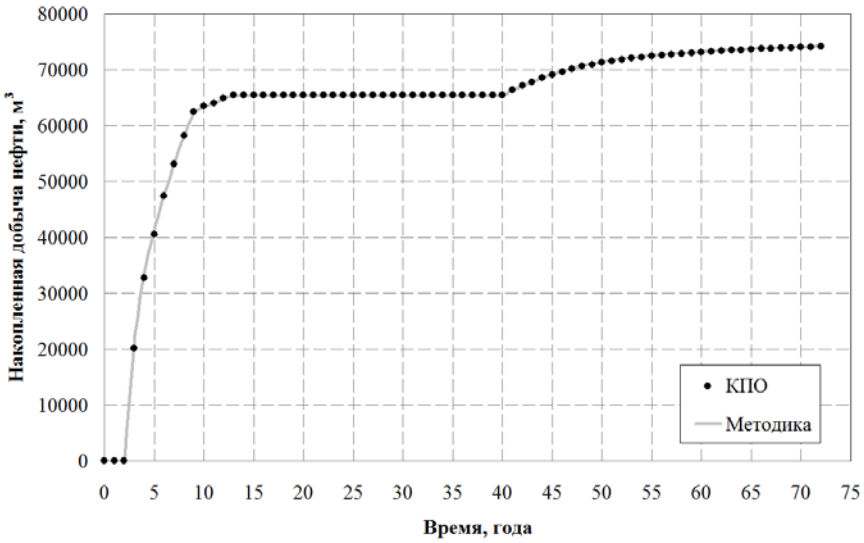


Рис. 5. Накопленная добыча нефти по скважине № 7. Методика и КПО

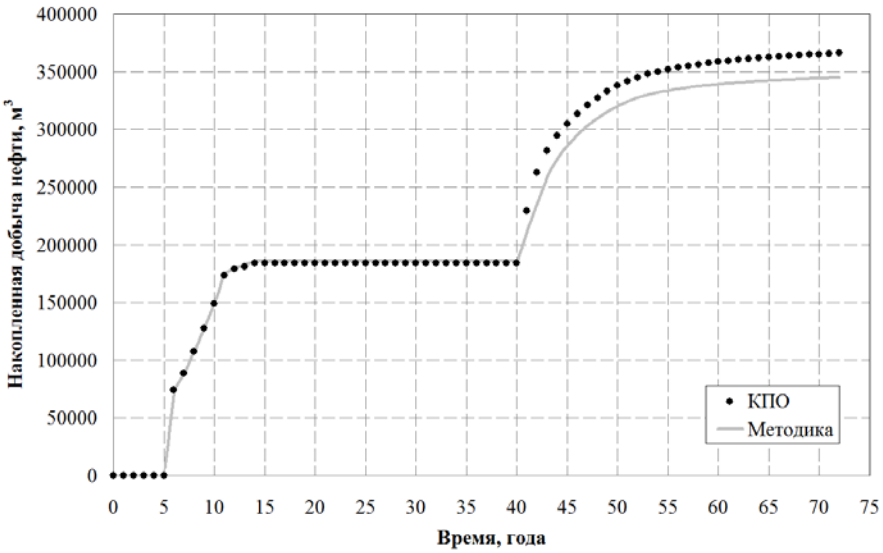


Рис. 6. Накопленная добыча нефти по скважине № 37. Методика и КПО

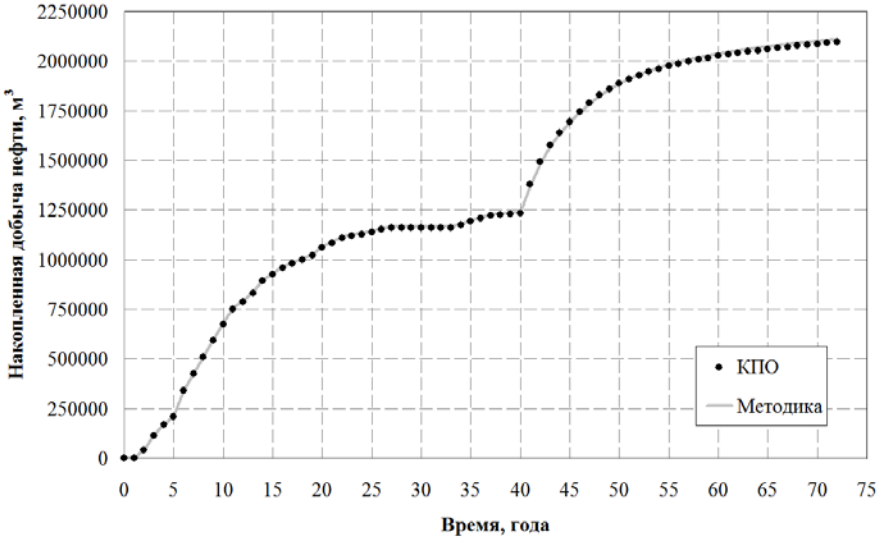


Рис. 7. Накопленная добыча нефти по пласту. Методика и КПО

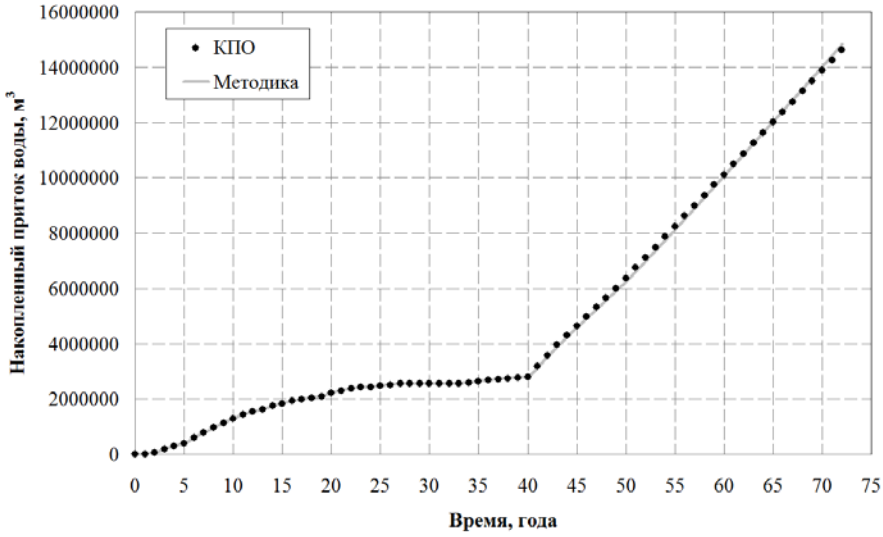


Рис. 8. Накопленный приток воды из аквифера (законтурной водоносной области)

Заключение

Основной целью выполненной работы является демонстрация развития методики, изложенной в работе [1], в виде возможности расчета модели реального месторождения. В статье приведены результаты расчета гидродинамической модели реального месторождения, разрабатываемого 12-ю скважинами с длительной историей разработки. Кроме того, сеточная модель месторождения имеет особенность в виде части активных ячеек от общего числа. Представленные результаты расчета хорошо согласуются с результатом, полученным по КПО (на момент окончания расчета отличие по накопленной добыче нефти по пласту составляет 0.66%). Это подтверждает работоспособность разработанной методики на данном классе задач.

Работа выполнена при поддержке гранта РФФИ 18-07-00671 А.

Special Methodology Application in Modelling of Two-Phase Fluid Flow in Porous Media on a Real Oil Field

O.I. Butnev, V.Ju. Kuznecov, V.A. Pronin, M.L. Sidorov, I.V. Afanaskin, A.V. Korolev, P.V. Jalov

Abstract: Results of hydrodynamic modelling on a real oil field are reported. Modelling was done by using a special methodology published earlier by authors. Proposed methodology of two-phase flow modelling is based on a “Black Oil” model with using a prismatic structured and unstructured grids and totally implicit finite-volumetric approach. Earlier this method was tested by using standard SPE8 test. In this article it was tested using real oil-field data and compared with results obtained by using standard commercial software.

Keywords: two-phase flow, finite-volumetric approach, prismatic grids

Литература

1. О.И.Бутнев, В.Ю.Кузнецов, В.А.Пронин, М.Л.Сидоров, Р.М.Кац, И.В.Афанаскин, А.В.Королев, П.В.Ялов. Методика расчета двухфазной фильтрации жидкости в пористых средах на призматических сетках различного вида // Вестник кибернетики. 2018. №3.
2. Kappa Engineering [Электронный ресурс] / Rubis – численное моделирование – Описание – Электрон. дан. – Режим доступа: https://www.kappaeng.com/software/rubis/overview (дата обращения 02.09.2018 г.)
3. M.J.Fetkovich. A Simplifed Approach to Water Influx Calculation – Finite Aquifer Systems // JPT. – July 1971. - P. 814-828.

Учёт факторов, влияющих на фазовое состояние «парафинов» в пластовых нефтях

К.Д.Ашмян [1], А.К.Пономарев [2], О.В.Ковалева [3]

[1, 2] ФГУ ФНЦ НИИСИ РАН. Москва. Россия, [3] АО «ВНИИнефть». Москва. Россия,

E-mail's: [1]kdashmyan@yandex.ru, [2]akponomarev@mail.ru, [3]olgakovaleva57@ mail.ru

Аннотация: Рассмотрен вопрос учета факторов, влияющих на фазовое состояние «парафинов» в пластовых нефтях. Приведены формулы и расчетная таблица для определения температуры насыщения нефти парафином при различном газосодержании и соответствующие графические зависимости. Показано влияние асфальто-смолистых компонентов на выпадение (образование) твердой фазы в пластовой нефти.

Ключевые слова: нефть пластовая, фазовое состояние, температура насыщения нефти парафином, газосодержание, смолы, асфальтены, парафин.

В статье «Развитие методов оценки фазового состояния парафинов в пластовых условиях [1] нами были рассмотрены вопросы образования твердой углеводородной фазы в пластовой нефти в результате изменения термобарических условий залегания нефти в нефтенасыщенном пласте. Основным тезисом статьи является то, что образование твердой фазы происходит при изменении пластовой температуры и пластового давления [2, 3].

Это достаточно общее положение потребовало более детального, дифференцированного подхода при определении температуры насыщения нефти парафином расчетным методом для нефтей с высоким содержанием асфальтенов и смолистых компонентов.

На рисунке 1 построена зависимость температуры насыщения дегазированной нефти парафином от содержания парафина в нефти, построенная по экспериментальным данным исследования нефтей более чем 500 залежей, расположенных в различных нефтедобывающих районах РФ. Использованные при построении данные находятся в диапазоне от 1,5 до 30% масс. содержания парафина в нефти.

По имеющимся данным была построена корреляционная зависимость, которая выражается уравнением:

$$t_{нас.дег.} = 15{,}345 \ln C + 12{,}03, \qquad (1)$$

где С – содержание парафина в нефти, % масс.

Однако, полученная по уравнению (1) величина температуры насыщения нефти парафином может характеризовать фазовое состояние нефти только в поверхностных условиях, т.к. в дегазированной нефти не содержится растворенный газ.

Для оценки влияния давления и количества растворенного в нефти газа в пластовых условиях на температуру насыщения нефти парафином применяется так же корреляционная формула [2, 3]:

$$t_{нас.пл.} = t_{нас.дег.} + 0{,}2P - 0{,}1G, \qquad (2)$$

где Р - текущее пластовое давление, МПа, G - газосодержание пластовой нефти, $м^3/м^3$.

Коэффициенты 0,2 для давления и 0,1 для газосодержания являются коэффициентами корреляции, которые были получены при обработке данных для различных месторождений, при выявлении зависимости влияния давления и газосодержания на температуру насыщения нефти парафином.

Углы наклона полученных зависимостей являются коэффициентами корреляции в данном уравнении (2).

Среднеквадратичное отклонение коэффициента, учитывающего влияние давления на температуру насыщения нефти парафином, составляет 2%, учитывающего влияние газосодержания - до 5% [2, 3].

Определение температуры насыщения нефти парафином пластовых нефтей по графическим зависимостям

На рисунке 1 представлены результаты экспериментальных определений температуры насыщения нефти парафином пластовых нефтей, полученных экспериментально и сгруппированных в зависимости от газосодержания.

Т.о., получены 3 новые кривые К2, К3 и К4, отличающиеся от К1 (дегазированная нефть) тем, что они характеризуют степень газонасыщения пластовой нефти.

Систематизация и анализ свойств пластовых нефтей, «потенциально насыщенных» парафином, выявили зависимость температуры насыщения пластовой нефти парафином от отношения газосодержания к пластовому давлению G/Рпл.

Полученная зависимость температуры насыщения пластовых нефтей парафином от содержания в них парафинов в интервале $1{,}5 \leq C \leq 30$ при различном отношении $G/P_{пл.}$, представлены в таблице 1.

Таким образом, в случае невозможности проведения прямых исследований температуры насыщения пластовой (газонасыщенной) нефти парафином, необходимые данные могут быть получены с использованием результатов по разгазированной нефти и расчетов по уравнению (2) (см. Рисунок 1 и таблица 1).

В статье [1] было сказано, что под «парафином» в нефтяной промышленности подразумевается конгломерат АСПО, состоящий из А- асфальтенов, С – смол и П-парафиновых углеводородов, а также примесей.

АСПО представляют собой сложную углеводородную смесь, состоящую из парафинов, асфальто-смолистых компонентов, а также масел, воды и механических примесей.

Содержащиеся в АСПО асфальтены – это вещества темно-бурого или черного цвета, плотностью более единицы, массовое содержание которых в нефти достигает 5%. Асфальтены являются наиболее тугоплавкой и малорастворимой частью отложений тяжелых компонентов нефти. При нагревании асфальтены не плавятся, становятся пластичными при температуре 300°С, при более высокой температуре разлагаются на газообразные и жидкие вещества и твердый остаток – кокс.

В составе смол, содержащихся в АСПО, входит большое число элементов, основными из которых являются углерод, водород, кислород, сера и азот. Выделенные из нефти смолы темно-коричневого цвета, имеют мазеобразную консистенцию. Плотность смол составляет около 1кг/м3. Содержание смол в нефти может достигать 30%.

Изучение состава АСПО различных нефтей РФ выявило, что на механизм выпадения «парафина» в основном влияет изменение фазового состояния высокопарафиновых углеводородов и в меньшей степени – асфальтенов [4].

АСПО в нефти представлен смесью алкановых углеводородов различной молекулярной массы, отдельные компоненты которой переходят в твердое состояние при индивидуальной температуре кристаллизации высокомолекулярных углеводородов парафинового ряда [5]. Максимальная температура кристаллизации высокомолекулярных парафиновых углеводородов не превышает +70°С.

Основным фактором, влияющим на образование твердой фазы, являются изменение фазового состояния высокомолекулярных углеводородов парафинового ряда, при этом «выпавший» парафин адсорбирует на своей поверхности асфальтены, смолы, а также механические примеси пластовой нефти. Перераспределяясь между частичками асфальтенов, выпавшие кристаллики парафиновых углеводородов начинают формировать кристаллический каркас в объеме, образуя структуру коагуляционно – кристаллического типа [5].

Образовавшийся конгломерат имеет размеры, сопоставимые с размером пор породы нефтяного пласта и может при фильтрации добываемой нефти значительно уменьшить добычу или же прекратить фильтрацию вообще.

Влияние содержащихся в пластовой нефти асфальтенов и смол на образование твердой фазы в жидкой пластовой нефти потребовало учета их содержания на зависимости, представленные на рисунке 2.

Проведенные экспериментальные исследования показали, что интенсивность выпадения асфальтосмолпарафиновых отложений (АСПО) в зависимости от содержания в нефти асфальтенов и смолистых компонентов [1, 2] показали, что

присутствие в нефтяной дисперсной системе асфальтенов и смол может так же приводить их к депрессорным эффектам.

В работе [4] нами показано, что процесс образования твердой фазы зависит от соотношения асфальтенов (А) и смолистых соединений (С) в составе нефти.

В связи с увеличением параметра А/С температура насыщения нефти парафином снижается – ассоциаты асфальтенов в нефти менее стабилизированы из-за недостатка стабилизирующих компонентов (смол), что приводит к уменьшению температуры насыщения.

Процесс кристаллизации парафинов таких нефтей подавляется ассоциатами, и отложения парафина не происходит.

При небольших значениях параметра А/С наоборот, температура насыщения возрастает, т.к. смолы не оказывают воздействия на парафинообразование, парафин свободно выделяется из нефти [5] .

Влияние количества асфальтенов и смол в дегазированной нефти на образование АСПО оценивается по соотношению (3):

$$П/(А+С) \qquad (3).$$

ЗАКЛЮЧЕНИЕ

В статье рассмотрен вопрос учета факторов, влияющих на фазовое состояние «парафинов» в пластовых нефтях. Приведены формулы для построения зависимости температуры насыщения нефти парафином от его содержания в дегазированной и пластовой нефти.

Приведена расчетная таблица для определения температуры насыщения нефти парафином при различном газосодержании и соответствующие графические зависимости.

Показано влияние асфальто-смолистых компонентов на выпадение (образование) твердой фазы в пластовой нефти.

Работа выполнена при поддержке гранта РФФИ 18-07-00679А.

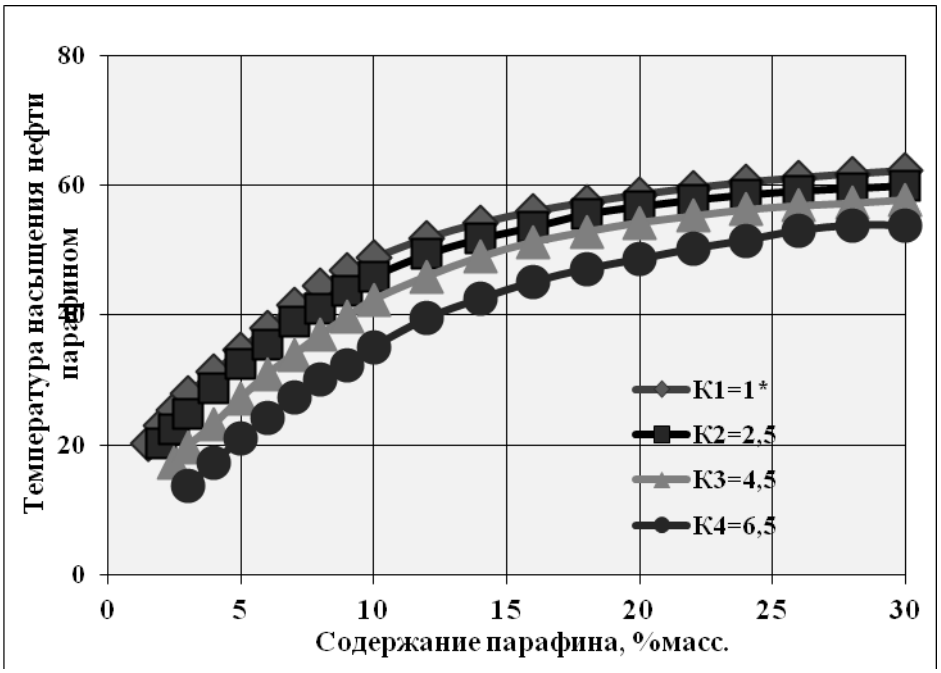


Рис. 1. Зависимость температуры насыщения пластовой нефти парафином от содержания парафина при различном отношении К=G/Рпл.

Таблица 1. Зависимость температуры насыщения пластовых нефтей парафином от содержания в них парафинов

Содержание парафина, %	Температура насыщения пластовой нефти парафином при различном отношении $К=G/P_{пл.}$			
	$К_1$=1*	**$К_2$=2,5**	**$К_3$=4,5**	**$К_4$=6,5**
1,5	**18**			
2	**23**	20,2		
2,5	**26**	22,4	17,2	
3	**29**	24,8	19,7	13,7
4	**33**	28,7	23,2	17,3
5	**37**	32,4	27,2	21,1
6	**40**	35,5	30,9	24,2
7	**42**	39	33,9	27,4
8	**44**	41	37	30
9	**46**	43,8	39,7	32,3
10	**47**	46	42,4	35,1
12	**50**	49,4	46	39,5
14	**53**	51,8	49	42,5
16	**55**	53,6	51,3	45,1
18	**56**	55,6	52,9	47,1
20	**58**	56,8	54,3	48,6
22	**59**	57,8	55,3	50,2
24	**61**	58,6	56,3	51,5
26	**62**	59,2	56,9	53
28	**63**	59,6	57,3	53,8
30	**64**	60	57,8	53,8

Примечание: * - для дегазированных нефтей

The factors that influence the phase state of "paraffin" in situ oil

K. D. Ashmyan, A. K. Ponomarev, A. K., O. V. Kovaleva

Abstract: The issue of taking into account factors affecting the phase state of "paraffins" in reservoir oils is Considered. Formulas and calculation table for determination of oil saturation temperature with paraffin at different gas content and corresponding graphic dependences are given. The influence of asphalt-resin components on the deposition (formation) of the solid phase in the formation oil is shown.

Keywords: oil reservoir, phase state, oil saturation temperature with paraffin, gas content, resins, asphaltenes, paraffin.

Литература

1. К.Д.Ашмян, А.К.Пономарев, О.В.Ковалева. Развитие методики оценки фазового состояния парафинов в пластовых нефтях // Труды НИИСИ РАН, 2018, т. 8, №2, стр. 120 – 124.
2. РД 39-0147035-226-88 «Методическое руководство по выявлению залежей нефтей, насыщенных парафином».
3. К.Д.Ашмян, О.В.Ковалева. И.Н.Никитина. Методическое руководство по выявлению залежей нефтей, насыщенных парафином, и оценке фазового состояния парафинов в пластовых нефтях // Вестник ЦКР РОСНЕДРА, № 6, 2011, с -11 - 14
4. К.Д.Ашмян, И.Н.Никитина, Е.Н.Носова. Факторы, влияющие на выпадение асфальтосмолистых отложений (АСПО) из нефти // Нефтяное хозяйство. - 2014. - № 11. - С. 126-128.
5. В.М.Татевский. Физико-химические свойства индивидуальных углеводородов. М. Гостоптехиздат. 1960. – 412 с.

Система управления виртуальной моделью марсохода

М.В. Михайлюк[1], Е.В. Страшнов[2], Л.А. Финагин[3]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1]mix@niisi.ras.ru, [2]strashnov_evg@mail.ru

Аннотация: В работе предлагается метод управления виртуальной трехмерной моделью марсохода с помощью виртуального пульта и функциональной схемы расчета управляющих сигналов. Управление позволяет осуществлять движение марсохода, раскрытие и сворачивание солнечных батарей, включение и выключение фар, повороты видео камер, а также выполнение операций забора грунта с помощью бура. Апробация системы управления моделью марсохода проводилась в рамках комплекса виртуального окружения, разработанного в ФГУ ФНЦ НИИСИ РАН.

Ключевые слова: система виртуального окружения, система управления, модель марсохода

Введение

Одним из важных направлений современной космонавтики является исследование планет с помощью мобильных роботов. Это связано, прежде всего, со сложностью полета и высадки космонавтов на поверхности планет, опасной для человека средой на этих планетах, обеспечением жизнедеятельности человека в космическом пространстве и т.д. Мобильные роботы позволяют изучить состав поверхности и атмосферы планеты, взять образцы грунта и воздуха, получить качественные снимки участков, передать накопленные данные на Землю или орбитальную станцию и т.д. Они могут двигаться как автономно, используя компоненты искусственного интеллекта, так и под управлением оператора. Обучение операторов управлению такими роботами на первом этапе целесообразно осуществлять на виртуальных трехмерных моделях [1, 2]. Это позволит избежать поломки дорогостоящего оборудования, отработать выполнение различных технологических операций, усовершенствовать эргономику пультов управления, уменьшить время и повысить качество обучения операторов. В работе [3] рассматривается задача моделирования движения модели робота-марсохода по виртуальной модели марсианской поверхности с целью оптимизации различных параметров его движения. В работе [4] описывается система управления марсоходом в рамках программы Экзомарс. В ФГУ ФНЦ НИИСИ РАН разработана система управления виртуальными объектами, которая включает создание пультов управления и функциональных схем расчета управляющих сигналов по степени воздействия оператором на элементы управления пульта [5]. В данной работе рассматривается задача моделирования управления трехмерной виртуальной моделью мобильного колесного робота (марсохода) с помощью виртуального пульта и функциональной схемы управления.

1. Трехмерная модель марсохода и участка поверхности Марса

Трехмерная модель марсохода (Рис. 1) создана в системе моделирования 3DS MAX, содержит более 80 тысяч полигонов (треугольников), а также специальные объекты (электрические двигатели и датчики). Модель перемещается на колесах, прикрепленных к корпусу марсохода с помощью подвесок продольно-осевого типа. Каждое колесо снабжено электрическим двигателем, который приводится в действие подаваемым на него напряжением от аккумулятора. Величина напряжения регулируется в процессе управления. На корпусе закреплены солнечные батареи, которые можно складывать и раскрывать, прожектор, камеры и антенны. Кроме того, на мачте робота установлена камера с широким углом обзора 120°, предназначенная для получения панорамного изображения поверхности Марса. В нижней части горизонтально расположен бур, пред-

назначенный для взятия проб грунта. На боковой поверхности бура располагается фара для подсвечивания участка бурения. Эти элементы также снабжены электрическими двигателями, с помощью которых можно перемещать их, используя пульт управления.

Модель марсохода находится на модели участка марсианской поверхности размером 450x450 метров, содержащей более 300 тысяч полигонов и включающей плоские и гористые элементы, а также камни различных размеров (Рис. 2). Технологии создания трехмерных моделей поверхностей планет описаны в [6].

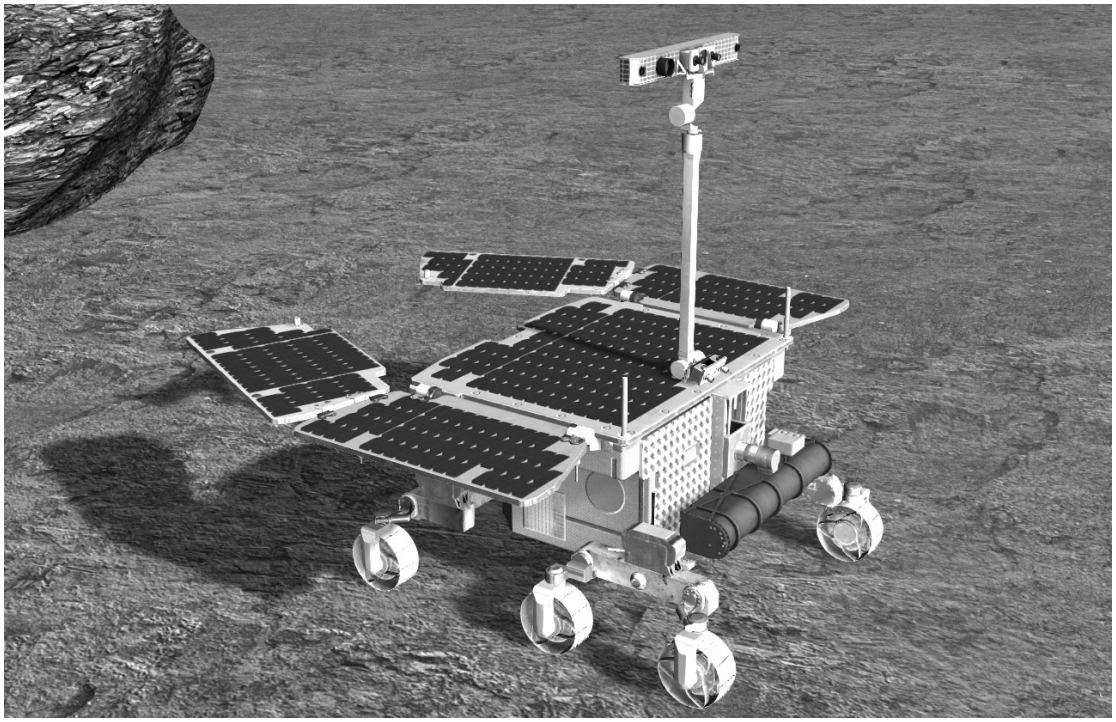


Рис. 1. Виртуальная трехмерная модель марсохода

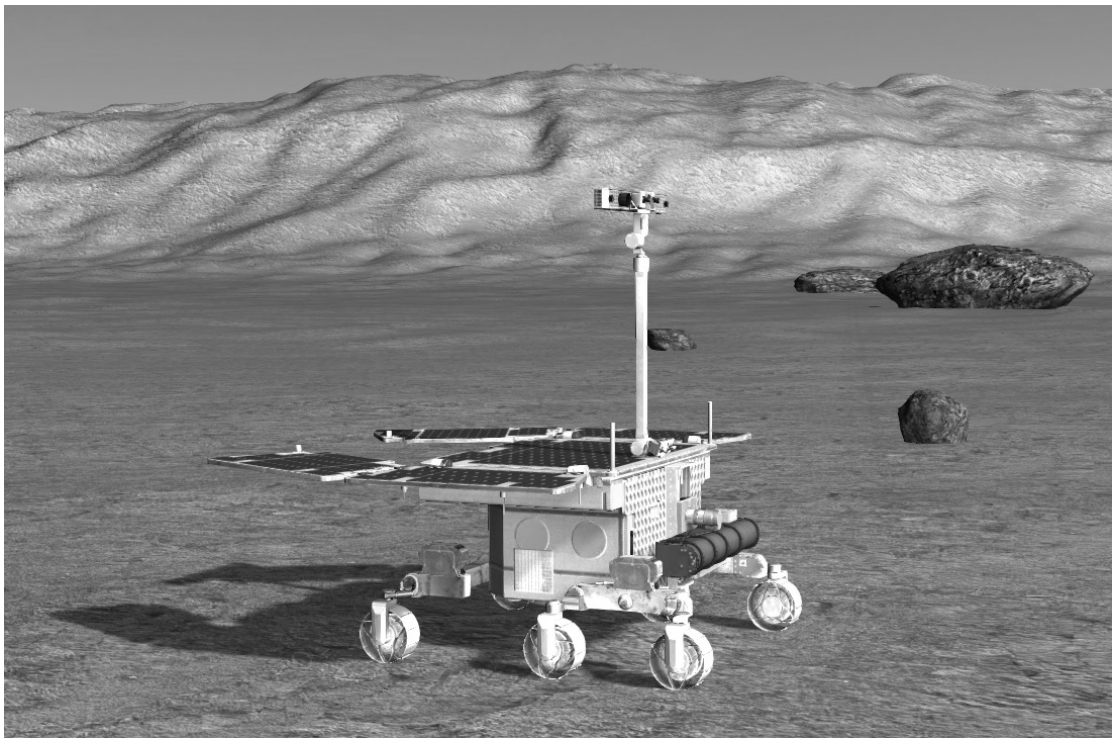


Рис. 2. Участок модели поверхности Марса

2. Система управления марсоходом

Виртуальный пульт управления (Рис. 3) содержит тумблер включения пульта и шесть блоков: управления марсоходом, управления камерами, управления мачтой, буром, солнечными батареями и фарой. Блок управления марсоходом в правой части панели содержит переключатель управления между шасси марсохода и его мачтой и кнопки включения двигателей, обеспечивающих движение марсохода вперед/назад и повороты влево/вправо. Блок управления камерами позволяет выбрать одну из трех камер, установленных в разных местах робота. Мачта содержит два сочленения: выбрав одно из них (или оба одновременно) можно поворачивать мачту с помощью тех же кнопок, которые используются для перемещения шасси, Управление буром осуществляется с помощью двух кнопок, одна из которых переводит его в вертикальное/горизонтальное положение, а другая (кнопка вкл/выкл) запускает процесс бурения. Кнопка вкл/выкл активна при вертикальном положении бура и после ее нажатия бур опускается и начинает бурение. Повторным нажатием кнопки вкл/выкл бурение прекращается, после чего с помощью бура осуществляется забор и перемещение грунта в специальный выдвижной контейнер. Эти операции выполняются автоматически путем изменения углов поворота и положений бура до целевых значений. Для солнечных батарей предусмотрены кнопки их сворачивания/разворачивания, а для фары – подсвечиваемая кнопка включения/выключения. При горизонтальном положении бура включается фара на корпусе робота, которая светит вперед, а при вертикальном положении включается фара на буре, которая подсвечивает участок бурения.

С пультом управления тесно связана функциональная схема, в соответствии с которой вычисляются управляющие сигналы, подаваемые на объекты управления в марсоходе. На Рис. 4 показан фрагмент этой схемы, управляющий раскрытием и сворачиванием одной солнечной батареи.

Рис. 3. Виртуальный пульт управления моделью марсохода

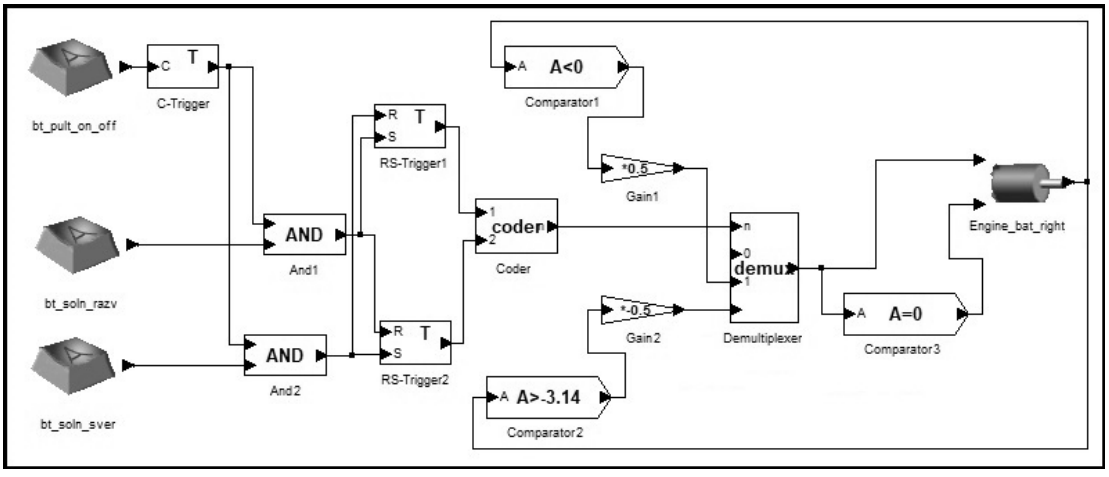


Рис. 4. Функциональная схема управления солнечной батареей

В левой части схемы расположены три кнопки (включения/выключения пульта, разворачивания и сворачивания батареи), каждая из которых на выходе формирует 0 или 1. Триггер C-Trigger изменяет значение своего выхода с 0 на 1 и обратно, если на вход подается 1. Блок AND реализует конъюнкцию входов, т.е. формирует на выходе 1 только тогда, когда на обоих входах 1. RS-триггеры RS-Trigger1 и RS-Trigger2 работают следующим образом: если на вход R подается 1, то на выходе получается 0; если на вход S подается 1, то на выходе формируется 1; если на обоих входах 1, то на выходе также 0. Т.е. если пульт включен и батарею надо развернуть, то на выходе верхнего триггера будет 1, а нижнего – 0. Шифратор (Coder) выдает наименьший номер входа, на который подана 1. Компараторы Comparator1, Comparator2 и Comparator3 сравнивают входной сигнал с заданной константой: если условие выполнено, то на выход подается 1, в противном случае – 0. Блоки Gain1 и Gain2 производят умножение входа на константу 0.5. Демультиплексор Demultiplexer работает по следующей схеме: если на вход n подается целое неотрицательное число i, то на выход будет подан сигнал с входа i.

Рассмотрим работу этой схемы. Если пульт включен кнопкой bt_pult_on_off, то на выходе C-Trigger формируется 1. Аналогично, если кнопкой bt_soln_razv задано развертывание солнечных батарей, то на вход блока And1 поступает 1 и на его выходе также формируется 1. Так как эта 1 поступает на вход S триггера RS-Trigger1, то на его выходе будет 1. Аналогично на выходе триггера RS-Trigger2 будет 0. Шифратор Coder на выходе сформирует минимальный номер входа с 1, т.е. в данном случае 1. Демультиплексор Demultiplexer передаст на свой выход сигнал с 1-го входа, на котором через обратную связь формируется величина угла поворота

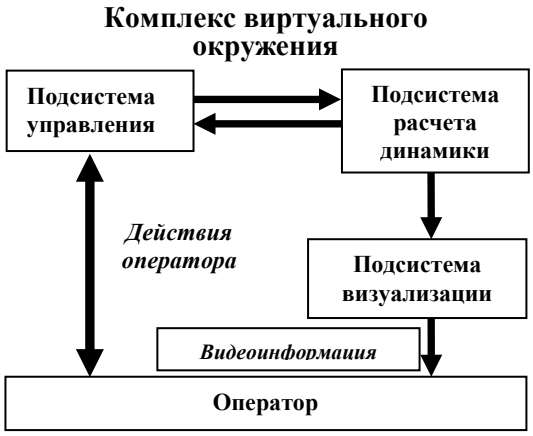


Рис. 5. Структура комплекса виртуального окружения

двигателя солнечной батареи. Если этот угол положительный, то Comparator1 выдаст 0, который поступит на 1-й вход демультиплексора и, соответственно, на двигатель. Кроме того, Comparator3 выдаст значение 1, которое поступит на вход тормозной муфты двигателя. Это будет означать, что на двигатель перестает поступать напряжение, и

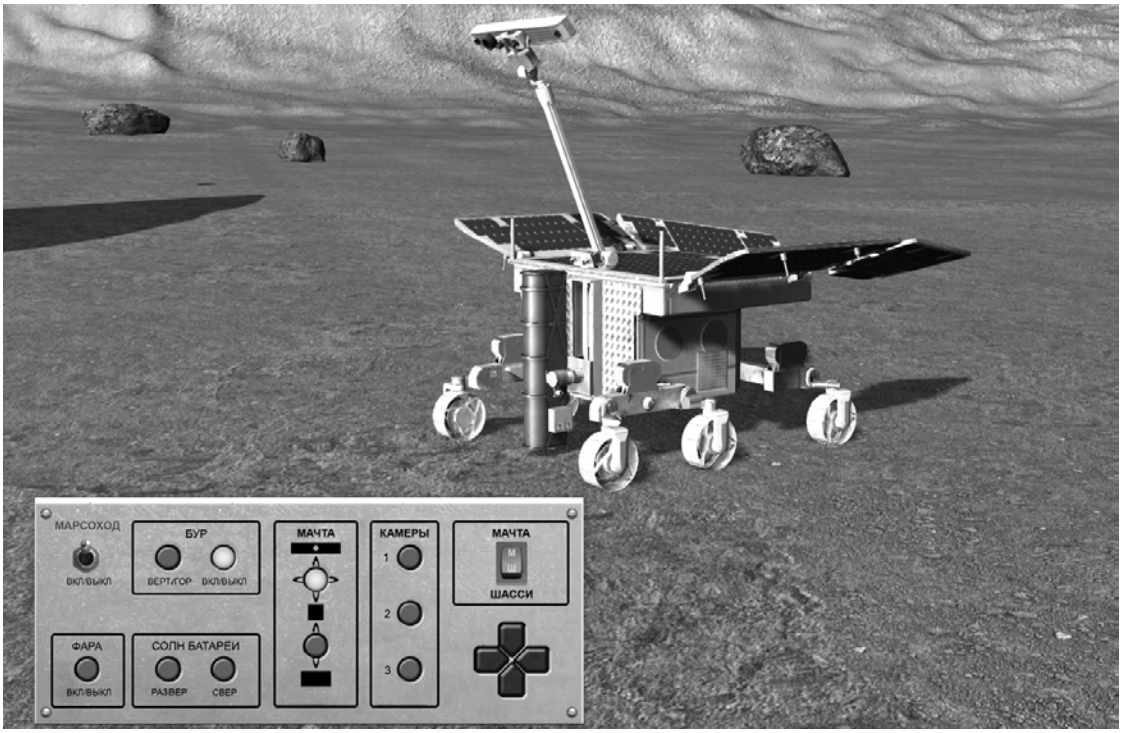


Рис. 6. Управление марсоходом с помощью пульта

он остановится, т.е. батарея полностью раскрыта. Если же угол станет отрицательным, то на двигатель будет подаваться напряжение 0.5 и батарея продолжит раскрытие.

Апробация системы управления

Апробация предложенной системы управления марсоходом проводилась в рамках комплекса виртуального окружения, разработанного в ФГУ ФНЦ НИИСИ РАН. Структура этого комплекса показана на Рис. 5. Из нее видно, что вычисленные подсистемой управления сигналы передаются в подсистему расчета динамики, которая вычисляет новые положения и ориентации всех динамических (в том числе управляемых оператором) объектов [7] и передает их в подсистему визуализации. Последняя синтезирует изображение из выбранной оператором виртуальной камеры на мониторе [8]. Воспринимая это изображение, оператор принимает решение о дальнейшем управлении марсоходом. Рассмотренный цикл занимает около 40 мсек, что обеспечивает непрерывное управление и движение марсохода и его составляющих. На Рис. 6 показан процесс управления марсоходом с помощью виртуального пульта управления. Нажатия на кнопки пульта управления производятся с помощью компьютерной мыши.

Видно, что мачта с камерами слежения наклонена, бур приведен в рабочее состояние и солнечные батареи раскрыты. Апробация показала адекватность предложенных методов управления поставленным задачам.

Заключение

В данном исследовании предложен метод управления виртуальной моделью марсохода на модели участка поверхности Марса, основанный на использовании виртуальных пультов управления и функциональных схем расчеты управляющих сигналов. Апробация метода проводилась в рамках комплекса виртуального окружения, разработанного в ФГУ ФНЦ РАН и показала его адекватную работу в масштабе реального времени.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) «34.9. Системы виртуального окружения: технологии, методы и алгоритмы математического моделирования и визуализации». (0065-2018-0020).

Control System for Virtual Mars Rover

M.V. Mikhaylyuk, E.V. Strashnov, L.A. Finagin

Abstract: The paper proposes the method for control of virtual mars rover model by means of virtual control panel and functional scheme for control signals computation. The control allows to provide rover movement, opening and closing of solar panels, switching on and off headlights, cameras rotation as well as performing ground sampling operations with the help of a drill. Control system approbation of the rover model was carried out within the virtual environment complex, developed at SRISA RAS.

Keywords: virtual environment system, control system, mars rover model

Литература

1. М.В. Михайлюк, В.И. Брагин. Технологии виртуальной реальности в имитационно-тренажерных комплексах подготовки космонавтов. Пилотируемые полеты в космос, ФГБУ «НИИ ЦПК имени Ю.А.Гагарина», № 2(7), 2013 стр. 82-93.
2. Е.В. Страшнов, М.А. Торгашев. Имитационное моделирование виртуального робота-кентавра в космических тренажерах // Труды НИИСИ РАН. – 2017. – т. 7, № 4. – С. 73-77. DOI: 10.25682/NIISI.2018.4.9982
3. Alexandre Carvalho Leite, Bernd Schäfer. On multi-objective optimization of planetary exploration rovers applied to ExoMars-type rovers. In: ASTRA 2011, ESA/ESTEC. 11th Symposium on Advanced Space Technologies in Robotics and Automation, 12-14 April 2011, Noordwijk, Niederlande. https://elib.dlr.de/70411.
4. R. Sánchez-Beato, M. Barrera, F. Martínez, L. Joudrier, P. Franceschetti. ExoMars 2020: Rover Operations Control System (ROCS). Design as part of the Rover Operations Control Center (ROCC). 2018 SpaceOps Conference, 28 May - 1 June 2018, 2018, Marseille, France.
https://arc.aiaa.org/doi/pdfplus/10.2514/6.2018-2405/
5. М.В. Михайлюк, М.А. Торгашев. Визуальный редактор и модуль расчета функциональных схем для имитациono-тренажерных комплексов. Программные продукты и системы, № 4, 2014, стр. 10-15.
6. A.V. Maltsev, P.Yu. Timokhin. Seamless Synthesis of Extra Large Planet Textures. Mathematica Montisnigri 2015, V. 32, p. 140-148.
7. M.V. Mikhaylyuk, E.V. Strashnov and P.Yu. Timokhin. Algorithms of multibody dynamics simulation using articulated-body method, Mathematica Montisnigri, Vol. XXXIX, pp. 133-145, 2017.
8. М.В. Михайлюк. Система визуализации для имитационно-тренажерных комплексов. Радиоэлектронные технологии № 2, 2016, стр. 72–76.

Программный комплекс визуализации цифровой модели керна

М.В. Михайлюк[1], П.Ю. Тимохин[2], И.Н. Мироненко[3]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1]mix@niisi.ras.ru, [2]webpismo@yahoo.de

Аннотация: В статье описывается разработанная в ФГУ ФНЦ НИИСИ РАН альфа-версия программного комплекса визуализации цифровой модели керна, полученной с помощью рентгеновской компьютерной томографии. Созданный комплекс позволяет визуализировать в масштабе реального времени поровое пространство в выделенном объеме цифровой модели керна, синтезировать изображения поперечных сечений цифровой модели керна, исследовать структуру модели керна с помощью кросс-сечений. Комплекс включает в себя разработанную графическую оболочку, позволяющую эффективно объединять в единую систему различные методы визуализации и расширять ее новыми методами. Была проведена успешная апробация разработанного комплекса на цифровой модели образца керна баженовской свиты, которая подтвердила корректность предлагаемых решений и их применимость в исследованиях специалистов нефтегазовой отрасли.

Ключевые слова: цифровая модель, керн, поровое пространство, поперечное сечение, кросс-сечение, визуализация, OpenGL.

Введение

Рентгеновская компьютерная томография (РКТ) широко применяется для исследований геологических образцов начиная с 1990-х годов [1]. Основная идея ее использования состоит в том, что при прохождении сквозь породу рентгеновские лучи теряют мощность и эта потеря пропорциональна плотности породы. Если регистрировать эти лучи после прохождения породы, то можно получить цифровой снимок. Множество снимков, произведенных из разных ракурсов, можно обработать и получить цифровую трехмерную модель образца [2]. Если в качестве образца берется керн (кусочек горной породы из нефтяного месторождения), то с помощью данной технологии получается цифровая модель керна. Используя эту модель, можно изучать особенности строения горной породы, строение порового пространства нефтегазоносных пород, их фильтрационные свойства, распределение включений различных минералов и т.д. Одним из методов изучения модели является ее 3D визуализация. Результаты исследований позволят в конечном итоге разработать новые методы повышения нефтеотдачи месторождений.

В данной работе рассматривается созданная в ФГУ ФНЦ НИИСИ РАН альфа-версия керносимулятора - программного комплекса визуализации цифровой модели керна, включающего единый универсальный пользовательский интерфейс, визуализатор порового пространства выделенного объема керна, а также визуализаторы поперечных сечений и кросс-сечений керна.

Общая структура программного комплекса

В работе использовалась цифровая модель керна баженовской свиты, полученная с помощью рентгеновского томографа HeliScan microCT (micro-computed tomography). Модель представляет собой трехмерную сетку размером 2560x2560x8600 ячеек, в каждой из которых записан коэффициент поглощения рентгеновского излучения материалом керна в этой ячейке. Коэффициенты представлены в виде 16 разрядных целых чисел со знаком. Если значение коэффициента томографом не определено, то в ячейку записывается (-1), в противном случае значением коэффициента является положительное число. Размер квадратной ячейки равен 0,0015276 миллиметра. Данные представлены в виде 215 файлов, каждый из которых содержит 40 слоев размером 2560x2560 ячеек. Таким образом, для получения нужных данных может понадобиться чтение большого числа файлов.

Программный комплекс построен в виде одного приложения, исполняемого в операционной среде Windows, написан на языке C++ с использованием графической библиотеки OpenGL, языка GLSL программирования шейдеров - программ, исполняемых на графическом процессоре (GPU), а также комплекса Qt средств разработки графических интерфейсов пользователя. Основное окно приложения показано на рис. 1.

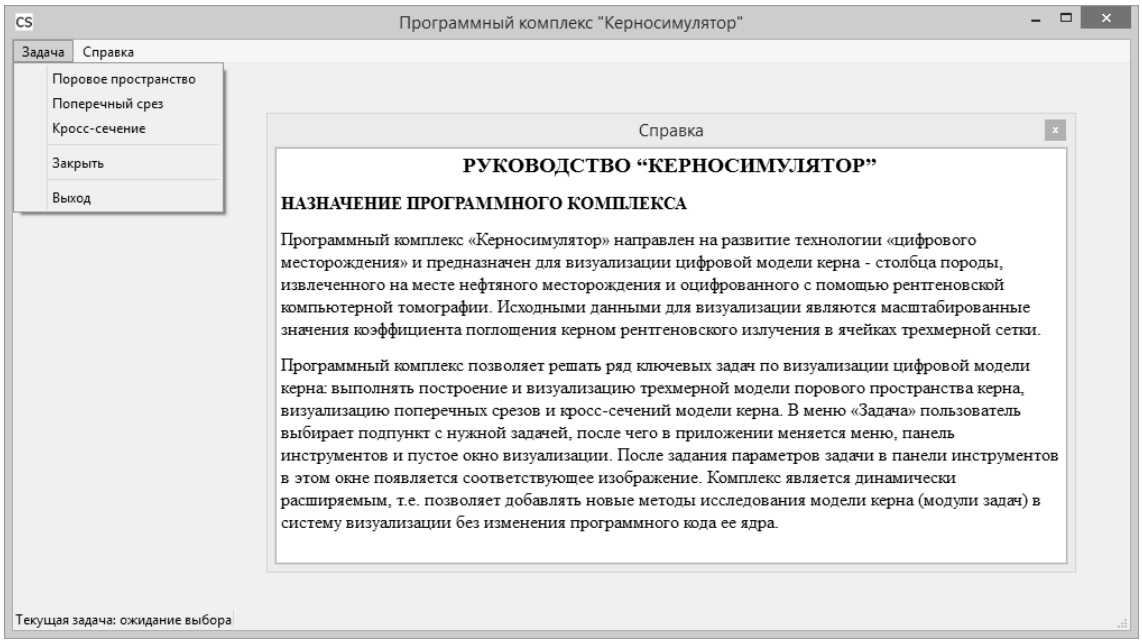


Рис. 1. Основное окно программного комплекса визуализации цифровой модели керна

После открытия меню приложения «Задачи» выводится список всех задач, доступных для выполнения в данный момент. Набор задач (и их список) жестко не фиксирован, а формируются динамически из набора dll-модулей, находящихся в директории Tasks приложения. При запуске приложение считывает эти динамические библиотеки и формирует список доступных для выполнения задач. При выборе каждой задачи подключается соответствующая динамическая библиотека, которая формирует на панели инструментов интерфейсные элементы задачи и справку, в которой описывается порядок выполнения работы по этой задаче. Подробно технология создания мультизадачной графической оболочки комплекса визуализации цифровой модели керна описана в работе [3]. В общем случае пользователь должен сначала выбрать директорию, в которой находятся данные цифровой модели керна, затем задать необходимые параметры и запустить задачу на выполнение. После получения результата визуализации в окне приложения он может анализировать их, при этом имея возможность вращать и перемещать изображение для получения удобного ракурса. В настоящий момент для решения доступны три задачи: визуализация порового пространства выделенной области керна, визуализация произвольного поперечного сечения и визуализация кросс-сечения (т.е. сечения тремя плоскостями, перпендикулярными осям координат модели керна). Рассмотрим эти задачи более подробно.

Визуализация порового пространства выделенной области керна

Для решения этой задачи пользователь в панели инструментов должен выбрать директорию, в которой находятся файлы цифровой модели, задать координаты левого нижнего угла и размеры (количества ячеек) визуализируемой области (параллелепипеда) цифровой модели керна, а также диапазон значений коэффициентов поглощения рентгеновского излучения, которые соответствуют поровому пространству. После нажатия на кнопку «Ok» в окне приложения выполняется построение и визуализация в масштабе реального времени трехмерной модели порового пространства образца керна (см. рис. 2). Цвет каждой ячейки полученной модели порового пространства соответствует значению коэффициента поглощения в этой ячейке и рассчитывается с помощью RGB-палитры с учетом освещенности. Построение геометрии и закраска модели порового пространства осуществляются полностью на

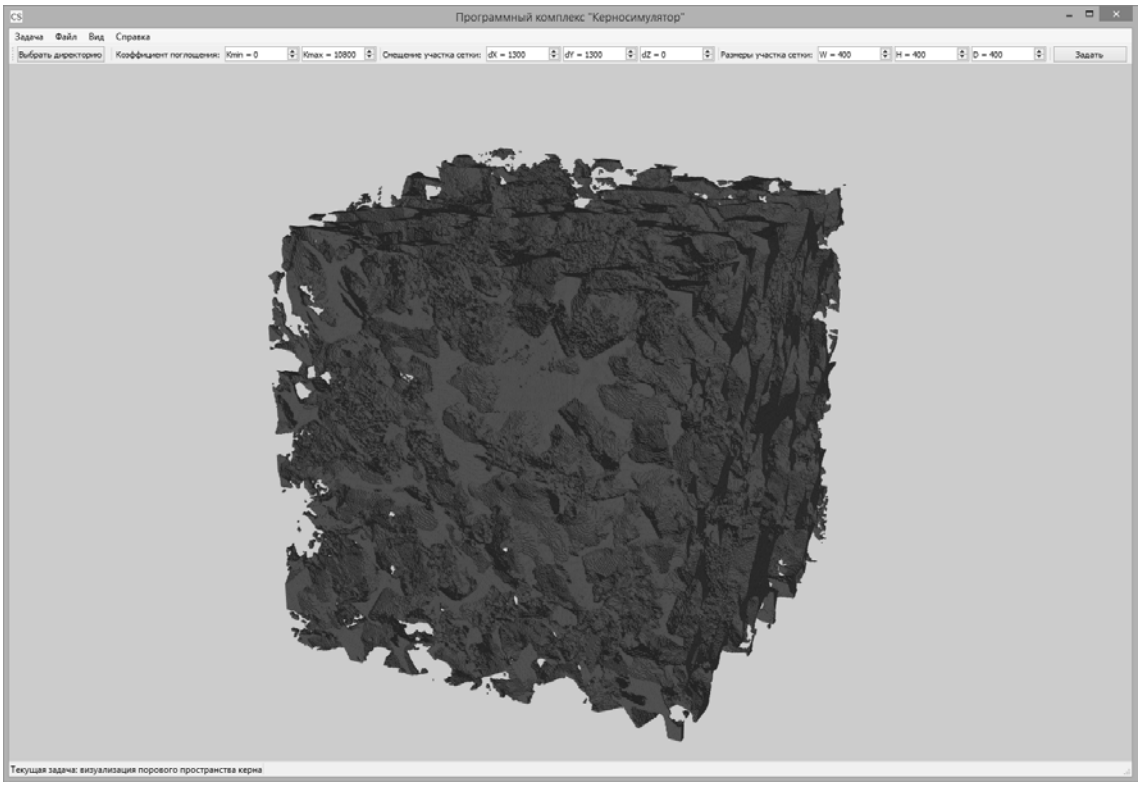


Рис. 2. Визуализация порового пространства выделенного объема керна

GPU с помощью комплекса разработанных шейдерных программ.

В процессе визуализации пользователь может перемещать модель порового пространства вдоль ее осей координат с помощью клавиатуры, а также с помощью мыши вращать ее вокруг этих осей, приближать и отдалять. Более подробное описание алгоритма решения этой задачи приведено в работах [4, 5].

Визуализация поперечного сечения керна

Для решения этой задачи пользователь в панели инструментов должен выбрать директорию, в которой находятся файлы цифровой модели, задать номер слоя (среза), который следует визуализировать, координаты X, Y левого нижнего угла лупы и выбрать одну из двух доступных палитр: RGB (полноцветная) или RED (оттенки красного). После подтверждения выбора с помощью кнопки «Ok» в окне слева появится изображение сечения в цветах выбранной палитры, в центре - изображение самой палитры и справа - изображение выделенного лупой участка (см. рис. 3). Лупа показывается на левом изображении белым квадратом, содержимое которого выводится справа. Особенностью лупы является то, что в ней каждой ячейке моделирования соответствует в точности один пиксел изображения, в то время как на всем изображении среза пиксел может соответствовать нескольким ячейкам и его цвет вычисляется как среднее значение их цветов. При изменении размеров окна приложения меняется также размер лупы с тем, чтобы взаимно однозначное соответствие ячеек и пикселов в ней сохранилось. Лупу можно передвигать с помощью стрелок на клавиатуре. Обе палитры являются монотонными, т.е. большему значению коэффициента поглощения соответствует большее значение цвета. Отсутствие значения коэффициента в исходных данных задается значением (-1) и визуализируется черным цветом (это указывается черной полоской под палитрой). В этой задаче мы используем модель без источников освещения, т.е. непосредственно указывается цвет каждого элемента. Подробное описание соответствующих алгоритмов приводится в работе [6].

Визуализация кросс-сечения керна

Для визуализации кросс-сечения цифровой модели керна пользователь должен в панели инструментов выбрать директорию, в которой находятся данные модели, затем задать диапазон слоев (начальный и конечный слои), номера вертикальных слоев по X и по Y, а также номер горизонтального (поперечного)

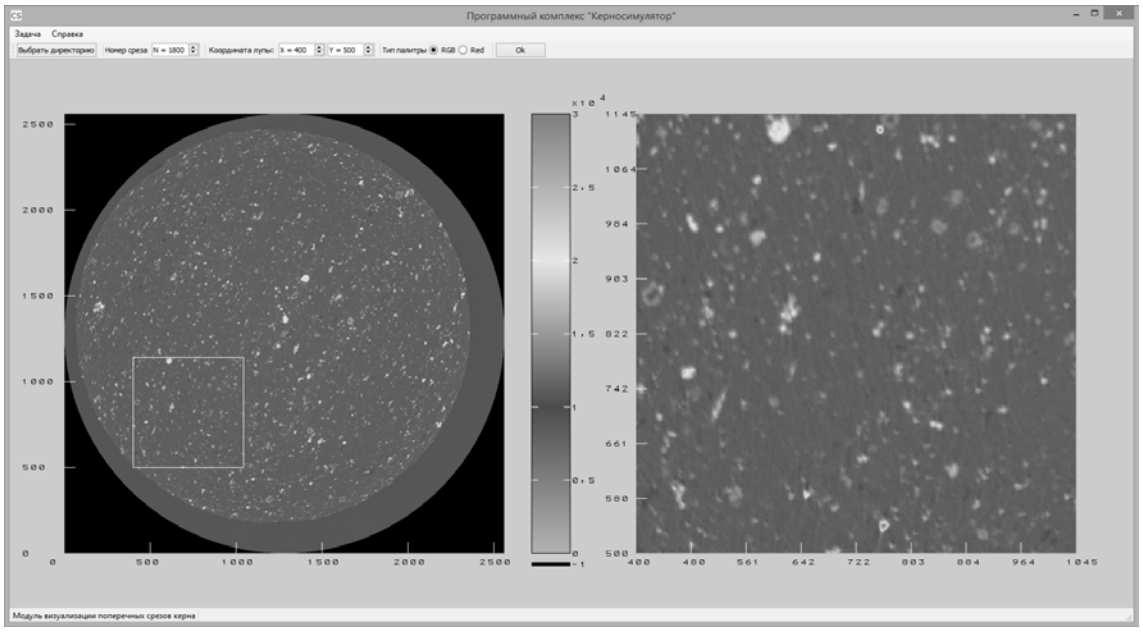


Рис. 3. Визуализация поперечного сечения керна

слоя. Кроме того, необходимо выбрать палитру (RGB или RED), которые устроены так же, как в предыдущем пункте. После нажатия на кнопку «Ok» производится загрузка необходимых данных и на экран выводится кросс-сечение (см. рис. 4) в цветах заданной палитры и сама палитра. Пользователь с помощью стрелок на клавиатуре может вращать изображение вокруг осей координат. В отличие от визуализации поперечных сечений, здесь необходимо использовать модель с одним источником освещением, т.к. в противном случае сечения сливаются в одно пятно, на котором трудно разглядеть детали. Мы используем фоновую и диффузную составляющие источника: первую для общей освещенности сцены, а вторую – для

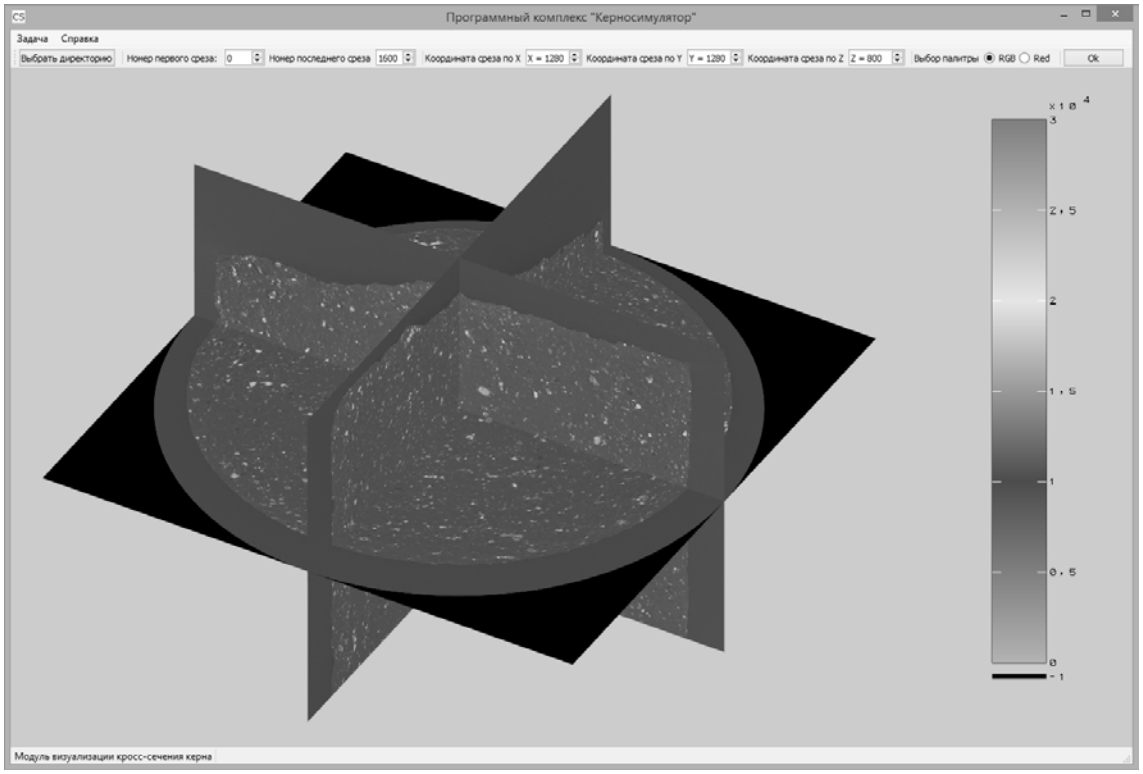


Рис. 4. Визуализация кросс-сечения керна

получения границ между сечениями за счет различной освещенности по-разному ориентированных прямоугольников сечений. Для этих составляющих задаются интенсивности излучения источника и коэффициенты отражения материала прямоугольников сечений.

Заключение

В данной работе описан созданный в ФГУ ФНЦ НИИСИ РАН программный комплекс визуализации цифровой модели керна (альфа-версия).

Разработанный комплекс позволяет исследовать в масштабе реального времени поровое пространство в выделенном объеме модели керна с помощью трехмерной визуализации, выполнять построение и визуализацию поперечных сечений цифровой модели керна, исследовать структуру цифровой модели керна с помощью кросс-сечений.

Взаимодействие пользователя с программным комплексом осуществляется с помощью разработанной графической оболочки, которая позволяет эффективно объединять в единую систему различные задачи визуализации цифровой модели керна, а также поддерживает расширение существующего набора методов визуализации без изменения программного кода ядра комплекса.

С помощью цифровой модели образца керна баженовской свиты была проведена успешная апробация созданного программного комплекса, которая подтвердила корректность предлагаемых решений, их применимость для исследования петрофизических характеристик нефтяных пластов, а также для анализа поровых пространств, полученных в результате гидроразрыва породы.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) «34.9. Системы виртуального окружения: технологии, методы и алгоритмы математического моделирования и визуализации» (0065-2018-0020).

A program complex for visualization of digital model of the core material

M.V. Mikhaylyuk, P.Yu. Timokhin, I.N. Mironenko

Abstract: The paper describes developed in SRISA RAS alpha-version of a program complex for visualization of digital model of the core material, obtained by means of X-ray computed tomography. The complex created allows the pore space in the allocated volume of core model to be visualized in real-time, to synthesize images of transverse sections of the core model, and to investigate the structure of the core model using cross-sections. The complex comprises a developed graphical shell allowing various visualization methods to be effectively combined into a single system and expand this system with new ones. A successful approbation of the complex developed, using digital model of the core sample of Bazhenov suite, was carried out. It confirmed the correctness of the solutions proposed and their applicability in researches of the specialists of oil and gas industry.

Keywords: digital model, core material, pore space, transverse section, cross-section, visualization, OpenGL.

Литература

1. S.L. Wellington, H.J. Vinegar. X-ray computerized tomography // Journal of Petroleum Technology. – 1987. – V. 39. – PP. 885-898.
2. Я.В. Савицкий. Современные возможности метода рентгеновской томографии при исследовании керна нефтяных и газовых месторождений // Вестник Пермского национального исследовательского политехнического университета. Геология. Нефтегазовое и горное дело. – 2015. – № 15. – С. 28–37. DOI: 10.15593/2224-9923/2015.15.4.
3. П.Ю. Тимохин, М.В. Михайлюк. Технология создания мультизадачной графической оболочки системы визуализации цифровой модели керна // Вестник кибернетики. – 2018. – № 3 (принято в печать).
4. М.В. Михайлюк, А.В. Мальцев, М.А. Торгашев. Моделирование и визуализация порового пространства керна // Труды НИИСИ РАН. – 2017. – Т. 7, № 4. – С. 78-82.
5. М.В. Михайлюк, Д.А. Кононов, Д.М. Логинов. Визуализация порового пространства в цифровой модели керна // Электронный научный журнал «ИТ-Стандарт». – 2017. – Т. 1, № 3-1 (12). – С. 7-10. URL: http://journal.tc22.ru/wp-content/uploads/2018/02/vizualizaciya_porovogo_prostranstva_v_cifrovoy_modeli_kerna.pdf (дата обращения: 16.08.2018).
6. М.В.Михайлюк, П.Ю. Тимохин, Л.А. Финагин. Визуализация поперечных сечений керна // Электронный научный журнал «ИТ-Стандарт». – 2018. – Т. 1, № 1-1 (14). – С. 14-18. URL: http://journal.tc22.ru/wp-content/uploads/2018/05/vizualizaciya_poperechnih_secheniy_kerna.pdf (дата обращения: 16.08.2018).

Алгоритм решения задачи планирования вычислений в многопроцессорной системе с нефиксированными параметрами

М.Г. Фуругян

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

Аннотация: Рассматривается задача составления расписания без прерываний и переключений в многопроцессорной системе для случая, когда работы имеют общий директивный интервал, а длительности их выполнения линейно зависят от объема выделенных им дополнительных ресурсов. Разработан алгоритм решения задачи, который является псевдополиномиальным при фиксированном числе процессоров.

Ключевые слова: многопроцессорная система, допустимое расписание без прерываний, директивный интервал, распределение ресурсов.

.

1. Введение

Одна из основных задач, возникающих при разработке математического и программного обеспечения многопроцессорных вычислительных систем, главным образом, систем реального времени, заключается в разработке алгоритмов построения расписания, показывающего, когда и какому программному модулю следует выделять те или иные вычислительные ресурсы.

В случае, когда прерывания и переключения с одного процессора на другой не допускаются, такие задачи, как правило, являются NP-трудными и все известные в настоящее время точные алгоритмы их решения являются переборными, а точных полиномиальных (эффективных) алгоритмов не известно. Поэтому актуальным является вопрос о поиске псевдополиномиальных алгоритмов.

Задачи нахождения допустимых расписаний без прерываний и переключений и без дополнительного ресурса широко освещены в литературе. Отметим, например, такие методы их решения, как метод ветвей и границ[1], муравьиные алгоритмы [2], вероятностные алгоритмы [3], генетические алгоритмы [4], различные эвристические алгоритмы [5, 6], алгоритмы агрегирования [6]. Задачи с дополнительным ресурсом ранее рассматривались для составления расписаний с прерываниями и переключениями [7 – 10].

В настоящей статье рассматривается задача составления многопроцессорного расписания без прерываний и переключений для случая, когда задан общий для всех работ директивный интервал. Кроме того, предполагается, что помимо процессоров, имеется несколько типов дополнительных ресурсов, а длительности выполнения работ линейно зависят от количества выделенных им этих ресурсов. Для случая, когда процессоры идентичные и их число фиксировано, предлагается псевдополиномиальный алгоритм построения допустимого расписания.

2. Постановка задачи

Рассматривается вычислительная система, состоящая из m идентичных процессоров, и множество работ $A = \{\overline{1,n}\}$, подлежащее выполнению. При выполнении работ не допускаются прерывания и переключения с одного процессора на другой. В фиксированный момент времени каждый процессор может выполнять не более одной работы, а каждая работа может выполняться не более чем одним процессором. Для всех работ $i \in A$ задан общий директивный интервал $[0; T]$, в котором они могут выполняться. Помимо процессоров в системе имеется K типов дополнительных ресурсов невозобновляемого типа. Суммарное количество k-го типа этого ресурса составляет R_k, $k = \overline{1,K}$. Если работе i выделено r_{ik} единиц дополнительного ресурса k-го типа, $i \in N$; $k = \overline{1,K}$, то ее длительность составляет

$$t_i = d_i - \sum_{k=1}^{K} a_{ik} r_{ik}, \qquad (1)$$

где

$$r_{ik} \in [0, \overline{r}_{ik}],\ \overline{r}_{ik} \le R_k,\ \ i \in A,\ \ k = \overline{1,K}, \qquad (2)$$

$$\sum_{i\in N} r_{ik} \le R_k, \quad i \in A, \tag{3}$$

a_{ik}, d_i, $\overline{r}_{ik}$ – заданные величины, $a_{ik} \ge 0$, $0 < d_i \le T$, $\overline{r}_{ik} > 0$, d_i – длительность выполнения работы i в случае, если ей дополнительных ресурсов не выделено, $a_{ik}, d_i, r_{ik}, \vec{r}_{ik} \in Z$, Z – множество целых чисел, $d_i - \sum_{k=1}^{K} a_{ik}\overline{r}_{ik} > 0$. Таким образом, $t_i \in [d_i - \sum_{k=1}^{K} a_{ik} r_{ik};\ d_i]$ при всех $i \in A$, причем $t_i \in N$, N – множество натуральных чисел.

Требуется найти такое распределение ресурсов $r_{ik}^0, \quad i \in N; \quad k = \overline{1, K}$, удовлетворяющее ограничениям (2), (3), при котором существует допустимое расписание (т.е. такое расписание, когда каждая работа полностью выполняется на одном процессоре без прерываний и переключений в директивном интервале [0, T]), или установить, что такого распределения ресурсов не существует. Примеры подобных задач содержатся в [8, 9].

Под расписанием выполнения работ будем понимать разбиение множества A на m непересекающихся подмножеств

$$A_1, A_2, ..., A_m \ (A = \bigcup_{j=1}^{m} A_j;\ A_{j_1} \bigcap A_{j_2} = \varnothing$$

при $j_1 \ne j_2$).

Работы из множества A_j приписываются процессору j и выполняются на нем одна за другой в произвольном порядке. Как известно [11], данная задача является *NP*-трудной даже при отсутствии дополнительных ресурсов и фиксированном m, а при произвольном m – *NP*-трудной в сильном смысле.

3. Алгоритм решения задачи

Для работы $i \in A$ определим U_i как множество всех векторов распределения ресурсов $X = \{r_{i1}, r_{i2}, ..., r_{iK}\}$, удовлетворяющих условиям (2), а V_i – множество всех различных значений t_i, вычисленных под формуле (1) для каждого такого вектора из U_i. Для определенности будем полагать, что $V_i = \{v_{i1}, v_{i2}, ..., v_{ih_i}\}$, а число векторов в U_i также равно h_i. Действительно, число элементов в множестве U_i не меньше числа элементов в V_i. Если же для некоторого значения v_{ip}, $1 \le p \le h_i$, существует несколько векторов X, то оставим любой один из них, а остальные исключим из множества U_i. Пусть $H = \max_{i\in N} h_i$. Из постановки задачи следует, что $v_{ih} \in N$ при всех $i \in A, \ h = \overline{1, h_i}$. Будем строить допустимое расписание с помощью точек m-мерного куба E^m с ребром T, имеющих целочисленные координаты. Множество этих точек разбивается на n уровней (возможно, пересекающихся). Каждая точка уровня l, $1 \le l \le n$, соответствует одному из всех возможных вариантов распределения дополнительных ресурсов между работами $i = \overline{1, l}, \ l \le n$, и вариантов распределения этих работ по процессорам при заданном директивном сроке T. Кроме того, точке уровня l приписывается номер уровня (т.е. l) и матрица $\|r_{ik}\|, \ i = \overline{1, l}, \ k = \overline{1, K}$, распределения дополнительных ресурсов между работами $i = \overline{1, l}$, где r_{ik} удовлетворяют условиям (2), (3). Точки первого уровня соответствуют множеству всех вариантов распределения, в которых первая работа назначена на один из m процессоров. Каждой такой точке соответствует m-мерный вектор загруженности процессоров, j-я компонента которого равна временной загруженности j-го процессора. Таким образом, точкам первого уровня соответствуют m-мерные векторы загруженности процессоров (t_1, 0, …, 0), (0, t_1, 0, …, 0), …, (0, …, 0, t_1) (всего m векторов), где

$$t_1 = d_1 - \sum_{k=1}^{K} a_{ik} r_{ik} \in V_1, \quad (r_{11}, r_{12}, ..., r_{1K}) \in U_1.$$

С каждым таким вектором связан номер уровня $l = 1$ и матрица распределения ресурсов $(r_{11}, r_{12}, ..., r_{1K})$. Для каждого $X = (r_{11}, r_{12}, ..., r_{1K})$ имеем m m-мерных векторов загруженности процессоров, а поскольку число векторов в U_1 равно h_1, то число точек первого уровня равно mh_1.

Каждая точка первого уровня куба E^m связана не более чем с mh_2 точками второго уровня, соответствующими множеству всех вариантов распределения ресурсов и работ, в которых первые две работы назначены на один или два из имеющихся m процессоров. Например, точка (0, …, 0, t_1, 0, …, 0) уровня l = 1 с матрицей распределения ресурсов

$(r_{11}, r_{12}, ..., r_{1K}) \in U_1$ связана не более чем с mh_2 точками уровня $l = 2$, которым соответствуют m-мерные векторы загруженности процессоров (t_2, 0, …, 0, t_1, …, 0), (0, t_2, 0, …, 0, t_1, …, 0), …, (0, …, t_2, t_1, 0, …,0), (0, …, t_1+t_2, 0,…,0),…,(0, …, t_1, t_2, …, 0), (0, …, t_1, 0, …, t_2) и матрица распределения ресурсов $\|r_{ik}\|,\ i = \overline{1, 2},\ k = \overline{1, K}$, удовлетворяющая соотношениям (2), (3). Здесь

$$t_2 = d_2 - \sum_{k=1}^{K} a_{2k} r_{i2} \in V_2,$$

$$(r_{21}, r_{22}, ..., r_{2K}) \in U_2.$$

Далее, с точками второго уровня связаны точки третьего уровня и т.д. В общем случае, каждая точка $(c_1, c_2,…, c_m)$ уровня $l - 1$, $l \le n$, куба E^m и соответствующая ей матрица распределения ресурсов

$$\|r_{ik}\|,\ i = \overline{1, l-1},\ k = \overline{1, K},$$

связаны не более чем с mh_l точками уровня l, которым соответствуют векторы загруженности процессоров $(c_1+ t_l, c_2,…, c_m)$, $(c_1, c_2 + t_l,…, c_m)$, …, $(c_1, c_2,…, c_m + t_l)$ и матрица распределения ресурсов $\|r_{ik}\|,\ i = \overline{1, l},\ k = \overline{1, K}$, удовлетворяющий соотношениям (2), (3). Здесь

$$t_l = d_l - \sum_{k=1}^{K} a_{lk} r_{lk} \in V_l, \quad (r_{l1}, r_{l2}, ..., r_{lK}) \in U_l.$$

Если хотя бы одна компонента m-мерного вектора загруженности процессоров превышает величину T, то соответствующая точка куба E^m исключается из дальнейшего рассмотрения. Допустимое расписание в поставленной задаче существует в том случае, когда хотя бы одна точка n-го уровня содержится в кубе E^m.

Для построения допустимого расписания используется следующая процедура. Если точка уровня l, $2 \le l \le n$, куба E^m соответствует вектору загруженности процессоров $(c_1, ..., c_{j-1}, c_j + t_k, c_{j+1}, ..., c_m)$ и матрице распределения ресурсов $\|r_{ik}\|,\ i = \overline{1, l},\ k = \overline{1, K}$, и была получена из точки куба E^m уровня $l - 1$, заданной вектором $(c_1, ..., c_{j-1}, c_j, c_{j+1}, ..., c_m)$ и матрицей $\|r_{ik}\|,\ i = \overline{1, l-1},\ k = \overline{1, K}$, то работе $l \in A$ выделяется вектор дополнительных ресурсов $(r_{l1}, r_{l2}, ..., r_{lK}) \in U_l$, ее длительность составляет

$$t_l = d_l - \sum_{k=1}^{K} a_{lk} r_{lk} \in V_l$$

и она назначается на процессор j. Указанная процедура выполняется для $l = \overline{n, 2,}$ в результате чего работы $i = \overline{n, 2}$ будут назначены на процессоры и, кроме того, будут определены векторы $(r_{i1}, r_{i2}, ..., r_{iK}) \in U_i$ дополнительных ресурсов, выделяемых этим работам. Тогда, если в результате работы этой процедуры была получена точка уровня $l = 2$ куба E^m, связанная с вектором загруженности процессоров (0, …, 0, t_1, 0, …, 0) уровня $l = 1$, в котором j-я компонента $t_1 \ne 0,$ и матрицей распределения ресурсов $(r_{11}, r_{12}, ..., r_{1K})$, то первой работе выделяется вектор дополнительных ресурсов $(r_{11}, r_{12}, ..., r_{1K})$, ее длительность составляет

$$t_1 = d_1 - \sum_{k=1}^{K} a_{1k} r_{1k}$$

и она назначается на j-й процессор.

Определим вычислительную сложность предложенного алгоритма. Для каждого уровня $l = \overline{1, n}$ число точек куба E^m, которые могут быть получены в результате работы алгоритма, не превосходит числа точек куба E^m с натуральными координатами, т.е. величины T^m. Для каждой точки уровня $l - 1$ строится не более mH точек уровня l ($l = \overline{2, n}$) и не более mH соответствующих им матриц распределения ресурсов размером $l \times K$. Таким образом, сложность алгоритма составляет $O(mHT^m)$ операций с m-мерными векторами и матрицами размером $l \times K$, $l = \overline{1, n}$. Выполнение одной такой операции с m-мерным вектором загруженности процессоров заключается в вычислении величины $t_l = d_l - \sum_{k=1}^{K} a_{lk} r_{lk}$ и сложении ее с одной из компонент вектора. А выполнение одной операции с матрицей распределения дополнительных ресурсов заключается в добавлении к ней строки $(r_{l1}, r_{l2}, ..., r_{lK})$. Поскольку $t_l \le T,\ r_{lk} \le R_k$ при всех $l \in A,$ то при фиксированном m предложенный алгоритм является псевдополиномиальным.

The algorithm for solving the problem of planning calculations in a multiprocessor system with non-fixed parameters

M.G. Furugyan

Abstract. The problem of scheduling without interruptions and switches in a multiprocessor system is considered for the case when the jobs have a common directive interval, and the duration of their execution depends linearly on the amount of additional resources allocated to them. The algorithm of solving the problem, which is pseudopolinomial with a fixed number of processors was developed.

Keywords: multiprocessing system, feasible schedule without interruptions, directive interval, distribution of resources

Литература

1. О.Г. Алексеев. Комплексное применение методов дискретной оптимизации. М., Наука, 1987.
2. С.Д. Штовба. Муравьиные алгоритмы. «ExponentaPro. Математика в приложениях», (2003), № 4, 70 – 75.
3. R. Raghavan. Probabilistic Construction of Deterministic Algorithms: Approximating Packing Integer Programs. «J. Computer and System Sciences», v. 37 (1988), 130 – 143.
4. В.А. Костенко, Р.Л. Смелянский, А.Г. Трекин. Синтез структур вычислительных систем реального времени с использованием генетических алгоритмов. «Программирование», (2000), № 5, 63 – 72.
5. P. Brucker. Scheduling Algorithms. Heidelberg, Springer, 2007.
6. Д.В. Красовский, М.Г. Фуругян. Алгоритмы решения минимаксной задачи составления расписания. «Изв. РАН. ТиСУ», (2008), №5, 732 – 736.
7. Е.О. Косоруков, М.Г. Фуругян. Некоторые алгоритмы распределения ресурсов в многопроцессорных системах. «Вестн. МГУ. Сер. 15, Вычисл. математика и кибернетика», (2009), № 4, 34 – 37.
8. М.Г. Фуругян. Планирование вычислений в многопроцессорных АСУ реального времени с дополнительным ресурсом. «АиТ», (2015), № 3, 144 – 150.
9. М.Г. Фуругян. Составление расписаний в многопроцессорных системах с несколькими дополнительными ресурсами. «Изв. РАН. ТиСУ», (2017), № 2, 57 – 66.
10. М.Г. Фуругян. Планирование вычислений в многопроцессорных системах с несколькими типами дополнительных ресурсов и произвольными процессорами. «Вестн. МГУ. Сер. 15, Вычисл. математика и кибернетика», (2017), № 3, 38 – 45.
11. Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. Алгоритмы. Построение и анализ. (Второе издание.). М., Вильямс, 2005.

Принцип наименьшего действия и термодинамика ориентированного ациклического графа

А.Л.Круглый

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail: akrugly@mail.ru

Аннотация: Рассмотрена дискретная модель конечной области непрерывного пространства-времени на микроуровне, которая представляет собой конечный связный ориентированный ациклический граф, каждая вершина которого имеет полустепени захода и исхода не более двух, называемый x-граф. Рассмотрено статистическое описание x-графов. Различные макроскопические состояния пространства-времени соответствуют различным подмножествам x-графов. Принцип наименьшего действия означает, что в макроскопическом пределе реализуется наиболее вероятный вариант состояния пространства-времени. Построено множество x-графов, соответствующее одиночной массивной частице и порождаемому ей гравитационному полю. Показано, что для этой модели действие имеет стандартный вид, постулируемый в ОТО. Определены топологические характеристики x-графа, которые можно идентифицировать с интегралом от скалярной кривизны пространства-времени, массой и собственным временем частицы.

Ключевые слова: принцип наименьшего действия, общая теория относительности, термодинамика, статистическая физика, ориентированный ациклический граф.

1. Введение

Вскоре после создания теории относительности была построена релятивистская термодинамика. Она строилась как феноменологическая теория и решала два круга вопросов: вопрос о преобразовании термодинамических величин при переходе от одной системы отсчета к другой и вопрос об описании термодинамических систем при наличии гравитационного поля [1]. Далее был предсказан ряд новых эффектов. Определена энтропия черной дыры, которая пропорциональна площади горизонта [2], наличие излучения черной дыры и соответствующей температуры [3]. Открыт эффект Унруха, который заключается в том, что наблюдатель, движущийся с постоянным собственным ускорением, видит наличие у вакуума пространства-времени Минковского отличную от нуля температуру, пропорциональную ускорению [4].

Указанные эффекты могут свидетельствовать о глубокой связи гравитации и термодинамики. Якобсон продемонстрировал, что уравнение Эйнштейна может рассматриваться как уравнение состояния [5]. Схожие идеи развиваются в работах [6, 7]. Хорошим обзором по этой теме является работа [8].

На фундаментальном, а не феноменологическом уровне термодинамика должна выводиться из статистики микросостояний. В настоящей работе построена модель микросостояний для простой системы одиночной макроскопической массивной частицы и создаваемого ею гравитационного поля и принцип наименьшего действия рассматривается как следствие термодинамики.

2. 4-мерная термодинамика

Базовым понятием нерелятивистской статистической физики является микросостояние в фиксированный момент времени. Нерелятивистская статистическая физика изучает статистику этих микросостояний, которые могут реализовываться с некоторой вероятностью. Макросостояния являются множествами микросостояний, а вероятности макросостояний являются суммами по вероятностям соответствующих микросостояний. Нерелятивистская термодинамика является следствием нерелятивистской статистической физики. Она изучает соотношения между макроскопическими величинами, которые

характеризуют макросостояние и являются средними по соответствующим микросостояниям. Построим аналогичную схему в релятивистском случае.

Сразу же возникает проблема с определением микросостояния в момент времени, так как в общем случае в искривленном пространстве-времени нельзя задать одновременность в конечной области даже в фиксированной системе отсчета. По-видимому, с этим связано построение релятивистской термодинамики как феноменологической теории без связи со статистической физикой. Однако проблема все равно проявляется. Имеются сложности даже с преобразованием температуры при переходе из одной инерциальной системы отсчета в другую (см., например, [9]).

Мы должны рассматривать четырехмерные состояния и их вероятности, то есть состояния не в пространстве, а в пространстве-времени. Это процессы. Для краткости назовем эту статистику 4-статфизикой. Она должна вести к четырехмерным макропеременным (4-термодинамике), которые описывают процессы, а не состояния. В 4-термодинамике должны рассматриваться другие величины, чем в термодинамике.

3. Принцип наименьшего действия

Почти вся физика построена на описании объектов и их состояний в момент времени и эволюции этих состояний во времени. Процессное описание должно оперировать величинами, характеризующими процесс. Такой величиной является действие. Фундаментальную роль в физике играет принцип наименьшего действия, который постулируется.

Основной гипотезой, рассматриваемой в настоящей работе, является предположение, что действие является монотонно убывающей функцией вероятности 4-макросостояния. Максимальной вероятности соответствует минимальное действие. В этом случае **принцип наименьшего действия имеет ясный физический смысл: в макроскопическом пределе процесс идет по наиболее вероятному варианту.** При достаточно остром максимуме вероятности, что типично для термодинамики, выбор варианта эволюции процесса практически детерминирован.

В качестве простейшего примера рассмотрим одиночную точечную массивную частицу и создаваемое ею гравитационное поле. Действие имеет вид:

$$S = \int R\, d\Omega - m \int ds, \qquad (1)$$

где первое слагаемое есть интеграл по 4-мерному объему от скалярной кривизны R, $d\Omega$ - элемент 4-мерного объема, второе слагаемое есть интеграл по мировой линии частицы, ds – элемент интервала вдоль этой мировой линии, m - масса частицы. В формуле (1) знак скалярной кривизны принят аналогично [10]. В [11] принят противоположный знак.

В настоящей работе принята естественная система единиц, в которой равны 1 скорость света, постоянная Планка и гравитационная постоянная. Числовые множители типа π также считаются включенными в определение физической величины.

Предлагается следующая интерпретация действия. Интеграл от скалярной кривизны является монотонно убывающей функцией вероятности 4-макросостояния. Его минимум означает максимум вероятности процесса. Второе слагаемое является условием, определяющим, какую систему мы рассматриваем. Длина мировой линии τ считается заданной динамикой 4-микросотояний. Таким образом, имеется вариационная задача на условный экстремум изопериметрического типа. Масса является неопределенным множителем Лагранжа, который в изопериметрических задачах является константой. Обозначим первое слагаемое в (1) через S_g. Имеем для его минимального значения $S_g(min)$.

$$m = \frac{dS_g(min)}{d\tau} \qquad (2)$$

Уравнение Эйнштейна рассматриваемой системы получается варьированием функционала (1). Оно является соотношением между величинами 4-термодинамики, то есть уравнением состояния 4-термодинамики, а не 3-термодинамики, как предполагается в [5].

Рассмотрим конкретную модель 4-микросостояния и покажем, как описанные в настоящем параграфе величины и

соотношения 4-термодинамики могут быть получены из 4-статистики.

4. Модель 4-микросостояния

Модель 4-микросотояния рассматривалась автором в ряде предыдущих работ (см., например, [12]).

Предположим, что пространство-время на микроуровне имеет конечную делимость, дискретно и представляет собой ориентированный ациклический граф. Вершины графа соответствуют элементарным неделимым событиям. Все ребра графа ориентированы и соответствуют элементарным неделимым причинно-следственным связям. Ацикличность означает, что отсутствуют ориентированные циклы, что отражает принцип причинности. Ребра и вершины неделимы, не имеют внутренних свойств. Все свойства связаны с топологией графа. Любой физический объект является структурой графа, и все свойства объекта являются топологическими характеристиками этой структуры.

Ограничимся рассмотрением конечных связных графов. Такой граф является дискретным аналогом конечной области непрерывного пространства-времени. Несложно определить дискретные аналоги различных геометрических объектов непрерывного пространства-времени. Ориентированный маршрут является дискретным аналогом времениподобной мировой линии. Две вершины причинно связаны, причем первая является причиной, а вторая – следствием, если имеется ориентированный маршрут из первой во вторую. Для фиксированной вершины множество вершин – ее следствий (причин), составляют дискретный аналог светового конуса будущего (прошлого) этой вершины. Множество вершин, попарно причинно не связанных, являются дискретным аналогом пространственноподобной гиперповерхности. Таким образом, рассматриваемый граф имеет причинно-следственную структуру, аналогичную конечной области непрерывного пространства-времени. Однако любая структура конечного графа может содержать только конечное число событий.

Отметим, что важной задачей построения модели является идентификация физических объектов и их свойств со структурами графа и их топологическими характеристиками. При этом нельзя просто идентифицировать объекты непрерывной геометрии и их дискретные аналоги, что рассматривается ниже. В принятой естественной системе единиц числовое значение физической величины равно числовому значению соответствующей топологической характеристики графа.

Рассмотрим динамику модели. Конечный связный граф дает всю информацию о соответствующем конечном этапе моделируемого процесса. Задача динамики предсказать будущий ход процесса, или реконструировать прошлый. Это означает достроить граф в будущее или в прошлое. Минимальная часть графа – это вершина. Достраивать граф можно последовательно, добавляя вершины одну за другой. Такая динамика называется динамикой последовательного роста. Добавление вершины в будущее (в прошлое) означает, что добавляемая вершина является следствием (причиной) некоторых существующих вершин и не связана причинно с остальными. Не рассматриваются варианты добавления новой вершины, являющейся причиной для некоторых имеющихся вершин и следствием для некоторых других имеющихся вершин.

Последовательный рост интерпретируется не как физическое рождение вершин, а как процесс получения информации наблюдателем. При такой интерпретации рост графа в прошлое не приводит к нарушению причинности.

Имеются различные варианты добавления новой вершины. Предполагается, что конкретный вариант добавления вершины определяется не однозначно, а каждый вариант может быть выбран с некоторой вероятностью. Варианты не равновероятны, а вероятности зависят от топологии уже имеющегося графа.

Алгоритм вычисления вероятности добавления новой вершины играет роль уравнения движения модели. В настоящее время он неизвестен. Однако можно получить ряд результатов по 4-термодинамике из некоторых достаточно общих предположений о статистике графов. Ситуация аналогична нерелятивистской термодинамике. Общие законы термодинамики газов не зависят от деталей динамики молекул. От этих деталей зависят конкретные значения различных величин, например, теплоемкости.

Ограничимся рассмотрением графов специального вида, у которых каждая вершина инцидентна не более чем двум входящим и двум исходящим ребрам. Назовем такой граф x-графом, так как каждая вершина и инцидентные ей ребра и свободные валентности образуют структуру в виде буквы «X».

Обозначим через n число свободных входящих валентностей x-графа. X-граф может быть интерпретирован как n неделимых частиц, которые испытывают ряд парных столкновений, описываемых вершинами. Входящая валентность описывает начало участия частицы в известной части процесса, а соответствующая вершина – ее первое столкновение. Аналогично исходящая валентность описывает прекращение участия частицы в известной части процесса, а соответствующая вершина – ее последнее столкновение. Поскольку число частиц в каждом столкновении не меняется, то число исходящих валентностей тоже равно n. Этот результат легко доказать математически без рассмотренной наглядной интерпретации. Назовем n шириной x-графа. При добавлении новой вершины в ходе последовательного роста она может соединяться с имеющимся x-графом только ребрами, которые добавляются вместо имеющихся свободных валентностей.

Обозначим через N число вершин x-графа, а через $N(e)$ - число ребер. Очевидно, имеет место соотношение

$$N(e) = 2N - n. \qquad (3)$$

Рассмотрим последовательный рост x-графа, начиная с пустого множества. На первом шаге по определению появляется единственная вершина. Число вершин x-графа равно номеру шага роста. Легко доказать, что в результате последовательного роста может быть получен любой конечный связный x-граф. Пронумеруем x-графы так, что каждый имеет уникальный номер, который будем обозначать латинским индексом. В остальном нумерация произвольна. Обозначим через $X_i(N)$ x-граф, имеющий N вершин, через $P(X_i(N))$ вероятность его получения в ходе последовательного роста, начиная с пустого множества и через $\{X_i(N)\}$ множество всех x-графов, имеющих N вершин. Имеем

$$\sum_{\{X_i(N)\}} P(X_i(N)) = 1 \qquad (4)$$

Формула (4) означает, что какой-то x-граф будет получен на шаге номер N.

Тот факт, что алгоритм последовательного роста описывает закономерности вселенной, означает, что вероятности $P(X_i(N))$ описывают, с какой частотой подграфы $X_i(N)$ встречаются в x-графе всей вселенной. Мы не будем формулировать это утверждение более строго, так как это выходит за рамки настоящей статьи.

Фиксированный x-граф является моделью фиксированного 4-микросостояния рассматриваемой системы.

5. Вероятность 4-макросостояния

По определению 4-макросостояние – это подмножество 4-микросостояний, имеющих равные некоторые макроскопические характеристики. Для определения вероятности 4-макросостояния необходимо определить статистический ансамбль 4-микросостояний, подмножеством которого является рассматриваемое 4-макросостояние. Рассмотрим два подхода.

Первый подход основан непосредственно на динамике последовательного роста (динамике 4-микросостояний). В качестве статистического ансамбля 4-микросостояний рассматриваются все x-графы, которые могут быть получены на некотором шаге роста, и тем самым имеющие одинаковое число вершин. Обозначим через $P_k(X_i(N_j\text{-}1))$ вероятность варианта номер k добавления новой вершины на шаге номер N_j к x-графу $X_i(N_j\text{-}1)$, содержащему $N_j\text{-}1$ вершин. При последовательном росте, начиная с пустого множества, число вершин в x-графе совпадает с номером шага роста. Один и тот же x-граф может быть получен в результате различных вариантов последовательного роста, которые различаются порядком, в котором добавляются вершины. Пронумеруем эти варианты индексом m. Для вероятности $P_m(X_i(N))$ варианта m последовательного роста имеем

$$P_m(X_i(N)) = \prod_{N_j=1}^{N} P_k\left(X_i\left(N_j - 1\right)\right) \quad . \quad (5)$$

Вероятность $P(X_i(N))$ получения x-графа $X_i(N)$ является суммой по всем этим вариантам:

$$P(X_i(N)) = \sum_m P_m\left(X_i(N)\right) \quad . \quad (6)$$

Это вероятность 4-микросотояния $X_i(N)$. 4-макросотояние является некоторым подмножеством множества $\{X_i(N)\}$ всех x-графов, имеющих N вершин. Это можно описать как принадлежность индекса i некоторому множеству индексов I. Вероятность 4-макросотояния, обозначаемая через P_I, получается суммированием вероятностей по всем 4-микросостояниям, соответствующим этому 4-макросостоянию.

$$P_I = \sum_{i \in I} P(X_i(N)) \quad . \quad (7)$$

На последовательный рост могут быть наложены дополнительные условия. Например, рассмотрен последовательный рост, начиная с какого-то непустого x-графа. В этом случае статистический ансамбль является собственным подмножеством множества всех x-графов с одинаковым числом вершин. Вероятность P должна быть соответственно перенормирована.

Эволюция процесса по варианту с максимальной вероятностью означает вариант с максимальной вероятностью P_I. В общем случае максимальная вероятность P_I не означает, что максимальны вероятности 4-микросостояний, входящих в сумму (7). Большая величина P_I может достигаться за счет большого числа слагаемых в (7).

Нерелятивистская термодинамика имеет затруднения с описанием эволюции сложных систем. Различные подходы развиваются в синергетике. Предпринимаются попытки модифицировать термодинамику. Например, в работе [13] предлагается принцип максимума производства энтропии. Эта идея близка к предлагаемому принципу максимума вероятности процесса, а не состояния. В нерелятивистской термодинамике система эволюционирует из менее вероятных состояний к более вероятным. Можно предположить, что возможны системы, для которых эта эволюция не совпадает с эволюцией по наиболее вероятному варианту процесса, или вероятности состояний не могут быть определены.

6. Связь вероятности 4-макросотояния и числа вершин

Рассмотренный вариант построения 4-макросостояния основан на детальном знании динамики последовательного роста (динамики 4-микросостояний). Экспериментально наблюдаемой ситуации более соответствует вариант, когда динамика 4-микросотояний неизвестна. В частности, неизвестно число вершин. Измерены только некоторые макроскопические величины. Требуется определить наиболее вероятные значения других макроскопических величин без детального анализа динамики последовательного роста. Так задача ставится и в нерелятивистской термодинамике. Предполагается, что этой постановке соответствует действие (1). Заданными являются длинна мировой линии и масса частицы (второе слагаемое).

Рассмотрим вероятность 4-макросотояния, заданного некоторыми макроскопическими величинами. Этому макроскопическому процессу и фиксированным величинам на микроскопическом уровне соответствует некоторое множество x-графов, которые будем называть допустимыми x-графами. В общем случае допустимые x-графы могут иметь различное число вершин в интервале от $N(min)$ до $N(max)$, а x-графы с иным числом вершин не соответствуют рассматриваемому процессу и значениям фиксированных величин. Динамика последовательного роста задает вероятности x-графов при фиксированном числе вершин. Макроскопический наблюдатель не может определить число вершин. Поэтому положим, что число вершин может равноправно принимать любое значение из допустимого интервала от $N(min)$ до $N(max)$.

Рассмотрим получаемые статистические ансамбли x-графов, их вероятности и нормировку этих вероятностей. Обозначим через $Y_i(N)$ допустимый x-граф и через $\{Y_i(N)\}$ множество всех допустимых x-графов, имеющих N вершин. Обозначим через $\{Y\}$ множество всех допустимых x-графов с числом вершин от $N(min)$ до $N(max)$ и через W

статистическую сумму этого множества. Имеем

$$W = \sum_{\{Y\}} P(Y_i(N)) \quad . \qquad (8)$$

В сумму (8) входят вероятности, полученные в динамике последовательного роста. Это отражает указанную в разделе 4 гипотезу, что рассматриваемые вероятности описывают статистику 4-микросостояний во вселенной. Гипотеза о равноправном включении х-графов с различным числом вершин во множество $\{Y\}$ выражается тем, что вероятности х-графов с разным числом вершин входят в сумму (8) без дополнительных коэффициентов, зависящих от числа вершин. Обозначим через $p(Y_i(N))$ нормированную вероятность того, что рассматриваемый макроскопический процесс имеет микроскопическую структуру $Y_i(N)$. Имеем

$$p(Y_i(N)) = W^{-1} P(Y_i(N)) \quad . \qquad (9)$$

Если для рассматриваемого 4-макросостояния мы хотим узнать средние значения некоторых других макроскопических величин, помимо заданных изначально, то мы их должны вычислять с помощью статистического распределения (9). При этом если для некоторого небольшого подмножества х-графов их вероятности $p(Y_i(N))$ доминируют, то можно рассматривать только эти х-графы. Такая ситуация и соответствует 4-термодинамике.

Рассмотрим основную гипотезу о статистике х-графов, вероятности каких х-графов доминируют. Разобьем множество допустимых х-графов на подмножества с одинаковым числом частиц.

Выберем некоторое целое число

$$K << N(max) - N(min).$$

Объединим в одно подмножество х-графы числом вершин от $N(min)$ до $N(min) + K$. Предположим, что можно выбрать число K так, что суммарная вероятность всех х-графов, не вошедших в рассматриваемое подмножество много меньше 1. Тем самым, с вероятностью близкой к 1 х-граф, соответствующий рассматриваемому макроскопическому процессу, имеет минимально возможное число вершин, или число вершин, отличающихся от минимального на микроскопическую величину.

Проиллюстрируем сделанное предположение на модели множества симметричных монет. Рассмотрим монеты, на которых с одной стороны указана цифра 1, а с другой -1.

Пусть с равной вероятностью имеется некоторое число монет в интервале от 2 до $C >> 1$.

Число монет является аналогом числа вершин. Монеты были однократно подброшены и определено среднее выпавшее число. Пусть известно, что оно равно 0. Это аналог фиксированной макроскопической переменной. Мы имеем стандартное биномиальное распределение (см., например, [14]). Требуется определить, какое число монет наиболее вероятно было подброшено. Минимально возможное число 2. В этом случае заданное среднее выпадает с вероятностью 0,5. Вероятность для трех монет равна нулю. При четырех монетах вероятность заданного среднего равна ⅜. Вероятность заданного среднего при четном числе монет падает с увеличением числа монет. Эффект связан с тем, что с ростом числа монет растет число допустимых вариантов, но общее число вариантов растет быстрее. Также имеются осцилляции вероятности при четном и нечетном числе монет.

Чтобы сгладить эти осцилляции объединим в одно подмножество варианты с четным и следующим за ним нечетным числом монет. Это аналог объединения х-графов с числом вершин, отличающимся на K при $K = 1$. В итоге получаем, что вероятность возможного числа монет монотонно падает с ростом числа монет, и наиболее вероятно минимальное число монет.

Различные х-графы с одинаковым числом вершин имеют различные вероятности, которые определяются деталями динамики последовательного роста. Однако предполагается, что справедлив эффект, рассмотренный на примере монет. Для х-графов с макроскопическим числом вершин с ростом числа вершин х-графа число допустимых х-графов растет намного медленнее общего числа х-графов. В результате суммарная вероятность допустимых х-графов быстро падает с ростом числа вершин рассматриваемых х-графов. Имеем

$$1-\sum_{N=N(min)}^{N(min)+K} p(V_i(N)) \ll 1 \quad . \qquad (10)$$

Таким образом, минимум действия для некоторой физической системы означает минимум числа вершин (с точностью до микроскопических отклонений) соответствующего х-графа, то есть максимум вероятности 4-микросостояния реализуется при минимуме числа событий.

7. Скалярная кривизна

В разделе 3 сделано предположение, что минимум действия для гравитационного поля (первое слагаемое в (1)) означает максимум вероятности процесса. В модели х-графов максимум вероятности достигается при минимуме числа вершин. Предположим, что интеграл от скалярной кривизны в терминах х-графа есть удвоенное число вершин. Смысл введения коэффициента 2 будет показан в следующем разделе.

$$2N = \int R\, d\Omega \quad . \qquad (11)$$

Гипотеза (11) может быть строго доказана только при наличии полной теории, включающей конкретный алгоритм последовательного роста и процедуру перехода от х-графа к непрерывному пространству времени. В отсутствие такой теории равенство (11) является постулатом. Однако можно привести некоторые предварительные рассуждения в пользу его физической обоснованности.

Рассмотрим физический смысл скалярной кривизны. В стандартных учебниках он почему-то или не рассматривается [11, 15], или рассматривается вне связи с гравитацией [10]. Это довольно странно, так как скалярная кривизна играет фундаментальную роль в ОТО. Рассмотрим точку M_1 в четырехмерном псевдоримановом пространстве-времени, два не коллинеарных бесконечно малых смещения δx^i и dx^j в этой точке таких, что, образуемая ими, бесконечно малая плоскость не является изотропной, и вектор $a^j(M_1)$, коллинеарный dx^j. Латинские индексы пробегают значения от 0 до 3. Параллельно перенесем a^j (M_1) на δx^i в точку M_2 и получим b^j (M_2). Параллельно перенесем a^j (M_1) и b^j (M_2) на dx^j в точки M_3 и M_4 соответственно, получив a^j (M_3) и b^j (M_4). Параллельно перенесем a^j (M_3) на δx^i в точку M_4, получив a^j (M_4). В искривленном пространстве a^j (M_4) и b^j (M_4) не совпадают. Две бесконечно близкие мировые линии, которые были параллельны в начальных точках M_1 и M_2, уже не параллельны после бесконечно малого смещения вдоль этих линий. При положительной кривизне векторы, касательные к этим линиям, поворачиваются друг к другу, что и описывает притяжение. Имеем для угла между a^j (M_4) и b^j (M_4) в плоскости, задаваемой смещениями δx^i и dx^j [15]

$$\varphi \approx \tan\varphi = K(ij)\sigma \quad , \qquad (12)$$

где $K(ij)$ – кривизна в двумерном направлении, задаваемом смещениями δx^i и dx^j, а σ – площадь параллелограмма, построенного на смещениях δx^i и dx^j. Для кривизны имеем [10]

$$R = \sum_{ij} K(ij) \quad . \qquad (13)$$

Суммирование в (13) означает суммирование по всем возможным четырем направлениям смещения начальной точки M_2 второй мировой линии относительно M_1 и суммирование по всем возможным четырем направлениям мировых линий. Таким образом, имеем для среднего угла ‹φ› справедливо:

$$\langle\varphi\rangle = \frac{R\sigma}{16}. \qquad (14)$$

Скалярная кривизна описывает усредненное притяжение, называемое гравитацией.

В терминах х-графа взаимодействие двух его подграфов A и B должно означать наличие каких-то маршрутов между ними. В графах других связей и не существует. Не будем спекулятивно рассуждать о возможных деталях такой связи.

Для целей настоящей работы важно, что связь можно охарактеризовать некоторым числом, которое может являться функцией типа образующих связь маршрутов, их числа, длинны и так далее. Назовем это число интенсивностью связи.

Если мы интересуемся только свойствами подграфа A, то произведем усреднение интенсивности связи по всем возможным подграфам B.

Зафиксируем число вершин подграфа A и произведем усреднение по всем возможным его топологиям.

Формула (11) означает гипотезу, что полученная усредненная интенсивность связи

пропорциональна числу вершин подграфа *A*. Гипотеза выглядит правдоподобной. Гравитация является усредненным взаимодействием, которое в ОТО описывается как свойство геометрии.

При этом учет влияния на интенсивность связи конкретной топологии подграфа *A* может описываться включением дополнительных слагаемых в действие.

При описании в терминах непрерывного пространства-времени они интерпретируются как не гравитационные силовые поля, а соответствующие топологические свойства - как заряды.

Приведенные рассуждения носят предварительный характер и должны рассматриваться как вопросы для дальнейших исследований.

8. Масса и прообраз действия

Построим модель макроскопической времениподобной мировой линии. Идентифицируем ее конечный макроскопический отрезок с x-графом специального вида, для которого

$$N(e) >> n.$$

Рассмотрим среднее число T ребер в ориентированных маршрутах рассматриваемого x-графа, которые начинаются входящей валентностью и заканчиваются исходящей валентностью. В соответствии с указанным выше соглашением о системе единиц постулируем, что в макроскопическом пределе интервал собственного времени τ рассматриваемого отрезка мировой линии равен T.

Построим прообраз соответствующего действия (1) для рассматриваемой модели. При этом считается, что динамика 4-микросостояний фиксирует две макропеременные: интервал собственного времени и ширину x-графа, физический смысл которой будет рассмотрен ниже.

$$S = 2N - mT \ . \qquad (15)$$

В (15) на топологическом языке осталась не выражена масса частицы. Подставим (3) в (15)

$$S = N(e) + \ n - mT \ . \qquad (16)$$

Свободная валентность означает ребро, которым рассматриваемый x-граф соединяется с остальной вселенной. Свободные валентности – это дискретный аналог гиперповерхности, являющейся границей рассматриваемого четырехмерного объема. При рассмотрении действия (1) применяется стандартный прием выделения из первого слагаемого поверхностного члена. На гиперповерхности действие не варьируется и этот член отбрасывается.

Равенство (16) имеет аналогичный смысл для дискретной модели. Число свободных валентностей n фиксировано.

Таким образом, это слагаемое можно отбросить.

В итоге имеем

$$S = N(e) - mT \ . \qquad (17)$$

Рассмотрим приращение ΔT длины мировой линии T. Аналогично (2) имеем

$$m = \frac{\Delta N(e)}{\Delta T} \ . \qquad (18)$$

Рассмотрим ΔT =1, то есть приращение средней длины ориентированных маршрутов на одно ребро. Для этого длину всех ориентированных маршрутов надо увеличить на одно ребро. Это легко сделать при четном n. Выберем увеличение длины маршрутов в будущее (или в прошлое) и к каждой паре исходящих (входящих) валентностей присоединим новую вершину. При этом все маршруты удлинятся на одно ребро и их число удвоится. При нечетном n останется одна свободная валентность. При большом n если неиспользованную валентность выбрать так, что она является концом минимального числа маршрутов, то приращение средней длины будет близко к 1. Таким образом, число ребер увеличивается на n. Ширина x-графа – это топологическая характеристика, которая соответствует собственной массе частицы. Окончательно имеем

$$S = N(e) - nT \ , \qquad (19)$$

или

$$S = 2N - nT \ . \qquad (20)$$

Соотношения (19 - 20) являются прообразом действия для дискретной модели массивной частицы и создаваемого ею гравитационного поля. Макроскопический наблюдатель фиксирует собственную массу частицы и собственное время вдоль ее мировой линии. Принцип наименьшего действия в этом случае означает, что с вероятностью, очень близкой к единице, частице соответствует 4-микросостояние с

минимальным числом вершин (ребер), что макроскопически означает минимизацию интеграла от скалярной кривизны.

9. Модель 4-микросостояния массивной частицы

Рассмотрим модель типичного 4-микросостояния точечной частицы, для чего рассмотрим топологию х-графа, которая минимизирует число его вершин при фиксированной ширине и средней длине ориентированных маршрутов. Что эквивалентно, рассмотрим обратную задачу поиска х-графа с максимальной средней длиной ориентированных маршрутов при фиксированных ширине n и числе вершин N. Возьмем $Q= N-n+2$ вершин. Построим из них х-граф, представляющий собой последовательность вершин, соединенных двойными ребрами. Этот х-граф содержит $2^{(Q+1)}$ ориентированных маршрутов, каждый длинной Q-1 ребер. Это единственная топология при ширине 2. Построим х-граф ширины 4, добавив две вершины, которые присоединены одним ребром каждая одна к исходящей свободной валентности, а другая к входящей. Это приведет к увеличению ширины до 4. Мы получили х-граф из Q+2 вершин, имеющий максимальную среднюю длину ориентированных маршрутов

$$T_{max(n=4)} = \frac{2^{Q-1}(9Q+3)}{9 \times 2^{Q-1}+4} \approx Q \quad . \qquad (21)$$

Наличие цифры 4 в знаменателе связано с появлением четырех ориентированных маршрутов нулевой длины. Несложно проверить перебором всех вариантов увеличения ширины х-графа до 4, что другие способы присоединения двух вершин к свободным валентностям последовательности двойных ребер не приводят к большей средней длине маршрутов. Очевидно, что любое добавление в середину последовательности двойных ребер двух вершин, связанных одним ребром, приводит к уменьшению длины некоторых ориентированных маршрутов, а, следовательно, и средней длины.

Рассмотренное построение дает алгоритм построения х-графа заданной ширины и максимальной или близкой к максимальной средней длине ориентированных маршрутов.

К концам последовательности вершин, связанных двойными ребрами последовательно добавляется n-2 вершин, связанных с имеющимся х-графом одним ребром. Очевидно, что х-графы такого вида имеют минимальное или близкое к минимальному число вершин при фиксированной средней длине ориентированных маршрутов.

Существенным недостатком построенной модели является явная неоднородность по времени. Если такую однородность потребовать, то получаем следующую модель точечной частицы. Имеется последовательность вершин, связанных двойными ребрами. Изредка имеется топологическая особенность, состоящая в том, что пара вершин связана одинарным ребром вместо двойного. Вместо второго ребра могут иметься следующие структуры. Во-первых, может иметься свободная валентность. Во-вторых, может иметься ребро, связывающее рассматриваемую частицу с другой частицей, то есть другим подграфом. В-третьих, может иметься ребро, одним концом инцидентное вершине из одной пары, образующей топологическую особенность, а другим концом инцидентное вершине из другой пары, образующей топологическую особенность, то есть длинное ребро, идущее параллельно последовательности двойных ребер. Полученная модель массивной частицы согласуется с современными представлениями, что мы экспериментально наблюдаем не голую частицу, а окруженную облаком виртуальных частиц. При этом последовательность двойных ребер соответствует некоторому ядру с минимальной затравочной массой. Основная масса создается облаком длинных ребер, параллельных ядру.

Модель описывает не какую-то определенную элементарную частицу, а является общей моделью для всех лептонов и кварков. Модель конкретной частицы должна иметь специфические топологические свойства, которые идентифицируются с квантовыми числами. В настоящей работе они не рассматриваются и сейчас не известны. Однако, все лептоны и кварки обладают одинаковым квантовым числом – спином ½. В теории элементарных частиц важную роль играют симметрии. В случае х-графа все симметрии являются симметриями относительно перестановок вершин и/или

ребер. Выскажем гипотезу, что спин ½ описывает симметрию х-графа относительно перестановки ребер в двойном ребре.

10. Обсуждение

В предлагаемом подходе пространство-время и частицы рассматриваются как проявления единой дискретной микроструктуры. Близкой моделью является гипотеза причинностного множества (causal set) [16, 17, 18], где предполагается, что пространство-время в микромире дискретно, состоит из элементарных событий. В этой модели элементарные события также могут быть представлены как вершины ориентированного ациклического графа. Соответствие графа и непрерывного пространства-времени строится с помощью вложения графа в пространство-время с соблюдением следующих условий. Каждой вершине графа ставится в соответствие одна и только одна точка пространства-времени. Вершины графа причинно связаны тогда и только тогда, когда причинно связаны соответствующие им точки пространства-времени. В среднем 4-объем пропорционален содержащемуся в нем числу вершин. Однако в рамках этой модели пока не построена процедура перехода от дискретной структуры к непрерывному пространству-времени. Настоящей работе более близки идеи, рассматриваемые в работе [19].

В предлагаемой модели соответствие х-графа и непрерывного пространства-времени более сложное. Не предполагается непосредственного соответствия на микроуровне вершин графа и точек пространства-времени. Соответствие имеется только на уровне макроскопических усредненных характеристик. Состоящие из ребер ориентированные маршруты составляют мировую линию точечной частицы. Макроскопической длине мировой линии соответствует средняя длина этих маршрутов. Число вершин в виде интеграла от скалярной кривизны распределено по всему 4-мерному объему. Отметим, что в литературе обсуждается идея о том, что скалярная кривизна не является не самостоятельной физической сущностью, а является частью материальных частиц [20].

Модель частицы строится до пространства-времени, что рассматривалось в настоящей работе. При этом производится идентификация некоторых средних топологических характеристик х-графа с наблюдаемыми физическими величинами. В ходе дальнейших исследований дискретных моделей частиц предполагается идентификация их топологических характеристик с зарядами и другими квантовыми числами. Направлением дальнейших исследований является построение спектра реальных частиц. Он является следствием алгоритма последовательного роста. Алгоритм должен приводить к формированию дискретного спектра устойчивых топологических структур х-графа, повторяющихся в ходе последовательного роста. Возможность возникновения богатого спектра устойчивых структур в простой дискретной модели продемонстрирована в игре «Жизнь» (см., например, [23]).

Соотношение (19 - 20) названо прообразом действия, а не действием, так как в модели не простроено пространство-время. Имеется дискретная модель мировой линии частицы. В этом смысле построена дискретная модель второго слагаемого в действии (1). Для первого слагаемого в (1) получен дискретный аналог значения интеграла, но нет, ни модели области интегрирования, ни подынтегральной функции.

В дальнейшем предполагается построение пространства-времени, как описания макроскопической сети взаимодействующих частиц. В гладком четырехмерном многообразии задание длин всех времениподобных и изотропных мировых линий в каждой бесконечно малой окрестности фиксирует метрику псевдориманова пространства-времени [21] (достаточно только времениподобных мировых линий [22]). При этом фиксируется и скалярная кривизна, для ее варьирования не остается возможности. Однако, в рассматриваемой модели задаются длины только на мировых линиях реальных частиц, что, возможно, позволяет сохранить необходимый произвол в выборе кривизны. Это вопрос для дальнейших исследований. При этом следует получить и размерность 3+1 пространства-времени.

Предлагаемая модель строится в рамках стандартной теории вероятности. По мнению автора особенности квантового описания

связаны с тем, что дискретные структуры микромира описываются в терминах непрерывного пространства-времени, и на последовательно дискретном языке они получат ясную интерпретацию.

Топологические характеристики графа описываются целыми числами и отношениями целых чисел.

Топологические характеристики повторяющихся структур графа могут описываться комплексными числами при использовании анализа Фурье. Однако вопрос о классическом или квантовом характере динамики х-графа не влияет на результаты настоящей работы.

Пример нерелятивистской термодинамики показывает, что законы термодинамики не зависят от того, классической или квантовой статистике подчиняются микросостояния, так как квантовые эффекты проявляются на макроуровне только для специфических систем, например, сверхпроводников.

Автор выражает благодарность профессору Ю.Г. Рудому, обсуждение релятивистской термодинамики с которым стало отправной точкой настоящего исследования, а также М.Г. Мещерякову за ценные критические замечания по статистическим ансамблям х-графов.

Работа выполнена по теме «Развитие методов математического моделирования распределенных систем и соответствующих методов вычисления» (0065-2018-0004), регистрационный номер в ЦИТиС АААА-А18-118041290146-7.

Principle of least action and thermodynamics of directed acyclic graph

A.L.Krugly

Abstract: The discrete model of a finite part of continuous spacetime is considered. In microscopic level this model is a finite connected directed acyclic graph. Each vertex has indegree and outdegree no more than 2. Statistical description of graphs is considered. Different macroscopic states are different sets of graphs. The action has the form as in general relativity theory. The principle of least action means the maximum of probability of state. We get the set of graphs that describe a massive particle and its gravitational field. Some topological properties of the graph can be identified as integral of scalar curvature, mass and proper time of the particle.

Keywords: principle of least action, theory of relativity, thermodynamics, statistical physics, directed acyclic graph.

Литература

1. Р. Толмен. Отноительность, термодинамика и космология. Изд. 2-е, испр. М.: Книжный дом «ЛИБРОКОМ», 2009.
2. J.D. Bekenstein. Black Holes and Entropy. Phys. Rev. D 7 (1973) 2333.
3. S.W. Hawking, Particles Creation by Black Holes. Commun. Math. Phys. 43 (1975) 199.
4. W.G. Unruh. Notes on Black Hole Evaporation. Phys. Rev. D 14 (1976) 870.
5. T. Jacobson. Thermodynamics of Spacetime: The Einstein Equation of State. Phys. Rev. Lett. 75 (1995) 1260, (gr-qc/9504004).
6. T. Padmanabhan. Thermodynamical Aspects of Gravity: New insights. Rep. Prog. Phys. 73 (2010) 046901, (gr-qc/09115004).
7. E.P. Verlinde. On the origin of gravity and laws of Newton. JHEP 1104 (2011) 029, (arXiv:1001.0785).
8. D. Moustos. Gravity as a thermodynamic phenomenon. M.Sc. Thesis. Department of Physics, University of Patras, Greece, (arXiv:1701.08967).

9. Ю.Г. Рудой. Роль Макса Планка в создании релятивистской термодинамики. В сб. Исследования по истории физики и механики. 2009 – 2010, М., Физматлит, 2010, 59 – 91.
10. В. Паули. Теория относительности. М.: Наука, 1983.
11. Л. Д. Ландау, Е.М. Лифшиц. Теория поля. Изд. 6-е, исправленное и дополненное. Серия «Теоретическая физика», том II, М.: Наука, 1973.
12. A.L. Krugly. A sequential growth dynamics for a directed acyclic dyadic graph. Вестник Университета Дружбы Народов. Серия: Математика, Информатика, Физика. 2014, No 1, 124-138 (arXiv: 1112.1064).
13. А.М. Хазен. Введение меры информации в аксиоматическую базу механики. М.: Рауб, 1998.
14. В. Феллер. Введение в теорию вероятностей и ее приложения. Т.1. М.: Мир, 1984.
15. П.К. Рашевский. Риманова геометрия и тензорный анализ. Изд. 5-е, стереотипное. М.: Едиториал УРСС, 2006.
16. G. 't Hooft. Quantum gravity: a fundamental problem and some radical ideas, in Recent Development in Gravitation, Proceedings of the 1978 Cargese Summer Institute, ed. by M. Levy and S. Deser, Plenum, New York/London, 1979, 323-345.
17. J. Myrheim. Statistical Geometry, CERN preprint TH-2538, 1978.
18. Bombelli, L., Lee, J., Meyer, D. and Sorkin, R.D. Space-time as a causal set. Phys. Rev. Lett., 59 (1987), 521-524.
19. M. Cortês, L. Smolin. Energetic causal sets. Phys. Rev. D 90, 044035 (2014), (arXiv 1308.2206).
20. I.E. Bulyzhenkov. Einstein's Gravitation for Machian Relativism of Nonlocal Energy-Charges. Int. J. Theor. Phys. 47 (2008) 1261-1269.
21. S.W. Hawking, A.R. King, P.J. McCarthy. A new topology for curved space-time which incorporates the causal, differential and conformal structures. Journal of mathematical physics, 17 (1976) 174-181.
22. D.B. Malament. The class of continuous timelike curves determines the topology of space-time. Journal of mathematical physics, 18 (1977) 1399-1404.
23. J. Hemmingsen. Consistent Results on Life. Physica D80 (1995) 80.

Инструментарий подготовки блочно-структурированной сетки для проведения расчетов методом RANS/ILES

Л.А. Бендерский[1], Д.А. Любимов[2], А.А. Рыбаков[3]

Межведомственный суперкомпьютерный центр РАН - филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия,

E-mail's: [1] leosun.ben@gmail.com, [2] dalyubimov@ya.ru, [3] ybakov.aax@gmail.com

Аннотация: Метод RANS/ILES с явным разрешением турбулентных вихрей успешно применяется для расчета нестационарных турбулентных течений. Применение данного метода требует большого количества вычислительных ресурсов, которое может обеспечить только суперкомпьютер. При использовании суперкомпьютера остро встает вопрос распараллеливания вычислений и их масштабирования на расчетные сетки, содержащие большое количество ячеек. В данной статье описываются подходы к подготовке блочно-структурированной расчетной сетки, позволяющие эффективно проводить вычисления на суперкомпьютере с использованием RANS/ILES метода.

Ключевые слова: RANS/ILES, суперкомпьютер, масштабирование, блочно-структурированная расчетная сетка, дробление расчетной сетки, распределение вычислительной нагрузки.

Введение

Комбинированные RANS/ILES методы с явным разрешением турбулентных вихрей успешно применяются для расчетов силовых установок высокоскоростных летательных аппаратов [1]. Данные методы требуют применения разностных схем для конвективных членов уравнений Навье-Стокса, имеющих низкий уровень схемной вязкости. Обычно для вихреразрешающих методов используются схемы с высоким порядком аппроксимации параметров на гранях ячеек, что приводит к увеличению объема данных, передаваемых между вычислительными процессами. Также вихреразрешающие методы требуют использования сеток с существенно большим количеством ячеек, чем методы RANS для явного разрешения турбулентных вихрей. Кроме того, турбулентные течения нестационарны, и для их описания также необходимо решать численно нестационарные уравнения. При этом для явного разрешения вихрей шаг по времени должен быть мал, а время интегрирования должно быть достаточным для того, чтобы получить с требуемой точностью осредненные параметры течения и турбулентности. Из сказанного выше следует, что для эффективного численного решения задач такого класса требуется применение суперкомпьютеров [2].

Описание RANS/ILES метода

В комбинированном методе RANS/ILES около стенок для расчета течения решаются нестационарные уравнения Навье−Стокса с моделью турбулентности Спаларта−Аллмараса. Вдали от стенок течение описывается с помощью LES с неявной SGS-моделью − ILES. При таком подходе отсутствует явная SGS-модель турбулентности, а ее функцию выполняет схемная вязкость разностной схемы. Для расчета конвективных потоков на гранях расчетных ячеек используется схема Роу, предраспадные параметры для которой вычисляются с помощью схемы 9-го порядка MP9. Для реализации данной схемы необходимо знать значения параметров течения из 5 соседних ячеек с каждой стороны рассматриваемой грани. Диффузионные потоки на гранях ячеек определяются с помощью аппроксимации с центральными разностями со вторым порядком. Интегрирование по времени выполняется с помощью технологии интегрирования по двойному времени. Интегрирование по физическому времени выполняется со вторым порядком точности.

В области около стенок, где течение описывается с помощью RANS, конвективные потоки на гранях расчетных ячеек в разностном аналоге уравнения для модели турбулентности вычисляется с помощью схемы WENO5. В области ILES модель турбулентности Спаларта–Аллмараса изменяется таким образом, чтобы турбулентная вязкость равнялась нулю [3].

Вычисления методом RANS/ILES проводятся на блочно-структурированной расчетной сетке. RANS/ILES метод показал свою состоятельность при расчетах высокоскоростных турбулетных течений на суперкомпьютере и продемонстрировал близкую к линейной масштабируемость, а на количестве вычислительных узлов до 20 – даже сверхлинейную масштабируемость [4].

Внутреннее представление блочно-структурированной сетки

Блочно-структурированные расчетные сетки более предпочтительны для проведения вычислений, чем неструктурированные, так как состоят из отдельных блоков, ячейки в каждом из которых упорядочены в памяти. Это позволяет быстрее производить вычисления и снижает требования к объему памяти. Однако построение блочно-структурированных сеток значительно затруднено по сравнению с неструктурированными [5]. Построение же блочно-структурированных расчетных сеток вручную приводит к неоптимальному распределению вычислительной нагрузки при проведению расчетов, так как зачастую такие сетки содержат блоки с большим количеством ячеек.

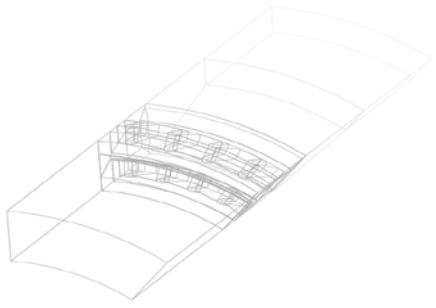


Рис. 1. Пример расчетной сетки с выраженнымикрупными блоками.

Для повышения эффективности использования блочно-структурированных сеток при проведения расчетов RANS/ILES методом было разработано внутреннее представление сетки [6] и программный комплекс для подготовки сетки к вычислениям.

Основным объектом блочно-структурированной расчетной сетки является блок (Block), содержащий трехмерный массив ячеек. В блоке хранится информация о газодинамических параметров всех ячеек, а также координаты всех вершин ячеек. Каждый блок обладает собственной системой координат (I, J, K). Отдельные блоки расчетной сетки могут соприкасаться друг с другом, касание двух блоков по ненулевой площади описывается парой интерфейсов (Iface). Кроме координат касания граней блоков пара интерфейсов также устанавливает взаимное соответствие систем координат касающихся блоков. На границе блока кроме другого блока может находиться также граница расчетной области, в этом случае на данной границе можно задать граничное условие (BCond).

На рис. 1 представлена блочно-структурированная расчетная сетка, представляющая 36-градусный сектор элемента силовой установки высокоскоростного летательного аппарата, состоящая из 30 блоков, 152 интерфейсов и 204 граничных условий.

Измельчение крупных блоков расчетной сетки

В блочно-структурированной сетке все объекты обладают каркасом, который представляет собой двумерный или трехмерный массив узлов. Данный каркас может быть разрезан вдоль любого направления (I, J или K) по линии между соседними ячейками сетки. Таким образом любой объект расчетной сетки может быть разрезан по любому направлению. Дробление блока сетки является важнейшим действием, позволяющим обеспечить равномерное распределение блоков сетки между обрабатывающими параллельными процессами. Наиболее простой стратегией дробления сетки является дробление наиболее крупных блоков сетки на две

равные части [7]. После выполнения очередного разреза наиболее крупного блока по наиболее протяженному направлению выполняется попытка равномерно распределить получившиеся блоки между вычислительными процессами. При этом показателем качества распределения является отклонение наиболее загруженного процесса от среднего – значение, на которое отличается максимальное количество ячеек, обрабатываемое одним процессом, от среднего значения по всем процессам.

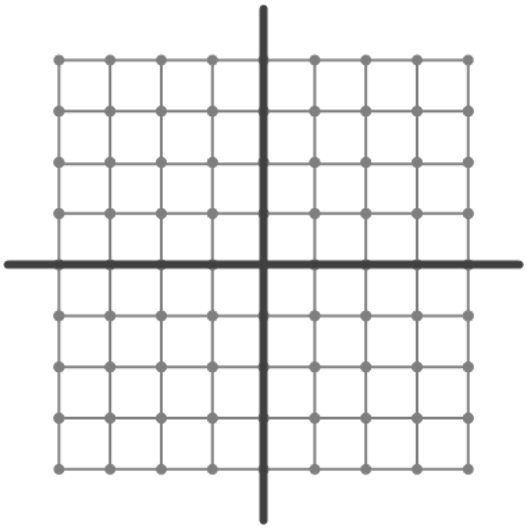


Рис. 2. Возможные варианты разреза при делении крупного блока на две равные части.

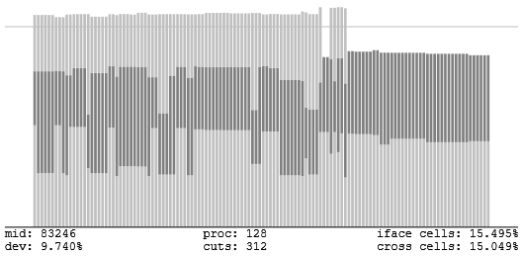


Рис. 3. Деление крупных блоков для распределения сетки на 128 процессов (312 разрезов при отклонении 9.74%).

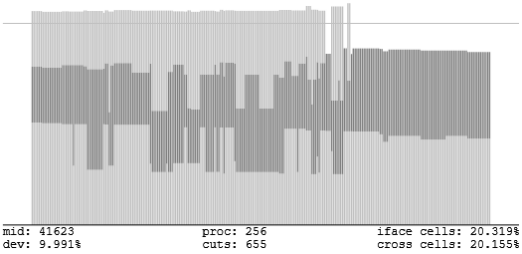


Рис. 4. Деление крупных блоков для распределения сетки на 256 процессов (655 разрезов при отклонении 9.99%).

Минимизация количества разрезов расчетной сетки

Дробление расчетной сетки путем разрезания наиболее крупных блоков выполняется достаточно быстро, однако такой подход обладает рядом недостатков. Зачастую данный метод требует выполнения большого количества разрезов (несмотря на то, что производятся они достаточно быстро), что приводит к возрастанию объема данных (и увеличению самого количества сообщений), пересылаемых между вычислительными процессами во время проведения расчетов.

Поэтому был предложен другой подход, во время которого на каждом шаге дробления блоков рассматривается все доступное множество разрезов всех блоков (см. рис. 5) и из этого множества выбирается наиболее оптимальный по некоторому набору эвристик. Данный метод подробно описан в [4].

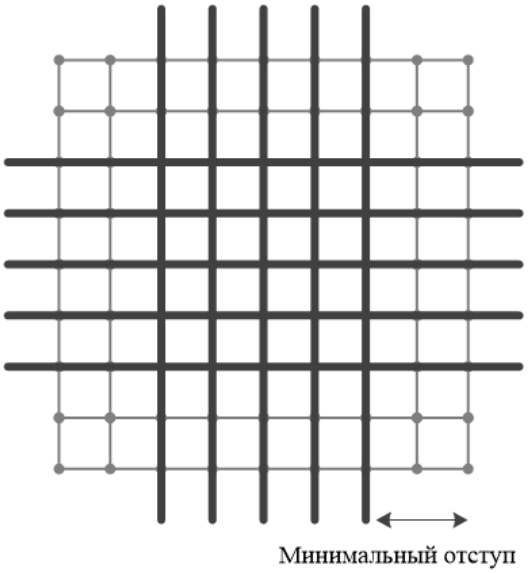


Рис. 5. Возможные варианты разреза при минимизации количества дроблений.

На рисунках 6 и 7 представлены распределения блоков расчетной сетки на 128 и 256 процессов методом минимизации количества разрезов для рассматриваемой в данной статье тестовой сетки. Применение данного метода демонстрирует примерно двукратное сокращение количества разрезов по сравнению с обычным методом последовательных разрезов крупных блоков.

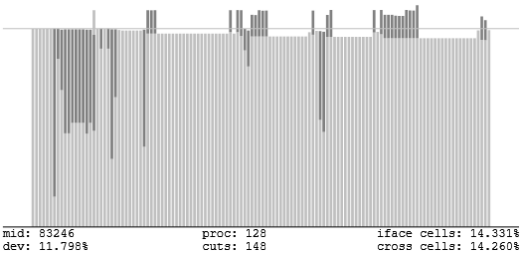


Рис. 6. Минимизация количества дроблений для распределения сетки на 128 процессов (148 разрезов при отклонении 11.798%).

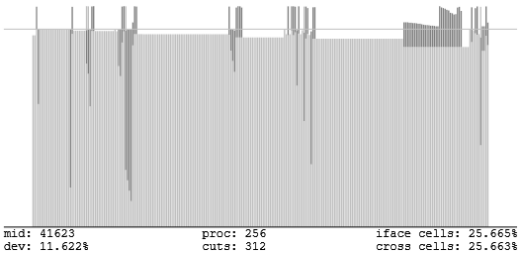


Рис. 7. Минимизация количества дроблений для распределения сетки на 256 процессов (312 разрезов при отклонении 11.622%).

Реализация связанных граничных условий

При расчете газодинамических процессов внутри осесимметричной области целесообразно производить прямой расчет лишь части области, а результаты в дополнительной области устанавливать в соответствии с симметрией. На рис. 8 представлена расчетная осесимметричная сетка, состоящая из десяти одинаковых 36-градусных сегментов (расчетная сетка, соответствующая одному сегменту, представлена на рис. 1).

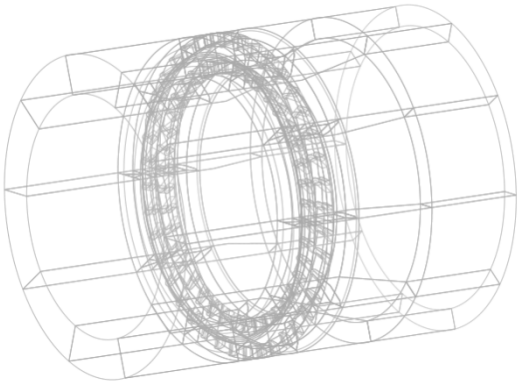


Рис. 8. Дополнение сетки с рис. 1 до 360-градусного объекта с помощью использования связанных граничных условий.

Для того, чтобы было возможно проводить расчет только внутри одного 36-градусного сегмента, необходимо поддерживать синхронное изменение данных на границах сегмента с его симметричными соседями. Это приводит к тому, что появляются так называемые связанные граничные условия – граничные условия, заданные сразу на паре пространственно разнесенных граней блока. Данная пара граничных условий имеет одинаковые размеры и при дробление одного граничного условия из этой пары приводит к автоматическому дроблению его отражения (Mirror BCond).

В программном комплексе реализован механизм связанных граничных условий с автоматическим поиском этих условий. Поиск связанных граничных условий определяется типом движения, с помощью которого связанные граничные условия совмещаются друг с другом (реализованы два типа движения – параллельный перенос и поворот вокруг заданной оси вращения).

Реализованный механизм позволяет для сетки, изображенной на рис. 8 производить прямой расчет только одного сектора, что ускоряет расчеты ровно в 10 раз на рассматриваемой тестовой сетке.

Заключение

В статье рассмотрены важные инструменты подготовки блочно-структурированной расчетной сетки к проведению высокопараллельных расчетов RANS/ILES методом на суперкомпьютере. Рассмотрены два метода распределения блоков расчетной сетки по вычислительным процессам, позволяющие распределить вычислительную нагрузку на 128, 256 и более процессов с отклонением, не превышающим 10% от среднего значения. Рассмотрен механизм использования связанных граничных условий, позволяющий ускорить в разы вычисления, проводимые на осесимметричных расчетных сетках.

Работа выполнена в рамках проекта «Исследование с использованием суперкомпьютера физических особенностей нестационарных турбулентных течений и

газодинамического управления этими течениями в элементах силовой установки гиперзвукового летательного аппарата» по программе фундаментальных исследований президиума РАН «Фундаментальные основы прорывных технологий в интересах национальной безопасности». В работе использовался суперкомпьютер МВС-10П, находящийся в МСЦ РАН.

The toolkit for preparing a block-structured grid for RANS/ILES calculations

L.A. Benderskiy, D.A. Lyubimov, A.A. Rybakov

Abstract: The RANS/ILES method with an explicit resolution of turbulent vortices is successfully used to calculate nonstationary turbulent flows. The application of this method requires a large amount of computing resources, which can be provided only by a supercomputer. When using a supercomputer, the problem of parallelizing computations and their scaling to computational grids containing a large number of cells is acute. This article describes approaches to the preparation of a block-structured computational grid, which allow efficient calculation on a supercomputer using the RANS/ILES method.

Keywords: RANS/ILES, supercomputer, scaling, block-structured grid, cutting of calculated grid, computational load distribution.

Литература

1. Д.А. Любимов, И.В. Потехина. Исследование нестационарных режимов работы сверхзвукового воздухозаборника RANS/ILES-методом. ТВТ. 2016. Т. 54. № 5. С. 784–791.
2. О.С. Аладышев, Е.А. Киселев, Г.И. Савин, П.Н. Телегин, Б.М. Шабанов. Влияние характеристик внешней памяти суперкомпьютерных комплексов на выполнение параллельных программ. Системы и средства информатики. 2014. Т. 24. № 4. С. 111-123.
3. Д. А. Любимов. Разработка и применение метода высокого разрешения для расчета струйных течений методом моделирования крупных вихрей. ТВТ. 2012. Т. 50. № 3. С. 450-466.
4. Л.А. Бендерский, Д.А. Любимов, А.А. Рыбаков. Анализ эффективности масштабирования при расчетах высокоскоростных турбулентных течений на суперкомпьютере RANS/ILES методом высокого разрешения. Труды НИИСИ РАН. Т. 7. № 4. 2017. С. 32-40.
5. V. Liseikin. Grid generation methods. Springer. 2010.
6. А.А. Рыбаков. Внутреннее представление и механизм межпроцессного обмена для блочно-структурированной сетки при выполнении расчетов на суперкомпьютере. Программные системы: Теория и приложения. № 1 (32). 2017. С. 121-134.
7. А.А. Рыбаков. Распределение вычислительной нагрузки между узлами суперкомпьютерного кластера при расчетах задач газовой динамики с дроблением расчетной сетки. Международный научный журнал «Современные информационные технологии и ИТ-образование». Т. 12. № 2. 2016. С. 101-107.

О цифровой экономике

С.Г. Романюк

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

Аннотация: В 2017 году Правительством России была принята программа перехода к цифровой экономике, однако во многих публикациях признаётся, что общепринятого определения термина «цифровая экономика» не существует. В статье говорится о том, какой смысл может приобрести словосочетание «цифровая экономика», если отталкиваться от значения слова «цифровая», в котором оно употребляется в информатике и вычислительной технике, и какие из этого могут быть сделаны практические выводы.

Ключевые слова: компьютер, компьютерная программа, информация, аналоговые данные, цифровые данные, неосязаемость, оцифровка, интерпретация, искусственный интеллект, Интернет, электронный документ.

1. Введение

Для начала заметим, что в словосочетании «цифровая экономика» ключевым является первое слово, «цифровая», и приведем некоторые элементарные сведения и трактовку основных терминов из книги [1].

Информатика, по-немецки — Informatik, по-английски — Computer Science. Как следует из названий, содержанием, предметом исследования этой науки является ***информация и компьютерные программы***. (Прямой перевод Computer Science звучит как «Наука о компьютерах», но этот перевод явно неудовлетворителен, и как справедливо заметил голландский математик Э. Дейкстра, «Computer Science — это такая же наука о компьютерах, как астрономия — наука о телескопах».

Сигналы — изменения в мире, которые могут быть зарегистрированы органами чувств человека или приборами.

Данные — результат регистрации сигналов.

Информация — результат интерпретации данных.

Про данные обычно говорят, что их «обрабатывают» или «анализируют», но правильнее говорить «интерпретируют».

Интерпретация — это наделение данных смыслом, и делать это может только человек, причем каждый делает это по-своему. (В голове у каждого — «свой собственный интерпретатор»). Нередко разные люди приписывают одним и тем же данным разный смысл, и добиваться одинаковости интерпретации — самая главная задача на пути к установлению взаимопонимания. По выражению Ф. Тютчева, «Нам не дано предугадать, как слово наше отзовётся».

Из приведенных определений следует, например, что информацию нельзя «принять» или «передать», «прочитать» или «записать», «купить», «продать» или «украсть», что информация не может быть «конфиденциальной», «секретной» или «закрытой», эти слова применимы к ***данным***, хотя в обиходной речи говорят, например, «проинформируйте». Кстати, в Советском Союзе в официальных документах не говорилось «секретная информация», говорилось «секретные сведения» (т.е. *данные*) или «сведения, составляющие тайну». Хороший пример — машинный перевод с одного языка на другой. Процесс машинного перевода начинается с разбиения текста на предложения, после чего выполняется грамматический разбор, а выделенные грамматические части располагаются в порядке, который налагает логический смысл на предложение. Раз это сделано, каждая часть может быть переведена посредством словаря на другой язык, после чего предложение пересобирается согласно грамматическим правилам другого языка. Это просто, если предложения простые. Но предложения

бывают сложные, и более того, двусмысленные, многосмысленные и даже бессмысленные. Люди находят смысл в предложении только потому, что налагают на него свои культурные пристрастия, жизненный опыт, и могут, вообще говоря, отыскать смысл там, где его нет. Точный машинный перевод невозможен просто потому, что компьютер (точнее, компьютерная программа) не способен установить, есть ли в предложении смысл. (Кстати, для того чтобы точнее передать смысл, переводчик-профессионал может выбрать вариант перевода какого-то слова, отсутствующий в словарях. Это — к вопросу об «искусственном интеллекте».)

Довольно часто люди говорят «информация», подразумевая «данные». Такие сочетания как «объем информации» или «скорость передачи информации» становятся более понятными, если слово «информация» заменять словом «данные». Расхожее утверждение о том, что «в современном мире люди захлебываются в потоке информации» неверно: люди захлебываются в потоках *данных,* а истинная проблема в том, что скорость обмена данными между людьми возросла благодаря Интернету в сотни или тысячи раз, в то время как скорость их осмысления (осознания, интерпретации, *понимания*) практически не изменилась, поскольку определяется в конечном счете скоростью химических реакций в головном мозге и степенью подготовленности (образованности) читателя (или слушателя). Смысл одной и той же фразы одни улавливают мгновенно, другим требуется несколько секунд, третьим — несколько месяцев. Это «время извлечения информации из данных» зависит от степени натренированности памяти, что является по сути одной из целей обучения.

Заметим, что данные (но не информация!) могут быть неверными, заведомо ложными, искаженными. Принцип «свободы распространения информации» означает по существу ничем не ограниченную свободу распространения *данных*, однако данные могут быть секретными, конфиденциальными, персональными, так что без ограничения свободы обойтись невозможно и указанный принцип на практике не реализуем. Сторонники этого принципа по существу выражают интересы разведывательных служб, задача которых — добыча данных и пересылка их тем, кто способен эти данные интерпретировать, а также, в качестве «побочного эффекта» — криминальных элементов. В статье [2] такого рода принципы называются «полезным идиотизмом», а их проповедники — «полезными идиотами».

По существу, следовало бы говорить о защите людей от потоков бесполезных *данных*, зашумляющих каналы связи и отвлекающих людей от того, что им важно и интересно (характерный пример — реклама). Темп поступления данных должен быть согласован с темпом их обработки (т. е. осмысления, интерпретации), в противном случае информация, заключенная в данных, не доходит до сознания. И вообще, «прочитать» или «услышать» — ещё не значит «понять».

Наконец, отметим ещё эмоциональный аспект «понимания». Дело выглядит так, как если бы в каждой человеческой голове существовал свой «резонатор», возбуждающийся на разное: одно и то же сообщение (данные) кого-то оставляет равнодушным, а у кого-то вызывает инфаркт. Это — опять к вопросу об искусственном интеллекте: у компьютерной программы инфарктов не бывает.

2. Данные аналоговые, цифровые, компьютерные, сетевые

Данные регистрируются (фиксируются) на каком-либо **физическом носителе**. Это может быть любой материальный объект: классная доска, скала, стена здания, ткань, бумага, магнитная лента или диск, радиоволна, память компьютера, мозг человека. При разрушении (уничтожении) носителя данные исчезают. При старении (износе) носителя данные искажаются (частично утрачиваются).

Аналоговые данные — это те, которые зафиксированы на носителе непосредственно, или «в аналоговой форме», например текст на бумаге, написанный чернилами, изображение на фотопленке, музыкальная запись на пластинке или на магнитной ленте. Аналоговые данные могут фиксироваться вручную или посредством специальной аппаратуры (фотоаппарат, магнитофон), и тогда для воспроизведения тоже может потребоваться специальная аппаратура.

Цифровые данные можно представлять себе как последовательность нулей и единиц.

Понятие цифровых данных лучше всего объясняется на примере азбуки Морзе. Каждой букве алфавита, цифре, знаку препинания и специальному символу ставится в соответствие уникальная группа точек и тире (аналог нулей и единиц). Таким образом, любой текст удаётся записать в виде последовательности точек и тире (или нулей и единиц). Со временем были придуманы способы записи в цифровом виде фотографий и звуков. Традиционный пример цифровых данных — текст, набранный с клавиатуры компьютера в каком-либо текстовом редакторе.

Цифровые данные могут быть нанесены на те же физические носители, что и аналоговые (например, их можно записать на бумаге или мелом на доске), однако их преимущества проявляются в полной мере при использовании магнитных и полупроводниковых носителей, на которых помещается гигантское количество нулей и единиц.

Компьютерные данные — это цифровые данные, хранящиеся на носителях компьютера. ***Сетевые данные*** — это данные на носителях компьютера, входящего в сеть. Последние два термина не являются общепринятыми. Часто вместо *сетевые* или *компьютерные* данные говорят просто *цифровые*, подразумевая данные в цифровой форме, хранящиеся в компьютерах глобальной сети.

3. Оцифровка данных

Преобразование аналоговых данных в цифровые называется *оцифровкой*. Оцифровка выполняется с помощью специальных устройств (например, сканеров). При оцифровке неизбежна потеря информации. Например, при сканировании страницы текста или фотографии задаются параметры сканирования (разрешение, т. е. число точек на мм, число градаций цвета и др.). При разных значениях этих параметров получаются разные цифровые образы одного и того же аналогового оригинала. Точное обратное преобразование (из цифровой формы в аналоговую) невозможно. Можно, конечно, попробовать напечатать цифровой образ и сфотографировать его, но исходного, оригинального фото, с которого мы начали, не получится. Цифровая фотография отличается от аналоговой так же, как живая птица от ее чучела. (Поэтому процесс оцифровки данных можно было бы назвать «бальзамированием».)

Заметим, что аналогичная ситуация имеет место в соотношении устного и письменного языков. Написанный текст всегда можно произнести вслух, «озвучить», но при попытке записать сказанное неизбежно что-то теряется (информация?). Теряются какие-то оттенки речи, зафиксированные в мозгу слушающего, которые не находят отражения в записанном тексте, то ли потому, что записывающий не придал им значения, то ли потому, что не знал, как их отразить. Пожалуй, каждый может вспомнить пример, когда интонация или пауза изменяла смысл чуть ли не на противоположный.

Соотношение между цифровыми и аналоговыми данными можно уподобить также соотношению между рациональными и иррациональными числами, где каждому иррациональному числу соответствует бесконечно много рациональных приближений, ни одно из которых не является его «точной копией». Другими словами, цифровые данные представляют собой некоторое приближение аналоговых.

Практически все библиотеки и архивы (да и не только они) переводят хранящиеся печатные материалы в цифровую форму. Здесь возможны по крайней мере два пути: (1) ввод текстов книг и других документов с клавиатуры и (2) сканирование. В любом случае получается оцифрованный вариант, но с потерей *разной* информации. В первом случае исчезают графики, рисунки и другие нетекстовые данные, во втором теряется структура текста. Для её восстановления применяются программы распознавания текстов, однако качество восстановленного текста сильно зависит от исходного образца и параметров сканирования. Таким образом, широко распространенное мнение о том, что реальные книги можно без будущего ущерба уничтожить, если имеются их цифровые копии, глубоко ошибочно.

В заключение заметим, что в настоящее время во многих случаях данные порождаются сразу в цифровой форме, например, в камерах видеонаблюдения и цифровых фотоаппаратах, которые встраиваются теперь практически во все мобильные телефоны.

В издательском деле, научных экспериментальных исследованиях и в промышленном производстве процесс

перехода на цифровые данные развивается давно: оригинал-макеты книг в цифровой форме, датчики со встроенным АЦП (аналого-цифровым преобразователем) — примеры генерации цифровых данных «на лету».

4. Особенности цифровых данных

Напомним, что здесь речь идет о цифровых данных, находящихся в компьютерах глобальной сети. Несомненным достоинством этих данных является их глобальная доступность, а также неограниченные возможности и лёгкость объединения данных любой природы «в одном файле»: числа, тексты, видео, аудио и т. д.

4.1. Отчуждённость от носителя

При работе компьютера постоянно происходит переписывание данных, в том числе в технологических целях (таких как резервное копирование), поэтому любые данные существуют в неопределенном (и изменяющемся во времени) количестве *копий*. Невозможно сказать, на каком диске какого компьютера сети хранятся конкретные данные, да это никого и не интересует, важно, что по определенному запросу они будут выданы. Можно сказать, что носитель данных *обезличен*, а сами **данные *отчуждены* от носителя**. Это, в свою очередь, означает, что в **мире цифровых данных отсутствует понятие *оригинала* или *подлинника***, есть только *неразличимые копии*, ведь подлинник обладает некими уникальными свойствами, особенностями, позволяющими выделить его среди других образцов.

В мире аналоговых данных ситуация иная. В качестве примера возьмем книгу, отпечатанную тиражом 100 экземпляров. Их все можно считать подлинниками, поскольку они *принципиально* различимы: на экземпляр, попавший в библиотеку, ставится штамп библиотеки, экземпляр в частной коллекции подписывается владельцем, и даже если нет никаких «постпродажных» признаков, есть типографские дефекты, и вообще, две книги, находящиеся в разных шкафах, *отличимы* друг от друга по «месту пребывания».

4.2. Фальсифицируемость

Цифровые данные гораздо **более уязвимы для фальсификаций**, чем аналоговые. Чтобы изготовить фальшивый бумажный документ, банкноту или фотографию, нужно подобрать подходящую бумагу, особые краски, иметь доступ к специальному оборудованию и т. д. Для подделки цифрового документа ничего этого не требуется, нужны лишь соответствующие компьютерные программы. Примером подобной программы может служить обычный текстовый редактор, например, MS Word. Поскольку этот редактор общедоступен, подготовленный с его помощью документ может фальсифицировать (исказить) кто угодно. Одна из мер защиты от фальсификаций текстовых документов — создание программы-редактора, общедоступная версия которого допускает только чтение (пример — издательская система компании Adobe, формирующая файлы в формате pdf) . Тем самым фальсификация документа усложняется, но не исключается полностью. И вообще, на всякую «анти-фальсификационную» меру изобретательные программисты придумывают контрмеру.

Аналогичная ситуация имеет место для любых цифровых данных, например, фотографий (напомним об общедоступных программах ретуширования фотографий и фотомонтажа). Изготовленные с их помощью фальшивки грубы, но будучи размещенными в Интернете, они своё действие, бывает, оказывают: как говорится, «многие верят», и можно догадаться о существовании более изощренных, не афишируемых программ.

Для защиты цифровых данных от фальсификаций была предложена и активно рекламируется т. н. «технология blockchain». Не вдаваясь в подробности (вопрос заслуживает отдельного рассмотрения), заметим, что эта технология имеет довольно узкую сферу применений и лишь *затрудняет* фальсификацию, но не исключает её.В мире аналоговых данных, которые неразрывно связаны с физическим носителем, подлинность может быть установлена экспертизой, исследующей свойства носителя. В мире цифровых данных, которые от носителя отчуждены, этот способ неприменим.

Существует вроде бы обнадёживающий пример: подлинность «Слова о полку Игореве» [3] была установлена лингвистами

путём анализа собственно текста, а не носителей, однако этот способ вряд ли применим в «промышленных масштабах», когда нужно выбрать «подлинный» текст из двух или нескольких почти одинаковых (классический пример — фраза «Казнить нельзя помиловать», смысл которой зависит от положения запятой).

Считается, что для предохранения цифровых документов от фальсификаций можно использовать криптографию. А именно, вместо исходного документа в компьютер помещается его шифрограмма, которая формируется с помощью закрытого ключа, а расшифровывается с помощью общедоступного открытого.

Таким образом, тот факт, что шифрограмму удалось расшифровать, можно считать доказательством подлинности документа. Тема криптографической защиты требует отдельного обсуждения, здесь же отметим только, что ключи шифрования — это тоже цифровые данные, существующие в неизвестном количестве копий, одной из которых злоумышленник потенциально может воспользоваться для изготовления шифрограммы поддельного документа.

4.3. Неосязаемость

Ещё одно свойство цифровых данных — ***неосязаемость***. Эти данные не воспринимаются органами чувств человека непосредственно, обязательно нужны устройства-посредники (экраны, принтеры и т. д.). Впрочем, это во многих случаях имеет место и для аналоговых данных (например, на магнитной ленте). Важное отличие состоит в том, что кроме устройств требуются ещё многочисленные ***программы-посредники***, преобразующие цифровые данные в образы, воспринимаемые человеком, и если изменена программа, то изменится образ. Далеким от программирования людям невозможно представить, какой гигантский потенциал фальсификаций и просто недоразумений здесь таится!

5. К определению цифровой экономики

Если попытаться выделить нечто общее из опубликованного на эту тему, то краткий ответ на вопрос, в чём суть цифровой экономики, может звучать так: ***тотальный переход на цифровые (компьютерные, сетевые) данные во всех сферах человеческой деятельности***. Здесь слово «переход» означает замену аналоговых данных цифровыми.

Нужно сказать, что переход на цифровые данные начался сразу после появления ЭВМ и во многих областях уже завершился или далеко продвинулся, только раньше этот процесс назывался «автоматизацией»: автоматизация проектирования (САПР), автоматизация управления (АСУ), автоматизация технологического процесса (АСУ ТП) и т. д. Относительно автоматизации собственно экономики и финансов можно вспомнить язык программирования COBOL (Common Business Oriented Language) и систему команд IBM/360 (конец 1960-х), в которую были включены команды, предназначенные для денежных расчетов (т. н. «двоично-десятичная арифметика»). Похоже, в те годы контакты экономистов со специалистами в области Computer Science были гораздо более тесными, чем сейчас.

Ключевые факторы, обусловившие всплеск интереса к этой тематике — появление персонального компьютера, что повлекло беспрецедентный рост числа пользователей ЭВМ, а также Интернета, обеспечившего возможность обмена данными (цифровыми) в глобальном масштабе. Обострение интереса со стороны экономистов, наблюдающееся в последнее время, вызвано массовым внедрением компьютера в сферу торговли и денежных расчетов, прежде всего — в сфере услуг (в самом широком, современном смысле, когда к услугам относятся не только традиционные коммунальные, транспортные, услуги связи и пр., но и здравоохранение, образование и т. д.). Процесс перехода на цифровые данные в каждой конкретной области, где применяется компьютер, будет развиваться естественным образом, а задача экономистов, как и представителей других профессий, — выработка требований к архитектуре и программному обеспечению компьютера, как это было в 1960-е годы.

Теперь напомним о некоторых распространенных заблуждениях, связанных с цифровыми данными.

5.1. Об «информационной ёмкости»

Считается, что компьютерные носители способны хранить громадные объёмы ***информации*** при беспрецедентно малых размерах (сообщалось, что вся Библиотека

конгресса США помещается чуть ли не на одной современной микросхеме памяти). Заблуждение состоит в том, что на микросхему переносится не вся ***информация*** из Библиотеке конгресса, а только её ***оцифрованная часть*** (причем зависящая от метода оцифровки). То, что хранится в Библиотеке конгресса — реальность, и она — бесконечна, как ***полное десятичное разложение*** числа ***e***, или «неисчерпаема, как электрон и атом», а вот то, что оцифровано и помещается на микросхему памяти, аналогично ***рациональному приближению*** числа ***e*** и поддается численной оценке. Таким образом, подспудно навязываемое мнение о том, что оцифровкой можно сохранить архив, ошибочно. Уничтожение аналоговых данных необратимо.

5.2. Парадокс «времени жизни» цифровых данных

На первый взгляд может показаться, что цифровые данные вечны, поскольку отчуждены от физического носителя, подверженного старению, порче, износу и т. д. Однако 70-летний опыт работы с компьютерными данными свидетельствует об ином. Дело в том, что материальные носители данных и устройства чтения-записи непрерывно совершенствуются, изнашиваются и подвергаются замене. За семьдесят лет существования цифровых вычислительных машин сменилось несколько десятков типов носителей и устройств для них: магнитные ленты (разной ширины, скорости перемотки и плотности записи), гибкие диски (разного диаметра), флеш-память, жесткие диски и дисковые массивы, твердотельные накопители и т.д. Таким образом, время жизни цифровых данных определяется временем жизни устройств хранения, которое составляет всего несколько лет и имеет тенденцию к сокращению. До истечения срока жизни устройства хранящиеся на нём данные необходимо скопировать на новый носитель, ещё выпускающийся промышленностью, и если этого не сделать, они будут утрачены. Это означает непрерывный рост эксплуатационных расходов на хранение данных. Для сравнения: срок жизни бумажных книг — десятки и сотни лет. Таким образом, широко распространенное мнение о том, что если есть доступ к Интернету, то нет смысла держать домашнюю библиотеку, поскольку «Яндекс знает всё», — ошибочно.

5.3. Об электронных документах

Электронная документация или «электронный документооборот» — современная формулировка старой идеи «безбумажного документооборота», при котором расход бумаги должен был быть существенно снижен (чего, как известно, не произошло).

Электронный (или цифровой) документ может быть получен либо оцифровкой обычного документа, либо генерацией с помощью компьютерной программы.

В любом случае возникает цифровая ***копия*** документа.

Известно, что во многих жизненных ситуациях ***копия не имеет юридической силы оригинала***, и в этих ситуациях цифровые документы неприменимы.

Таким образом, вопреки бытующему мнению, электронный документ не может служить полноценным эквивалентом обычного.

Об этом завуалированно говорится в отчете [4] (послужившем, по-видимому, прототипом Программа перехода к цифровой экономике РФ): «раскрытие потенциала blockchain в значительной мере сдерживается техническими и концептуальными (policy) проблемами».

6. Заключение

Компьютер настолько глубоко проник в жизнь современного человека, что можно говорить о «цифровом мире», в котором все сведения хранятся в глобальной сети в цифровой форме. Этот мир является макетом, моделью, приближением реальности, «действительности», её искаженным образом, он скрыт от человека, недаром его называют «виртуальным». Для проникновения в этот мир человеку требуются устройства (например, экран) и компьютерные программы, т. е. *посредники*, вносящие дополнительные искажения. Замена аналоговых данных на цифровые означает попадание человека в зависимость от этих посредников, что служит причиной ограничения распространения «электронных удостоверений личности» или «электронных паспортов».

В цифровом мире отсутствует понятие «оригинала» или «подлинника», есть только копии (для изготовления которых нужны лишь компьютерные программы), а это влечёт юридические проблемы. Поскольку программирование становится всё более

массовым занятием, риск появления искаженных (фальсифицированных) копий растет, что подтверждает современная практика: одним из самых часто звучащих в новостях слов является fake, фальшивка.

В статье было показано, как на основании анализа различий между аналоговыми и цифровыми данными можно сделать практические выводы о целесообразности перехода на цифровые данные и пределах их применимости.

Публикация выполнена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН (выполнение фундаментальных научных исследований ГП 14) по теме № 0065-2018-0004 "Развитие методов математического моделирования распределенных систем и соответствующих методов вычисления." (№ АААА-А18-118041290146-7).

About the Digital Economy

S.G. Romanyuk

Abstract: In 2017, the Government of Russia adopted a program of transition to the digital economy, but many publications recognize that there is no generally accepted definition of the term "digital economy". The article describes the meaning of the word "digital economy" if we start from the meaning of the word "digital", in which it is used in computer science and computer engineering, and what practical conclusions can be drawn from this.

Keywords: computer, computer program, information, analog data, digital data, intangible, digitization, interpretation, artificial intelligence, Internet, electronic document.

Литература

1. Ф.Л.Бауэр, Г.Гооз. «Информатика». Москва, Мир, 1976. Перевод с немецкого под редакцией А.П.Ершова. (Технический материал этой книги в значительной мере устарел, поскольку компьютеры 1970-х заметно отличаются от современных, однако основы изложены блестяще. Как написал в предисловии к русскому изданию редактор перевода академик А.П.Ершов, книга «возрождает традиции добротной немецкой научной прозы»).

2. Bob Hughes. «From Useful Idiocy to Activism: a Marxist interpretation of computer development». Proceedings of the 4th decennial conference on Critical computing: between sense and sensibility. Pages 157-160. Aarhus, Denmark — August 20 - 24, 2005. ACM New York, NY, USA ©2005

3. А.А.Зализняк — «Слово о полку Игореве»: взгляд лингвиста. Рукописные памятники древней Руси, Москва, 2008.

4. OECD (2017), OECD Digital Economy Outlook 2017, OECD Publishing, Paris.
http://dx.doi.org/10.1787/9789264276284-en

Векторизация сильно разветвленного управления с помощью инструкций AVX-512

А.А. Рыбаков[1], С.С. Шумилин[2]

Межведомственный суперкомпьютерный центр РАН - филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия,

E-mails: [1] rybakov.aax@gmail.com, [2] shumilin@jscc.ru

Аннотация: Векторизация вычислений является важнейшей оптимизацией, с помощью которой может быть достигнуто кратное ускорение расчетных кодов. По мере развития современных микропроцессоров длина вектора в векторных операциях постоянно увеличивается. В современных линейках микропроцессоров Intel x86 (Xeon Phi KNL и Xeon Skylake) длина вектора уже достигает 512 бит. Однако зачастую программный код написан таким образом, что автоматическое применение векторизации невозможно. Причинами отказа от применения векторизации может стать наличие зависимостей между операциями в коде, вызовы функций или непостоянное количество итераций циклов в векторизуемых гнездах. В статье рассматривается актуальная проблема векторизации циклов, тела которых содержат сложное управление. Предложены два подхода к применению векторизации для циклов с условием и рассмотрено применение данных подходов на практической задаче с сильно разветвленным управлением.

Ключевые слова: оптимизация, векторизация цикла, предикатный режим исполнения, AVX-512, функции-интринсики, сильно разветвленное управление.

Введение

В настоящее время одновременно с развитием вычислительных систем и возрастанием их вычислительной мощности возникают вопросы об эффективности их применения. Использование векторных инструкций является наиболее низкоуровневым направлением создания высокопроизводительного кода. При выполнении векторизации программного кода необходимо учитывать факторы, которые могут негативно повлиять на итоговую производительность.

Самый простой вид циклов для векторизации – плоские циклы. Это циклы без побочных эффектов и зависимостей между итерациями, в которых эти итерации могут выполняться параллельно. Такие циклы хорошо поддаются векторизации, и при выполнении необходимых условий по выравниванию компилятор применяет векторизацию.

При добавлении в плоский цикл команд ветвления (операторы if, switch) компилятор icc также без труда справляется с векторизацией, используя масочные операции и операции слияния по условию. Сложнее дело обстоит, если внутри векторизуемого цикла появляется другой цикл с непостоянным количеством итераций. Такие конструкции не поддаются автоматической векторизации. Однако в случае независимости итераций вложенного цикла друг от друга охватывающий цикл все же можно векторизовать вручную, заменив все операции на векторные аналоги, использующие в качестве операнда вектор предикатов продолжения выполнения каждой из объединенных итерации исходного вложенного цикла [1].

В случае наличия в векторизуемом цикле маловероятных ветвей исполнения можно выполнить расщепление цикла по условию, получив два цикла, в первом из которых будет сохраняться только условие ухода на маловероятную ветку, во второй попадет ее реализация. После преобразования первый цикл, избавившись от маловероятного кода, может быть эффективно векторизован [2]. Другие эквивалентные преобразования циклов, в результате которых цикл может быть приведен к виду, пригодному для автоматической векторизации, можно найти в фундаментальных трудах [3, 4].

В данной статье рассматривается проблема векторизации кода для процессоров с поддержкой набора инструкций AVX-512 применительно к

циклам, тела которых содержат сложное управление, и приводится описание двух различных подходов к векторизации циклов, содержащих условие. Также описано объединение данных подходов в комбинированный метод векторизации цикла со сложным управлением и рассмотрено применение этого метода к практической задаче.

Особенности инструкций AVX-512

Множество инструкций AVX-512 представляет собой 512-битное расширение 256-битных инструкций AVX архитектуры Intel x86 [5], которое поддерживается в семействах микропроцессоров Intel Xeon Phi Knights Landing (KNL) [6] и Intel Xeon Skylake.

В микропроцессорах KNL поддерживаются следующие подмножества инструкций AVX-512: AVX-512F – основной набор векторных инструкций с поддержкой маскирования, AVX-512PF – инструкции предварительной подкачки данных из памяти, AVX-512ER – команды для вычисления экспоненты и обратных значений, AVX-512CD – инструкции для определения совпадения элементов вектора друг с другом, предназначенные для динамического определения конфликтов по доступу в память.

Для поддержания выборочного применения операций над упакованными данными к конкретным элементам аргументов-векторов используются маски. Большинство инструкций из набора AVX-512 могут использовать специальные регистры масок (k0 - k7).

Длина каждой маски составляет 64 бита, она может быть использована для маскирования векторов любых типов, содержащих от 8 до 64 элементов. Маски k0 - k7 используются в командах для осуществления условной операции над элементами упакованных данных (если бит маски выставлен в 1, то результат операции записывается в соответствующий элемент вектора назначения), для слияния элементов данных в регистр назначения, для выборочного чтения и записи в память элементов векторов и для аккумулирования результатов логических операций над элементами векторов.

Операции AVX-512 можно условно разделить на несколько групп согласно количеству и виду их аргументов и результата. Упакованные операции с одним операндом zmm (512-битный вектор) и одним результатом zmm (извлечение корня, округление, операции сдвигов и другие), упакованные операции с двумя операндами zmm и одним результатом zmm (поэлементные арифметические операции, операции сдвига на переменное количество разрядов и другие), упакованные операции с двумя операндами zmm и результатом маской (поэлементное сравнение двух векторов), операции конвертации, предназначенные для преобразования элементов вектора из одного формата в другой (cvt, pack), упакованные комбинированные операции (поэлементное вычисление значений вида ±a·b±c), операции перестановок, операции пересылок (в частности операции пересылки элементов данных расположенных с произвольными смещениями от заданного базового адреса в памяти, а также операции пересылки с дублированием элементов), операции предварительной подкачки данных, и другие операции с более сложной логикой, среди которых можно отметить определение класса вещественного числа, реализацию логических функций от трех аргументов, операции определения конфликтов и другие.

Для упрощения векторизации исходного кода для компилятора icc разработаны специальные функции-интринсики [7], определенные в заголовочном файле immintrin.h. Они покрывают не все инструкции AVX-512, однако избавляют от необходимости вручную писать ассемблерный код и позволяют использовать встроенные типы данных для 512-битных векторов (__m512, __m512i, __m512d). Некоторые интринсики соответствуют не отдельной команде, а целой последовательности, например для операции сложения всех элементов вектора (группа функций-интринсиков reduce). Из множества интринсиков можно выделить следующие группы функций, схожие по структуре. Функции swizzle, shuffle и

permute осуществляют перестановку элементов вектора и раскрываются в последовательность операций, в которой присутствует операция shuf и пересылка по маске. Для большего числа операций AVX-512 реализованы соответствующие инстринсики, раскрывающиеся в одну конкретную операцию. Некоторые инстринсики, особенно предназначенные для выполнения упакованных трансцендентных операций, раскрываются просто в вызов библиотечной функции (например _mm512_log_ps, _mm512_hypot_ps, а также тригонометрические функции).

Исследование эффективности векторизации цикла с одним условием

Рассмотрим простой цикл for с счетчиком, принимающим значения от 0 до n - 1. Тело цикла состоит из одного условного оператора if и двух линейных участков block 1 и block 2, которые выполняются в зависимости от данного условия. Будем считать, что цикл не содержит межитерационных зависимостей, все итерации цикла могут выполняться независимо друг от друга. Как вычисление условия, так и выполнение всех операций в блоках block 1 и block 2 зависят только от значения i на данной итерации цикла, возможны обращения в память только вида a[i].

```
for (i = 0; i < n; i++)
{
    if (<cond(i)>)
    {
        <block 1> // t1, p1
    }
    else
    {
        <block 2> // t2, p2
    }
}
```

Пусть длина линейного участка block 1 равна t_1, а вероятность его выполнения – p_1. Аналогично длина линейного участка block 2 равняется t_2, а вероятность его выполнения – p_2. Единицы измерения длины участка условны, будем понимать под ними количество операций в результирующем коде без учета задержек между ними. В нашем случае длины участком обозначены через t_1 и t_2, так как длина линейного участка пропорциональна времени его исполнения. В дальнейшем условимся считать длину участка и время его исполнения тождественными понятиями. Время исполнения одной итерации цикла совпадает с математическим ожиданием длины исполненного линейного участка, тогда общее время выполнения цикла равно

$$T^{orig} = \left(\delta + p_1 t_1 + p_2 t_2\right) n,$$

где через δ обозначено время вычисления условия, и им, вообще говоря можно, пренебречь.

Рассмотрим возможности по векторизации данного цикла. Одной из важнейших особенностей некоторых микропроцессорных архитектур является наличие предикатного (условного) режима исполнения отдельных операций. Особенно это характерно для архитектур с явно выраженной параллельностью [8]. Наличие предикатного режима исполнения операций позволяет избежать создания избыточных операций передачи управления путем планирования в одном линейном участке команд под различными предикатами. Предикатных режим исполнения поддержан к таких архитектурах, как ARM [9] или «Эльбрус» [10]. В архитектуре x86 предикатный режим поддержан частично, в ней присутствуют специальные команды условной пересылки (CMOVcc, conditional move). Так как в языке программирования C отсутствуют явные синтаксические конструкции для применения предикатного режима исполнения, то необходима поддержка со стороны компилятора, способного создавать предикатный код. Примерами таких оптимизаций могут послужить преобразование if nodes conversion для архитектуры «Эльбрус» [11] или оптимизация условных пересылок для архитектур с поддержкой таких операций. В предикатной форме исходный цикл может быть переписан в следующем виде:

```
for (i = 0; i < n; i++)
{
    p = <cond(i)>;
```

```
    <block 1> // p
    <block 2> // ~p
}
```

В данном случае под обозначением ~p понимается предикат, обратный предикату p. В результате данного преобразования тело цикла содержит один линейный участок, и общее время выполнения цикла равно

$$T_1^{pred} = (\delta + t_1 + t_2)n.$$

Набор инструкций AVX-512 обладает важным достоинством, заключающимся в наличии масочных аргументов в подавляющем большинстве векторных операций. Использование масок позволяет производить вычисления выборочно, над отдельными элементами векторов. При использовании векторизации с длиной вектора v суммарное время исполнения векторизованного кода будет равно

$$T_1^{vect} = (\delta + t_1 + t_2)nv^{-1},$$

при этом v равняется 8 в случае использования вещественных вычислений с типом double, v равняется 16 при использовании вещественных вычислений с типом float (именно данный случай мы будем в дальнейшем анализировать).

Конечно при оценке эффективности векторизации необходимо учитывать показатели латентности отдельных операций (у векторных инструкций они выше, чем у скалярных), это имеем важное значение при наличии длинных критических путей исполнения в программном коде.

Также важно учитывать пропускную способность для этих операций (количество операций данного вида, которые могут быть исполнены за один такт).

Некоторые операции AVX-512 имеют крайне низкую пропускную способность (например, операции деления, а также инструкции gather/scatter), и их присутствие в коде может послужить причиной возникновения узких мест и полностью нивелировать всю выгоду от оптимизации.

В данной статье мы не будем учитывать эти факторы, ограничившись простым числом операций в качестве времени выполнения кода (в том случае, кода код состоит из простых арифметических операций и не содержит длинных критических путей исполнения, это предположение можно считать верным).

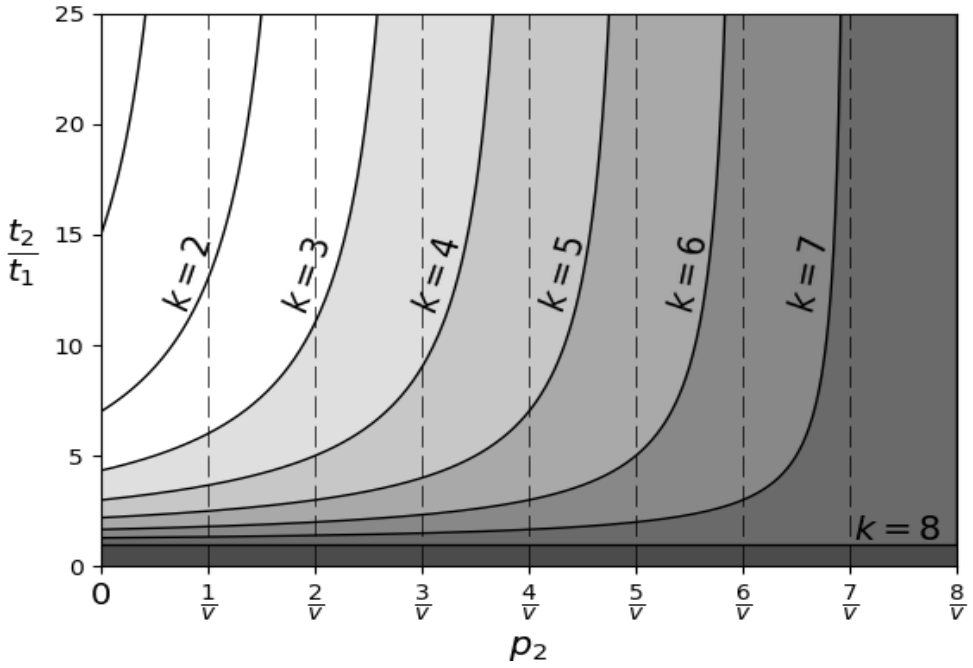


Рис. 1. Оценка эффективности векторизации при слиянии ветвей исполнения в теле цикла.

Для анализа эффекта от применения векторизации рассмотрим условие, при котором в наших предположениях эффективность применения векторизации не превышает k раз, то есть векторизованный код быстрее оригинального не более, чем в k раз (величину δ в данном случае опустим):

$$T^{orig} \le kT_1^{vect}$$

$$p_1 t_1 + p_2 t_2 \le \frac{(t_1 + t_2)k}{v}$$

Введем обозначение q=k/v, тогда последнее неравенство может быть переписано в виде

$$p_1 t_1 + p_2 t_2 \le (t_1 + t_2) q,$$

что после преобразования может быть переписано в виде

$$t_1(p_1 - q) + t_2(p_2 - q) \le 0$$

Для того, чтобы это условие выполнялось, необходимо, чтобы вероятность хотя бы одного из линейных участков block 1 или block 2 была достаточно мала. Пусть маловероятной веткой является block 2, тогда имеем $p_2 \le q$ (в данном случае q выступает в роли верхней границы маловероятной ветки, при которой эффективность векторизованного кода мала). Вычислим, во сколько раз длина маловероятного линейного участка должна превышать длину основной ветки, чтобы положительный эффект от применения векторизации пропал.

$$d_1(p_2) = \frac{t_2}{t_1} \ge \frac{p_1 - q}{q - p_2} = \frac{1 - p_2 - q}{q - p_2} = \frac{(1-q) - p_2}{q - p_2}$$

Будем рассматривать значения k от 1 до 16 (значение v у нас равняется 16) при $p_2 \le 0.5$.

Проанализируем графики, которые приведены на рис. 1. Для k < 8 график зависимости $d_1(p_2)$ представляет собой возрастающую ветвь гиперболы. При увеличении значения p_2 величина d_1 значительно возрастает, то есть чем выше вероятность исполнения маловероятного участка, тем длиннее он должен быть для того, чтобы применение векторизации оказалось неэффективно, а начиная со значения $p_2 = q$ векторизованный вариант более эффективен при любом соотношении длин линейных участков block 1 и block 2. При k = 8 график представляет собой константу, равную единице. Это означает, что при условии $t_2 > t_1$ невозможно получить теоретическое ускорение векторизованного кода в 8 раз. Для оставшихся случаев k > 8 график также представляет собой ветку гиперболы, однако загибающуюся вниз и уходящую в область отрицательных значений. Это означает, что при использовании векторизации предикатного кода, полученного вследствие слияния отдельных ветвей исполнения, сложно или невозможно достигнуть ускорения в k раз.

Теперь рассмотрим другой подход к векторизации цикла, основанный на вынос маловероятного условия в отдельный цикл. Снова будем считать, что маловероятной веткой является линейный участок block 2. Модифицируем цикл таким образом, чтобы полностью избавиться от выполнения маловероятного кода block 2 в цикле. Вместо этого заведем временный массив с булевыми переменными, в котором будем сохранять признаки необходимости исполнения маловероятного кода. Обработку же маловероятного кода вынесем в отдельный цикл.

```
for (i = 0; i < n; i++)
{
    tmp[i] = <cond(i)>;

    if (tmp[i])
    {
        <block 1> // t1, p1
    }
}

for (i = 0; i < n; i++)
{
    if (!tmp[i])
    {
        <block 2> // t2, p2
    }
}
```

Такой подход оправдывается тем, что после преобразования первый цикл содержит только вероятную ветку и небольшой объем вспомогательных вычислений, а второй цикл содержит только маловероятные вычисления, которыми можно пренебречь. Общее время работы данного модифицированного кода равно

$$T_2^{\text{mod}} = (\delta + \alpha + p_1 t_1) n + (\beta + p_2 t_2) n.$$

Запишем первый цикл в предикатной форме. При этом второй цикл не требуется векторизовать, поэтому переводить его в предикатную форму не нужно, поэтому приведем представление только первого цикла, который теперь выглядит следующим образом:

```
for (i = 0; i < n; i++)
{
    p = <cond(i)>;
    tmp[i] = p;
    <block 1> // p
}
```

Общее время работы данного участка кода в предикатной форме равно

$$T_2^{pred} = (\delta + \alpha + t_1) n + (\beta + p_2 t_2) n,$$

однако теперь первый цикл поддается векторизации, что в итоге дает время исполнения векторизованного кода, равное

$$T_2^{vect} = (\delta + \alpha + t_1) n v^{-1} + (\beta + p_2 t_2) n.$$

Аналогично предыдущему подходу, проанализируем ситуацию, когда векторизованный код быстрее оригинального не более, чем в k раз.

$$T^{orig} \le k T_2^{vect}$$

$$p_1 t_1 + p_2 t_2 \le k \left(\frac{t_1}{v} + p_2 t_2 \right)$$

$$t_1 (p_1 - q) - t_2 p_2 (k-1) \le 0$$

Проанализируем получившееся неравенство. В отличие от предыдущего подхода, в данном случае чем ниже вероятность маловероятной ветви, тем выгоднее применение векторизации. Получим аналогичную оценку на отношение длины маловероятной ветки к длине вероятной ветки.

$$d_2(p_2) = \frac{t_2}{t_1} \ge \frac{p_1 - q}{p_2 (k-1)} = \frac{(1-q) - p_2}{p_2 (k-1)}$$

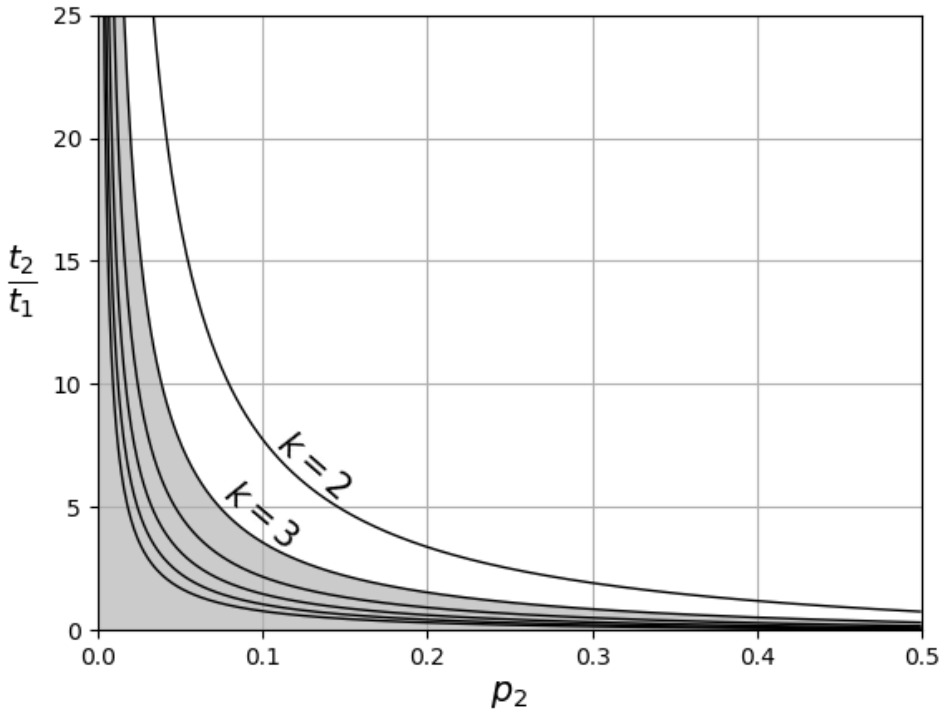


Рис. 2. Оценка эффективности векторизации при выносе маловероятной ветви исполнения из цикла.

Из графиков на рис. 2 видно, что применение выноса маловероятной ветви исполнения из цикла выгодно при очень низких значениях p_2.

Таким образом, описаны два подхода к векторизации циклов, состоящих из одного простого условия с двумя линейными участками: слияние двух ветвей исполнения и вынос маловероятной ветви из цикла. При k < 8 эффективность векторизации со слиянием двух ветвей повышается при росте вероятности маловероятной ветви, а эффективность векторизации с выносом маловероятной ветви из цикла наоборот понижается. То есть, в зависимости от вероятности маловероятной ветви исполнения нужно выбирать конкретный подход к векторизации. Значение вероятности p_2 для выбора одного из двух подходов определяется из соотношения

$$\frac{(1-q)-p_2}{q-p_2} = \frac{(1-q)-p_2}{p_2(k-1)},$$

откуда находим $p_2 = v^{-1}$. На рис. 3 представлены комбинированные графики оценки эффективности векторизации цикла с одним условием с учетом выбора конкретного подхода к векторизации.

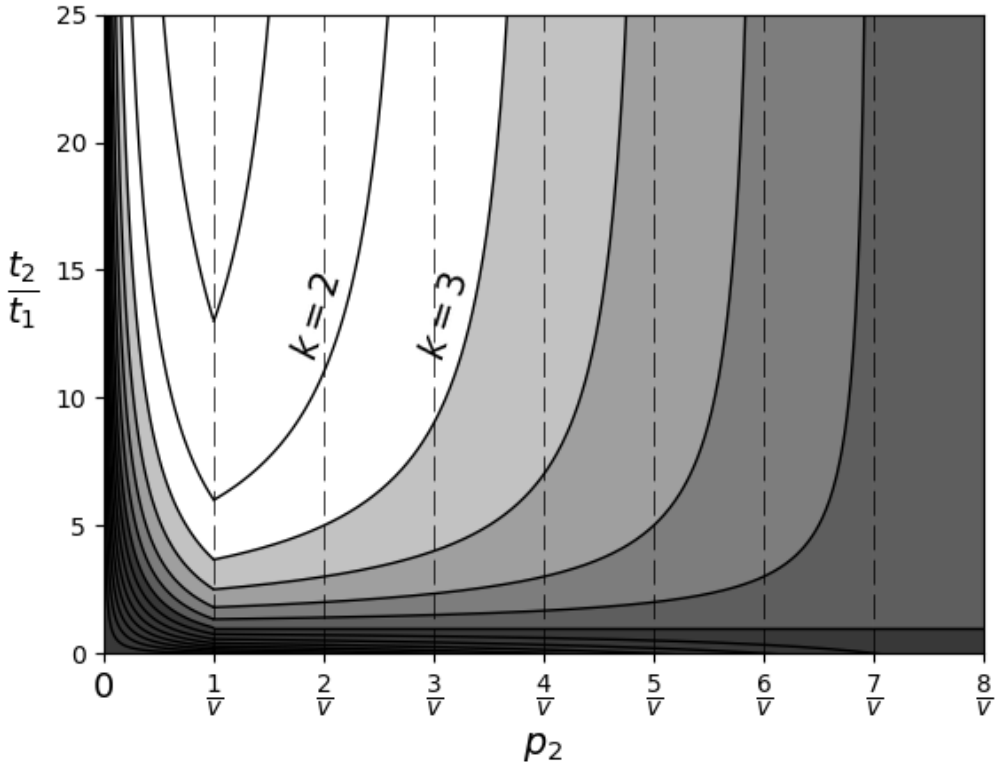


Рис. 3. Оценка эффективности векторизации цикла с одним условием с учетом выбора подхода к векторизации.

Практическое применение

Рассмотрим применение описанных выше методов векторизации кода со сложным управлением. Подходящий контекст для оптимизации будем искать в реализации точного решения задачи о распаде произвольного разрыва, которая является одной из классических фундаментальных задач математической физики [12].

Ввиду широкой известности программной реализации различных вариаций решения данной задачи не будем детально описывать ее, полную информацию касательно ее решения можно найти в [13], реализация на языке программирования FORTRAN находится в открытом доступе в сети Интернет [14].

Основная функция решения задачи о распаде произвольного разрыва имеет следующую сигнатуру:

```
void riemann
  (float dl, float ul, float pl,
   float dr, float ur, float pr,
   float &d, float &u, float &p)
```

Функция принимает на вход значение газодинамических параметров слева от разрыва (dl, ul, pl) и справа от него (dr, ur, pr) и определяет значения параметров в нулевой момент времени на самом разрыве (d, u, p). В процессе математического моделирования процессов газовой динамики данная задача решается множество раз на каждой грани каждой ячейки расчетной сетки, поэтому эффективная реализация данного кода имеет важное значение. Решение каждой отдельной задачи о распаде разрыва является независимым, решение всех таких задач может выполняться параллельно. На каждой временной итерации проведения расчетов вызовы функции riemann выполняются над отдельными элементами входных данных, объединенных в массивы. Условно это можно изобразить в следующем виде:

```
void riemann_multy
    (float *dl, float *ul,
     float *pl, float *dr,
     float *ur, float *pr,
     float *d, float *u,
     float *p)
{
    ..........
    for (i = 0; i < count; i++)
    {
      riemann(dl[i], ul[i], pl[i],
              dr[i], ur[i], pr[i],
              d_, u_, p_);
      d[i] = d_;
      u[i] = u_;
      p[i] = p_;
    }
    ..........
}
```

Для повышения эффективности целесообразно произвести inline-подстановку тела функции riemann в место вызова. Составной частью функции riemann является функция sample, которая на основании данных о газодинамических параметрах слева и справа от разрыва, а также вычисленных скорости и давлении газа в центральном регионе (star region), получает характер и газодинамические характеристики на разрыве в нулевой момент времени. Функция sample содержит сильно разветвленное управление, что является подходящим контекстом для применения описанных в данной статье подходов. Для повышения эффективности применения оптимизации произведем расщепление основного цикла функции riemann_multy таким образом, чтобы один из результирующих циклов содержал только тело функции sample. Данный цикл и представляет интересующий нас контекст для применения оптимизации, в дальнейшем будем рассматривать только этот участок кода.

```
for (i = 0; i < count; i++)
{
    ..........
    // should be inlined...
    sample(dl[i],ul[i],pl[i],cl[i],
           dr[i],ur[i],pr[i],cr[i],
           pm[i], um[i],
           d_, u_, p_);
    d[i] = d_;
    u[i] = u_;
    p[i] = p_;
    ..........
}
```

С учетом того, что тело рассматриваемого цикла содержит около полутора сотен строк кода, мы не будет приводить текст данного цикла, а обозначим только его граф потока управления. Тело рассматриваемого цикла содержит 19 линейных участков и 9 условий (максимальная вложенность условий равна четырем). При этом все итерации данного цикла независимы и могут быть выполнены параллельно. Также отметим, что на итерации с номером i осуществляются обращения в память только в элементам массивов вида a[i], из операций используются арифметические операции (сложение, вычитание, умножение, деление), сравнение, возведение в степень, извлечение квадратного корня, вычисление обратного значения. Все эти операции имеют векторные аналоги в множестве команд AVX-512, в тому же все эти векторные операции поддерживают предикатный режим исполнения.

Для определения лучшей стратегии векторизации для тела рассматриваемого цикла был собран профиль исполнения полученный на стандартных тестовых задачах: задача Сода, задача Лакса, задача о слабой ударной волне, задача Эйнфельдта, задача Вудвода-Колелла, задача Шу-Ошера [15, 16] и другие. На рис. 4 представлен граф потока управления (control flow graph, CFG) для тела рассматриваемого цикла. Узлами данного графа являются линейные участки, а ребрами – переходы между ними.

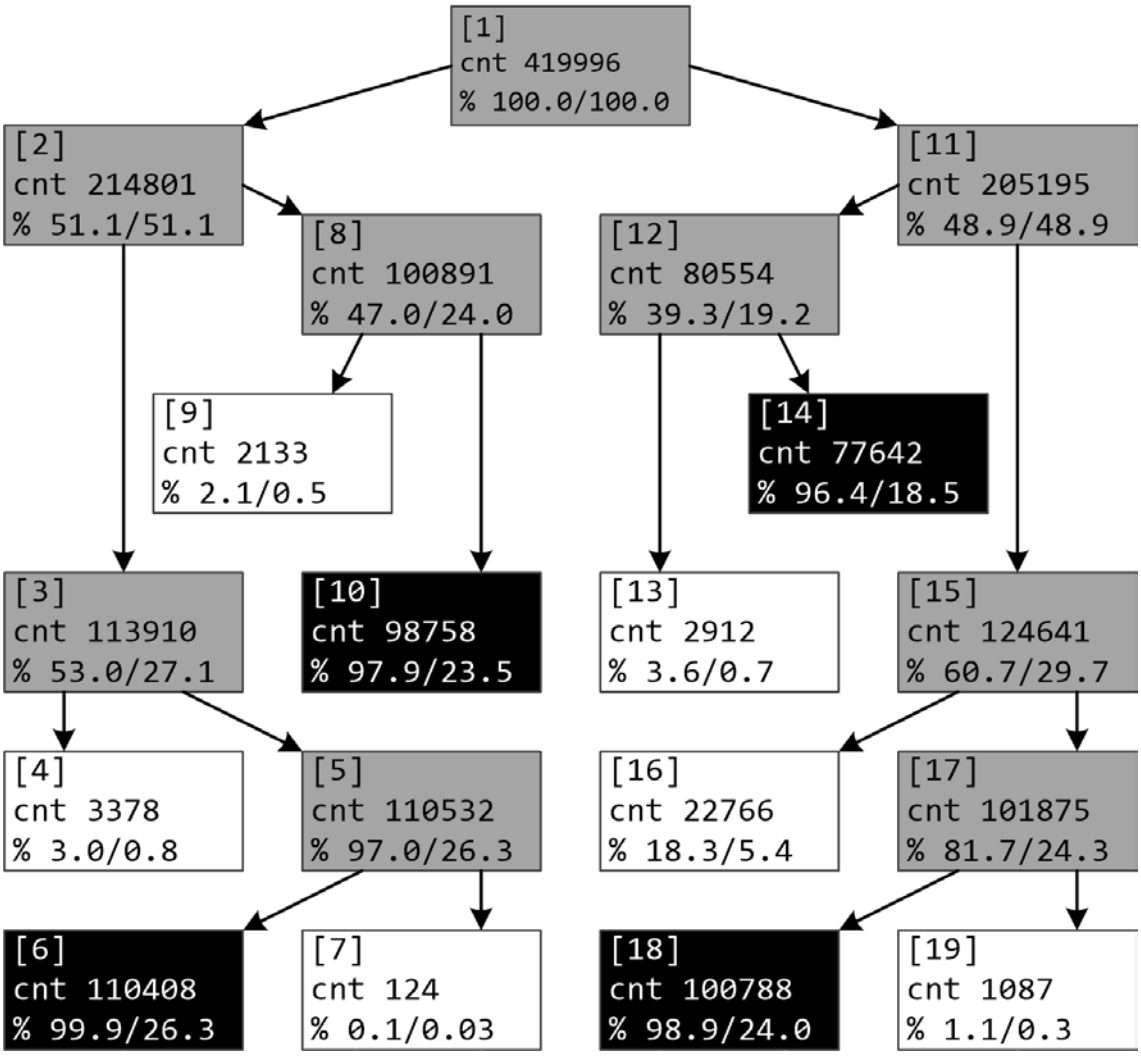


Рис. 4. Граф потока управления тела векторизуемого цикла со сложным управлением.

Для каждого узла CFG с рис. 4 приведены следующие его характеристики: номер узла (в квадратных скобках), счетчик или количество исполнений линейного участка (после слова cnt), локальная вероятность исполнения (отношение счетчика линейного участка к счетчику его непосредственного доминатора, умноженное на 100%), глобальная вероятность исполнения (отношение счетчика линейного участка к счетчику головы цикла, умноженное на 100%). Все узлы CFG окрашены в три цвета. Белым цветом закрашены маловероятные участки кода, черным – наиболее вероятные листовые линейные участки, серым – все остальные линейные участки.

Код данного цикла был подготовлен для векторизации – была декларирована независимость всех массивов, участвующих в вычислениях (все массивы были переданы в функцию с модификатором __restrict__), было декларировано выравнивание расположения массивов в памяти на 64 байта. После подготовки программного кода

к векторизации от компилятора icc был получен отказ по эффективности.

Таким образом, компилятор смог построить векторизованный код, однако его теоретическая эффективность оказалась слишком низкой ввиду слишком большого количества ветвей исполнения.

Для поддержки ручной векторизации были предприняты следующие действия.

Прежде всего было замечено, что 4 линейных участка содержали определение газодинамических параметров d, u и p, которое можно было изменить на инициализацию с помощью выноса операций присваивания вверх по коду цикла. Таким образом, линейные участки с номерами 4, 9, 13, 16 были удалены, что привело к ускорению кода примерно на 10% (см. рис. 5).

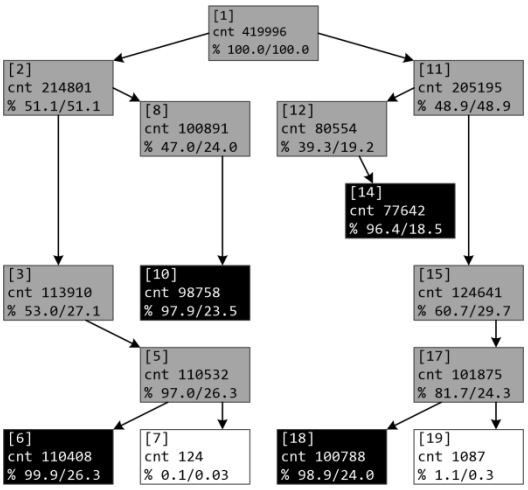


Рис. 5. Удаление линейных участков из тела цикла путем выноса операций присваивания вверх.

Далее было отмечено, что функция sample обрабатывает одинаковым образом правый и левый профиль распада разрыва с незначительными изменениями.

С помощью простой замены переменных удалось выполнить слияние кода для двух поддеревьев узла с номером 1 (поддеревья, опирающиеся на узлы с номерами 2 и 11).

Данное преобразование вдвое уменьшило объем расчетного кода и ожидаемо сократило время исполнения еще на 45% (см. рис. 6).

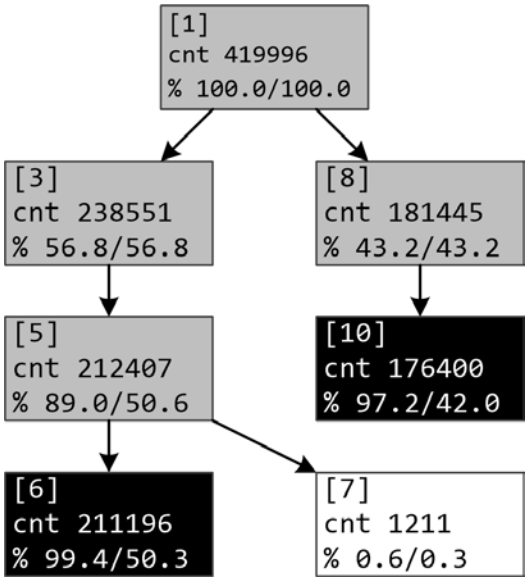


Рис. 6. Выполнение слияния симметричных линейных участков в теле цикла.

Однако даже после сокращения тела цикла до 7 линейных участков, связанных 4 условиями переходов автоматическая векторизация оказалась неэффективной, и компилятор выдал отказ в проведении данной оптимизации.

Для достижения положительного эффекта от применения векторизации следует воспользоваться описанными в предыдущем разделе подходами. Во-первых, обратим внимание на линейный участок с номером 7. Это линейный участок с крайне низким счетчиком, однако обладающий большой длиной, так как он содержит тяжелые операции (вещественное деление и возведение в степень). К данному линейному участку целесообразно применение выноса маловероятной ветки из цикла.

Так как тело цикла само по себе достаточно объемное, то мы будем применять не вынос маловероятной ветки, а локализацию этой ветки. Локализация маловероятной ветки выполняется следующим образом. Если в оригинальном варианте выполнение маловероятной ветки обслуживалось предикатом p, то в векторизованном варианте соответствующий векторный предикат в большинстве случаев нулевой, а значит перед выполнением векторизованного кода сначала следует проверить векторный предикат на наличие хотя бы одного ненулевого элемента. Такой подход выгоден только для маловероятных

веток, так как одной проверкой отсекается сразу 16 итераций тяжелого кода, который и не должен быть выполнен. К тому же локализация маловероятной ветки избавляет от необходимости содержания дополнительного временного массива булевых переменных.

После локализации маловероятной ветки остаются две равнозначные ветки с практически одинаковой вероятностью, к которым следует применить слияние в предикатный код, который поддается векторизации, что и представлено на рис. 7.

На рис. 7 представлен результирующий код для рассматриваемого векторизуемого цикла. Дадим некоторые краткие пояснения. Сначала в строках 4-5, 8-11 выполняется загрузка требуемых векторов с помощью LD (_mm512_load_ps).

```
1  for (int j = 0; j < TESTS_COUNT; j += 16)
2  {
3      // Load.
4      v_pm = LD(&pm[j]);
5      v_um = LD(&um[j]);
6      // d/u/p/c/ums
7      cond_um = _mm512_cmp_ps_mask(v_um, z, _MM_CMPINT_LT);
8      d = _mm512_mask_blend_ps(cond_um, LD(&dl[j]), LD(&dr[j]));
9      u = _mm512_mask_blend_ps(cond_um, LD(&ul[j]), LD(&ur[j]));
10     p = _mm512_mask_blend_ps(cond_um, LD(&pl[j]), LD(&pr[j]));
11     c = _mm512_mask_blend_ps(cond_um, LD(&cl[j]), LD(&cr[j]));
12     ums = LD(&um[j]);
13     u = _mm512_mask_sub_ps(u, cond_um, z, u);
14     ums = _mm512_mask_sub_ps(ums, cond_um, z, ums);
15     // Calculate main values.
16     pms = DIV(v_pm, p);
17     sh = SUB(u, c);
18     st = _mm512_fnmadd_ps(POW(pms, g1), c, ums);
19     s = _mm512_fnmadd_ps(c, SQRT(_mm512_fmadd_ps(g2, pms, g1)), u);
20     // Conditions.
21     cond_pm = _mm512_cmp_ps_mask(v_pm, p, _MM_CMPINT_LE);
22     cond_sh = _mm512_mask_cmp_ps_mask(cond_pm, sh, z, _MM_CMPINT_LT);
23     cond_st = _mm512_mask_cmp_ps_mask(cond_sh, st, z, _MM_CMPINT_LT);
24     cond_s = _mm512_mask_cmp_ps_mask(~cond_pm, s, z, _MM_CMPINT_LT);
25     // Store.
26     d = _mm512_mask_mov_ps(d, cond_st, MUL(d, POW(pms, igama)));
27     d = _mm512_mask_mov_ps(d, cond_s, MUL(d, DIV(ADD(pms, g6), _mm512_fmadd_ps(pms, g6, v1))));
28     u = _mm512_mask_mov_ps(u, cond_st | cond_s, ums);
29     p = _mm512_mask_mov_ps(p, cond_st | cond_s, v_pm);
30     // Low prob - ignnore it.
31     cond_sh_st = cond_sh & ~cond_st;
32     if (cond_sh_st)
33     {
34         u = _mm512_mask_mov_ps(u, cond_sh_st, MUL(g5, _mm512_fmadd_ps(g7, u, c)));
35         uc = DIV(u, c);
36         d = _mm512_mask_mov_ps(d, cond_sh_st, MUL(d, POW(uc, g4)));
37         p = _mm512_mask_mov_ps(p, cond_sh_st, MUL(p, POW(uc, g3)));
38     }
39     // Final store.
40     u = _mm512_mask_sub_ps(u, cond_um, z, u);
41     ST(&od[j], d);
42     ST(&ou[j], u);
43     ST(&op[j], p);
44 }
```

Рис. 7. Итоговый вариант векторизованного тела цикла для функции sample.

В строках 8-13 выполняется изначальная инициализация результатов с помощью операций blend. Далее в строках 16-19 вычисляются значения, вынесенные вверх из вероятных листовых линейных участков. Строки 21-24 нужны для вычисления условий, по которым определяется листовой линейный участок, который должен исполниться. В строках 26-29 выполняется условная пересылка результатов функции для разных веток исполнения. Строки 32-38 содержат локализацию маловероятной ветки

исполнения. Заключительные строки 41-44 записывают результат в память.

Отметим отдельно строки 13, 14 и 40, обеспечивающие замену переменных, которая позволила получить 45% ускорение.

Заключение

В статье рассмотрены вопросы векторизации программного кода, содержащего сложное управление. Описаны подходы, основанные на слиянии равновероятных ветвей исполнения, а также на выносе из цикла и локализации маловероятной ветви исполнения. Рассмотрено применение описанных методов к части программного кода, входящего в состав решения задачи о распаде произвольного разрыва. Финальный векторизованный код, представленный на рис. 7 показал сокращение времени исполнения на 91% (более, чем в 10 раз) по сравнению с оригинальной версией при исполнении на микропроцессоре Intel Xeon Phi KNL. Если исключить из данного значения ускорение, полученное вследствие удаления линейных участков и замены переменных, то полученное ускорение, прямо связанное с применением векторизации равняется около 82% (более, чем в 5 раз).

Работа выполнена в МСЦ РАН в рамках государственного задания по теме «Разработка архитектур, системных решений и методов для создания вычислительных комплексов и распределенных сред мультипетафлопсного диапазона производительности, в том числе нетрадиционных архитектур микропроцессоров». Для расчетов при проведении исследований использовался суперкомпьютер МВС-10П, находящийся в МСЦ РАН.

Vectorization of highly branched control using AVX-512 instructions

A.A. Rybakov, S.S. Shumilin

Abstract: Vectorization of computations is one of the most important optimizations, with the help of which a multiple acceleration of the calculation codes can be achieved. With the development of modern microprocessors, the vector length in vector operations is constantly increasing. In modern families of Intel x86 microprocessors (Xeon Phi KNL and Xeon Skylake) the vector length already reaches 512 bits. However, often the program code is written in such a way that the automatic application of vectorization is impossible. The reasons for not using vectorization may be the presence of dependencies between operations in the code, function calls or a non-constant number of iterations of loops in vectorized sockets. The current problem of vectorization of loops whose bodies contain complex control is discussed in the article. Two approaches to the application of vectorization for loops with a condition are proposed and the application of these approaches to a practical task containing highly branched control is discussed.

Keywords: optimization, loop vectorization, predicated execution, AVX-512, intrinsic functions, highly branched control.

Литература

1. А.А. Рыбаков, П.Н. Телегин, Б.М. Шабанов. Проблемы векторизации гнезд циклов с использованием инструкций AVX-512. Электронный научный журнал: Программные продукты, системы и алгоритмы. 2018. № 3. С. 1-11.
2. А.А. Рыбаков. Оптимизация задачи об определении конфликтов с опасными зонами движения летательных аппаратов для выполнения на Intel Xeon Phi. Программные продукты и системы. 2017. Т. 30. № 3. С. 524-528.
3. S. Muchnick. Advanced compiler design and implementation. Morgan Kaufmann. 1997.
4. R. Allen, K. Kennedy. Optimizing compilers for modern architectures. Morgan Kaufmann. 2001.
5. Intel 64 and IA-32 architectures software developer's manual. Combined volumes: 1, 2A, 2B, 2C, 2D, 3A, 3B, 3C, 3D and 4. Intel Corporation. 2017.
6. J. Jeffers, J. Reinders, A. Sodani. Intel Xeon Phi processor high performance programming. Knights Landing edition. Morgan Kaufmann. 2016.
7. Intel C++ compiler 16.0 user and reference guide. Intel Corporation. 2015.
8. В.Ю. Волконский, С.К. Окунев. Предикатное представление как основа оптимизации программы для архитектур с явно выраженной параллельностью. Информационные технологии. 2003. № 4. С. 36-45.
9. A. Sloss, D. Symes, C. Wright. ARM System developer's guide. Designing and optimizing system software. Morgan Kaufmann. 2004.
10. А.К. Ким, В.И. Перекатов, С.Г. Ермаков. Микропроцессоры и вычислительные комплексы семейства «Эльбрус». Питер. 2013.
11. А.А. Рыбаков, М.В. Маслов. Быстрый регионный компилятор системы двоичной трансляции для архитектуры «Эльбрус». V Международная научно-практическая конференция «Современные информационные технологии и ИТ-образование», сборник избранных трудов, под редакцией проф. В.А. Сухомлина. Москва. 2010. С. 436-443.
12. А.Г. Куликовский, Н.В. Погорелов, А.Ю. Семенов. Математические вопросы численного решения гиперболических систем уравнений. М.: ФИЗМАТЛИТ. 2001.
13. E.F. Toro. Riemann solvers and numerical methods for fluid dynamics. A practical introduction. 2nd edition. Springer. 1999.
14. https://github.com/dasikasunder/NUMERICA. Дата обращения 14.08.2018.
15. П.В. Булат, К.Н. Волков. Одномерные задачи газовой динамики и их решение при помощи разностных схем высокой разрешающей способности. Научно-технический вестник информационных технологий, механики и оптики. 2015. Том 15. № 4. С. 731-740.
16. S. Brill. Verification of the wavelet adaptive multiresolution representation method to a prescribed error threshold. Department of Aerospace and Mechanical Engineering University of Notre Dame, AME 48491 Undergraduate research, June 4, 2014.

Об одном методе конфигурирования сети научного суперкомпьютерного центра

А.В.Баранов[1], А.П.Овсянников[2], В.Ф.Огарышев[3], В.И.Федоров[4]

[1,2,3]Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН,

[4]Московский физико-технический институт (государственный университет), Москва, Россия,

E-mails: [1]antbar@mail.ru, [2]apo@jscc.ru, [3]ogaryshev@jscc.ru, [4]petrenko091@mail.ru

Аннотация: В статье рассматривается подход к автоматизации решения одной из важных повседневных задач системного администратора суперкомпьютерного научного центра – реконфигурации оборудования сегментированной локальной вычислительной сети. Авторы предлагают использовать специальный язык разметки сети Network Markup Language (NML), на котором возможно составить описание сети с необходимой полнотой. С помощью разработанного авторами программного средства администратор может преобразовать NML-описание сети в набор конфигурационных файлов сетевых устройств. За счёт модульной архитектуры спектр поддерживаемых программным средством сетевых устройств может быть расширен путём разработки и добавления в систему новых модулей. В статье также приводится методика проверки корректности сгенерированной конфигурации, основанная на применении среды имитационного моделирования GNS3, и даётся предварительная оценка эффективности предложенного авторами подхода.

Ключевые слова: настройка сети, Network Markup Language, NML, VLAN, суперкомпьютерный центр, GNS3

Введение

На сегодняшний день в состав современных суперкомпьютерных центров может входить несколько высокопроизводительных вычислительных установок, сетей и систем хранения данных, рабочих станций пользователей и сотрудников. Объединяет все перечисленные компоненты развитая сетевая инфраструктура [1], которая, как правило, разделяется на отдельные сегменты (участки), соединяющиеся между собой маршрутизаторами и/или выполняющими функцию маршрутизации коммутаторами 3-го уровня. Такой подход к организации сети называется сегментацией.

Особенность сегментирования сети состоит в том, что сетевой трафик между сегментами в общем случае запрещён. Сегментация служит для повышения безопасности сети, сокращения широковещательного трафика от активных устройств, расширения возможностей управления всей сетью в целом.

С точки зрения безопасности, сегментирование сети позволяет скрыть её внутреннее устройство от внешнего наблюдателя. В правильно сегментированной сети локализуются потенциальные атаки, при которых компрометируется одно или несколько устройств сети. Примером является разделение веб-серверов, серверов баз данных и пользовательских машин, при котором каждый будет находиться в своем сегменте.

Для сегментации сети на канальном уровне используется технология виртуальных локальных сетей VLAN (Virtual Local Area Network). VLAN представляет собой группу устройств в сети, которые взаимодействуют так, как если бы они были подключены к одному домену широковещательной рассылки, независимо от их физического местонахождения. VLAN имеет те же свойства, что и физическая локальная сеть, но позволяет конечным узлам, например, рабочим станциям пользователей, группироваться вместе, даже если они не находятся в одном физическом сегменте.

VLAN обладает следующими преимуществами:

– является эффективным способом группировки сетевых пользователей в виртуальные рабочие группы, несмотря на их физическое размещение в сети;

– обеспечивает возможность контроля широковещательных сообщений;

– позволяет повысить безопасность сети, определив с помощью фильтров, настроенных на коммутаторе или маршрутизаторе, политику взаимодействия пользователей из разных виртуальных сетей.

Помимо технологии VLAN, в практике организации вычислительных сетей часто используются технологии LAG (Link Aggre-

gation) объединения нескольких физичеких каналов в одну линию связи и ее развитие - MLAG (Multi-Switch Link Aggregation – технология агрегирования каналов), которая даёт возможность независимым коммутаторам выглядеть одним логическим коммутатором для других устройств без объединения в стек.

Сегментирование сети и агрегирование каналов всегда выполняются в соответствии с текущими требованиями к вычислительной сети, которые, как показывает практика, постоянно изменяются в зависимости от производственной ситуации.

Это вызывает необходимость перестраивать и/или перенастраивать существующую инфраструктуру сети с большим количеством устройств и конечных узлов. Подобное конфигурирование сети, включающее настройку сетевых устройств и связей между ними, является одной из повседневных задач системного администратора. Помимо собственно настройки устройств, администратор должен обеспечить аккуратное документирование изменений сетевой конфигурации и следить за соответствием документации реальным настройкам сети.

Условно работу системного администратора по конфигурированию и настройке вычислительной сети можно разделить на три этапа.

На первом каким-либо образом определяются изменения, которые необходимо произвести в существующей сети.

На втором этапе определяется последовательность команд для перенастройки сетевых устройств. На третьем эти последовательности команд выполняются на сетевых устройствах.

Ручное выполнение указанных этапов сопряжено с большим числом однотипных операций для каждого из устройств, что, с одной стороны, делает задачу конфигурирования весьма трудоёмкой, с другой стороны, становятся неизбежными ошибки. Одно из решений заключается в формировании описания структуры сети на некотором формализованном языке и разработке процедуры автоматизации получения на базе сформированного описания последовательностей команд настройки или конфигурационных файлов сетевого оборудования.

Цель настоящей статьи – разработка метода программной автоматизации работы системного администратора по реконфигурации вычислительной сети, основанного на использовании формализованного описания сети.

Системы управления инфраструктурой центров обработки данных (ЦОД)

Для решения задачи реконфигурации сетей могут быть применены системы комплексного управления инфраструктурой ЦОД – Data Center Infrastructure Management Systems (DCIM). DCIM предназначены для удалённого мониторинга использования ресурсов ЦОД, состояния окружающей среды, перемещения активов (как физических: серверов, коммутаторов, так и информационных: виртуальных машин, томов данных, сетей) и других параметров, а также для регулирования отдельных компонентов инфраструктуры ЦОД: от вентиляторов внутри серверов до системы безопасности [2, 3].

Применение DCIM позволяет автоматизировать, как минимум, функции сбора данных, моделирования инфраструктуры и генерации аналитической отчётности. Сбор информации в DCIM осуществляется в процессе постоянного мониторинга таких характеристик устройств из состава оборудования ЦОД, как потребление энергии, температура, влажность и воздушные потоки. На основе этих данных создаётся ключевой элемент DCIM-решения – представляемая в удобном графическом формате цифровая модель инфраструктуры, которая обновляется по мере изменения инфраструктурных элементов. Интерпретацией собираемых в DCIM данных занимается аналитическая подсистема, которая позволяет, например, выявить проблемы в сети или определить причины неэффективности обмена данными в ней.

На сегодняшний день существует большое количество поставщиков DCIM-решений, наиболее крупными из них являются Nlyte Software, Emerson Network Power, Schneider Electric и Panduit.

Несмотря на то, что DCIM упрощают планирование мощностей и распределение ресурсов, помогают обеспечить максимально эффективное использование электроэнергии, оборудования и площади, отмечается [4], что эти решения сложны в использовании, обладают относительно низкими эффективностью и скоростью развёртывания. Кроме этого, следует учитывать немалую стоимость ПО DCIM. В основном поставщики формируют цену на основе числа «конечных устройств», с которыми программное обеспечение обменивается информацией, будь то стойки с сетевым оборудованием, серверы или другие устройства. При этом средняя

стоимость проектов от разных поставщиков варьируется от $1000 до $2000 за стойку [5].

Для достижения цели настоящей работы применение DCIM возможно, так как эти решения содержат функционал моделирования инфраструктуры сети, обновляющейся по мере изменения её элементов. Однако, в рамках настоящей работы не требуется наличие функций по сбору данных о состоянии сети и её составляющих и формированию аналитической сводки по этим данным. При этом из комплекта ПО DCIM невозможно выделить подсистему моделирования инфраструктуры, DCIM-решения возможно приобрести лишь целиком. Применение DCIM эффективно и выгодно для больших коммерческих ЦОД, обладающих обширным парком оборудования от различных производителей. Научные суперкомпьютерные центры, подобные МСЦ РАН, как правило, не обладают такими размерами, не решают коммерческих задач и имеют сравнительно ограниченный бюджет. DCIM-решения для таких центров являются достаточно дорогим инструментом, при этом обладающим избыточным функционалом.

Язык разметки сети NML и его расширение

Альтернативой подсистеме моделирования сети DCIM-решения может стать наличие некоего формального описания сети, которое в любой момент может быть модифицировано и дополнено. Для описания сети можно использовать язык разметки сети NML (Network Markup Language). Стандарт языка был представлен в мае 2013 года сообществом пользователей Open Grid Forum [6].

XML-подобный язык разметки NML предназначен для создания функционального описания многоуровневых и многодоменных сетей. Примером многоуровневой сети может быть виртуализированная сеть. Описание многодоменной сети может включать агрегированные или абстрактные сетевые топологии. NML позволяет описывать не только статическую сетевую топологию, но и её потенциальные возможности (службы) и их конфигурацию. NML нацелен на описание логических сетевых топологий, в частности VLAN. Недостатком основной схемы является отсутствие объектов или свойств для работы со сложными таблицами маршрутизации. NML содержит механизмы расширения для упрощенного связывания со стандартами резервирования и мониторинга сети.

Основная схема NML описывает информационную модель для компьютерных сетей. Эта схема умышленно сделана обобщённой, с возможностью расширения для описания информации, специфичной для каждого слоя. Схема состоит из объектов, атрибутов, отношений и параметров. Объекты описывают устройства сети, отношения – связи между объектами, атрибуты – свойства объектов. Параметры, подобно атрибутам, являются свойствами объектов. Атрибуты и параметры представлены в описании каждого из объектов.

Отношения определяют, как различные объекты сети связаны друг с другом. Например, к какому устройству принадлежит описываемый порт (объект Port) или линк (объект Link). В общем случае отношения могут переходить от любого объекта к любому другому. В NML существуют два типа отношений: допустимые и определённые. При этом отношение может быть одновременно и допустимым, и определённым. Термин «допустимое отношение» означает, что применение отношения к объектам схемы допустимо. Определённое отношение означает, что оно имеет конкретное значение.

Проведённый авторами анализ показал, что базовая схема NML не позволяет описать все необходимые для настройки сетевого оборудования параметры. Например, отсутствует возможность назначить интерфейсу виртуальную сеть VLAN для тегированного и нетегированного трафика. Другими важными параметрами сетевой конфигурации, не предусмотренными в стандарте NML, являются IP-адреса сетевых интерфейсов устройств. Без этого корректно сконфигурировать оборудование сегментированной с помощью технологии VLAN сети не представляется возможным. Поэтому авторами было произведено расширение базовой схемы и добавлены два объекта (VLAN - определяет параметры виртуальной сети на устройстве, MLAG - определяет параметры агрегации каналов на устройстве), три отношения и семь дополнительных атрибутов для стандартных объектов Port и Link.

Введённые дополнительные атрибуты определяют связанные с физическим интерфейсом устройства VLAN и режим работы, в котором будет функционировать интерфейс.

Введённые дополнительные отношения определяют наличие IP-адреса у физического или виртуального интерфейса устройства,

наличие у устройства одной или нескольких виртуальных сетей VLAN, наличие у физического интерфейса устройства атрибутов.

Требования к программному средству генерации конфигурационных файлов по NML-описанию сети

Использование языка разметки сети NML позволяет сравнительно быстро и легко изменить описание существующей сети. Однако, имея лишь описание сети, невозможно переконфигурировать само сетевое оборудование. На основе описания требуется сформировать файлы, содержащие команды CLI (Command Line Interface – интерфейс командной строки) определённого сетевого оборудования. Решением этой задачи может являться автоматическая генерация команд по заданному описанию и формирование конфигурационного файла для конкретного устройства. Сформированный конфигурационный файл должен позволять загрузить в устройство новые настройки во время кратковременной приостановки работы.

Авторами было реализовано собственное программное средство, выполняющее задачу генерации конфигурационных файлов по описанию на NML, которое было названо «Switches Configs Generation», или сокращенно – SCG.

Перечислим требования, которым должно удовлетворять программное средство SCG:

– программа должна разбирать XML-разметку, определённую стандартом NML;

– программа должна осуществлять интерпретацию NML-описания сети в команды CLI определённого сетевого оборудования;

– программа должна генерировать выходные данные в виде конфигурационных файлов для различного сетевого оборудования;

– наличие версионности выходных данных;

– возможность создавать модули для нового сетевого оборудования на основе реализованных модулей программы;

– совместимость с UNIX-подобными ОС.

Работа с XML-разметкой требуется в связи с тем, что NML использует синтаксис XML при описании объектов сети и отношений между ними.

Под версионностью подразумевается хранение всех предыдущих NML-описаний и сформированных по ним конфигурационных файлов для сетевого оборудования. Версионность выходных данных необходима для возможности восстановить последнюю рабочую конфигурацию оборудования в случае возможной ошибки системного администратора на этапе формирования описания сети.

Возможность создавать модули для нового сетевого оборудования требуется для универсализации процесса расширения программы SCG.

Требование к совместимости с UNIX-подобными ОС обусловлено тем, что именно они в настоящее время используются в большинстве суперкомпьютерных центров.

Для реализации программного средства авторами был выбран интерпретируемый язык Python, выбор обусловили интеграция интерпретатора языка в дистрибутивы UNIX-подобных ОС и наличие большой коллекции готовых библиотек, среди которых в том числе имеется библиотека для разбора XML-разметки.

Эта библиотека к тому же поставляется вместе с интерпретатором языка, что упрощает переносимость разработанного приложения.

Архитектура программного средства SCG

Программное средство SCG представляет собой набор Python-модулей. Модули можно условно разделить на три категории:

– модуль для разбора XML-разметки;

– модули для работы с сущностями NML;

– модули для генерации команд определённого сетевого оборудования.

В отдельную категорию можно выделить главный модуль, с помощью которого инициируется запуск программы.

Архитектура программного средства SCG представлена на рисунке 1.

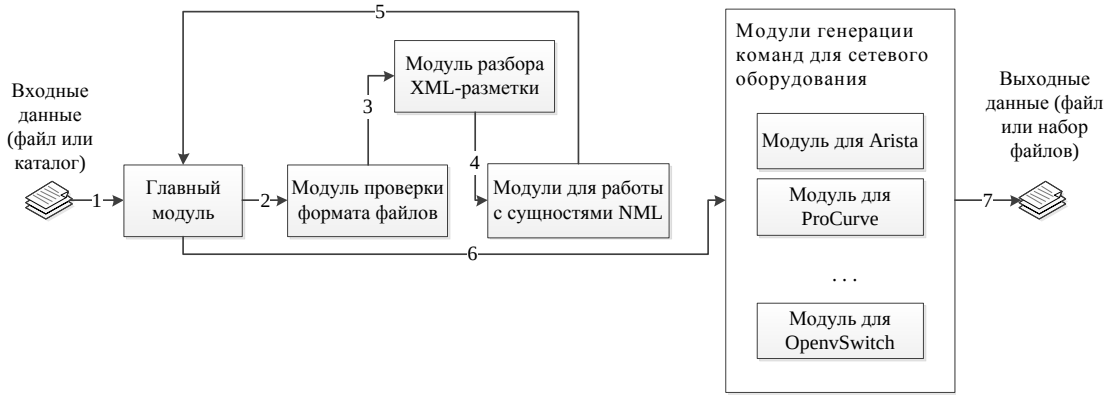


Рисунок 1. Архитектура разработанного программного средства SCG

Стоит отметить, что архитектура программного средства обеспечивает легкую расширяемость для новых сетевых устройств. Для этого будет необходимо создать модуль для нового сетевого оборудования (в качестве шаблона можно использовать любой существующий модуль) и на этапе выбора типа указать соответствующий модуль для генерации команд.

Для генерации конфигурационных файлов сетевого оборудования авторами был разработан специальный алгоритм, затрагивающий все модули программы. Алгоритм можно условно разбить на 3 этапа:

– получение и проверка файлов NML-описания сети;

– разбор XML-разметки и сохранение параметров описанной сети;

– генерация конфигурационных файлов, содержащих команды определённого сетевого оборудования.

На первом этапе происходят получение NML-файлов и проверка корректности XML-разметки в них.

На втором этапе поочередно осуществляется разбор разметки прошедших проверку файлов. Средствами Python-библиотеки minidom в файле происходит поиск XML-тегов, соответствующих объектам и отношениям NML. Атрибуты, принадлежащие тегу, считываются и записываются в объект класса, соответствующий тегу.

На третьем этапе, в зависимости от значения атрибута, указывающего на тип устройства, осуществляется создание файла с именем устройства в специальной иерархии каталогов. Иерархия каталогов была разработана авторами для удобства хранения созданных конфигурационных файлов и обеспечения их версионности. Верхний уровень иерархии составляют каталоги с датой генерации выходных файлов. Уровнем ниже располагаются каталоги, указывающие на время генерации выходных файлов. Каждый такой каталог содержит подкаталоги, хранящие файлы команд CLI для определённого сетевого оборудования и соответствующие им исходные NML-описания сети.

Результатом разработок стал расширяемый программный комплекс, включающий (см. рис. 1) модули генерации команд для сетевого оборудования OpenvSwitch, Arista Networks и HP Procurve.

Методика, средства и результаты проверки корректности сгенерированных настроек сети

Для проверки правильности настроек, сгенерированных средством SCG, авторами была применена среда GNS3 (Graphical Network Simulator – графический симулятор сети), разработанный в 2008 году коллективом GNS3 Technologies [7-9].

GNS3 позволяет создавать модели компьютеров или других устройств и запускать внутри моделей оригинальные операционные системы, например, ОС EOS коммутатора Arista [10].

При этом полнофункционально эмулируются все основные компоненты устройства, в том числе процессор, память и устройства ввода/вывода.

Вместе с ПО GNS3 поставляется виртуальная машина GNS3 VM, которая требуется для виртуализации объектов GNS3, например, коммутаторов.

Рассмотрим тестовый стенд, созданный авторами для проверки работоспособности сгенерированных средством SCG конфигурационных файлов сетевых устройств (рис. 2).

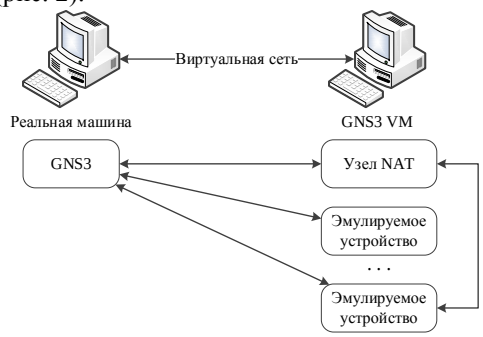


Рисунок 2. Схема тестового стенда

Реальная машина работает под управлением ОС семейства Windows. На ней было установлено программное обеспечение GNS3. Виртуальная машина GNS3 VM работает под управлением ОС семейства Linux.

Через виртуальный сетевой интерфейс реальная машина соединяется с эмулируемым узлом NAT, который может быть связан с одним из эмулируемых ВМ устройств. Такая связь позволяет посредством TFTP-сервера копировать файлы с реальной машины на эмулируемое устройство.

Для проверки корректности полученных конфигурационных файлов авторами была разработана методика, содержащая следующие этапы:

– загрузка файлов в эмулируемые коммутаторы;

– исполнение загруженного файла с целью переконфигурировать коммутатор;

– проверка работы коммутатора в имитационной сети внутри GNS3;

– проверка доступности узлов между собой утилитой ping.

Для загрузки файла в коммутатор необходимо соединить этот коммутатор с узлом NAT, который представлен как виртуальный сетевой интерфейс реальной машины.

Для проверки работоспособности коммутатора авторами в GNS3 была построена имитационная модель сети (см. рисунок 3).

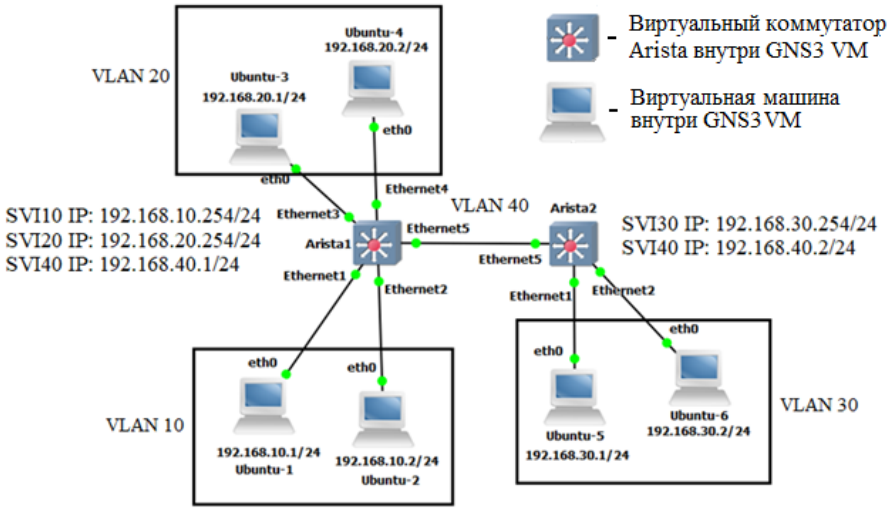


Рисунок 3. Построенная модель сети для проверки корректности настроек коммутаторов

Модель включает в свой состав два эмулируемых коммутатора (на рисунке – из состава оборудования Arista), работоспособность которых будет проверяться, и шесть эмулируемых конечных узлов, работающих на базе ОС Ubuntu.

Отметим, что модель сети была разработана таким образом, чтобы имелась возможность проверить корректное сегментирование сети коммутаторами. Для этого пары конечных узлов должны находиться в отдельных VLAN, а между самими коммутаторами порты должны быть настроены в режиме транка.

После загрузки производилась переконфигурация коммутатора путём исполнения файла в оболочке bash или непосредственно в CLI коммутатора. После этого для проверки корректности сегментирования сети коммутаторами с применением утилиты ping определялась взаимная доступность узлов в пределах одного VLAN коммутатора, в пределах разных VLAN коммутатора и в пределах разных VLAN двух коммутаторов.

Проверка показала, что конфигурационные файлы составлены корректно, и ошибки во время их исполнения отсутствуют.

Оценка эффективности предложенного решения

Эффективность предложенного в настоящей работе подхода можно оценить, рассмотрев этапы применения системным администратором разработанного авторами решения.

Первым этапом является составление описания сети на языке NML. На этом этапе системному администратору необходимо, используя XML-синтаксис NML, сформировать файл, содержащий NML-описание сети. В файле могут быть описаны параметры портов, VLAN, линков и параметры агрегирования линков. Несмотря на то, что данный этап в настройке устройств остаётся человекозависимым, описание системы на формализованном языке позволяет на уровне ПС SCG выявить многие ошибки. Кроме того, этот подход позволяет централизовано хранить, исправлять и дополнять конфигурационные параметры всех устройств в одном или нескольких файлах.

Это освобождает от необходимости вручную проверять текущую конфигурацию на коммутаторах и выгружать её с коммутаторов для последующего хранения. Однако некоторые ошибки выявляются только путем повторной проверки составленного файла NML-описания, что накладно с точки зрения затрат времени.

Однако, если бы описание отсутствовало, то системному администратору пришлось бы вручную проверять коммутаторы на наличие ошибок конфигурации.

Вторым этапом является автоматическая генерация ПС SCG конфигурационных файлов. На этом этапе системному администратору требуется лишь инициировать запуск программного средства, указав в качестве параметра файл или каталог с файлами, содержащими NML-описание сети. Этап практически не зависит от человека, и ошибки, которые могут быть допущены, сводятся к неправильному использованию программного средства, о чём будет выдана соответствующая диагностика.

С точки зрения эффективности этот этап значительно упрощает работу системного администратора, так как ему приходится не формировать самостоятельно файл с командами для коммутатора, а лишь правильно описать необходимые параметры команд в файле NML-описания и запустить генерацию конфигурационных файлов устройств.

Третьим этапом является непосредственное конфигурирование сетевых устройств.

На этом этапе системному администратору требуется загрузить сгенерированные конфигурационные файлы в устройства и исполнить их (или применить настройки – в зависимости от вида сетевого устройства).

Этот этап также является человекозависимым, и на нём могут возникнуть ошибки при загрузке конфигурационного файла от другого устройства.

Ошибки системного администратора могут заключаться в том, что в коммутатор некоторого типа вместо его конфигурационного файла будет загружен конфигурационный файл другого коммутатора.

Если синтаксис команд коммутаторов отличается, то неточность будет выявлена при исполнении файла на коммутаторе.

Если же синтаксис команд совпадает (коммутаторы одного типа), то ошибку можно выявить только на этапе проверки работы сконфигурированного коммутатора.

Третий этап является достаточно эффективным, так как позволяет сэкономить системному администратору значительное количество времени.

В случае, если бы конфигурационный файл с командами отсутствовал, администратору требовалось бы ввести каждую команду в отдельности, что в случае порядка нескольких сотен команд требует больших временных затрат.

Генерация конфигурационных файлов при помощи ПС SCG, во-первых, предотвращает наличие синтаксических ошибок в сгенерированных командах, во-вторых, требует от администратора лишь загрузить полученный конфигурационный файл в коммутатор и исполнить его, количество действий при этом уменьшается в несколько раз.

Действия системного администратора на четвёртом этапе – проверке корректности работы сконфигурированного коммутатора – не зависят от способа введения команд реконфигурации.

Таким образом, предложенный авторами подход к описанию сети и разработанное программное средство позволяют системному администратору существенно сократить время, затрачиваемое на изменение и настройку администрируемой сети.

Заключение

Для решения задачи автоматизации работы системного администратора по реконфигурации сети научного суперкомпьютерного центра авторами был предложен подход, заключающийся в автоматической генерации конфигурационных файлов сетевых устройств по составленному администратором на языке NML описанию сети. Для обеспечения возможности описания всех необходимых параметров сетевого оборудования были предложены соответствующие расширения языка NML.Были разработаны модульная архитектура и алгоритм работы программного средства SCG, осуществляющего генерацию конфигурационных файлов сетевого оборудования по NML-описанию сети. Архитектура SCG подразумевает возможность расширения спектра поддерживаемого сетевого оборудования за счёт разработки новых модулей. Авторская реализация SCG включает в себя модули для оборудования OpenvSwitch, Arista Networks и HP Procurve.

Для проверки правильности настроек, сгенерированных средством SCG, была разработана и опробована методика тестирования, основанная на применении среды моделирования GNS3.

Результаты оценки эффективности применения предложенного подхода показали, что разработанное решение позволяет с меньшими затратами времени переконфигурировать существующие сети.

В качестве направления развития работы представляется возможным реализация средств автоматической генерации файлов описания сети на языке NML, проверки корректности составленного NML-файла или конфигурационных файлов для сетевого оборудования.

Для того чтобы повысить удобство создания файлов описаний, представляется возможной реализация графического редактора построения сети с возможностью экспорта созданной сети в NML-файл.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) «Исследование и разработка методов сетевой интеграции ресурсов и сервисов научных организаций» (0065-2018-0404).

About one method of scientific supercomputer center network configuring

A.V.Baranov, A.P.Ovsyannikov, V.F.Ogaryshev, V.I.Fedorov

Abstract: The article considers the approach for automation of the system administrator routine activity for the network set up and reconfiguration in the supercomputer centers. The authors propose to use the Network Markup Language (NML) for the definition of the network. The software developed by the authors converts the NML-description of the network into a set of network devices configuration files. Due to the modular architecture of the software, the range of supported network devices expands by the adding of the new designed modules. The article also provides a technique for the verifying of the generated configuration files. It based on the usage of the graphical network simulator GNS3. The paper also gives the estimation of the efficiency of the proposed approach.

Keywords: network configuration, Network Markup Language, NML, VLAN, supercomputer center, GNS3, network simulator.

Литература

1. О.С. Аладышев, М.Р. Биктимиров, М.А. Жижченко, А.П. Овсянников, В.М. Опилек, Б.М. Шабанов, Н.Ю. Шульга. Особенности построения объединенной сети суперкомпьютерного центра. «Программные продукты и системы», (2008), № 2. С. 9-12.

2. M.Harris. Data Center Infrastructure Management. In Data Center Handbook, 2014, pp. 601-618. DOI: 10.1002/9781118937563.ch33.

3. Управление инфраструктурой ЦОД: в чем разница между DMaaS и DCIM? - Новости рынка ЦОД, обзор инженерных решений Дата-Центров [электронный ресурс] URL: http://telecombloger.ru/100621 (Дата обращения 10.07.2018).

4. Ситуация с DCIM: как изменилось управление инфраструктурой ЦОД за последние годы / Блог компании ИТ-ГРАД / Хабр [электронный ресурс] URL: https://habr.com/company/it-grad/blog/415385/ (Дата обращения 10.07.2018).

5. Рынок DCIM – сколько стоит ПО для управления инфраструктурой ЦОД [электронный ресурс] / URL: http://telecombloger.ru/16096 (Дата обращения 24.04.2018).

6. Jeroen van der Ham, Freek Dijkstra. Network Markup Language Base Schema version 1 [электронный ресурс] / Open Grid Forum, 2013. URL: https://www.ogf.org/documents/GFD.206.pdf (Дата обращения 30.11.2017).

7. Jason C. Neumann. The Book of GNS3: Build Virtual Network Labs Using Cisco, Juniper, and More. No Starch Press, 2015. 272 pgs. ISBN: 978-1-59327-554-9

8. GNS3 [электронный ресурс] / GNS3 Technologies, 2018. URL: https://github.com/GNS3 (Дата обращения 24.04.2018)

9. Getting Started with GNS3 [электронный ресурс] / GNS3 Technologies, 2018. URL: https://docs.gns3.com/1PvtRW5eAb8RJZ11maEYD9_aLY8kkdhgaMB0wPCz8a38/index.html (Дата обращения 24.04.2018)

10. Arista EOS User Guide [электронный ресурс] / Arista Networks, 2017. URL: https://www.arista.com/assets/data/docs/Manuals/EOS-4.20.1F-Manual.pdf (Дата обращения 24.04.2018).

Модернизация подсистемы сбора и обработки статистики центра коллективного пользования вычислительными ресурсами МСЦ РАН

А.В.Баранов[1], Е.А.Киселёв[2], Е.С.Кормилицин[3], В.Ф.Огарышев[4], П.Н.Телегин[5]

[1,2,4,5]Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, ,

[3]Московский физико-технический институт (государственный университет), Москва, Россия,

E-mails: [1]antbar@mail.ru, [2]kiselev@jscc.ru, [3]black686rebel@yandex.ru, [4]ogaryshev@jscc.ru, [5]ptelegin@jscc.ru

Аннотация: В статье рассматривается подсистема сбора и обработки статистической информации о работе суперкомпьютерного центра коллективного пользования на примере Межведомственного суперкомпьютерного центра Российской академии наук (МСЦ РАН). Авторами были выявлены недостатки существующей версии подсистемы сбора и обработки статистики, выбраны архитектура и инструментальные средства для реализации программного средства, устраняющего выявленные недостатки. В статье приводятся структура и алгоритмы работы разработанного программного средства, рассматриваются его возможности и отмечаются результаты опытной эксплуатации во внутренней сети МСЦ РАН.

Ключевые слова: высокопроизводительные вычисления, суперкомпьютеры, планирование заданий, статистика работы суперкомпьютера, «тонкий» клиент, web-интерфейс, база данных статистики

Введение

Одной из важнейших задач деятельности суперкомпьютерного центра коллективного пользования (СКЦ) является задача сбора и обработки различной статистической информации, позволяющей пользователям оценивать результаты своей работы на суперкомпьютере, системным администраторам – осуществлять мониторинг работоспособности и функциональности вычислительных систем, руководству СКЦ – анализировать эффективность работы суперкомпьютерного оборудования центра и планировать его дальнейшую деятельность. Для этих целей в СКЦ организуются специальные системы [1, 2], включающие как базы данных (БД), накапливающие соответствующую статистическую информацию, так и средства доступа к этой информации и ее анализа.

В Межведомственном суперкомпьютерном центре Российской академии наук (МСЦ РАН) более 15 лет организация коллективного пользования вычислительными ресурсами суперкомпьютеров осуществляется при помощи отечественной системы управления прохождением параллельных заданий (СУППЗ) [3], разработанной в Институте прикладной математики им. М.В.Келдыша (ИПМ РАН) совместно с МСЦ РАН. В состав СУППЗ входит подсистема сбора, хранения и обработки статистической информации о работе СКЦ (подсистема «Статистика»), включающая базу данных [4] и специальное программное средство (ПС) «МСЦ-КроСтат» [5].

Функциональные возможности и архитектура подсистемы «Статистика» развивались исторически в течение нескольких лет и на сегодняшний день морально устарели. В частности, подсистема «Статистика» обладает рядом следующих существенных недостатков:

– хотя формально для доступа к БД требуется авторизация, фактически для работы подсистемы используется единственная учетная запись, что как снижает информационную безопасность подсистемы, так и сужает ее функционал за счет отсутствия возможности многопользовательского доступа с разграничением ролей;

– ПС «МСЦ-КроСтат» позволяет генерировать статистические отчеты только в

виде таблиц, отсутствует возможность построения графиков и диаграмм;

– архитектура ПС «МСЦ-КроСтат» не позволяет расширять функциональность подсистемы «Статистика» за счет разработок новых модулей.

Целью работы является разработка современной версии подсистемы «Статистика», свободной от перечисленных недостатков. Результатом работы стало программное средство формирования и визуализации статистических отчетов о работе СКЦ, названное авторами iscStatistics и прошедшее опытную эксплуатацию в МСЦ РАН.

Сбор и обработка статистической информации о работе суперкомпьютерной системы

Рассмотрим структуру типовой суперкомпьютерной системы (рис. 1). Ее главной составляющей является решающее поле высокопроизводительных вычислительных модулей (ВМ), объединенных высокоскоростной коммуникационной сетью. Ресурсы решающего поля используются для высокопроизводительных расчетов с помощью пользовательских прикладных параллельных программ, которые могут требовать для своего выполнения от одного до всех ВМ решающего поля.

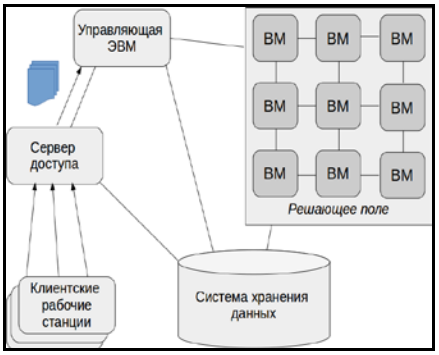


Рисунок 1. Типовая архитектура суперкомпьютерной системы

Для получения возможности производства расчетов на решающем поле суперкомпьютера пользователь должен сформировать т.н. вычислительное задание, в состав которого входят прикладная параллельная программа, исходные данные для расчетов и требования к ресурсам. В минимальный состав требований входят число ВМ и время, которые требуются параллельной программе для выполнения.

Сервер доступа предназначен для подключения к системе пользователей, которые подготавливают с его помощью параллельные программы, исходные данные, формируют вычислительные задания и направляют их на управляющую ЭВМ. Последняя принимает входной поток различных заданий от разных пользователей, ведет очередь заданий, осуществляет запуск заданий и контроль их выполнения на решающем поле суперкомпьютера.

Управление вычислительными ресурсами суперкомпьютера и пользовательскими заданиями осуществляется с помощью специальных программных систем управления заданиями (СУЗ), которые часто называют системами пакетной обработки (СПО) заданий [6]. Примерами СУЗ могут служить такие известные системы, как SLURM [7], PBS [8], Moab [9] и уже упоминавшаяся отечественная СУППЗ. Все СУЗ содержат в своем составе подсистемы сбора и обработки статистической информации о своей работе. Сбор информации заключается в фиксации статистически значимых событий, фиксируются время наступления события и его параметры. К основным статистически значимым событиям относятся:

– поступление нового вычислительного задания в систему (постановка задания в очередь);

– запуск задания на выполнение после прохождения очереди и, соответственно, занятие заданием определенного объема вычислительных ресурсов – подмножества ВМ решающего поля;

– завершение выполняющегося задания и, соответственно, освобождение занятых заданием вычислительных ресурсов;

– снятие задания из очереди до его запуска;

– старт и остановка СУЗ;

– смена режима работы СУЗ;

– изменение состава ВМ решающего поля по причине отказа (неисправности) определенных ВМ или, наоборот, их восстановления после отказа (возврата из ремонта).

Обработка собранной информации заключается в формировании статистических отчетов, отражающих за определенный период времени как потребление вычислительных ресурсов пользователями (биллинг), так и показатели эффективности работы суперкомпьютерной системы. К последним относятся утилизация ресурсов (или обратный

утилизации показатель – процент простаивающих ресурсов), среднее время нахождения задания в очереди, число обработанных заданий и др. Биллинговая информация может собираться отдельно по пользователям, по организациям, по научным проектам.

Рассмотрим организацию работы подсистемы «Статистика» СУППЗ (рис. 2).

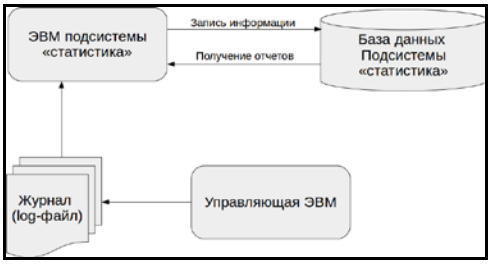


Рисунок 2. Схема работы подсистемы «Статистика»

Информация о происшедших статистически значимых событиях фиксируется СУППЗ на управляющей ЭВМ в специальном журнале событий (log-файле). С помощью ПС «МСЦ-КроСтат» информация из журналов событий заносится в специализированную БД [4] под управлением свободно распространяемой СУБД PostgreSQL. После разбора журналов событий и заполнения БД становится возможным формирование ряда статистических отчетов, содержащих как биллинговую информацию, так и информацию об утилизации ресурсов.

Программное средство «МСЦ-КроСтат» способно функционировать как в среде Linux, так и в среде MS Windows, поскольку реализовано при помощи кроссплатформенной библиотеки Qt. ПС «МСЦ-КроСтат» имеет развитый графический пользовательский интерфейс, и на момент создания программа отвечала всем современным требованиям. С течением времени непрерывно возрастало значение статистической информации о работе СКЦ, расширялся круг потребителей этой информации, в этот круг помимо руководства и системных администраторов СКЦ вошли представители учредителей научных организаций и пользователи СКЦ. ПС «МСЦ-КроСтат» не способно обеспечить многопользовательский режим работы с возможностью разграничения ролей потребителей статистической информации, в ПС отсутствует механизм расширения функционала и добавления новых статистических отчетов.

ПС «МСЦ-КроСтат» построено на основе архитектуры «толстого» клиента, в соответствии с которой логика и алгоритмы обработки статистической информации сосредоточены в клиентском приложении, серверная сторона в этом случае служит поставщиком данных для клиента. Недостатки архитектуры «толстого» клиента заключаются в необходимости предварительной установки и последующего сопровождения программного обеспечения на компьютере каждого потребителя информации, в трудности добавления в систему новых форм документов, а также в определенной сложности программного обеспечения на стороне клиента, влекущей повышенные требования к производительности клиентского компьютера.

Требования к разрабатываемому программному средству

Для преодоления недостатков ПС «МСЦ-КроСтат» было принято решение о разработке нового программного средства генерации статистических отчетов о работе СКЦ, соответствующего современному уровню развития технологий. Для разрабатываемого ПС, получившего условное название «iscStatistics», авторами были сформулированы следующие требования.

1. ПС «iscStatistics» должно обеспечивать доступ к статистической информации следующим категориям пользователей:

– системные администраторы – категория пользователей, осуществляющих настройку и сопровождение подсистемы «Статистика», пользователи этой категории имеют доступ ко всем функциональным возможностям ПС «iscStatistics», включая возможности реконфигурации, добавления новых и удаления потерявших актуальность статистических отчетов;

– руководители – категория, включающая руководство СКЦ и представителей учредителей научных организаций, пользователи этой категории имеют возможность генерировать и, соответственно, просматривать любые доступные статистические отчеты;

– пользователи – в этой категории представлены обычные пользователи СКЦ, которые могут генерировать и, соответственно, просматривать статистические отчеты только о собственном потреблении ресурсов СКЦ, биллинговая информация о других пользователях им недоступна.

2. ПС «iscStatistics» должно иметь web-интерфейс с авторизацией пользователей через службу LDAP (Lightweight Directory Access Protocol) – протокол прикладного уровня для доступа к службе каталогов, позволяющий производить операции аутентификации, поиска и сравнения, а также операции добавления, изменения или удаления записей. Служба каталога (англ. Directory Service) – средство иерархического представления ресурсов, принадлежащих некоторой отдельно взятой организации, и информации об этих ресурсах. Под ресурсами могут пониматься материальные ресурсы, персонал, сетевые ресурсы и т.д. В МСЦ РАН служба LDAP хранит информацию о всех учетных записях пользователей СКЦ.

3. ПС «iscStatistics» должно позволять строить инфографики для сформированных статистических отчетов.

4. Должна быть возможность добавления новых видов отчетов по готовым запросам к базе данных статистики.

5. Для редактирования и печати должна быть возможность экспортирования отчетов и графиков в другие программные средства, в частности, в MS Office.

Требование наличия web-интерфейса обуславливает построение ПС «iscStatistics» по архитектуре «тонкого» клиента. «Тонкий» клиент в компьютерных технологиях – компьютер или программа-клиент в сетях с клиент-серверной или терминальной архитектурой, где большая часть задач по обработке информации перенесена на сервер, и права доступа клиента строго ограничены.

Примером «тонкого» клиента может служить компьютер с браузером, использующийся для работы с web-приложениями.

Преимуществами «тонкого» клиента по сравнению с «толстым» клиентом являются:

– простота установки, конфигурации и администрирования программного обеспечения;

– простое и эффективное разграничение прав доступа пользователей;

– возможность организации удаленного доступа пользователей к системе.

Выбор инструментальных средств

Разрабатываемое программное средство можно разделить на клиентскую и серверную (функциональную) части.

Клиентская часть представляет собой web-интерфейс к функциональной части разрабатываемого ПС. В качестве инструментов для реализации web-интерфейса были выбраны стандартные инструменты web-разработки, в частности:

– HTML5 (HyperText Markup Language, version 5) – стандартизированный язык разметки документов для структурирования и представления данных;

– CSS3 (Cascading Style Sheets, version 3) – каскадные таблицы стилей для описания внешнего вида документов на языке разметки HTML;

– JavaScript – интерпретируемый язык программирования для обеспечения интерактивности web-интерфейса.

Серверная часть ПС «iscStatistics» определяет основную логику работы программного средства.

Для выбора языка программирования серверной части было осуществлено сравнение наиболее распространенных языков для web-разработки по версии GitHub [10].

Был проведен сравнительный анализ языков программирования на основе ряда критериев (возможность взаимодействия с LDAP, наличие средств взаимодействия с СУБД PostgreSQL и с MS Office).

Рассматривались языки PHP, Python, C# и Ruby, анализ показал, что все они удовлетворяют критериям выбора, и не определил однозначный выбор в пользу ни одного из вариантов.

Окончательное решение в пользу языка программирования PHP было принято авторами исходя из собственного опыта разработок и, соответственно, сроков освоения среды разработки.

Структура и алгоритмы работы ПС «iscStatistics»

Взаимодействие пользователя с программным средством «iscStatistics» осуществляется через web-интерфейс (см. рисунок 3).

Запросы от пользователя поступают на web-сервер под управлением Apache, который через соответствующие модули (добавления/удаления и формирования отчетов) предоставляет пользователю различные функциональные возможности в зависимости от принадлежности к той или иной кате-

гории (системных администраторов, руководителей, пользователей).

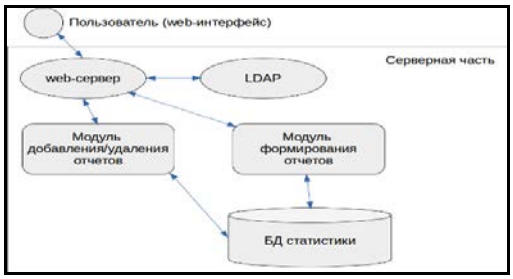


Рисунок 3. Структура программного средства «iscStatistics»

Авторизация пользователей осуществляется путем обращения web-сервера к LDAP-серверу, после чего осуществляется отнесение пользователя к одной из категорий.

Отдельного внимания заслуживает модуль добавления/удаления отчетов, реализующий подсистему подготовки и конфигурации новых модулей. Возможностью добавления новых отчетов обладают только системные администраторы, остальные категории пользователей, в т.ч. руководители, могут только генерировать и просматривать отчеты, сформированные администратором. В случае необходимости добавить новый отчет руководители или пользователи должны будут обратиться к системному администратору, который отвечает за корректность и информационную безопасность добавляемого отчета.

Статистический отчет в общем случае представляет собой заполненную данными таблицу, отражающую ту или иную информацию в зависимости от вида отчета. Для формирования статистического отчета необходимы данные, извлекаемые из БД статистической информации. Под вариантом отображения данных будем понимать ту или иную конфигурацию конкретного типа отчета. Все возможные конфигурации конкретного отчета будут образовывать модуль. Другими словами, модуль – это все множество конфигураций (вариаций отображения данных, имеющих одинаковый смысл) конкретного отчета.

Для реализации модульной структуры, позволяющей добавлять и удалять типы отчетов, была создана сущность в базе данных для хранения конфигураций отчетов. Созданная сущность содержит следующие атрибуты:

– название (тип) отчета;

– соотношение атрибутов, возвращаемых SQL-запросом, и подписей в шапке таблицы сформированного отчета; атрибуты хранятся в формате JSON, где ключом выступает атрибут, возвращаемый SQL-запросом, а значением – расшифровка этого атрибута (подпись в шапке таблицы);

– SQL-запрос к базе данных на извлечение необходимой для построения отчета информации;

– необходимые для выполнения SQL-запроса параметры, хранящиеся в формате JSON; ключом является название параметра, значениями выступают номер параметра в запросе и значение этого параметра.

Все кортежи сущности «Конфигурации отчетов», имеющие одинаковое значение атрибута «Тип отчета», представляют собой модуль, состоящий из различных конфигураций данного отчета.

Добавление новой конфигурации отчета доступно только пользователям категории администраторов.

Для добавления новой конфигурации какого-либо типа отчета системному администратору необходимо указать все перечисленные атрибуты сущности «Конфигурации отчетов».

Основная работа системного администратора заключается в конструировании SQL-запроса на извлечение данных, необходимых для формирования отчета, описании возвращаемых SQL-запросом атрибутов и определения необходимых для выполнения SQL-запроса параметров.

Модуль формирования отчетов предоставляет пользователю возможность для генерации статистических отчетов и их визуальных представлений.

Пользователю необходимо выбрать тип отчета и задать параметры, необходимые для выполнения SQL-запроса, указанного системным администратором в момент добавления соответствующей конфигурации отчета.

При этом системным администраторам и руководителям для извлечения и представления в виде отчета доступна любая информация из БД, а обычным пользователям – только информация об их собственной работе в СКЦ.

Алгоритм формирования статистических отчетов представлен на рисунке 4.

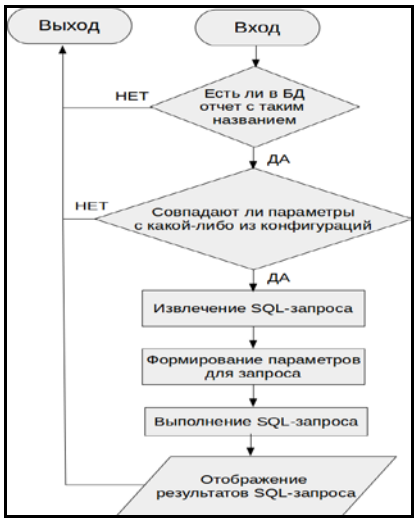


Рисунок 4. Алгоритм формирования отчета

Первым шагом алгоритма формирования отчета является определение его наличия в базе данных. Далее ищется соответствие введенных пользователем параметров с параметрами какой-либо конфигурации выбранного отчета. В случае нахождения соответствия происходит извлечение SQL-запроса соответствующей конфигурации. После чего в извлеченный SQL-запрос осуществляется подстановка введенных пользователем параметров, и производится выполнение запроса. На основе извлеченных запросом данных формируется таблица отчета, предоставляемая пользователю в виде html-страницы.

Опытная эксплуатация ПС «iscStatistics»

Для проведения опытной эксплуатации и демонстрации работоспособности разработанное программное средство было развернуто на отдельном компьютере под управлением Linux во внутренней сети МСЦ РАН. В качестве web-сервера был использован Apache. Макет был подключен к LDAP-серверу МСЦ РАН и БД «Статистика» под управлением СУБД PostgreSQL.

Доступ к ПС «iscStatistics» предоставляется из внутренней сети МСЦ РАН после прохождения авторизации через LDAP. Внешний вид страницы формирования отчетов ПС «iscStatistics» приведен на рисунке 5.

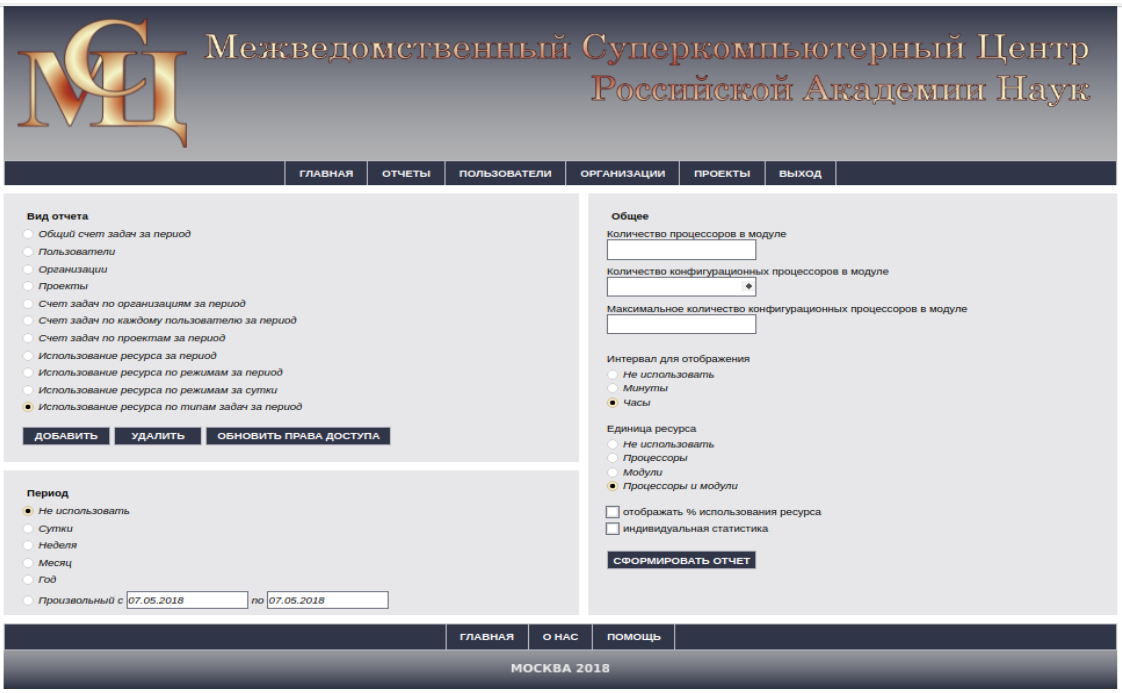


Рисунок 5. Страница формирования статистических отчетов

В опытной версии ПС «iscStatistics» с помощью подсистемы добавления новых отчетов были реализованы все отчеты, которые генерировались с помощью ПС «МСЦ-КроСтат», таким образом была проверена и отлажена работоспособность подсистемы. Модальное окно добавления нового отчета представлено на рисунке 6.

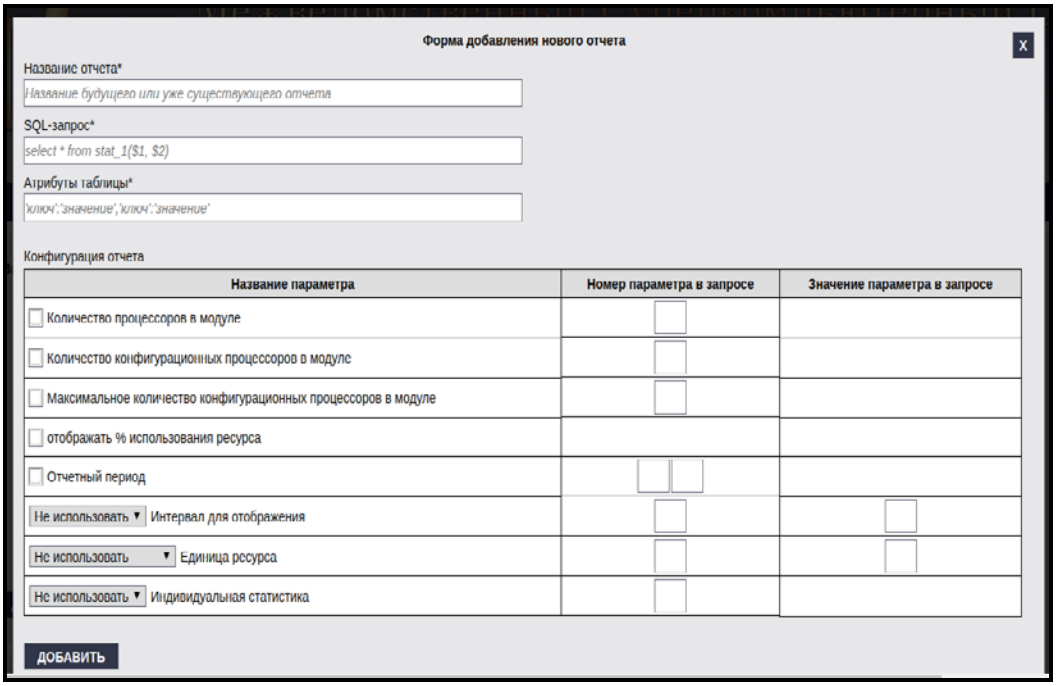


Рисунок 6 – Окно добавления новой конфигурации статистического отчета

После добавления нового отчета он становится доступным для извлечения из БД и представления статистической информации, соответствующего форме отчета. Для сформированного отчета возможно построение наглядных графиков и диаграмм. Для этого необходимо заполнить специальную форму (рис. 7), пример построенной диаграммы представлен на рисунке 8.

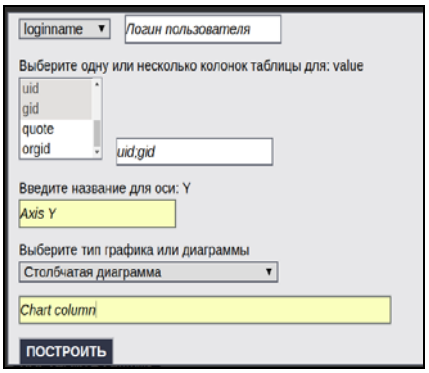


Рисунок 7. Пример заполненной конфигурационной формы для построения графика или диаграммы

Кроме этого, для построенного отчета существует возможность выгрузки отчетной таблицы и соответствующих ей графиков и диаграмм в файл формата MS Excel. Далее этот файл может быть использован для работы с отчетом в среде MS Office.

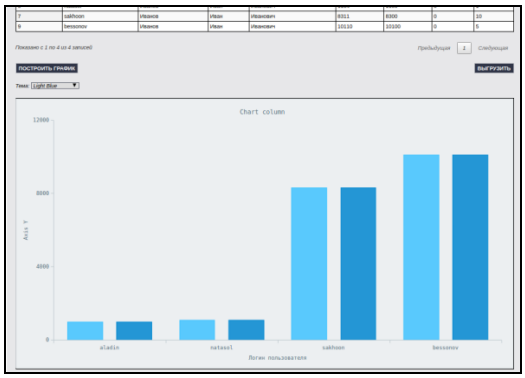


Рисунок 8. Пример построенной диаграммы

Заключение

В рамках модернизации подсистемы сбора и обработки статистической информации о работе центра коллективного пользования вычислительными ресурсами МСЦ РАН авторами было разработано специализированное программное средство «iscStatistics», отвечающее современным требованиям к подобного рода программным продуктам. В частности, ПС «iscStatistics» имеет web-

интерфейс, обладает возможностями авторизации пользователей через службу LDAP и разграничения ролей пользователей в зависимости от их категории.

Разработанное ПС «iscStatistics» имеет модульную структуру, реализованную по принципу «один отчет – один модуль», что делает возможным процесс осуществления реконфигурации ПС «iscStatistics». Реконфигурация заключается в возможности добавления и удаления статистических отчетов, что позволяет легко расширять функционал системы.

Разработанное ПС «iscStatistics» позволяет строить графики и диаграммы по сформированным статистическим отчетам, а также выгружать результаты в среду MS Office.

Опытная эксплуатация разработанного программного средства во внутренней сети МСЦ РАН показала его работоспособность и функциональность.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) «Разработка архитектур, системных решений и методов для создания вычислительных комплексов и распределенных сред мультипетафлопсного диапазона производительности, в том числе нетрадиционных архитектур микропроцессоров» (0065-2018-0409)

Modification of the statistic subsystem of the Joint Supercomputer Center of the Russian Academy of Sciences

A.V.Baranov, T.F.Kiselev, E.S.Kormilitsin, V.F.Ogaryshev, P.N.Telegin

Abstract: The article considers subsystem of collection and processing statistical information on the operation of a supercomputer center with the example of the Joint Supercomputer Center of the Russian Academy of Sciences (JSCC RAS). The authors identified drawbacks of current version of the subsystem. There were selected architecture and tools for software implementation to remove these drawbacks. The article describes the structure and algorithms of the developed software, considers its features and results of trial operation in the local network of the JSCC RAS.

Keywords: high performance computing, supercomputers, job scheduling, supercomputer operation statistics, thin client, statistic database

Литература

1. А.Ю.Сафонов, П.С.Костенецкий. Система сбора и отображения статистики о загрузке суперкомпьютеров ЛСМ ЮУрГУ // В сборнике: Параллельные вычислительные технологии (ПаВТ'2015) Труды международной научной конференции. Редакторы: Л.Б.Соколинский, К.С.Пан. 2015. С. 519.

2. В.Воеводин, С.Жуматий, С.Соболев, А.Антонов, П.Брызгалов, Д.Никитенко, К.Стефанов. // Открытые системы. СУБД. 2012, №7, с.36.

3. Система управления прохождением параллельных заданий (СУППЗ). Руководство программиста (пользователя) [электронный ресурс] / URL: http://www.jscc.ru/informat/СУППЗ_Руководство_пользователя.pdf (дата обращения 03.07.2018).

4. А.В.Баранов, С.А.Дбар. Статистика Системы управления прохождением параллельных заданий. Свидетельство о государственной регистрации базы данных № 2016621669 от 15 декабря 2016 года.

5. О.С.Аладышев, А.В.Баранов, Е.А.Киселёв, Р.И.Гришин. Система сбора и обработки статистики «МСЦ-МСЦ-КроСтат». Свидетельство о государственной регистрации программы для ЭВМ № 2016663606 от 13 декабря 2016 года.

6. А.В.Баранов, А.В.Киселёв, В.В.Старичков, Р.П.Ионин, Д.С.Ляховец. Сравнение систем пакетной обработки с точки зрения организации промышленного счета // В сборнике: Научный сервис в сети Интернет: поиск новых решений: Труды Международной суперкомпьютерной конференции (17-22 сентября 2012 г., г. Новороссийск). М.: Изд-во МГУ, 2012. С. 506-508.

7. A.Yoo, M.Jette, M.Grondona. (2003) SLURM: Simple Linux Utility for Resource Management. In: Feitelson D., Rudolph L., Schwiegelshohn U. (eds) Job Scheduling Strategies for Parallel Processing. JSSPP 2003. Lecture Notes in Computer Science, vol 2862. Springer, Berlin, Heidelberg. pp. 44-60. DOI: 10.1007/10968987_3

8. R.L.Henderson. (1995) Job scheduling under the Portable Batch System. In: Feitelson D.G., Rudolph L. (eds) Job Scheduling Strategies for Parallel Processing. JSSPP 1995. Lecture Notes in Computer Science, vol 949. Springer, Berlin, Heidelberg. pp. 279-294. DOI: 10.1007/3-540-60153-8_34

9. Moab HPC Suite Enterprise Edition [электронный ресурс] / URL: http://www.adaptivecomputing.com/products/hpc-products/moab-hpc-suite-enterprise-edition (дата обращения 12.07.2018).

10. 15 самых популярных языков программирования по версии GitHub [электронный ресурс] / URL: https://habrahabr.ru/post/310262/ (Дата обращения: 07.08.2018)

О росте числа анонсируемых подсетей в глобальной таблице маршрутизации Интернет

А.П.Овсянников[1], А.А.Шер [2]

Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия,

E-mails: [1] apo@jscc.ru , [2] sher@rasnet.ru

Аннотация: В статье затрагиваются вопросы увеличения размера глобальной таблицы маршрутизации, приводится попытка оценить масштабы последствий такого роста для научных телекоммуникационных сетей и операторов связи, использующих в своих сетях полноразмерные глобальные таблицы маршрутизации, а также рассматривается влияние роста числа маршрутов на стабильность доступа к ресурсам во всемирной сети Интернет.

Ключевые слова: маршрутизация, Интернет, глобальная таблица маршрутизации, BGP, подсети, IPv4, IPv6

Глобальная структура маршрутизации Интернет образована маршрутами на сети IPv4 и IPv6, анонсируемыми посредством протокола BGP операторами связи и провайдерами контента и сервисов: научными и исследовательскими институтами и университетами, органами государственной власти, средствами массовой информации и т.п.). Анонсы сетей обрабатываются маршрутизаторами, работающими на 3-м (сетевом) уровне эталонной модели OSI, согласно их настройкам, а затем помещаются в таблицы пересылки (Forwarding Information Base - FIB) и таблицы соседства (Adjacency Table) [1]. Таблицы хранятся в специализированной троичной ассоциативной памяти (TCAM, Ternary Content Addressable Memory) маршрутизатора, позволяющей производить очень быстрый поиск нужной строки по ее содержимому.

Память TCAM является дорогой в производстве, занимает физически больше места чем обычная оперативная память, потребляет больше электроэнергии, к тому же часто бывает сильно ограничена по скорости записи [2]. Поэтому проблема увеличения количества маршрутов в глобальной таблице маршрутизации затрагивает большое количество маршрутизирующего оборудования, а значит и эксплуатирующие его организации, управляющие крупными телекоммуникационными сетями.

Сама по себе данная проблема не нова с исторической точки зрения: операторы связи уже сталкивались с ней в 2008 и 2014 годах. В первом случае количество маршрутов превысило 256 000, а во втором – 512 000. Данные цифры стали знаковыми из-за большого распространения сетевого оборудования фирмы Cisco Systems, в частности маршрутизаторов серии Catalyst 6500 с супервизором Sup720-3B, имеющим ограничение размера FIB в 256 000 записей для протокола IPv4, и обновленной его версии Sup720-3BXL, имеющем ограничение уже в 1 000 000 записей для IPv4, правда разделяемое с записями для протоколов IPv6 и многоадресной рассылки, а потому настроенное по умолчанию на максимальное количество именно в 512 000 записей для IPv4 [3,4].

На протяжении многих лет определяющим фактором роста таблиц маршрутизации было распределение региональными интернет-регистраторами новых подсетей IPv4 локальным интернет-регистраторам, а в итоге – назначение их пользователям. Этот процесс в настоящее время является основной причиной роста глобальной таблицы для протокола IPv6, однако в случае с IPv4 ситуация кардинально поменялась еще в 2011 году, в связи с исчерпанием свободных блоков адресов IPv4. 31 января 2011 года IANA (Администрация адресного пространства Интернет) распределила последние два незарезервированных блока размером /8 в пользу Азиатско-Тихоокеанского сетевого информационного центра. 3-го февраля того же года IANA распределила 5 последних зарезервированных блоков /8 между всеми региональными интернет-регистраторами [5]. Таким образом, с этого времени основным фактором роста таблицы маршрутизации IPv4 стало дробление крупных блоков адресов на более мелкие – вплоть до /24 (256 адресов). Этот процесс продолжается и в настоящее время. Дробление на еще более мелкие подсети нецелесообразно, так как многие операторы широкополосного доступа в Интернет фильтруют прием маршрутов с маской более

/24. Для IPv6 же, согласно RFC 5375, подсеть для конечного пользователя всегда должна иметь маску /64 для корректной работы всех функций протокола [6]. Тем самым максимальная маска для подсети IPv6, которая может быть выдана конечному пользователю и будет маршрутизироваться в глобальной сети, - /64.

Исходя из вышесказанного, легко посчитать теоретический предел количества маршрутов в глобальной таблице для обоих протоколов (мы не учитываем зарезервированные блоки адресов специального назначения):

- Для IPv4: 2^24 = 16 777 216 маршрутов
- Для IPv6: 2^64 = 18 446 744 073 709 551 616 маршрутов

Здесь надо заметить, что даже для IPv4 такое количество маршрутов вряд ли будет достигнуто, а для IPv6 рекомендуемый размер подсети для назначения конечным пользователям долгое время был равен /48, и хотя сейчас считается допустимой маска большего размера, блоки /64 как отдельный маршрут в глобальной сети используются крайне редко [7]. Правда даже если считать, что все маршруты IPv6 будут иметь маску /48, таблица маршрутизации при условии распределения всех подсетей может теоретически составить 2^48 = 281 474 976 710 656 маршрутов (без учета зарезервированных блоков адресов специального назначения).

Надо сказать, что объективно оценить размер глобальной таблицы маршрутизации практически невозможно, т.к. для каждого оператора, и даже маршрутизатора размер таблицы является субъективным значением, зависящим от множества факторов: настроенных фильтров, транзитной политики, наличия внутренних маршрутов и т. д. Для сбора и оценки статистических данных был создан проект Route Views в Университете Орегона (данные доступны начиная с 1998 года), а позднее, в 2001 году, региональным интернет-регистратором RIPE NCC - проект RIS (Routing Information Service) [8]. Для данной статьи источником информации послужили материалы проекта Route Views, содержащего больший объем исторического материала по сравнению с RIS.

На сайте проекта Route View [9] данные представлены как в текстовом виде (для IPv4), так и в виде Multi-Threaded Routing Toolkit Routing Information Export Format (MRT - для IPv4 и IPv6), описанном в RFC 6396 с дополнениями в RFC 6397, RFC 8050, включающих дополнительные подтипы полей. Для работы с MRT использовалось свободно распространяемое ПО bgpdump2 [10] и mrtparse [11].

Ежегодный рост числа анонсируемых подсетей IPv4 достигал пиковых значений в 2000-2001 годах (до 34%), а, начиная с 2010 года, стал практически линейным и составлял порядка 10% в год, как это видно из Таблицы 1.

Таблица 1. Число подсетей IPv4

Год	Кол-во подсетей	Прирост по сравнению с предыдущим годом (в %)
1998	60604	-
1999	56087	-7
2000	70890	26
2001	94984	34
2002	115433	22
2003	126266	9
2004	154040	22
2005	167047	8
2006	191534	15
2007	216324	13
2008	248075	15
2009	292062	18
2010	322070	10
2011	353290	10
2012	403604	14
2013	453982	12
2014	498133	10
2015	552984	11
2016	614151	11
2017	677894	10
2018	741108	9

В 2018 году общая тенденция роста остается примерно на том же уровне (Таблица 2).

Таблица 2. Число подсетей IPv4 в 2018 г.

Месяц 2018 года	Кол-во подсетей	Прирост по сравнению с январем (в %)
янв	741108	-
фев	744430	0,4
мар	767594	3,6
апр	748622	1,0
май	754714	1,8
июн	752910	1,6
июл	768295	3,7
авг	760781	2,7

Более наглядно рост числа анонсируемых подсетей IPv4 представлен на рисунке 1. Кривая имеет вид параболы, в связи с чем с целью аппроксимации была построена линия тренда (обозначена пунктиром) с достоверностью R^2 = 0,998. Следуя данному прогнозу, количество маршрутов IPv4 достигнет 800 000 в конце 2018 – начале 2019 года, а 1 000 000 – в 2021-2022 годах.

Рисунок 1. Количество подсетей IPv4

В случае с IPv6 ситуация выглядит кардинально иным образом. Исходя из собранных данных, представленных в Таблице 3, ежегодный прирост, за исключением 2006-го года, составлял десятки процентов, в пике достигая 86%. Начиная с 2015 года тренд, по аналогии с IPv4, стал практически линейным, правда средний показатель ежегодного роста превосходил уже 30% в год.

Таблица 3. Число подсетей IPv6

Год	Кол-во подсетей	Прирост по сравнению с предыдущим годом (в %)
2004	500	-
2005	623	25
2006	748	20
2007	793	6
2008	1230	55
2009	1586	29
2010	2536	60
2011	4119	62
2012	7672	86
2013	11943	56
2014	16669	40
2015	22114	33
2016	28054	27
2017	37493	34
2018	49908	33

Такая же ситуация сохраняется и в 2018 году (Таблица 4).

Таблица 4. Число подсетей IPv6 в 2018 г.

Месяц 2018 года	Кол-во подсетей	Прирост по сравнению с январем (в %)
янв	49908	-
фев	52415	5
мар	52522	5
апр	53580	7
май	55349	11
июн	56350	13
июл	58416	17
авг	60142	21

График роста количества подсетей для протокола IPv6 (График 2) – более сложный, нежели таковой для протокола IPv4. В данном случае для аппроксимации были построены две линии тренда: полином 3-й степени, с достоверностью R2 = 0,9982, и полином 6-й степени – с достоверностью R2 = 0,9996. Первый сценарий является более оптимистичным: согласно ему, количество маршрутов IPv6 достигнет числа 100 000 в 2020-м году. Исходя же из пессимистичного второго сценария, таблица маршрутизации для протокола IPv6 будет составлять 100 000 маршрутов уже в 2019 году.

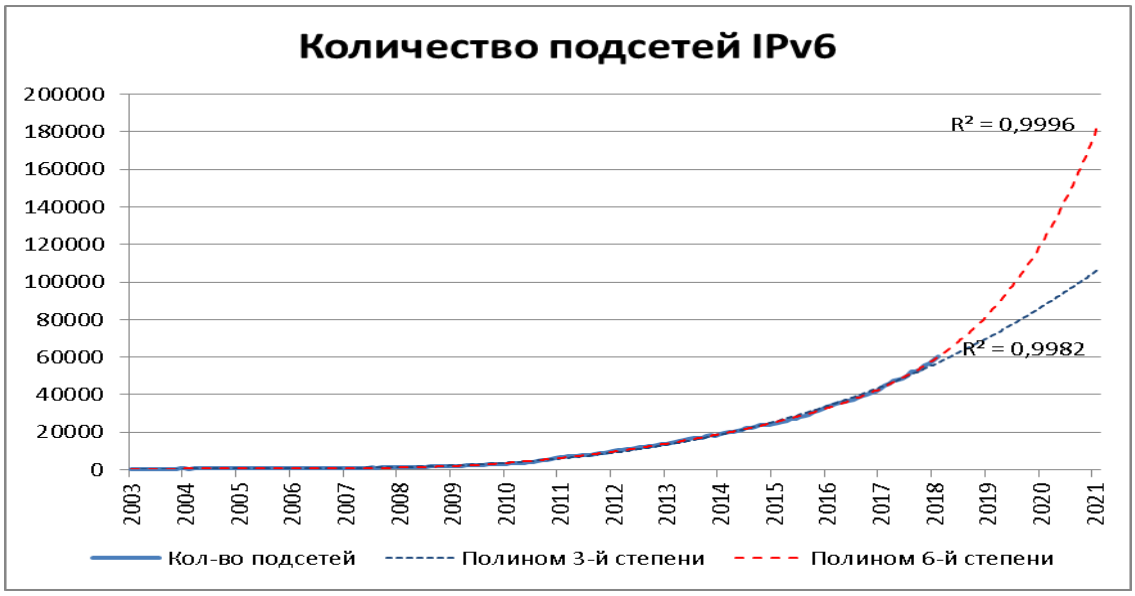


Рисунок 2. Количество подсетей IPv6.

Указанные выше цифры 800 000 и 1 000 000 для протокола IPv4 и 100 000 для IPv6 приведены не случайно. В Научной телекоммуникационной сети Межведомственного суперкомпьютерного центра Российской академии наук (МСЦ РАН) используются маршрутизаторы фирмы Cisco Systems серии Catalyst 6500 с супервизорами SUP-720-3BXL. Как уже было сказано выше, в данных супервизорах объем памяти TCAM позволяет хранить максимально либо 1 000 000 маршрутов IPv4, либо 512 000 маршрутов IPv6 – но не одновременно. В 2014 году, когда размер мировой таблицы маршрутизации для протокола IPv4 превысил 512 000 записей, настройки по-умолчанию на оборудовании сети были изменены на соотношение в 835 584 записей для IPv4 и 106 496 записей для IPv6. Согласно полученным выше данным аппроксимации, данное соотношение является оптимальным с точки зрения максимально продолжительного периода работоспособности маршрутизаторов без переполнения таблиц FIB.

По данным на 14 августа 2018 года, на граничном маршрутизаторе Научной телекоммуникационной сети МСЦ РАН (филиал НИИСИ РАН) число сохраненных записей в таблице FIB было 709 958 для протокола IPv4 и 53 418 – для IPv6, что составляло, соответственно, 85% и 50% от максимально возможных при указанных выше настройках оборудования. Меньший по сравнению с данными проекта Route Views объем таблиц маршрутизации объясняется, как упоминалось выше, его субъективным характером для каждого оператора и маршрутизатора.

Так, в Научной сети МСЦ РАН не принимаются подсети IPv4 с маской более чем /24. Плюс ко всему, согласно исследованиям Geoff Huston из APNIC (Asia Pacific Network Information Centre), таблицы маршрутизации, полученные от участников проекта Route Views в среднем на 30 000 больше, чем от участников проекта RIS, основная часть которых располагается в Европейской зоне [8]. К тому же для данного исследования использовалось общее количество уникальных маршрутов, полученных от разных участников Route Views, при этом у каждого отдельно взятого участника данное число всегда будет меньше, что соответствует, опять же, субъективному характеру данного параметра.

Таким образом, исходя из полученных трендов роста, при условии использования текущего оборудования, уже через год встанет вопрос либо его замены на новые модели, либо поиска вариантов ограничения принимаемых Научной сетью МСЦ РАН маршрутов от вышестоящих операторов связи. Первый вариант потребует значительных затрат на новое телекоммуникационное оборудование, а второй приведет к использованию неоптимальных путей прохождения трафика и повлечет риск недоступности отдельных ресурсов Интернет, как это случилось в 2014 году [4,12]. Изложенная проблема в полной мере затрагивает не только Научную сеть МСЦ РАН, но и всех операторов связи, использующих полноразмерные таблицы маршрутизации.

Статья подготовлена в рамках выполнения государственного задания ФГУ ФНЦ НИИСИ РАН.

On the Growth of the Number of Announced Subnets in the Internet Global Routing Table

A.P.Ovsyannikov, A.A.Sher

Abstract: This article considers the increasing of the the size of the global routing table. The authors attempt to estimate the scale of the growth and its impact for research networks and telecom operators using full-size global routing tables. The article also concerns the impact of the table growth to the stability of the access to the world wide web.

Keywords: routing, Internet, global routing table, BGP, subnets, IPv4, IPv6

Литература

1. Understanding Cisco Express Forwarding (CEF) [электронный ресурс] / Cisco Systems, 2006, URL: https://www.cisco.com/c/en/us/support/docs/routers/12000-series-routers/47321-ciscoef.html (Дата обращения 14.08.2018)

2. Content-addressable memory [электронный ресурс] / Wikipedia, 2018, URL: https://en.wikipedia.org/wiki/Content-addressable_memory#cite_note-4 (дата обращения 14.08.2018)

3. 2017 BGP Table Size Prediction and Potential Impact on Stability of Global Internet Infrastructure [электронный ресурс] / BGP Help, 2017, URL: http://bgphelp.com/2017/01/01/bgpsize/ (Дата обращения 14.08.2018)

4. Border Gateway Protocol [электронный ресурс] / Wikipedia, 2018, URL: https://en.wikipedia.org/wiki/Border_Gateway_Protocol#Routing_table_growth (Дата обращения 14.08.2018)

5. IPv4 address exhaustion [электронный ресурс] / Wikipedia, 2018, URL: https://en.wikipedia.org/wiki/IPv4_address_exhaustion (Дата обращения 14.08.2018)

6. Request for Comments 5375, IPv6 Unicast Address Assignment Considerations [электронный ресурс] / IETF, 2008, URL: https://tools.ietf.org/html/rfc5375#section-3 (Дата обращения 14.08.2018)

7. Request for Comments 6177, IPv6 Address Assignment to End Sites [электронный ресурс] / IETF, 2008, URL: https://tools.ietf.org/html/rfc6177 (Дата обращения 14.08.2018)

8. BGP in 2017 [электронный ресурс] / APNIC, 2018, URL: https://blog.apnic.net/2018/01/10/bgp-in-2017/ (Дата обращения 14.08.2018)

9. University of Oregon Route Views Archive Project [электронный ресурс] / University of Oregon, 2018, URL: http://archive.routeviews.org (Дата обращения 14.08.2018)

10. Bgpdump2: A Tool to Read and Compare the BGP RIB Dump Files. [электронный ресурс] / GitHub, 2016, URL: https://github.com/yasuhiro-ohara-ntt/bgpdump2 (Дата обращения 14.08.2018)

11. MRT format data parser [электронный ресурс] / GitHub, 2018, URL:https://github.com/t2mune/mrtparse (Дата обращения 14.08.2018)

12. Проблемы со связностью интернета из-за разрастания BGP full-view [электронный ресурс] / habr, 2014, URL: https://habr.com/post/233283/ (Дата обращения 14.08.2018)

Разработка представления неформатных документов с автоадаптивным шрифтом

И.Н. Чередниченко [1], К.Ю.Юдинцев [2]

Межведомственный суперкомпьютерный центр РАН -

филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1]inch@jscc.ru, [2] climenty@jscc.ru

Аннотация: Рассматривается задача приведения результатов работы процедуры ретроконверсии (восстановления) сканированных неформатных документов (рукопечатных или выполненных с использованием нерегулярных шрифтов) к современным веб-стандартам для задач формирования и использования цифровых фондов электронных библиотек. Описываются конечные стадии процедуры ретроконверсии и предлагается формат на основе JSON-стандарта для интеграции с веб-браузерами, поддерживающими стандарт HTML-5.

Ключевые слова: ретроконверсия, автоадаптивные шрифты, электронная библиотека, JSON, Фурье.

Введение

Хранение и представление данных в электронных библиотеках с веб-доступом пользователей, требуют новых подходов и развития существующих технологий работы с информацией. Одна из таких технологий - технология автоадаптивных шрифтов создавалась для решения задач обработки, ретроконверсии, хранения и веб-представления неформатных графических документов. Решения задач обработки и ретроконверсии уже подробно рассматривались в предыдущих работах [1], а вот соответствие стандартам и форматам представления выходных данных было уделено мало внимания.

В основу технологии формирования выходного документа, с использованием технологии автоадаптивного шрифта положены процедуры разбиения изображения на отдельные графические объекты документа, полученного в процессе сканирования, анализ графических объектов входного изображения неформатного, их автоматической кластеризации с учетом степени похожести, а также динамическое формирование характеристических представителей кластеров - суть элементов автоадаптивного шрифта.

Исходными положениями для построения и исследования процедур, применяемых в методе автоадаптивных шрифтов, выступает модель АВО (Алгоритм вычисления оценок), предложенная Ю.И. Журавлевым [2], в рамках которой графические объекты определяются векторами признаков фиксированной размерности, которые в совокупности образуют конечномерное векторное пространство. Каждому кластеру ставится в соответствие характеристический вектор той же размерности, компоненты которого отражают усредненную характеристику близости объединенных в кластер объектов. В качестве меры близости (функции оценок) для них может быть выбрана приемлемая для предметной области метрика, отвечающая интуитивному понятию похожести. Решающее правило о принятии решения об отнесении объекта к тому или иному кластеру имеет в своей основе оценку отклонения значений соответствующих компонент вектора признаков и характеристического вектора, которое должно не превышать некоторого заданного порога. Таким образом, основными составляющими модели распознавания являются векторы признаков, характеристические векторы классов объектов, функция оценок и пороги. Однако в выходных документах, подготовленных по технологии автоадаптивных шрифтов, содержится только часть этой конечной информации, которая необходима для полноценного восстановления вида исходного изображения.

Поскольку в качестве конечной цели обработки выходные документы с автоадаптивным шрифтом предполагается ретроконверсия, а ее результат в виде электронного документа нужно размещать на веб-сайтах, было проведено исследование форматов удобного представления данных, совместимых с современной средой отображения информации в браузерах.

Стандарт HTML-5 представляет собой язык для структурирования и представления содержимого веб-документов. Стандарт был окончательно сформирован и завершен только в 2014 году [3; 4], но уже с 2013 года частично поддерживался браузерами, а разработчики использовали его в качестве перспективного рабочего стандарта. Основной целью разработки HTML-5 было улучшение уровня поддержки мультимедиа-технологий с одновременным сохранением обратной совместимости, удобочитаемости кода для человека и простоты анализа для парсеров – программ, выполняющих автоматический разбор входной информации.

На данный момент можно рассматривать HTML-5 как совокупность современных технологий и стандартов, применяемых для разработки WEB-приложений. Одной из ключевых особенностей является поддержка языка JavaScript. С его помощью реализуется обращение к внутреннему API, реализованному в стандарте HTML-5. На данный момент уже действует стандарт HTML-5.1 [5].

В рамках языка JavaScript был разработан текстовый формат обмена данными JSON (JavaScript Object Notation). Как и многие другие текстовые форматы, JSON легко читаем. Формат был разработан Дугласом Крокфордом [6]. Несмотря на происхождение от JavaScript, формат считается независимым от языка и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания, обработки и конвертации данных в формате JSON. Из-за всех этих описанных выше особенностей, была поставлена задача скорректировать формат выходных документов с автоадаптивным шрифтом, для совместимости со стандартом JSON в целях более легкого интегрирования таких документов в среду электронных библиотек.

Описание задачи

Как уже упоминалось выше, метод автоадаптивных шрифтов был создан для обработки, представления и процедуры ретроконверсии неформатных графических документов [1]. Под термином “неформатные” документы понимаются изображения сканированных исторически значимых и библиотечных документов, которые плохо поддаются обработке современными программами оптического распознавания текстов из-за особенностей оформления и старинных наборов шрифтов. Метод основан на контурном анализе графических объектов исходного изображения документа и автоматической кластеризации всех таких объектов, входящих в документ. Так как в нашем случае множество подлежащих кластеризации объектов пополняется случайным образом, то для построения алгоритма использовалась адаптивная модель АВО (алгоритм вычисления оценок) [2], которая была дополнена динамической процедурой формирования компонент характеристических векторов и построения функции оценок и порогов на основе статистической обработки случайного входного потока сканированных объектов, подлежащих кластеризации.

Все исследуемые объекты математически описываются в виде набора контуров. Каждый контур представляет собой множество точек, принадлежащих параметрической кривой, заданной вектор-функцией:

$$W(t)=\big(x(t),y(t)\big),\qquad t=1,2\ldots n$$

Для корректной работы алгоритмов требовалось решить проблемы построения такого описания графических объектов, которое инвариантно относительно процедур сдвига и решить проблему масштабирования, поскольку модель АВО предполагает манипуляции с входными векторами признаков, имеющими фиксированную размерность. При решении задачи нормализации описания всех рассматриваемых графических объектов, был использован переход в Фурье пространство и далее все операции проводились с Фурье-образами объектов:

$$\vec{v}=\big(b_1,\ldots,b_i\big),\qquad i=1,2\ldots n,$$

где b_i есть коэффициенты разложения Фурье для кусочно-линейной функции:

$$b_k=\frac{(-1)^k 2f(N)}{k\pi}+\frac{2N}{k^2\pi^2}\sum_{i=0}^{N-1} f_i(\sin kt'_{i+1}-\sin kt'_i)$$

Такие сформированные нормализованные вектора признаков поступали на вход алгоритма формирования автоадаптивного шрифта и над ними проводились операции автоматической кластеризации и формирования характеристических векторов кластеров.

Таким образом, выходной элемент полученного автоадаптивного шрифта

содержит информацию о количестве контуров в символе, смещение каждого контура относительно точки привязки и набор Фурье компонент, достаточный для восстановления этого контура с заданной точностью.

Процедура восстановления или другими словами – конечного этапа ретроконверсии, описывается формулой:

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{N} b_k \sin kt$$

Таким образом, выходная информация о документе после процедуры ретроконверсии содержит помимо информации о словаре символов еще и данные о координатах каждого символа и ссылке на элемент словаря.

Структура словаря

Чтобы представить один элемент словаря в стандарте JSON, была выбрана следующая структура:

```
{  "Shift": [0,0],
   "Fill": "black",
   "xF":[40.66,…],
   "yF":[87.0,…]       },
 { "Shift": [30,10],
   "Fill": "white",
   "xF":[24.0,…],
   "yF":[40.69,…]      },
```

где поле ***"Shift"*** содержит информацию о координатах смещения данного контура относительно точки привязки символа, поле ***"Fill"*** фактически содержит информацию о том, является ли данный контур внешним или внутренним, а массивы ***"xF"*** и ***"yF"*** содержат значения Фурье компонент, необходимых для восстановления контуров с заданной точностью.

Сам словарь представляется массивом элементов с необходимыми дополнительными данными (см. Рис 1).

Поле ***"nSymb"*** содержит информацию о количестве элементов в словаре, а ***"lSymb"*** – количество компонент Фурье разложения.

Такой подход к организации информации, содержащейся в словаре, позволяет не накладывать ограничения ни на количество элементов словаря, ни на количество элементов разложения Фурье, определяющих конечную точность восстановления исходной информации.

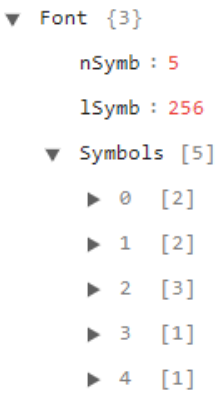


Рис.1. JSON-структура словаря.

Структуры представления страниц электронного документа

Как уже было сказано выше, в результате работы алгоритма автоадаптивного шрифта, на выходе получается, что каждый отдельный графический элемент, попавший в словарь, представляется набором из трех чисел.

Во-первых, это ***x*** и ***y*** координаты расположения символа на странице. Эти координаты представляют из себя целые числа и позиционируют с точностью до пикселя расположение стартовой точки или точки привязки этого символа на странице.

Рис.2. JSON-структура страниц.

Во-вторых, это ссылка на номер элемента словаря, в котором содержится

информация, необходимая для восстановления символа.

Этих данных вполне достаточно для работы алгоритма ретроконверсии исходного неформатного графического документа. Однако, в документе может быть гораздо больше чем одна страница. Именно из этих соображений была введена JSON-структура, отвечающая за массив страниц (см. Рис. 2).

В приведенном примере описывается структура документа, в котором содержится одна страница с шестью графическими символами, которые описаны в словаре автоадаптивного документа.

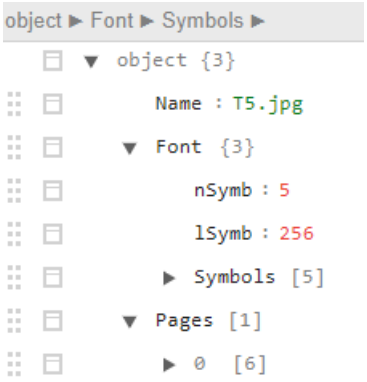


Рис.3. JSON-формат документа с автоадаптивным шрифтом.

Для большей наглядности, приведем традиционную блок-схему выходного документа с автоадаптивным шрифтом, сформированного с учетом стандартов JSON:

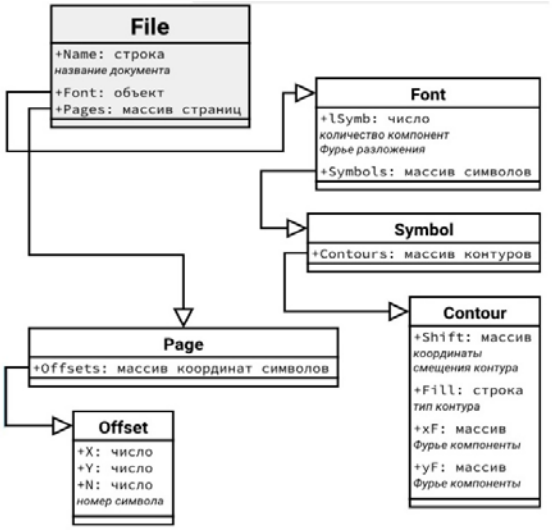


Рис.4. Блок схема формирования выходного документа с автоадаптивным шрифтом.

Особенности реализации процедуры ретроконверсии

Полученные в результате обработки выходные документы с автоадаптивным шрифтом, предназначены для размещения на веб-сайтах. В связи с современными тенденциями распределенной обработки информации, было принято решение о том, что финишная процедура ретроконверсии, основанная на обратном Фурье-преобразовании, будет выполняться на клиентской машине в локальной среде веб-браузера.

Восстановление содержимого происходит динамически с помощью программы на языке JavaScript, встроенном в современные веб браузеры, и не требует установки дополнительных плагинов. Так как формат JSON представляет данные в текстовом виде, то передаваемые структуры могут быть разобраны посредством встроенных в язык функций или иным способом, соблюдающим синтаксис формата.

После получения данных в вышеописанном формате, выполняется восстановление контуров символов с заданной точностью. В нём каждый символ представлен набором значений достаточных для его отображения посредством линейных примитивов. Затем выделяется указатель контекста рисования на область страницы, зарезервированную для отображения документа. И эта область заполняется последовательностью символов, соответственно полученному описанию текущей странице документа. При переходе на новую страницу, область рисования очищается, а процесс заполнения повторяется на основе информации о текущей странице.

В этом же подходе практически реализована и возможность управления качеством выходного документа, отображаемого в браузере клиента. При изменении количества Фурье-компонент, которые используются в восстановлении документа, будет изменяться и качество восстановления:

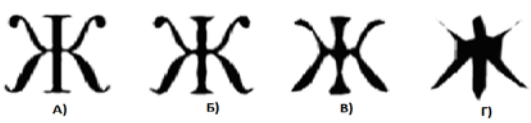


Рис. 5. Изменение качества восстановления.

На рисунке 5 показана зависимость результата работы алгоритма ретроконверсии от использованного количества Фурье компонент:

А) Число использованных компонент n=256; Б) Число использованных компонент n=128; В) Число использованных компонент n=64; Г) Число использованных компонент n=16;

При этом, соответственно, уменьшается и размер выходного документа.

Заключение

В результате приведения выходного стандарта документа с автоадаптивным шрифтом к современным стандартам, принятым в веб-технологиях, решается задача комплексной интеграции таких документов с интерфейсами, используемыми в среде разработки электронных библиотек.

Работа отражает результаты исследований на стыке нескольких тем, проводимых в МСЦ РАН, филиале ФГУ ФНЦ НИИСИ РАН по проекту РФФИ № 17-07-00400 и Государственному заданию по теме «Исследование и разработка технологий, методов и средств формирования и использования интегрированных электронных информационных ресурсов» (№ государственной регистрации 01201464391).

В работе использовались вычислительные ресурсы МСЦ РАН.

Development of non-format documents submission with auto-adaptive font.

I.N. Cherednichenko, C.U. Yudintcev

Abstract: The article considers solving of the task of retroconversion operation for scanned non-format documents (hand-printed or executed using irregular fonts) to modern web standards for the tasks of forming digital collections of digital libraries. The final stages of the retroconversion procedure are described and a JSON-based format is proposed for integration with web browsers that support the HTML-5 standard.

Keywords: retroconversion, auto-adaptive fonts, electronic library, JSON, Fourier.

Литература

1. А.Н.Сотников, И.Н.Чередниченко. Построение автоадаптивного фонта в документах электронных библиотек. Тверь: Программные продукты и системы, №2(82), 2008 г., стр. 16-19. ISSN 0236-235X.

2. Ю.И.Журавлев. Об алгебраическом подходе к решению задач распознавания и классификации. Вып. 33, М : Проблемы кибернетики., 1978 г., стр. 5–68.

3. HTML5 IS A W3C RECOMMENDATION. https://www.w3.org/blog/news/archives/4167. [В Интернете]

4. Open Web Platform Milestone Achieved with HTML5 Recommendation. https://www.w3.org/2014/10/html5-rec.html.en. [В Интернете]

5. https://www.w3.org/TR/html51/. www.w3.org. [В Интернете] 2018 г.

6. JSON Redux AKA RFC7159. https://www.tbray.org/ongoing/When/201x/2014/03/05/RFC7159-JSON. [В Интернете]

Технология сбора данных самоконтроля для программ параллельной обработки сигналов в мультипроцессорных комплексах реального времени

Т.К. Грингауз[1], А.Н. Онин[2]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1]gring@niisi.ras.ru, [2]alexii@niisi.ras.ru

Аннотация: Рассматриваются мультипроцессорные комплексы реального времени на базе процессоров с архитектурой КОМДИВ, функционирующих в коммутируемой коммуникационной среде RapidIO [1]. Мультипроцессорные приложения разрабатываются на основе библиотеки параллельной обработки сигналов (БПОС) [2]. Для самоконтроля системы целесообразно в цикле вычислительного конвейера анализировать измерения параметров, характеризующих состояние объектов БПОС. Результаты измерений, осуществляемых на локальных процессорах, должны собираться на едином аккумулирующем процессоре на каждой итерации цикла. Разработана технология, основанная на использовании транзакций типа NWRITE в среде RapidIO в качестве транспортного механизма.

Ключевые слова: самоконтроль системы, библиотека параллельной обработки сигналов, вычислительная стадия, группа процессоров, поток данных, таймер, порт потока, стадия-аккумулятор, функция-обработчик, коммуникационная среда RapidIO, транзакция типа NWRITE.

1. Введение

Самоконтроль системы – это один из возможных вариантов реализации принципа контролируемого выполнения сложных систем [3], цель которого – обеспечение выполнения системой своей миссии в нештатных условиях функционирования (при наличии внутренних ошибок или неблагоприятных внешних воздействий). Согласно [3], «самоконтроль состоит в проверке значений параметров, отражающих уровень безопасности состояния, в котором находится система, и реагировании в случае, если состояние оказалось небезопасным (некорректным)». Будем называть параметры, подлежащие проверке, «параметрами самоконтроля». Результаты измерений параметров самоконтроля в процессе функционирования системы будем называть «данными самоконтроля». Критерий безопасности системы можно себе представить как булеву функцию на множестве данных самоконтроля. Критериев безопасности может быть несколько, и проверяться они могут в разных местах программы. В общем виде схема самоконтроля такова. В процессе функционирования система осуществляет измерение своих параметров самоконтроля. С заданной частотой осуществляется сбор накопленных данных самоконтроля, проверка критериев безопасности состояния системы, выработка реакции в случае нарушения безопасности. Механизмы измерения параметров самоконтроля, сбора данных самоконтроля, проверки критериев безопасности, выработки реакции встраиваются в систему на этапе ее проектирования.

В статье рассматривается перспектива введения самоконтроля в мультипроцессорные комплексы реального времени, создаваемые на основе аппаратных и программных средств, разрабатываемых в НИИСИ РАН [4,5].

В качестве аппаратной платформы таких комплексов используются процессоры архитектуры КОМДИВ, объединенные коммутируемой коммуникационной средой RapidIO [1].

Комплексы функционируют под управлением операционной системы реального времени семейства ОС РВ Багет [6] (далее – «ОС РВ»).

Рекомендуемая НИИСИ РАН технология разработки функционального программного обеспечения (ФПО) комплексов основана на применении библиотеки параллельной обработки сигналов (БПОС) [2], реализующей парадигму параллельного программирования.

В модели вычислений БПОС задача цифровой обработки сигналов представляется в виде последовательности стадий вычислительного конвейера, принимающих и передающих потоки данных, поступающих с фиксированной частотой.

В статье предлагается подход к созданию самоконтролируемых программ с архитектурой БПОС, при котором сбор и обработка данных

самоконтроля встраиваются в цикл вычислительного конвейера. Параметры самоконтроля измеряются в процессе выполнения итерации цикла на каждом процессоре. По завершении итерации на локальном процессоре накопленные за итерацию данные самоконтроля передаются на выделенный «аккумулирующий» процессор, выполняющий специальную стадию-аккумулятор. Стадия-аккумулятор на каждой итерации принимает данные самоконтроля от всех процессоров, осуществляет их семантический разбор и накопление в «структуре метаданных», заполняемой по принципу кольцевого буфера. Доступ к метаданным предоставляется посредством пользовательских функций-обработчиков, вызываемых стадией-аккумулятором при положительном результате проверки предопределенных условий в предопределенных местах стадии-аккумулятора. Для привязки функции-обработчика к проверяемому условию используется специальный механизм «регистрации» пользовательских обработчиков. Функция-обработчик выполняется в контексте потока управления, выполняющего стадию-аккумулятор. В рамках обработчика событий пользователь может проинтерпретировать данные самоконтроля, вычислить значение критерия безопасности и реализовать соответствующую реакцию системы.

Центральное звено изложенного подхода - этап сбора и передачи локальных данных самоконтроля на аккумулирующий процессор. В статье описан пример его реализации - программа «Сбор и обработка статистических данных» (далее – «СОСД»). В состав СОСД входят библиотека интерфейсных функций для сбора данных самоконтроля и типовая стадия-аккумулятор. Для передачи данных самоконтроля на аккумулирующий процессор используется специфический протокол, основанный на транзакциях типа NWRITE в среде RapidIO [1]. В статье описаны принципы реализации предлагаемой технологии.

2. Краткая справка по архитектуре и принципу функционирования БПОС

Ниже приводятся сведения о БПОС в объеме, необходимом для понимания описываемой технологии.

В соответствии с [2], основные объекты БПОС - это стадии, потоки, группы процессоров.

Вычислительная стадия определяется как совокупность следующих сущностей: множество входов для приема данных («входные потоки»), множество выходов для отправки данных («выходные потоки»), алгоритм преобразования входных данных в выходные. Под «потоком» понимается объект, описывающий однонаправленную передачу «отсчетов» (трехмерных массивов фиксированной размерности) от стадии-источника к стадии-приемнику на каждой итерации цикла с разбиением массива на источнике или на приемнике по указанной пользователем размерности. Стадия может выполняться параллельно на нескольких процессорах, на одном процессоре могут выполняться несколько стадий. Процессоры разбиваются на группы так, что на всех процессорах группы выполняется один и тот же набор стадий. Количество процессоров в разных группах может отличаться.

В формализме БПОС процессор идентифицируется «логическим номером», т.е. сквозным номером в упорядоченном множестве {{<номер группы>, <номер процессора в группе>}}. Средства ОС РВ обеспечивают соответствие логических номеров и идентификаторов процессоров в среде RapidIO [1] (РИО-идентификаторов) посредством функции mp_init(), входящей в состав ОС РВ.

Технология БПОС позволяет создавать унифицированный код мультипроцессорной программы.

На все процессоры системы загружается одна и та же программа, содержащая код всех вычислительных стадий. Состав и последовательность стадий, исполняемых на локальном процессоре, определяются динамически на основании уникального идентификатора процессора.

В каждой итерации цикла на локальном процессоре перебирается последовательность локальных стадий, и для каждой из них выполняется одна итерация стадии. Все локальные стадии выполняются в едином потоке управления операционной системы.

Логика работы вычислительной стадии определяется ее методами. К обязательным методам относятся:

- init (инициализация стадии). Метод вызывается только для локальных стадий,
- config (конфигурирование выходных потоков). Метод вызывается на всех процессорах независимо от того, является ли стадия локальной,
- iter (выполнение одной итерации стадии). Метод вызывается только для локальных стадий,
- fini (завершение работы стадии). Метод выполняется для локальных стадий по завершении всех итераций.

На каждой итерации конвейерного цикла стадия выполняет типовой набор операций:

прием отсчета из каждого входного потока, обработка принятых отсчетов, отправка результатов обработки в выходные потоки. Порция отсчета, принимаемая на процессоре или отправляемая с процессора, называется локальным отсчетом. Локальные отсчеты буферизуются: с потоком связаны очередь входных отсчетов и очередь выходных отсчетов. Схема приема данных: захват буфера входного потока, обработка локального отсчета, сдача буфера в очередь входных отсчетов. Если разбиение отсчета на стадии-источнике не совпадает с разбиением на стадии-приемнике (далее – «нетривиальное разбиение»), поток осуществляет сборку данных во входном отсчете на стадии-приемнике (данные копируются из промежуточных буферов в результирующий массив). Схема передачи данных: захват свободного буфера выходного потока, копирование локального отсчета в буфер, сдача буфера в очередь выходных отсчетов. Захват и сдача буферов осуществляются интерфейсными функциями БПОС, копирование данных при нетривиальном разбиении скрыто от прикладных программистов.

Поскольку вычислительная стадия может исполняться параллельно на нескольких процессорах, поток, соединяющий две стадии, определяет множество параллельно исполняемых передач «точка-точка» между парами процессоров. Для программной реализации таких передач в ОС РВ используются специфические каналы передачи сообщений mp (message passing). В качестве механизма передачи сообщений по RapidIO используются транзакции типа MESSAGE [1]. Для отправки и приема сообщений по mp-каналу на процессоре-источнике и на процессоре-приемнике создаются структуры данных - «порты». Порт содержит кольцо буферов для отправляемых (принимаемых) сообщений. Таким образом, с потоком БПОС связаны два множества портов (входных и выходных). Поток БПОС предоставляет интерфейс для приема и передачи данных, скрывающий обращение к интерфейсу mp-каналов, но информация о mp-портах потока может быть получена с помощью функций БПОС.

3. Предопределенные параметры самоконтроля и средства их измерения, встроенные в БПОС

БПОС в процессе функционирования производит измерение временных характеристик, позволяющих следить за ходом выполнения программы, а также позволяет опрашивать состояние очередей входных и выходных потоков данных на локальном процессоре. Такие характеристики могут быть положены в основу набора параметров самоконтроля программы. Для хронометража выполнения задачи в БПОС предназначены специальные объекты – «группы таймеров». Одна или несколько групп таймеров могут быть привязаны к стадии БПОС. Таймеры в группе идентифицируются номером и символическим именем. Указатели на группы таймеров сохраняются в структурах стадий БПОС, показания таймера доступны через интерфейсные функции БПОС. Таймер можно активировать и останавливать многократно. Время каждого периода активности таймера обрабатывается БПОС, результаты обработки сохраняются. Посредством интерфейсных функций БПОС доступны следующие показания таймера:

- общее время активности (сумма всех периодов активности),
- минимальное и максимальное значения периодов активности,
- номера итераций, для которых были зарегистрированы минимальное и максимальное времена активности.

Примечание. В течение некоторого числа начальных («разогревочных») итераций статистическая обработка показаний таймеров не производится. Группы таймеров могут создаваться в пользовательской программе для хронометража произвольных участков кода (далее - «пользовательские таймеры»). Кроме того, в БПОС предопределены таймеры для измерения специфического набора времен, связанных с функционированием основных объектов БПОС (далее – «системные таймеры»). Создание, активация и останов таких таймеров происходит автоматически (для этого не нужно писать специальный код в пользовательской программе). Посредством системных таймеров измеряются следующие времена:

- время исполнения итерации стадии на процессоре,
- общее время исполнения итерации всех локальных стадий на процессоре,
- времена, связанные с входными и выходными потоками локальных стадий.

С входным потоком связаны таймеры для измерения следующих времен:

- время ожидания данных;
- время сдачи обработанного буфера;
- время, затраченное на копирование данных при использовании нетривиального разбиения потока.

С выходным потоком связаны таймеры для измерения следующих времен:

- время ожидания свободного буфера;

- время сдачи обработанного буфера.

Посредством интерфейсных функций БПОС доступна следующая диагностическая информация о портах входных и выходных потоков:

- число сбоев при передаче данных по заданному порту;
- номер последнего успешно переданного отсчета;
- число буферов с доступными для чтения данными (для входного порта),
- число свободных буферов (для выходного порта).

Показания таймеров (системных и пользовательских), диагностическую информацию о портах потоков БПОС, а также значение системной переменной «errno» (код ошибки) будем далее называть «системными» данными самоконтроля.

4. Принципы реализации технологии сбора данных самоконтроля

4.1 Поддерживаемые типы данных самоконтроля. Системные типы данных

Библиотека интерфейсных функций СОСД позволяет собирать и передавать на аккумулирующий процессор «системные» и «пользовательские» данные. Тип данных определяет их структуру, семантику, способ сбора и форматирования на локальном процессоре для передачи на аккумулирующий процессор, способ интерпретации и обработки переданных данных на аккумулирующем процессоре. Для интерпретации данных, переданных на аккумулирующий процессор, необходимо передать информацию об их типе. Протокол передачи и формат передаваемых данных основаны на специфических алгоритмах маркировки данных при пересылке и их идентификации на приемной стороне. Алгоритмы для системных и пользовательских типов данных различаются.

4.1.1 Системные типы данных

К системным типам данных относятся:

- данные таймера БПОС,
- данные порта потока БПОС,
- значение системной переменной «errno».

Семантика типов «данные таймера» и «данные порта» описана в разделе 3 настоящей статьи.

Значение переменной «errno» - код системной ошибки, передаваемый на аккумулирующий процессор с каждым пакетом NWRITE (см. раздел 4.3.3 настоящей статьи).

Системные типы данных маркируются при пересылке и распознаются стадией-аккумулятором автоматически, без дополнительного кода в пользовательской программе. Для этого при реализации СОСД использован следующий подход.

Структуры локальных стадий и потоков БПОС к моменту завершения методов config() и init() на локальном процессоре содержат информацию, на основании которой можно сформировать «статические данные» для всех таймеров и для всех mp-портов процессора. «Статические данные» - это структура, содержащая идентифицирующую, семантическую и другую информацию, необходимую для обработки «динамических» данных таймера или порта. Под динамическими данными имеются в виду значения параметров самоконтроля, измеренные во время исполнения итерации. Для отправки данных с локального процессора на аккумулирующий процессор предназначена функция СОСД statistic_send . На первой итерации после «разогрева» БПОС функция формирует два массива статических данных (один для локальных таймеров, другой для локальных портов) и отправляет их на аккумулирующий процессор. Стадия-аккумулятор, получив статические данные таймеров и портов процессора-отправителя, записывает их в структуру метаданных, соответствующую этому процессору. При формировании массивов динамических данных, отправляемых с локального процессора, поддерживается тот же порядок таймеров (или портов), что и в статических массивах. Данные таймера или порта, переданные в массиве динамических данных, интерпретируются стадией-аккумулятором в соответствии с его статическими данными. Ключ для поиска в массиве статических данных на аккумулирующем процессоре - индекс элемента в массиве динамических данных.

Для работы с системными данными стадия-аккумулятор поддерживает «структуры метаданных», описываемые типом:

```
typedef struct struct_proc_info statistic_info;
```

Структура метаданных создается и поддерживается для каждого процессора, передавшего данные на аккумулирующий процессор. Структура содержит следующую информацию:

- значение переменной errno, полученное в заголовке последнего на текущий момент пакета, принятого от процессора,
- таблицу данных таймеров процессора,
- таблицу данных портов процессора,

- служебную информацию.

Каждая строка таблицы таймеров (портов) предназначена для хранения данных, накопленных за одну итерацию БПОС на локальном процессоре. Таблицы данных таймеров и портов рассчитаны на хранение данных 1000 итераций. Таблицы заполняются по принципу кольцевого буфера по мере поступления системных данных на аккумулирующий процессор.

Доступ пользователя к системным данным осуществляется в рамках «обработчиков событий» (механизм событий описан в разделе 4.2 настоящей статьи). Для интерпретации метаданных предназначены интерфейсные функции СОСД (см. раздел 4.4 настоящей статьи).

4.1.2 Пользовательские типы данных

«Пользовательские данные» - это данные, семантика и структура которых определяются разработчиком пользовательской программы.

В пользовательской программе может быть определено не более 1000 пользовательских типов данных. Данные одного типа могут создаваться в разных местах программы. Пользовательский тип данных характеризуется совокупностью следующих сущностей: имя типа, длина данных в байтах, функция-обработчик типа данных. Имя типа - это строка символов (не более 12 байт). Имя типа должно быть уникальным. Длина данных не должна превышать 232 байта. Для пользовательского типа пользователь должен реализовать функцию-обработчик с прототипом:

```
void *handler (statistic_data* sdata, void*
data, unsigned int length);
```

где

- sdata – указатель на структуру – «описатель типа»;
- data – данные пользователя, выровненные на 8 байт.
- length – длина данные пользователя в байтах.

Обработчик пользовательского типа данных вызывается стадией-аккумулятором в момент обнаружения данных этого типа в составе данных, поступивших на аккумулирующий процессор. Обработчик предназначен для интерпретации данных указанной длины в соответствии с их типом и для обработки их на аккумулирующем процессоре.

Пользовательские типы данных подлежат «регистрации», т.е. внесению в специальную таблицу (массив структур типа statistic_data) с присвоением уникального цифрового идентификатора.

Для регистрации типа предназначена интерфейсная функция statistic_register_data. Функция присваивает регистрируемому типу уникальный цифровой идентификатор, записывает его, а также имя, длину и указатель на функцию-обработчика в строку таблицы (далее - «описатель типа»), возвращает указатель на описатель типа.

Для присвоения цифрового идентификатора используется однозначное преобразование 12-символьного имени в пару целых чисел.

При попытке повторного использования имени типа данных регистрация не выполняется, в консоль процессора выводится сообщение об ошибке.

Регистрация пользовательского типа данных должна быть выполнена на каждом процессоре, независимо от того, используется ли этот тип в локальных стадиях.

Все пользовательские типы должны быть зарегистрированы до начала исполнения цикла итераций вычислительного конвейера БПОС. Рекомендуется регистрировать пользовательские типы данных в головной функции пользователя main() или в методе config() стадии БПОС.

На каждом процессоре создается своя таблица описателей типа, содержащая записи всех зарегистрированных пользовательских типов.

Указатели на описатель типа данных и на функцию-обработчика для одного и того же типа на разных процессорах различаются, цифровой идентификатор неизменен.

В связи с этим указатель на описатель типа используется для идентификации типа на локальном процессоре, цифровой идентификатор – для маркировки пересылаемых пользовательских данных и для опознания их на аккумулирующем процессоре.

Работа с пользовательскими данными происходит следующим образом.

Локальная стадия, создавшая данные зарегистрированного типа, подготавливает их к отправке на аккумулирующий процессор путем вызова функции statistic_add_data, в которую передаются указатель на описатель типа и указатель на данные.

Функция форматирует отправляемые данные, маркируя их цифровым идентификатором, записанным в описателе типа.

Стадия-аккумулятор, обнаруживает входные данные, маркированные цифровым идентификатором пользовательского типа.

По цифровому идентификатору стадия-аккумулятор находит строку в таблице описателей типа на своем процессоре и

вызывает обработчик данных по указателю, записанному в описателе типа.

В контексте обработчика пользовательского типа данных можно получить указатель на структуру метаданных процессора и таким образом «привязать» системные данные к пользовательским.

4.2 Доступ к принятым данным самоконтроля на аккумулирующем процессоре. События стадии-аккумулятора

На аккумулирующем процессоре выполняется стадия-аккумулятор accum_stat. Стадия выполняет сбор и обработку данных самоконтроля, полученных от всех процессоров, выполняющих программу на основе БПОС. Стадия-аккумулятор обеспечивает следующие виды доступа к данным самоконтроля:

- доступ к пользовательским данным и к структуре метаданных процессора в рамках обработчика пользовательского типа данных,
- опциональный вывод в консоль аккумулирующего процессора таблицы с результатами статистической обработки системных данных (частота вывода таблицы конфигурируется),
- доступ к данным в рамках обработчиков событий.

Событие - это предопределенное условие, проверяемое в предопределенной точке стадии-аккумулятора. При выполнении условия («при наступлении события») вызывается специальная функция-обработчик, связанная с этим событием. Пользователь должен написать реализацию функции-обработчика события. Типы поддерживаемых событий определяются типом перечисления StatisticTypeEvent. Ниже приведен фрагмент перечисления:

```
typedef enum
{
 STATISTIC_EVENT_REGISTRATION,
// первый приход данных от процессора

STATISTIC_EVENT_ITERATION_RELEASED,
//завершение очередной итерации на всех процессорах, передающих данные самоконтроля
            …
 STATISTIC_EVENT_NO_DATA,
//от процессора перестали поступать данные
 STATISTIC_EVENT_PORT_FAIL,
//ошибка передачи данных по mp-каналу потока БПОС
 STATISTIC_EVENT_ALL_BUSY,
//заняты все выходные буфера порта
 STATISTIC_EVENT_TRANSFER_SLOWLY,
//на стороне отправителя свободно не более одного выходного буфера, на стороне приемника свободны все входные буфера
 STATISTIC_EVENT_ERRNO,
// изменение значения переменной errno на процессоре, передавшем данные самоконтроля
 STATISTIC_EVENT_OVERTIME,
// процессор тратит на счет более 99% времени
 STATISTIC_EVENT_DATA_LOST,
// потерян пакет NWRITE
 STATISTIC_EVENT_USER_DATA_LOST,
// потерян пакет NWRITE с данными пользователя
 STATISTIC_EVENT_DATA_INVALID,
// переданные данные недействительны
 STATISTIC_EVENT_FLOW_IRREGULARY,
// «неравномерность» потока БПОС: в одном порту все буфера заняты, в другом – все свободны
            …
 ,
} StatisticTypeEvent;
```

Привязка обработчика события к номеру события осуществляется путем вызова функции регистрации обработчика события statistic_set_handler.

Эта функция должна быть вызвана на аккумулирующем процессоре, например, в головной функции пользователя main.

При регистрации необходимо указать пользовательскую функцию-обработчик и соответствующее ей тип события из перечисления StatisticTypeEvent.

На все перечисленные события, за исключением STATISTIC_EVENT_ITERATION_RELEASED, определены обработчики по умолчанию. Обработчики, установленные по умолчанию, выводят в консоль аккумулирующего процессора диагностическую информацию о событии.

Пользователь может переопределить обработчики, установленные по умолчанию, с помощью функции statistic_set_handler.

Каждому типу события соответствует свой прототип функции-обработчика.

Все перечисленные события, за исключением STATISTIC_EVENT_ITERATION_RELEASED, информируют о локальных ситуациях, сложившихся на процессоре, передавшем данные. В связи с этим независимо от типа события функции-обработчику в качестве первого входного параметра стадия-аккумулятор передает указатель на структуру метаданных процессора, к которому событие относится.

Событие STATISTIC_EVENT_ITERATION_RELEASED

информирует о глобальной ситуации. Прототип его функции-обработчика имеет вид:

void handler (statistic_info* infos[], unsigned int count, unsigned int iter);

где

infos – массив указателей на структуры метаданных процессоров;

count – количество процессоров, от которых были получены данные самоконтроля;

iter – номер итерации, на которой произошло событие.

В обработчике данного события пользователь может осуществить совместную обработку данных, переданных с разных процессоров, с упорядочением их по номеру итерации. Интерпретация метаданных в функции-обработчике осуществляется посредством интерфейсных функций СОСД.

Примечание 1. Массив infos в некоторых позициях вместо указателя на структуру statistic_info может содержать значение «ноль». Это означает, что от процессора, соответствующего данной позиции, не были получены данные самоконтроля. Для поиска процессоров, от которых приходили данные, необходим перебор элементов массива infos с пропуском нулевых элементов до тех пор, пока количество обнаруженных отправителей не совпадет со значением count.

Примечание 2. События обрабатываются в потоке управления, выполняющем стадию accum_stat. Прикладной программист должен обеспечить такое время выполнения функции-обработчика, чтобы стадия-аккумулятор успевала вычитывать поступающие данные.

4.3 Тракт данных самоконтроля

4.3.1 Сбор данных на локальном процессоре

Данные, подлежащие отправке на аккумулирующий процессор, на локальном процессоре накапливаются в двух буферах: буфер системных данных и буфер пользовательских данных. Пользовательские данные записываются в свой выходной буфер в процессе выполнения итерации конвейера БПОС, системные данные накапливаются в структурах БПОС и по завершении итерации записываются в выходной буфер системных данных.

Для записи данных в область отправки используются интерфейсные функции СОСД. Пользовательские данные записываются посредством функции statistic_add_data, которая вызывается из программы пользователя. Системные данные записываются посредством функции statistic_send(), которая выполняет следующие действия:

- переписывает системные данные в предопределенной последовательности из структур БПОС в выходной буфер системных данных,
- отправляет данные, находящиеся в выходном буфере пользовательских данных, на аккумулирующий процессор,
- отправляет данные, находящиеся в выходном буфере системных данных, на аккумулирующий процессор.

Примечание 1. Для отправки системных данных функция statistic_send должна вызываться по завершении реализованного в коде БПОС цикла перебора локальных стадий. В связи с этим с целью тестирования предлагаемой технологии была создана специальная экспериментальная версия БПОС, содержащая вызов statistic_send.

Всюду далее под БПОС будем иметь в виду упомянутую экспериментальную версию. Функция statistic_send может опционально использоваться для «досрочной» отправки пользовательских данных.

В этом случае ее можно вызывать из пользовательских стадий.

Данные, накопленные в пользовательской области к моменту вызова функции, будут отправлены и удалены из пользовательской области.

Примечание 2. Если процессор сбора данных статистики, выполняющий стадию-аккумулятор, еще не успел проинициализировать структуры, предназначенные для приёма данных самоконтроля, то локальный процессор не отправляет данные.

В этом случае данные начинают накапливаться на процессоре-отправителе в областях отправки. По исчерпании места в областях отправки новые данные теряются.

4.3.2 Транспортный механизм передачи данных. Адресация пакетов NWRITE

Для передачи данных самоконтроля на аккумулирующий процессор разработан протокол, основанный на транзакциях типа NWRITE протокола RapidIO («отложенная» (posted) запись данных в память удаленного процессора без подтверждения получения) [1].

Такой выбор обусловлен высоким быстродействием транзакций NWRITE, а также их асинхронностью: нештатные замедления

стадии-аккумулятора не должны замедлять весь вычислительный конвейер.

Поскольку транзакция NWRITE не требует подтверждения от процессора-приемника, теоретически возможна пропажа пакетов при пересылке данных.

Формат пакетов для пересылки системных данных СОСД позволяет идентифицировать пропавшие данные и парировать их отсутствие на приемной стороне.

В случае пропажи пользовательских данных устанавливается только факт пропажи, но пропавшие данные не идентифицируются.

За время тестирования СОСД случаи пропажи пакетов NWRITE не наблюдались.

Данные передаются в пакетах типа NWRITE [1], состоящих из «системного» заголовка и тела размером 256 байт.

В системном заголовке указываются РИО-идентификатор приемного процессора и адрес в памяти, по которому пакет должен быть записан [1].

В качестве аккумулирующего процессора должен использоваться процессор с архитектурой КОМДИВ, реализованный в одной из микросхем: 1890ВМ6Я, 1890ВМ8Я, 1890ВМ9Я.

На аккумулирующем процессоре для приема пакетов NWRITE выделяется буфер объемом 16 Мбайт.

Буфер разделяется на 256 равных частей, предназначенных для приема пакетов NWRITE от 256 процессоров соответственно.

Каждая часть используется как кольцевой буфер, рассчитанный на 256 пакетов NWRITE (далее - «входной буфер»).

Привязка входного буфера к процессору-источнику осуществляется в соответствии с логическим номером процессора в формализме БПОС (базовые адреса буферов упорядочены в соответствии с логическими номерами процессоров).

Передающий процессор вычисляет базовый адрес входного кольцевого буфера на аккумулирующем процессоре в соответствии со своим логическим номером.

В заголовках отправляемых пакетов адрес инкрементируется «по кругу».

Заполнение входных буферов приходящими пакетами NWRITE осуществляется на аппаратном уровне.

Пакеты NWRITE хранятся в кольцевом буфере до тех пор, пока их не обработает стадия-аккумулятор.

Во избежание потери данных стадия-аккумулятор должна успевать обрабатывать приходящие пакеты на частоте вычислительного конвейера БПОС.

В случае переполнения кольцевого буфера старые (необработанные) данные затираются новыми пакетами NWRITE.

4.3.3 Формат передаваемых данных

Пакеты NWRITE, используемые для пересылки данных самоконтроля, форматируются по-разному в зависимости от типа пересылаемых в них данных.

Тело пакета NWRITE размером 256 байт разбивается на два поля: поле прикладного заголовка (16 байт) и поле данных (240 байт). Поле данных форматируется в соответствии с типом пакета PacketType, указываемом в прикладном заголовке. Тип пакета определяется типом пересылаемых в нем данных. Однотипные данные передаются в пакетах одного типа. В одном пакете могут помещаться несколько (целое число) однотипных данных.

Используются следующие типы пакетов:

- пакет со статической информацией по таймерам (тип PACKET_TIMER_STAT);
- пакет со статической информацией по портам потоков (тип PACKET_PORT_STAT);
- пакет с данными таймеров (тип PACKET_TIMER);
- пакет с данными портов потоков (тип PACKET_PORT);
- пакет с пользовательскими данными (тип PACKET_DATA).

В заголовке передаётся следующая информация:

- время формирования пакета типа NWRITE;
- номер итерации на локальном процессоре,
- РИО-идентификатор локального процессора;
- код ошибки системной переменной «errno» в момент формирования пакета;
- тип пакета PacketType;
- признак «последний пакет по текущей итерации».

4.3.4 Прием данных на аккумулирующем процессоре

На аккумулирующем процессоре функционирует типовая стадия accum_stat (стадия-аккумулятор). Стадия осуществляет прием пакетов NWRITE из входных буферов, заполняемых на аппаратном уровне, и осуществляет их обработку.

Стадия не имеет ни входных, ни выходных потоков БПОС. Итерацию стадии accum_stat можно описать следующим образом.

Прием данных начинается по завершении «разогревочных итераций» БПОС.

На каждой итерации стадия аккумулятор последовательно перебирает входные буфера пакетов NWRITE на аккумулирующем процессоре. Из каждого буфера вычитываются все «новые» пакеты. Опознание «нового» пакета основано на анализе метки времени в заголовке пакета: время нового пакета должно превосходить время предыдущего нового пакета в текущем буфере. Ведутся счетчики принятых и пропавших пакетов в буфере. Если текущий пакет «старый», а следующий за ним – «новый», то считается, что текущий пакет утерян, и генерируется событие STATISTIC_EVENT_DATA_LOST.

Если два последовательных пакета «старые», то считается, что все новые пакеты на текущий момент вычитаны.

Каждый новый пакет обрабатывается в момент обнаружения.

Обработка начинается с анализа полей прикладного заголовка.

Значение переменной errno из заголовка пакета копируется в структуру метаданных, соответствующую локальному процессору, от которого получен пакет.

Если значение errno в новом пакете изменилось по сравнению с предшествующим значением, то генерируется событие STATISTIC_EVENT_ERRNO.

По признаку «последний пакет»» фиксируется факт завершения сбора данных от очередной итерации на локальном процессоре. Далее обрабатывается поле данных пакета в соответствии с типом пакета PacketType, указанным в прикладном заголовке пакета.

В случае типов PACKET_TIMER_STAT, PACKET_PORT_STAT, PACKET_TIMER, PACKET_PORT поле данных пакета копируется в структуру метаданных с учетом номера итерации из прикладного заголовка пакета.

В случае типа PACKET_DATA перебираются содержащиеся в пакете пользовательские данные.

Эти данные маркированы порядковым номером и цифровым идентификатором пользовательского типа.

Для каждого обнаруженного пользовательского типа вызывается функция-обработчик.

На основе анализа порядкового номера пользовательских данных проверяется целостность переданных данных, в случае фиксации пропажи генерируется событие

STATISTIC_EVENT_USER_DATA_LOST.

В процессе поиска и обработки новых пакетов стадия-аккумулятор генерирует предопределенные события и вызывает их обработчики.

В частности, событие

STATISTIC_EVENT_DATA_INVALID

генерируется в случае, если данные пакета признаны недействительными.

Проверке на действительность подлежат РИО-идентификатор процессора-отправителя, номер таймера стадии, номер порта потока.

Пропажа или недействительность переданных системных данных автоматически парируется стадией accum_stat следующим образом.

В случае пропажи пакета типа PACKET_TIMER с данными таймеров соответствующие значения времен для текущей итерации в структуре метаданных вычисляются путем сложения значения таймера на предыдущей итерации и среднего времени таймера за итерацию.

Пропавшие на текущей итерации данные портов потоков не корректируются.

При обработке используются их старые значения (с отставанием на 1000 итераций).

По завершении перебора всех входных буферов стадия-аккумулятор проверяет, произошло ли событие STATISTIC_EVENT_ITERATION_RELEASED(был ли завершен сбор данных итерации на всех локальных процессорах по текущей итерации или по итерации с превосходящим номером).

Примечание. Поскольку допускается возможность пропажи пакета с признаком «последний пакет итерации», не исключена ситуация, при которой окончание сбора полного комплекта системных данных по какой-либо итерации никогда не будет зафиксировано.

В этом случае сбор данных по текущей итерации считается завершенным, если собран полный комплект данных по итерации с номером, превосходящим номер текущей итерации.

Если событие

STATISTIC_EVENT_ITERATION_RELEASED

не произошло, то перебор входных буферов с поиском новых пакетов повторяется.

Если событие произошло, то выполняются следующие действия:

- опционально с конфигурируемым шагом по номеру итерации формируется и выводится в консоль аккумулирующего процессора сводная таблица с результатами статистической обработки системных данных самоконтроля;
- вызывается пользовательская функция-обработчик события

STATISTIC_EVENT_ITERATION_RELEASED.

На этом итерация стадии accum_stat завершается.

4.4 Интерпретация метаданных функциями-обработчиками событий и пользовательских типов данных

Стадия-аккумулятор, вызывая пользовательскую функцию-обработчика, передает в нее в качестве первого входного параметра указатель на структуру метаданных процессора, с которым связано обрабатываемое событие, или от которого пришли обрабатываемые пользовательские данные (структура statistic_info* info).

В случае события STATISTIC_EVENT_ITERATION_RELEASED в функцию-обработчика передается массив указателей на все структуры метаданных.

Для интерпретации метаданных предназначен набор интерфейсных функций, названия которых начинаются с префикса «statistic_get_», например:
int statistic_get_errno (statistic_info* info); (функция получения кода системной переменной «errno»). Функции могут иметь от одного до трех параметров:

-<имя функции>(statistic_info* info);

-<имя функции>(statistic_info* info, unsigned int iter);

-<имя функции>(statistic_info* info, unsigned int iter, unsigned int index);

где

- info – указатель на структуру метаданных,
- index – индекс таймера в массиве таймеров процессора если функция интерпретирует данные таймера, или индекс порта в массиве портов процессора если функция интерпретирует данные порта. Индекс таймера – это сквозной номер по всем таймерам всех стадий, функционирующих на указанном процессоре info. Индекс порта – это сквозной номер по локальным портам всех потоков, функционирующих на указанном процессоре info,
- iter – номер итерации, по которой нужно получить информацию.

Примеры функций с одним параметром: statistic_get_errno, statistic_get_iter, statistic_get_received, statistic_get_lost, statistic_get_proc_name, statistic_get_timer_count, statistic_get_port_count.

Перечисленные функции возвращают соответственно:

- код системной переменной «errno»,
- номер последней итерации, выполненной на процессоре,
- количество принятых и потерянных пакетов на процессоре,
- имя процессора в символике БПОС: <имя группы>_<номер в группе>,
- общее количество таймеров, используемых на процессоре,
- общее количество портов, используемых на процессоре.

Функция
int statistic_has_iter (statistic_info* info, unsigned int iter);
проверяет доступность информации по указанной итерации в структуре метаданных.

В приведенных ниже функциях с двумя или тремя параметрами индекс таймера (порта) должен указываться в диапазоне [0, statistic_get_timer_count (info)) ([0, statistic_get_port_count(info))) соответственно.

Функции с двумя параметрами позволяют получить статическую или аккумулированную информацию по индексу таймера или порта. Функции с двумя параметрами для таймеров:
statistic_get_stage_index_by_timer,
statistic_get_timer_name,
statistic_get_timer_time_min,
statistic_get_timer_time_max,
statistic_get_timer_iter_min,
statistic_get_timer_iter_max.

Перечисленные функции возвращают соответственно:

- индекс стадии БПОС stages[] по индексу таймера,
- название таймера стадии БПОС,
- значение минимального времени по таймеру,
- значение максимального времени по таймеру,
- номер итерации с минимальным временем таймера,
- номер итерации с максимальным временем таймера.

Функции с двумя параметрами для портов:
statistic_get_port_name,
statistic_get_port_number,
statistic_get_port_linked_info,
statistic_get_port_linked_index,
get_stage_index_by_port.

Перечисленные функции возвращают соответственно:

- имя порта потока (имя порта генерируется программой СОСД и используется при выводе диагностических сообщений),
- номер порта в сквозной нумерации всех портов потока,
- указатель на структуру метаданных «связанного» процессора (под процессором, связанным с портом, подразумевается процессор-отправитель, если порт входной, и процессор-получатель, если порт выходной),
- индекс «связанного» порта потока (под портом, связанным с портом потока на

локальном процессоре, подразумевается порт на процессоре-отправителе, если поток входной, и порт на процессоре-получателе, если поток выходной),

- индекс стадии по индексу порта.

Функции с тремя параметрами возвращают данные таймера или порта с заданным индексом, полученные на указанной итерации.

Функции с тремя параметрами для таймеров:
statistic_get_timer_time,
statistic_get_timer_time_accum.

Перечисленные функции возвращают соответственно:

- значения времени таймера на указанной итерации,
- значения накопленного времени по таймеру на момент выполнения указанной итерации.

Функции с тремя параметрами для портов:
statistic_get_port_fail,
statistic_get_port_free,
int statistic_get_port_seq.

Перечисленные функции возвращают соответственно:

- число сбоев при передаче данных по заданному порту на момент выполнения указанной итерации,
- количество свободных буферов порта потока на указанной итерации,
- номер отсчёта данных, переданных через поток на указанной итерации.

5. Инициализация и завершение программы СОСД

5.1 Инициализация драйвера

Поток управления прикладной задачи на основе БПОС, использующей СОСД, функционирует в защищенном процессе POSIX операционной системы.

Для обеспечения вызова функции отправки пакетов NWRITE из процесса POSIX в СОСД реализован специальный драйвер - драйвер устройства «/dev/statistic».

Драйвер инициализируется посредством функции statistic_driver_init.

Вызов функции должен быть выполнен в пользовательском или системном потоке управления процесса KERNEL ОС РВ на каждом процессоре.

Прототип функции:

int statistic_driver_init (void);

Результат равен нулю в случае, когда инициализация и добавление устройства выполнено успешно.

5.2 Инициализация программы

Для инициализации программы предназначена функция statistic_init. Функция инициализации должна выполняться на каждом процессоре. Повторный вызов функции statistic_init не запрещён.

Прототип функции:

void statistic_init (int id);

где,

id – РИО-идентификатор процессора, который будет выполнять сбор данных на стадии accum_stat.

Возвращаемое значение – отсутствует.

5.3 Завершение программы

Для завершения функционирования программы на текущем процессоре предназначена функция statistic_finish. После вызова функции statistic_finish данные самоконтроля на текущем процессоре перестанут собираться и передаваться на аккумулирующий процессор.

Прототип функции:

void statistic_finish (void);

Возвращаемое значение – отсутствует.

6. Результаты численного эксперимента

Проверка функционирования программы СОСД проводилась с использованием тестовой задачи, выполнявшей три отладочные стадии БПОС (d_out, d_fwd, d_in [2]) и стадию-аккумулятор accum_stat. Стадии d_out, d_fwd, d_in выполняют соответственно генерирование, ретрансляцию и прием потоков данных БПОС. Задача выполнялась на 37 процессорах архитектуры КОМДИВ со следующим распределением : d_out – 8 процессоров КОМДИВ128-РИО, d_fwd - 20 процессоров КОМДИВ128-РИО, d_in - 8 процессоров КОМДИВ128-РИО, accum_stat- 1 процессор КОМДИВ64-РИО.

От стадии к стадии передавался поток данных 4,5 Мб. Задача выполнялась в двух вариантах: со сбором и без сбора данных самоконтроля. Задача работала на частоте 200 Гц. Всего было выполнено 10000 итераций, среднее время итерации – 5 мсек. Было установлено, что в среднем в режиме сбора данных самоконтроля сбор и отправка данных занимает на стадии d_out 0,1 мсек, на стадии d_fwd 0.7 мсек, на стадии d_in 0.2 мсек. Время, затраченное стадией accum_stat на сбор и обработку данных за одну итерацию, в среднем составляло 1.8 мсек.

Parallel digital signal processing program self-diagnostic data collection technology in multiprocessor real-time systems

T.K. Gringauz, A.N. Onin

Abstract: Multiprocessor real-time systems with RapidIO communications based on KOMDIV microprocessors are considered. Applied multiprocessor programs development is based on parallel digital signal processing library (ParT) [2]. For system self-diagnostics ParT objects parameters measurement analysis during computing conveyor cycle seems to be reasonable. Local measurements are to be collected by the single accumulating processor during cycle iteration. Special technology based on NWRITE transactions used as transport mechanism in RapidIO communication environment is developed.

Keywords: system self-diagnostics, parallel digital signal processing library, computing pipeline stage, processor group, data flow, timer, flow port, accumulator stage, handler function, RapidIO communication environment, NWRITE transaction.

Литература

1. RapidIO Interconnect Specification (Revision 1.3) Available from: http://www.rapidio.org/specs/current
2. Г.О.Райко. Библиотека параллельной обработки сигналов. //Труды НИИСИ РАН.- М., НИИСИ РАН, 2015, том 5 №1, с.64-69, ISSN 2225-7349
3. В.Б.Бетелин, В.А.Галатенко, К.А.Костюхин, М.Д.Дзабраев. Контролируемое выполнение сложных систем // Научный сервис в сети Интернет: труды XIX Всероссийской научной конференции (18-23 сентября 2017 г., г. Новороссийск). — М.: ИПМ им. М.В.Келдыша, 2017. — С. 63-71. URL: http://keldysh.ru/abrau/2017/72.pdf doi:10.20948/abrau-2017-72
4. Т.К. Грингауз, А.Н. Онин. Программное обеспечение для приема и передачи данных по высокоскоростному каналу в мультипроцессорных комплексах реального времени.//Труды НИИСИ РАН.-М., НИИСИ РАН, 2014, том 4 №2, с.22-32, ISSN 2225-7349
5. Т.К. Грингауз, А.Н. Онин. Инструментальное программное обеспечение для подготовки запуска задач в мультипроцессорных комплексах реального времени. //Труды НИИСИ РАН.- М., НИИСИ РАН, 2015, том 5 №2, с.122-129, ISSN 2225-7349
6. 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

О применении тиражируемой системы при автоматизации расчета заработной платы и кадрового учета

И.Б. Егорычев[1], А.А. Прилипко[2], Г.А. Прилипко[3], С.Г. Романюк[4], Д.В. Самборский[5]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1]egorychev@niisi.msk.ru, [2]aaprilipko@niisi.msk.ru, [3]prilipko@niisi.msk.ru, [4]sgrom@niisi.ras.ru, [5]dsambor@niisi.msk.ru

Аннотация: В статье описан распределенный комплекс информационных систем, успешно использованный при автоматизации кадрового учета и расчета заработной платы на крупном предприятии, ведущем коммерческую деятельность. Рассмотрены особенности учета, требующие наличия дополнительной детализации расчета и предоставления комплекса управленческой отчетности. Описаны используемые средства обмена и верификации данных между информационными системами, входящими в состав комплекса. Приведены примеры учетных и технических задач, которые были решены в процессе автоматизации.

Ключевые слова: распределенные системы, автоматизация учета, обмен данными, верификация данных, кадровый учет, расчет заработной платы

Введение

Ведение кадрового учета и расчета заработной платы необходимо практически на каждом предприятии. Кадровый учет регламентирует трудовые отношения между работником и предприятием, он предусматривает документальное отражение изменений кадровых состояний сотрудников, ведение их личных дел, графика отпусков, штатного расписания, предоставление периодической отчетности в различные контролирующие органы. Расчет заработной платы предусматривает учет начисления оплаты работникам предприятия, сопутствующих налогов и взносов, предусмотренных законодательством. Кроме того, он включает подготовку документов по оформлению выплат, формирование периодической отчетности в налоговую инспекцию и фонды. Многие аспекты кадрового учета и расчета заработной платы регламентированы и ведение этого вида учета мало отличается на различных предприятиях. В то же время отраслевая специфика, организационные особенности и учетная политика конкретного предприятия могут приводить к тому, что учет может содержать в себе как расширенные управленческие схемы, так и существенные отклонения от типовых схем, оставаясь при этом в рамках требований существующего законодательства. Используемые при автоматизации подходы и средства должны обеспечить оперативность, достоверность и полноту учетной информации. Они должны учитывать специфику учета конкретного предприятия, быть гибкими и масштабируемыми. В случае распределения функциональности между несколькими информационными системами, комплекс средств автоматизации должен включать в себя инструменты обмена и верификации данных.

Выбор средств автоматизации

Для автоматизации учета можно использовать как типовые тиражируемые информационные системы, так и уникальные, специально разработанные с учетом специфических требований конкретного предприятия. Тиражируемые учетные системы являются эффективным средством решения типовых задач и не требуют затрат на разработку. Затраты на автоматизацию учета с использованием таких систем заключаются только в их приобретении, внедрении, настройке и последующем обслуживании. Информационная система «1С: Зарплата и кадры государственного учреждения 8»

позволяет автоматизировать широкий круг задач, связанных с ведением кадрового учета и расчетом заработной платы [1].

Эта система широко используется для автоматизации учета в государственных учреждениях, поэтому производитель заинтересован в повышении ее надежности и регулярном выпуске обновлений для поддержания системы в актуальном состоянии в соответствии с текущим законодательством.

Специфика учета заработной платы на нашем предприятии предполагает отражение начислений сотрудников в разрезе выполняемых организацией заказов. Это необходимо для последующего предоставления заказчикам отчетности о том, какие сотрудники предприятия были задействованы при выполнении работ, какие суммы были потрачены на их заработную плату и сколько рабочего времени было затрачено каждым сотрудником.

Реализация функциональности для учета подобной специфики в типовой информационной системе может быть сопряжена с целым рядом проблем.

Прежде всего, разработка новой функциональности в типовой системе осложнена необходимостью придерживаться заложенных в ней методических и архитектурных принципов.

Кроме того, значительные изменения программного кода типовой системы заметно осложняют процесс ее обновления, обслуживания и снижают надежность в целом.

С другой стороны, использование уникальной системы позволяет эффективно решить круг задач, не охваченных функциональностью тиражируемой информационной системой.

Однако, сложность разработки уникальной системы напрямую зависит от объема ее предполагаемой функциональности.

Поэтому при автоматизации следует в первую очередь пытаться решать задачи доступными типовыми средствами, а разрабатывать специальные механизмы лишь тогда, когда применение типовых средств невозможно либо неэффективно.

Одним из эффективных решений при автоматизации учета может стать использование комплекса типовых и специализированных информационных систем, связанных средствами обмена и верификации данных.

Описание систем

Распределенный комплекс информационных систем, используемый для ведения кадрового учета и расчета заработной платы на нашем предприятии, состоит из компонентов, представленных на рисунке 1.
Первым компонентом является система «1С: Зарплата и кадры государственного учреждения 8» (далее «система 1С»). В ней решается большинство типовых учетных задач и формируется регламентированная отчетность.

Следующим компонентом является система управленческого кадрового учета и учета начислений (далее «система АСУ НИИСИ»). Она построена на основе инструментального комплекса, подробно описанного в [2]. В этой системе автоматизирован управленческий учет начислений заработной платы в разрезе выполняемых организацией заказов, расчет трудоемкости работ и формирование сопутствующей управленческой отчетности. Кроме того, в этой системе реализован отдельный учет штатного расписания.

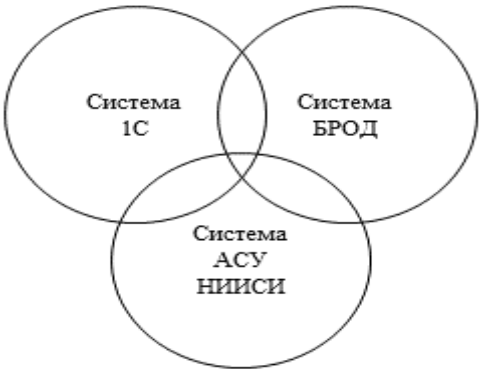


Рисунок 1. Комплекс информационных систем.

Еще одним компонентом является дополнительная система управленческого и бухгалтерского учета заработной платы (далее «система БРОД»). В этой системе ведется расчет дополнительных показателей заработной платы и формирование данных для отражения начислений в бухгалтерском учете. В ней реализована также функциональность по формированию комплекса разнообразной отчетности.

Описанные системы частично включают в себя функциональность друг друга. Такая избыточность позволяет дополнительно контролировать корректность данных, что особенно актуально в условиях работы с системами

большого количества операторов. Если в различных системах значения одного и того же показателя различаются, это может свидетельствовать об ошибке в данных и следует оповестить пользователя о необходимости уделить особое внимание такому участку расчета. Кроме того, использование комплекса информационных систем порождает целый ряд задач, связанных с необходимостью обмена и верификации данных, в том числе разнородных, когда одно и то же понятие в разных системах может трактоваться по-разному. Например, в системе 1С сущность «Статья финансирования» может быть привязана к организации, подразделению, сотруднику и конкретному начислению. Ее смысл заключается в указании источника средств при отражении начисления конкретной суммы для целей бухгалтерского учета. С другой стороны, организация ведет учет позиций штатного расписания с привязкой к статье финансирования в целях получения его разделов по каждой статье.

При этом предполагается, что если сотрудник занимает позицию штатного расписания, связанную с определенной статьей финансирования, то все его начисления, а также зависимые от них налоги и взносы, автоматически будут связаны с этой статьей. Однако, в системе 1С реализация такого ограничения отсутствует. Поэтому в системе АСУ НИИСИ ведется отдельный учет штатного расписания и одним из дополнительно описанных правил при верификации данных является контроль указания корректных статей финансирования при отражении начислений сотрудников.

Кроме статьи финансирования при учете начислений сотрудников на предприятии необходимо указание конкретного заказа или работы, из бюджета которых формируется фонд заработной платы. Рассмотрим подробнее структуру информации, которую необходимо учитывать при этом на примере отражения плановых начислений. Эта структура схематично представлена на рисунке 2.

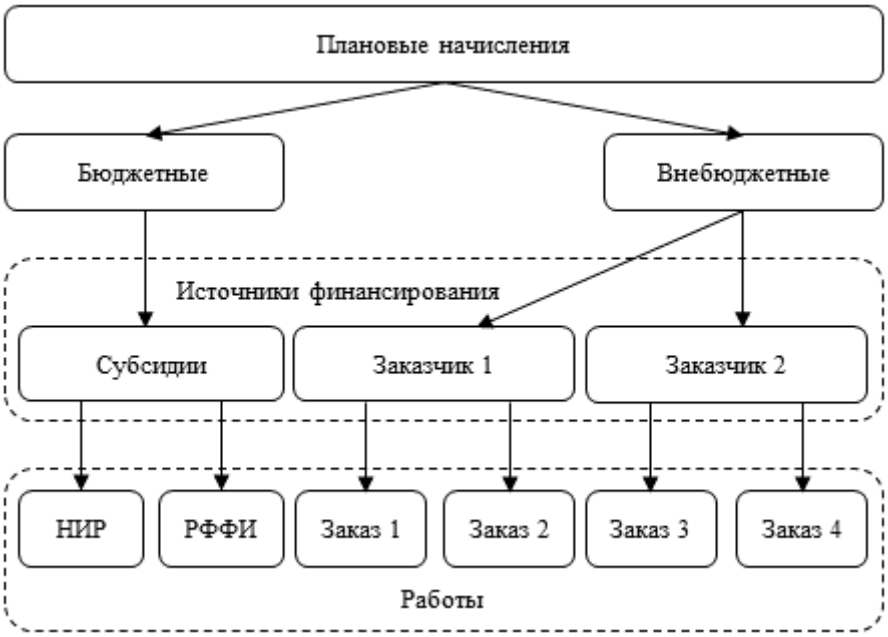


Рисунок 2. Отражение плановых начислений.

Средства предприятия, составляющие фонд заработной платы, можно разделить на бюджетные и внебюджетные. Бюджетные средства формируются за счет единственного источника финансирования — государственных субсидий. Фонд внебюджетных средств составляют оплаты за работы, которые предприятие выполняет в рамках договоров, заключенных с различными заказчиками. При учете начислений важно знать, из каких источников финансирования начисляется заработная плата каждого сотрудника, поскольку каждый источник финансирования, как правило, связан с определенным расчетным счетом и выплата денежных средств должна быть произведена с того расчетного счета, который соответствует источнику финансирования начисления.

Денежные средства, поступающие по каждому источнику финансирования, детализируются по работам, на которые они выделяются. В случае бюджетных средств детализация осуществляется по научно-исследовательским работам (НИР) и по грантам Российского фонда фундаментальных исследований (РФФИ). В случае внебюджетных средств детализация осуществляется по специальным работам или заказам. Требование вести учет начислений с такой детализацией обусловлено необходимостью формирования управленческой отчетности, используемой как для оценки деятельности внутри предприятия, так и для предоставления заказчикам. При отнесении начислений к определенному заказу важно, чтобы этот заказ был действующий. Ряд начислений при расчете распределяются по нескольким заказам. Примером таких начислений является оплата отпуска. Размер этого начисления рассчитывается исходя из заработной платы сотрудника за последние 12 месяцев. При этом учитываются все заказы, к которым были отнесены начисления за этот период и итоговое значение содержит в себе детализацию по ним. Поэтому технически возможна ситуация, когда начисленная оплата отпуска относится на заказы, которые уже закрыты и средства по ним не выделяются. Эта ситуация ошибочна, она должна быть своевременно обнаружена и исправлена.

Поскольку сведения о детализации начислений по работам не являются частью регламентированного учета, их учет ведется в системе 1С лишь частично. Более детальный учет распределения начислений по заказам и формирование комплекта управленческой отчетности осуществляется в специализированных системах АСУ НИИСИ и БРОД. При таком распределении функциональности одной из задач стало создание эффективного механизма обмена данными.

Интерфейс обмена данными

На раннем этапе автоматизации передача данных между информационными системами осуществлялась посредством загрузки и выгрузки файлов. На этом переходном этапе расчет заработной платы производился параллельно в системах 1С и БРОД. При этом за основу брался расчет, выполненный в системе БРОД, а разница между результатами расчетов формировалась в виде корректировочных файлов и загружалась в систему 1С. Со временем различий в расчетах становилось все меньше и за основу был взят результат, получаемый в системе 1С. Вместе с тем требования к размеру и составу данных, передаваемых между системами изменились, что потребовало переработки подходов к организации такого обмена. Одним из механизмов, реализованных в составе средств обмена и верификации данных между информационными системами кадрового учета и расчета заработной платы, стал универсальный интерфейс получения данных из системы 1С. Информационные системы, построенные на платформе 1С:Предприятие 8 могут предоставлять свою функциональность посредством публикации на веб-сервере [3]. На рисунке 3 изображена общая схема предоставления доступа к данным системы 1С посредством веб-сервера.

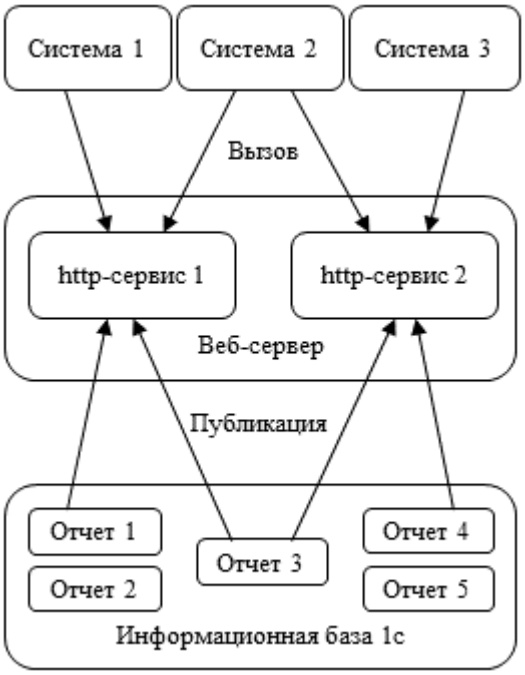


Рисунок 3. Схема доступа через веб-сервер.

В системе 1С реализованы средства для получения данных, обозначенные на схеме блоками «Отчет». На веб-сервере часть таких средств публикуется посредством http-сервисов, в каждом из которых производится формирование предоставляемых данных и их преобразование в формат xml или json. Прочие информационные системы, обозначенные на схеме блоками «Система» обращаются к веб-серверу, указывая параметры получения данных. Веб-сервер

анализирует входящие запросы, вызывает выполнение нужного http-сервиса и отправляет в ответ на запрос полученный блок информации.

В самой системе 1С был дополнительно реализован интерфейс для предоставления данных при запросах к http-сервисам. Каждый http-сервис может принимать запросы на выполнение отчета с указанием параметров, определяющих период получения данных и формат их представления. Например, рассмотрим такой запрос:

http://<host>/ZiKGU/hs/ReportRunner/СведенияОСотрудниках?Date=20180801&Format=xml

Он будет принят http-сервисом ReportRunner, который инициирует выполнение отчета СведенияОСотрудниках по состоянию на 1 августа 2018 года и скомпонует результат в формате xml.

На рисунке 4 представлена схема обработки запроса на получение данных. На первом шаге полученный запрос проходит этап верификации и анализа параметров. На этом этапе проверяется корректность значений параметров, их преобразование во встроенные типы системы 1С, проверяется состав параметров, исходя из описания интерфейса указанного отчета. Возникновение ошибки на данном этапе приводит к формированию http ответа с описанием проблемы. Если верификация и анализ данных выполнены успешно, информация передается в блок подключения обработчиков.

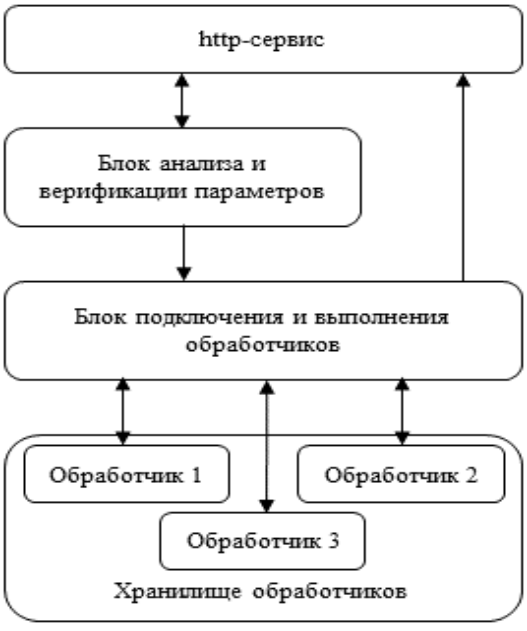


Рисунок 4. Схема обработки запроса.

Данные для формирования отчета могут быть скомпонованы из информации, полученной при выполнении целого ряда обработчиков. Их состав предопределен для каждого отчета. Все обработчики хранятся в виде алгоритмов в определенном справочнике и не входят в состав метаданных системы, поэтому если требуется изменить алгоритм получения данных, достаточно в пользовательском режиме загрузить новую версию обработчика, не приостанавливая работу всей системы. Блок подключения и выполнения обработчиков запускает необходимые модули, анализирует полученные массивы данных и компонует в требуемом формате результат, который затем передается в качестве ответа на запрос http-сервиса.

Создание обработчиков

Выборка данных из системы 1С при помощи обработчиков может осуществляться тремя основными способами, каждый из которых имеет свои преимущества и недостатки.

Первый способ заключается в создании обработчика на основе типового отчета. Отчеты в системе 1С являются основным способом получения данных. Их состав охватывает большую часть потребностей пользователей, а полученная в результате их выполнения информация часто может быть представлена в табличном виде, что удобно для ее сериализации (сериализация — преобразование данных в формат, подходящий для сохранения в файл или передачи по сети) и последующей передачи в другие системы. Создание обработчика на основе типового отчета не требует значительных затрат. Недостатками этого способа являются необходимость отслеживать изменения алгоритма типового отчета и согласование этих изменений с алгоритмом, реализованном в обработчике.

Второй способ заключается в использовании типовых интерфейсов получения данных. Для некоторых случаев в системе 1С реализованы удобные интерфейсы получения различных данных. Их использование позволяет не разбираться в логической схеме хранения информации в учетных регистрах и отслеживать изменения в ней, что ускоряет процесс разработки и повышает надежность алгоритма. К сожалению, такие интерфейсы реализованы для получения лишь некоторых сведений,

например, для кадровых данных и для сведений, используемых при отражении заработной платы в бухгалтерском учете. Кроме того, эти интерфейсы не документированы и могут быть изменены разработчиком системы 1С без предупреждения.

При третьем способе происходит получение данных напрямую из учетных регистров системы 1С. Этот вариант создания обработчика используется в тех случаях, когда не удается задействовать алгоритм стандартного отчета или какой-либо из программных интерфейсов. Он требует значительных затрат и менее надежен, чем другие способы, поскольку логическая схема учетных регистров в системе 1С недостаточно документирована, может быть некорректно интерпретирована разработчиком алгоритма обработчика и изменена разработчиком системы без предупреждения. Учитывая многочисленные недостатки этого способа, его использование не является основным, и он применяется лишь в исключительных случаях.

Задачи, решаемые при обмене данными

Рассмотрим некоторые задачи, для решения которых используется описанный механизм обмена данными с системой 1С.

Одной из главных задач, возникающих при расчете заработной платы является расчет налогов и взносов с последующим формированием соответствующей отчетности по установленным формам и правилам. Эта задача относится к регламентированному учету, поэтому расчет необходимых показателей производится в системе 1С. В то же время, полученные при расчете данные используются при отражении заработной платы в бухгалтерском учете. Для этого они должны быть переданы в систему БРОД, где данные детализируются и дополняются сведениями о распределении начислений по источникам финансирования и заказам предприятия. В процессе детализации из-за округления числовых показателей могут возникать различия между итоговыми значениями начислений и суммами их детализированных частей. Одной из задач верификации данных является контроль за возникновением этих различий.

Важной учетной задачей на предприятии является расчет трудоемкости выполнения работ. Эта информация включается в комплект управленческой отчетности, предоставляемой предприятием заказчикам. Сведения о характеристиках времени работы сотрудников являются частью кадровых данных и используются как в управленческом, так и в регламентированном учете. Они формируются на основании кадровых документов, которые отражаются прежде всего в системе 1С. В ней же ведется учет штатного расписания. Эти сведения передаются в систему АСУ НИИСИ, где происходит привязка сотрудников к выполняемым предприятием заказам для вычисления трудоемкости по каждой работе. В системе АСУ НИИСИ также ведется учет штатной расстановки и производится сверка данных о кадровых состояниях сотрудников с данными, полученными из системы 1С.

Средства верификации данных в системе 1С

В системе 1С производителем реализованы средства контроля целостности данных и их верификации исходя из правил, предусмотренных методологией регламентированного учета. Однако, для решения наших задач этих средств оказалось недостаточно, поэтому для контроля корректности данных в информационных системах дополнительно к основной функциональности нами был создан целый комплекс средств верификации и контроля логической целостности данных. Рассмотрим некоторые наиболее востребованные проверочные средства, дополнительно разработанные нами в системе 1С.

Прежде всего, это средства контроля сведений о внутренних совместителях. Внутренним совместителем является сотрудник, занимающий в организации более одной позиции штатного расписания. В системе 1С для отражения факта изменения состояния сотрудника необходимо вносить отдельный документ по каждой занимаемой позиции. Поэтому может возникнуть ситуация, когда, например, по одной позиции сотруднику оформлен отпуск, а по другой —не оформлен. Такая ситуация по принятой в организации методике учета является ошибочной. Для ее выявления был разработан отчет по контролю корректности ввода сведений о кадровых изменениях внутренних совместителей. В нем

производится сравнение состояний сотрудника по каждой занимаемой позиции на каждый момент времени и, в случае выявления различий, информация о них предоставляется пользователю для последующего анализа и внесения необходимых исправлений.

Следующим средством является модуль проверки адресов и лицевых счетов сотрудников. При ведении учета требуется предоставление ряда сведений о сотрудниках во внешние организации. Например, при передаче платежных ведомостей на перечисление заработной платы в банк, в частности, необходимо указание лицевого счета сотрудника, на который будут зачислены денежные средства. В случае, если сотруднику в банке открыто несколько лицевых счетов, причем некоторые из них могут не использоваться, для корректного проведения платежа важно указание правильного номера счета. Другой задачей является передача сведений по форме 2-НДФЛ в налоговую инспекцию. До недавнего времени в эти сведения включался адрес проживания сотрудника. Этот адрес должен был соответствовать классификатору адресов Российской Федерации, составленному согласно Государственному реестру адресов ФНС России. В то же время система 1С допускает указание адреса в произвольной форме. Дополнительно реализованные нами программные средства позволяют контролировать корректное указание лицевых счетов, а также выполнять проверку адресов и их приведение в соответствие классификатору.

Важным компонентом системы верификации данных является механизм контроля соответствия начисленной и выплаченной заработной платы. При начислении заработной платы помимо расчета сумм, которые будут перечислены сотрудникам, производится расчет налогов и взносов. Важно, что все эти сведения содержат в себе не только значения суммовых показателей, но и информацию об источнике финансирования и ряд других данных. Необходимо, чтобы все эти значения совпадали как при отражении начислений, так и при отражении выплат. Для исключения ситуаций, когда заработная плата сотрудника начислена по одному источнику финансирования, а налоги и взносы — по другому, а также ряда других возможных проблем, нами были разработаны расширенные средства контроля. Пользователь системы, ответственный за начисление и выплату заработной платы, в любой момент времени может получить развернутые результаты сравнения расчетов для своевременного выявления и исправления ошибочных ситуация.

Еще одним средством верификации является модуль контроля информации об отсутствии сотрудников. Если сотрудник не вышел на работу по неизвестной причине, этот факт отражается специальной записью в системе 1С. Впоследствии сотрудник может предоставить больничный лист, подтверждающий факт болезни в период отсутствия на рабочем месте. Больничный лист также отражается в системе 1С и является основанием для начисления сотруднику пособия за период его отсутствия. Возможна ситуация, когда по какой-то причине факт отсутствия сотрудника в системе отражен, а больничный лист не введен, хотя был предоставлен. В этом случае за период отсутствия сотруднику не будут начислены ни заработная плата, ни пособие. Реализованные средства контроля позволяют выполнить проверку соответствия между отметками об отсутствиях сотрудников по причине болезни и наличием оформленных больничных листов.

Заключение

Использование комплекса информационных систем для автоматизации кадрового учета и расчета заработной платы на предприятии позволило решить поставленные задачи. Для решения большинства задач, связанных с ведением регламентированного учета, применена тиражируемая информационная система, находящаяся на сопровождении производителя. В ней дополнительно к базовой функциональности нами были реализованы расширенные средства верификации данных, а также интерфейс взаимодействия с другими системами. Для автоматизации специфических задач, связанных с ведением управленческого учета, а также для решения задач, связанных с дополнительной детализацией регламентированного учета, нами были разработаны и успешно применены специализированные информационные системы. Используемые системы частично дублируют друг друга в части функциональности и учетных данных, имеют развитые средства обмена данными и

их верификации, что позволяет обеспечить полноту, достоверность и оперативность учетной информации.
Публикация выполнена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН (выполнение фундаментальных научных исследований ГП 14) по теме № 0065-2018-0004 "Развитие методов математического моделирования распределенных систем и соответствующих методов вычисления." (№ АААА-А18-118041290146-7).

On the use of a standard system in the automation of payroll and personnel accounting

I.B. Egorychev, A.A. Prilipko, G.A. Prilipko, S.G. Romanyuk, D.V. Samborskiy

Abstract: The article describes the distributed information systems set which was successfully applied for automation of payroll and personnel accounting at large enterprise with commercial activities. It considers extended accounting detailing and management reporting which are required according to enterprise particular features. Data exchange between information systems and data verification tools are described. Examples of accounting and technical problems that were solved in the automation process are given.

Keywords: distributed systems, automation, data exchange, data verification, personnel accounting, payroll

Литература

1. И.В. Гейц, Е.А. Кадыш. Учет и оплата труда работников государственных и муниципальных учреждений: актуальные вопросы. Применение «1С:Зарплата и кадры государственного учреждения 8». М.: ООО «1С-Паблишинг», 2017
2. И.Б. Егорычев. Инструментарий для построения автоматизированных учетных систем с WEB-интерфейсом. «Математическое и компьютерное моделирование систем: теоретические и прикладные аспекты: Сб. науч. тр. НИИСИ РАН. Под ред. акад. В.Б. Бетелина», М., НИИСИ РАН, 2009, 81-90
3. Пример создания HTTP-сервисов на платформе «1С:Предприятие». https://its.1c.ru/db/metod8dev#content:5756:hdoc

Методы создания распределенного комплекса информационных систем для автоматизации управленческого и бухгалтерского учета

А.Б. Бетелин[1], И.Б. Егорычев[2], А.А. Прилипко[3], Г.А. Прилипко[4], С.Г. Романюк[5], Д.В. Самборский[6]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия, E-mail's: [1]ab@niisi.msk.ru, [2]egorychev@niisi.msk.ru,

[3]aaprilipko@niisi.msk.ru, [4]prilipko@niisi.msk.ru, [5]sgrom@niisi.ras.ru, [6]dsambor@niisi.msk.ru

Аннотация: В статье рассмотрены методы и опыт создания распределенного комплекса информационных систем, успешно используемого на протяжении более 20 лет при автоматизации управленческого, кадрового и бухгалтерского учета крупного государственного учреждения. Особое внимание уделяется проблемам точного и своевременного расчета заработной платы в соответствии со спецификой учета. Описаны подходы к применению как тиражируемых, так и специально разработанных систем, а также созданию средства обмена и верификации данных между ними. Приведены примеры задач, решенных в процессе автоматизации учета и модернизации комплекса.

Ключевые слова: распределенные системы, автоматизация учета, обмен данными, верификация данных, кадровый учет, расчет заработной платы

Введение

Перед любым учреждением, особенно крупным, возникает целый ряд сложных проблем, связанных с организацией управленческого и бухгалтерского учета, в том числе и предоставления большого количества разнообразных отчетов.

В результате возникает необходимость создания предназначенного для их решения целого комплекса тесно взаимодействующих между собой информационных систем. Часть из них может быть разработана специально, а часть основана на использовании тиражируемых систем с дополнительной их настройкой и расширением их функциональности для учета специфики работы учреждения. В процессе эксплуатации этих систем необходима постоянная адаптация их возможностей для отражения изменений как в учете, так и в законодательстве. Одной из основных возникающих при этом проблем является обеспечение непрерывности функционирования всего комплекса и правильное распределение выполняемых задач между информационными системами, организации обмена и верификации данных между ними.

Хотя разработка собственных систем требует значительных затрат времени и усилий высококвалифицированных сотруд-ников, часто она оказывается оправдана тем, что позволяет полнее учесть специфику учета в учреждении и решать поставленные задачи более быстро и эффективно, особенно в тех ситуациях, когда соответствующих тиражируемых систем либо просто нет, либо их модификация нецелесообразна или требует еще больших усилий, затрат и времени, а также организации тесного и постоянного взаимодействия с внешними организациями и разработчиками.

Одной из важных и трудоемких задач для любого учреждения, особенно с большим числом сотрудников является своевременный расчет их заработной платы с последующим отражением его результатов в бухгалтерском и налоговом учете. При наличии нескольких видов деятельности учреждения и наличия различных источников их финансирования дополнительно возникает задача планирования и учета затрат по ним, в том числе и по зарплате, а также возникновение целого ряда дополнительных и довольно специфических видов форм отчетности. При этом необходимо решить целый ряд задач, связанных с кадровым учетом, в том числе и со штатным расписанием.

Создание системы БРОД

Для решения перечисленных выше задач, в том числе и задач бухгалтерского учета, а впоследствии и налогового учета в ФГУ ФНЦ НИИСИ РАН (в те годы НИИСИ РАН) более 20 лет назад был создан и с января 1997 года внедрен специально разработанный и постоянно развиваемый комплекс программ под общим

названием БРОД, который позволял не только решать типовые задачи, но и учитывать целый ряд специфических требований.

До 2005 года эта система использовалась не только для расчета зарплаты, но и для решения задач бухгалтерского учета.

В частности, это было связано еще и с тем, что в те годы практически отсутствовали тиражируемые системы для автоматизации бухгалтерии государственного учреждения, предоставляющие возможность их модификации своими силами.

Разработка осуществлялась на базе реляционной СУБД в операционной системе Linux.

При создании и эксплуатации собственных бухгалтерских систем возникают задачи отслеживания и постоянного учета изменений в законодательстве, создания большого количества отчетов и файлов специального формата для обмена персонифицированной информацией с системами пенсионного фонда и налоговых инспекций.

Интересными особенностями системы БРОД являются примененная в ней методика для компактного представления данных в базе данных [1] и наличие развитых средств для генерации отчетов [2].

Кроме того, был создан развитый набор средств для дополнительной верификации полученных результатов, в частности, при расчете зарплаты.

Система планового отдела

Для решения задач планового отдела, начиная с 1995 года применялась специально разработанная на базе реляционной СУБД система для ведения договоров (заказов).

В частности, она позволяла планировать и осуществлять учет затрат на зарплату сотрудников.

На основании подготовленного в ней распределения по заказам и учета данных об отработанном сотрудниками рабочем времени бухгалтерией и производился окончательный расчет зарплаты, налогов и накладных расходов.

Между системой планового отдела и системой БРОД был налажен механизм для обмена и верификации данных, что позволило существенно облегчить процесс расчета зарплаты и переноса информации.

Внедрение тиражируемых информационных систем

Для решения задач кадрового учета в отделе кадров была внедрена типовая система «1С: Зарплата и кадры 7.7» (далее называемой 1С: ЗиК7.7). При этом в ней пришлось сделать целый ряд доработок, позволяющих учесть соответствующие дополнительные требования для учреждений Российской Академии Наук.

В результате был создан распределенный комплекс информационных систем, которые позволили обеспечивать необходимую функциональность, однако требовалась постоянная и достаточно трудоемкая их поддержка и модернизация, в первую очередь для учета новых требований, например, изменений в законодательстве, отчетов.

Параллельно с этим, начиная с 2003 года начал рассматриваться вопрос о постепенном переходе на тиражируемую систему.

С 1 января 2005 года должен был произойти переход на новый план бухгалтерских счетов для бюджетных предприятий. Требовалась коренная переработка существующей системы в очень сжатые сроки. К этому времени появилась соответствующая версия программы для ведения бухгалтерского учета государственных учреждений «1С: Бухгалтерия 7.7 ПРОФ для бюджетных учреждений». В результате было принято окончательное решение о ее внедрении для решения задач бухгалтерского учета, что в первую очередь было связано с накопленным опытом ее использования, наличием в ней инструментальных средств для расширения ее функциональности собственными силами без привлечения внешних разработчиков, распространенностью, а также наличием возможности обучения и поддержки пользователей.

При этом необходимо было решить целый ряд довольно трудоемких задач по переносу в нее информации из системы БРОД, в частности, остатков по счетам, данных об основных средствах и материалах, однако в итоге применение тиражируемой программы бухгалтерского учета позволило сразу же обеспечить необходимую базовую функциональность и сосредоточиться в первую очередь на задачах, связанных со спецификой работы учреждения, и обеспечении взаимодействия с другими уже используемыми системами.

В то же время выяснилось, что количество доработок, которые потребовались бы для перехода на использование только

тиражируемой информационной системы 1С: ЗиК7.7, оказалось настолько значительным, что было принято решение продолжать использование уже существующих систем.

В 2012 году был осуществлен переход на систему «1С: Бухгалтерия государственного учреждения 8» (далее называемой 1С: БГУ) на платформе 1С: Предприятие 8.

Обмен и верификация данных

Для взаимодействия между 1С: БГУ и системой БРОД были разработаны специальные средства для передачи и верификации данных. Для обмена использовались как выгрузка информации из системы 1С: БГУ по сети с помощью специально созданных для этого сервисов, так и выгрузка специализированных xml файлов. В 1С: БГУ с помощью специальной процедуры была обеспечена возможность для загрузки документов. На основе изучения их структуры и использования выгруженных из нее словарей с помощью системы БРОД создаются xml файлы с документами «Отражение зарплаты в учете» в формате для загрузки в 1С: БГУ. В них содержатся сводные данные о зарплате, налогах и удержаниях в разрезе заказов. Загрузка этих документов, а при необходимости и сравнение с уже существующими документами, позволяет избежать большого объема ручного ввода данных и неизбежно возникающих при этом ошибок.

Создание АСУ НИИСИ

В связи с необходимостью создания большого количества информационных систем для решения различных задач бухгалтерского и управленческого учета было принято решение о создании набора инструментальных средств для их разработки, в связи с чем в 2007 году была разработана среда, названная АСУ НИИСИ [3]. Она позволила обеспечить ввод и доступ к данным через web-интерфейс. За счет введения понятия доменов (подсистем) и развитых средств контроля доступа [4], вплоть до конкретных полей в таблицах, в том числе и управления их визуализацией, удалось создать механизмы для настройки интерфейса пользователя таким образом, чтобы он мог выполнять только необходимые ему действия. Это позволило решить проблему разграничения данным.

Для разработки информационных систем в АСУ НИИСИ используется язык SQL.

Необходимая дополнительная функциональность обеспечивается с помощью комментариев особого типа при создании таблицы и возможностью при изменении значения содержимого полей вызова sql-процедур, имена которых сформированы по специальным правилам. Все это существенно упростило разработку информационных систем.

Одними из первых задач, которые были решены с помощью АСУ НИИСИ, было создание систем для финансового мониторинга и планирования учета затрат, а также постепенный отказ от разработанной ранее системы планового отдела и создание соответствующего набора подсистем в АСУ НИИСИ (далее называемой АСУПО). Отметим, что на данный момент в рамках АСУ НИИСИ создано порядка 30 различных информационных систем для решения задач бухгалтерского, производственного [5,6] и управленческого учета.

Таким образом, сформировался комплекс информационных систем, включающий как типовые системы на базе платформы «1С: Предприятие», так и специально разработанные системы БРОД и АСУПО с верификацией и обменом информации между ними. Распределение зарплаты по заказам осуществлялось в АСУПО, после чего в системе БРОД производился расчет зарплаты, результаты которого переносились в АСУПО для того, чтобы впоследствии формировались приказы о распределении зарплаты по заказам, отчеты по трудоемкости и т.п. При этом часть информации для расчета зарплаты сотрудников, в первую очередь об отработанных сотрудниками рабочих днях, отпусках и больничных в систему БРОД вводилась вручную по документам, сформированным в отделе кадров, поэтому одной из важных задач стало решение проблемы организации получения данных из системы, использованной в отделе кадров.

В связи с появлением новых задач и необходимостью организации полноценного обмена информацией с другими системами было принято решение о переходе в отделе кадров с 2016 года на систему «1С: Зарплата и кадры государственного учреждения 8» (далее называемой 1С: ЗиКГУ). Это позволило обеспечить автоматизированный перенос кадровой информации в систему БРОД, а в дальнейшем создать предпосылки для последующего перевода расчета зарплаты в систему 1С: ЗиКГУ.

Перевод расчета зарплаты в систему 1С: ЗиКГУ

Отметим, что расчет зарплаты по существу является задачей «реального времени», поскольку должен быть выполнен вовремя и без ошибок. Одной из возникающих при этом задач является правильный расчет отпусков и командировок, для чего требуется информация о зарплате и отработанных рабочих и календарных днях за предыдущие 12 месяцев. Кроме того, для правильного расчета налогов требуется информация о зарплате и налогах с начала расчетного года. Поэтому возникла необходимость наличия достаточно длительного переходного периода, когда зарплата рассчитывается одновременно в двух системах. Дополнительно потребовались доработки тиражируемой системы 1С: ЗиКГУ для учета особенностей расчета зарплаты на предприятии, в частности, для распределения зарплаты по заказам и научно-исследовательским работам, а также для получения необходимых данных из АСУПО. В целом процесс перехода занял год и девять месяцев, а окончательный переход на расчет зарплаты в системе 1С: ЗиКГУ был осуществлен в сентябре 2017 года.

Важно отметить, что при этом возникла уникальная возможность обеспечения дополнительного контроля правильности расчета зарплаты: наличие двух систем, одна из которых (1С: ЗиКГУ) выполняет основной, а вторая (БРОД) контрольный расчет, после чего их результаты сравниваются и в случае обнаружения разногласий уточняется, из-за чего они возникли.

Остановимся более подробно на некоторых проблемах, которые возникли в процессе перехода на расчет зарплаты в тиражируемой программе 1С: ЗиКГУ:

1. Перенос кадровой информации

На первом этапе, начиная с января 2016 года был осуществлен перевод кадровой информации из 1С: ЗиК7.7 в 1С: ЗиКГУ. Это потребовало создания специальных средств для ее переноса, переобучения сотрудников, а также добавление необходимой дополнительной функциональности в 1С: ЗиКГУ. В результате удалось при расчете зарплаты в системе БРОД начать использовать кадровые данные из 1С: ЗиКГУ и с помощью специально разработанных средств через web-интерфейс автоматизировать ввод данных о рабочем времени сотрудников, командировках и отпусках, а также об изменениях зарплаты и должностей сотрудников.

2. Расширение функциональности

Для учета специфики расчета, связанной с отнесением зарплаты сотрудников на заказы и научно-исследовательские работы, в том числе и с учетом различных источников финансирования, потребовалось обеспечить возможность получения информации из АСУПО и создать целый ряд специальных обработок. Отметим, что при этом удалось избежать существенных изменений в 1С: ЗиКГУ, что позволило значительно облегчить ее дальнейшее сопровождение.

3. Перенос данных о зарплате

Поскольку для правильного расчета отпусков и командировок нужны данные о зарплате и отработанных днях за предыдущие 12 месяцев, в систему 1С: ЗиКГУ были введены данные о зарплате сотрудников, начиная с 2016 года. Для этого были разработаны специальные средства для переноса и сравнения результатов в обеих системах и создания корректирующих документов в 1С: ЗиКГУ. В результате уже с середины 2017 года удалось получать из нее ряд отчетов, в том числе и файлов с персонифицированными данными для передачи в пенсионный фонд и налоговые инспекции.

4. Переход и параллельный расчет зарплаты в двух системах

В сентябре 2017 года зарплата впервые была рассчитана и выплачена с помощью системы 1С: ЗиКГУ. Одновременно с этим в системе БРОД был выполнен свой расчет, что позволило не только проверить его результаты в 1С: ЗиКГУ, но и сохранить ту дополнительную функциональность, которая была разработана в системе БРОД в части ее взаимодействия с системами АСУПО и 1С: БГУ. Более того, до конца 2017 года даже стандартные отчеты по зарплате продолжали формироваться на основе данных из системы БРОД.

Для сравнения и анализа данных дополнительно был создан специальный набор программных средств, который позволял извлекать информацию о зарплате из обеих систем. В 1С: ЗиКГУ для этого используется дополнительно разработанный web-интерфейс. В результате обращения к нему получается несколько xml файлов, которые анализируются и загружаются в базу данных, где они сравниваются с данными, полученными от системы БРОД.

Дополнительно анализируется, нет ли проблем в начислениях, в частности, нет ли

отнесения отпуска на источник финансирования, который уже не используется, нет ли налогов, начисленных на пустую зарплату, учтены ли все приказы отдела кадров, совпадает ли число дней отпусков и командировок т.п. Кроме того, средства сравнения позволяют при необходимости понять, что именно изменилось после предыдущего расчета зарплаты.

Еще одной особенностью расчета зарплаты в 1С: ЗиКГУ является раздельное ведение учета начислений и удержаний, в результате чего, в частности, возникает дополнительная задача определения точной суммы начисленного подоходного налога, связанная с его округлением до рубля.

В конечном итоге зарплата и налоги сотрудников в системе БРОД распределяется таким образом, чтобы они в разрезе заказов и научно-исследовательских работ совпадали с расчетами в 1С: ЗиКГУ, а суммы удержаний у сотрудника совпадали в обеих системах по каждому из источников финансирования.

Поскольку после окончательного расчета зарплаты информация в обеих системах совпадает, система БРОД используется для анализа и формирования данных, в том числе и для обмена информацией с другими системами, в частности с 1С: БГУ и АСУПО. При этом стандартные отчеты могут быть получены в 1С: ЗиКГУ, а дополнительные - в системе БРОД.

Синхронизация штатного расписания в системах БРОД, 1С: ЗиКГУ и АСУПО

Отдельно хотелось бы остановиться на проблеме синхронизации и поддержки штатного расписания и штатной расстановки сотрудников в актуальном состоянии, что необходимо не только для обеспечения правильного расчета зарплаты, но и для отслеживания истории изменений, контроля начислений системой БРОД и распределения зарплаты по заказам в АСУПО. Для этого важно, чтобы во всех этих трех системах состояние штатного расписания трактовалось одинаково.

В силу ряда особенностей архитектуры 1С: ЗиКГУ в ней по умолчанию отсутствуют необходимые средства для полного и наглядного представления штатного расписания учреждения. Так, для ставок не был предусмотрен атрибут, определяющий из какого источника она финансируется (за счет бюджетных или внебюджетных средств), а при указании совмещения должностей не было возможности задать ставку совмещения для сотрудника. Кроме того, надо отметить, что принятая в системе 1С: ЗиКГУ документо-ориентированная парадигма представления истории изменений усложняет задачу определения состояния штатного расписание на заданный момент времени: для этого необходимо последовательно учесть данные из всех предшествующих ему приказов. Причем приказы могут иметь не только дату их вступления в силу, но и дату окончания их действия, например, приказы о совмещении должностей или изменении источника финансирования ставки могут автоматически перестать действовать и тогда состояние штатного расписания неявно изменится.

Эти и некоторые другие недостатки были устранены с помощью доработки и настройки 1С: ЗиКГУ, а для экспорта данных было разработано несколько отчетов, вызываемых через web-интерфейс.

Оказалось, что для полного и наглядного представления истории изменений штатного расписания нужно комбинировать данные отчетов, собирающих информацию из четырех разных регистров 1С: ЗиКГУ: (1) «плановые начисления сотрудников» и (2) «кадровая история сотрудников» предоставляют взаимодополняющую информацию о должностях, местах работы, ставках, окладах и дополнительных надбавках сотрудников; (3) «история бухгалтерского учета зарплаты сотрудников» сообщает об источниках финансирования (бюджетное/внебюджетное) сотрудников и как они изменялись; (4) «сведения о совмещении должностей» выводит историю всех совмещений должностей (постоянных и временных).

С помощью автоматизированной процедуры синхронизации штатного расписания эти данные запрашиваются из 1С: ЗиКГУ и помещаются в базу данных АСУПО в виде доступной для анализа сводной таблицы.

В подсистеме АСУПО дополнительно имеется еще одно независимое представление штатного расписания, которое поддерживается в актуальном состоянии ручным вводом подписанных приказов о кадровых изменениях. Для сравнения представлений штатного расписания в подсистемах 1С: ЗиКГУ, АСУПО, и БРОД в системе АСУПО создано несколько отчетов. Несмотря на дублирование информации штатного расписания в трех подсистемах, возможность ее сверки означает дополнительный контроль, позволяющий предупредить распространение ошибки на

ранней стадии, до основного расчета заработной платы. Опыт показал, что ошибки могут возникать при ручном вводе данных во все эти системы.

Заключение

В статье были рассмотрены основные методы создания и дальнейшего сопровождения распределенного комплекса систем для автоматизации управленческого и бухгалтерского учета, включающего как тиражируемые с возможностью их самостоятельной доработки, так и специально разработанные системы.

Важной особенностью данного комплекса является наличие развитых средств для обмена и верификации данных между входящими в него компонентами, что позволяет обеспечить необходимую полноту и достоверность учета, а также в ряде случаев значительно снижает объем вручную вводимой информации. Всё это позволяет на протяжении уже более 20 лет оперативно решать большое количество разнообразных задач, связанных с управлением работой крупного государственного учреждения.

Публикация выполнена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН (выполнение фундаментальных научных исследований ГП 14) по теме № 0065-2018-0004 "Развитие методов математического моделирования распределенных систем и соответствующих методов вычисления." (№ АААА-А18-118041290146-7).

Approaches for the design of distributed management and accounting information systems

A.B. Betelin, I.B. Egorychev, A.A. Prilipko,
G.A. Prilipko, S.G. Romanyuk, D.V. Samborskiy

Abstract: This article describes the approaches for the design of the distributed information systems which have been successfully used for more than 20 years in the automation of managerial, personnel, and financial accounting at large government institution. Particular attention is paid to the problems of accurate and timely calculation of salary in accordance with bookkeeping specifics. We evaluate deployment of purchased and in-house information systems as well as the processes of organizing data exchange and verification. The article also exemplifies the challenges of financial accounting automation and its gradual modernization.

Keywords: distributed systems, automation, data exchange, data verification, personnel accounting, payroll

Литература

1. Г.А. Прилипко. Компактное представление информации в реляционной базе данных. «Современное состояние естественных и технических наук. Материалы IX Международной научно-практической конференции (14.12.2012)», М., 2012, 110-112.
2. Г.А. Прилипко. Генератор отчетов и обработка текстовой информации. «Достижения и приложения современной информатики, математики и физики. Материалы Всероссийской научно-технической заочной конференции», Уфа, РИЦ БашГу, 2012, 69-71.
3. И.Б. Егорычев. Инструментарий для построения автоматизированных учетных систем с WEB-интерфейсом. "Программные продукты и системы", 2008, №4, 49-52
4. И.Б. Егорычев. О подходе к организации защиты данных в учетных системах. "Моделирование и визуализация. Многопроцессорные системы. Инструментальные средства разработки ПО. Сборник статей под редакцией академика РАН В.Б. Бетелина", М., НИИСИ РАН, 2009, 90-104.
5. А.Б. Бетелин. Программное обеспечение для автоматизации учетно-управленческой деятельности подразделения по изготовлению единичной продукции. «Труды НИИСИ РАН», т.5 (2015), №2, 79-85
6. А.Б. Бетелин. Поддержка версий изделий при автоматизации производственного процесса. «Труды НИИСИ РАН», т.8 (2018), №1, 14-18

Провидение цикла занятий "Алгоритмика" в летнем лагере для дошкольников и младшеклассников

А. Г. Кушниренко [1], А.Г. Леонов [2], И.Н. Грибанова [3], М. В. Райко [4]

[1,2,3,4] ФГУ ФНЦ НИИСИ РАН, Москва, Россия,

[1,2] ФГБОУВО «Московский государственный университет имени М.В.Ломоносова», механико-математический факультет, Москва, Россия,

[2] ФГБОУВО «Московский педагогический государственный университет», Москва, Россия,

E-mail: [1] agk@niisi.ras.ru, [2] dr.l@vip.niisi.ru, [3] nig@niisi,msk,ru, [4] mila.rayko@gmail.com

Аннотация: В настоящей статье описаны методика и программное обеспечение, позволяющие провести краткий курс "Алгоритмика" в разновозрастной группе летнего лагеря. Программное обеспечение и методика получены путем адаптации материалов годового курса "Алгоритмика для дошкольников"[1].

Ключевые слова: информатика, алгоритмика, программирование, ПиктоМир, школьные олимпиады.

Различия в условиях проведения годового курса «Алгоритмика» и его летней версии

Стандартный годовой курс «Алгоритмики» для подготовительных групп ДОУ рассчитан на 30 занятий с дошкольниками 6-летнего возраста, причем занятия проводятся раз в неделю и длительность одного занятия составляет 30 минут. Условия проведения курса в условиях летнего лагеря (далее «летний курс») отличаются от условий проведения годового курса в ДОУ, во-первых, продолжительностью, а во-вторых, разновозрастным составом группы. Несмотря на эти отличия, авторам удалось адаптировать годовой курс для использования в летнем лагере с сохранением значительной доли методического обеспечения.

Длительность одной смены летнего лагеря составляет 10 дней и в занятиях одновременно участвуют дети возрастов от 6,5 до 10 лет. Поэтому летний курс «Алгоритмики» был сокращен до 10 занятий длительностью 45 минут, с 5 минутным перерывом и двухминутной паузой для гимнастики для глаз во второй половине занятия.

Для летней версии курса было специально разработано программное обеспечение - мир «Летний лагерь» в учебной системе ПиктоМир.

В этом новом мире 10 игр по 8-10 уровней в каждой игре. Мир доступен для свободного просмотра на сайте https://piktomir.ru/online/.

Доработанное методическое обеспечение было опробовано авторами статьи в летнем лагере Семейного Клуба Жени Кац "Солнышково". Занятия проводились каждый день в течение двух недель (исключая выходные). Состав групп в ходе цикла менялся от 8 до 12 детей возрастов от 6.5 до 10 лет. Для летнего курса «Алгоритмики» такое количество детей является оптимальным при условии, что в составе группы есть, как минимум, 3-4 ребенка возраста 9-10 лет. Как и в базовом курсе, в летнем курсе занятия проводились и в бескомпьютерной и в компьютерной форме. На компьютерной части занятия каждому ребенку предоставлялся планшет с 10-дюймовым экраном.

В курсе использовался учебный робот «Ползун», предоставленный предприятием «ИнфоМир» [4].

Содержание летнего курса

Занятия начинаются с прохождения Темы 1 годового курса «Алгоритмики» -

кластера базовых понятий алгоритмики [2]. Эти понятия необходимо дать детям любого возраста, без этого невозможно изучение остальных тем курса, знакомящих с более сложными конструкциями программирования. В годовом курсе на изучение этой темы отводится 7 занятий (по 30 минут). Ниже полностью приводится содержание Темы 1 из годового курса, тема вошла в летний курс без каких-либо изменений.

1. Парадигма программного управления. Программное управление без обратной связи. Реальные и виртуальные роботы

Робот и его система команд. Обстановка, в которой действует робот. Передача команд роботу с помощью радиосигналов или звуковых сигналов. Выполнение команды роботом, подтверждение успешного выполнения, отказ. Реальный робот-игрушка и его виртуальный двойник. Реальная и виртуальная обстановки. Пиктограммное представление команд робота. Шаблон программы в системе ПиктоМир. Процесс управления роботом. Команды-приказы. Режим непосредственного управления роботом. Пульт управления роботом.

Программа – план действий по управлению роботом. Линейная программа. Процесс составления программы и процесс исполнения программы. Программист – человек, составляющий программы. Возможность исполнения программы человеком. Компьютер – устройство для автоматического выполнения программ по управлению реальными и виртуальными роботами. Принцип программного управления – разделение процессов составления и выполнения планов по управлению роботом.

Поскольку занятия в летнем лагере проводятся ежедневно и группа разновозрастная, освоение Темы 1 в летнем лагере можно ускорить, отведя на нее два 45-минутных занятия. Оказалось, что в условиях каждодневных занятий, двух занятий вполне достаточно для полного овладения материалом учащимися возраста 7 лет и старше. Оставшиеся 8 занятий летнего курса заняло изучение материала, изложенного в теме 2 годового курса. Следует отметить, что годовой курс отводит на изучение этой темы 20 получасовых занятий.

В разновозрастной группе летнего лагеря тему 2, несмотря на регулярный пропуск занятий многими детьми, удается пройти за восемь 45-минутных занятий.

Ниже полностью приводится содержание Темы 2 из годового курса, Тема 2 вошла в летний курс практически полностью (единственное изъятие выделено ниже курсивом).

2. Способы составления программ, позволяющие сделать программу короче – повторители и подпрограммы

Пиктограммы повторителей. Обозначение подпрограммы однобуквенной пиктограммой. Технология отладки программы: непрерывное и пошаговое исполнение программы.

Технология составления программы: запоминание последовательности выполненных команд в «копилке», откатка. Совместное использование повторителей и подпрограмм.

Параллельное управление несколькими однотипными исполнителями с помощью одной и той же программы без обратной связи (SIMD-режим).

Совместная работа двух роботов в общей обстановке. Кооперативное составление программ управления двумя роботами для решения общей задачи.

Параллельная работа двух программ. Отказы при параллельной работе двух роботов в общей обстановке.

Включение в систему команд каждого робота команды «мигнуть» (подождать один такт). Синхронизация действий двух роботов с помощью команд «мигнуть».

Структура занятий летнего курса. Заключительная олимпиада

Каждое занятие, кроме последнего, состояло из четвертьчасовой бескомпьютерной части и получасовой компьютерной, разделенных пятиминутным перерывом. В середине компьютерной части обязательно выполнялась двухминутная гимнастика для глаз. На каждом занятии каждому участнику выдавался планшет. На каждом из 9 первых занятий из 30 минут

компьютерной части 20 минут занимала индивидуальная работа на планшетах, а оставшиеся 10 минут занимало коллективное программирование реального робота.

В бескомпьютерной части был использован свободно распространяемый методический материал годового курса, размещенный на сайте ФГУ ФНЦ НИИСИ РАН https://www.niisi.ru/piktomir/m2016.pdf:

1. Занятие 2. Лабиринт;
2. Занятие 4. Рассуждения о программах;
3. Занятие 7. Делаем программы короче. Ленты с командами;
4. Занятие 8. Секретные пакеты;
5. Занятие 11. Делаем программу короче - подпрограммы;
6. Занятие 12. Играем вместе. Рисуем букву «Ф».

На заключительном десятом занятии проводилось соревнование - олимпиада по параллельному программированию. Методика проведения соревнований — олимпиад для дошкольников в рамках курса «Алгоритмика» изложена в [3]. В курсе для летнего лагеря был оставлен неизменным принцип исключения собственно соревновательного момента при проведении олимпиады. Каждый участник соревнуется с самим собой, проверяя свой уровень знаний.

Олимпиада проводилась в командном режиме. Дети были разбиты на 5 команд по два человека. Каждой команде был выдан один планшет. На этом планшете члены команды составляли две программы двумя виртуальными роботами, решающими некоторую общую задачу. Было предложено для решения 9 заданий, расположенных в порядке возрастания сложности. Было объявлено, что решение 6 задач из 9 будет считаться отличным, а решение 8 задач из 9 – выдающимся результатом.

Основываясь на опыте работы с детьми на предшествующих занятиях, авторы старались подобрать задачи так, чтобы хотя бы одна команда показала выдающийся результат, а все остальные команды показали отличный результат. Этот план удалось претворить в жизнь.

Одна команда была составлена из двух участников возраста 9 лет. Эта команда за 45 минут успешно выполнила все задания. Из детей возраста 8 и менее лет были составлены три команды, примерно с одинаковым уровнем подготовки. Эти три команды за 30 минут успешно выполнили первые 6 заданий и, получив "отличный" результат, не стали браться за оставшиеся три задачи.

Последняя команда была составлена из ребенка 7 лет, посетившего все занятия и ребенка 10 лет, до олимпиады посетившего только два занятия.

Эта команда проработала все 45 минут и также сумела выполнить первые 6 заданий. В итоге все дети были горды своими успехами, понимая, что успехи эти заслуженные.

По окончании соревнования каждому участнику олимпиады был вручен диплом об успешном окончании курса «Алгоритмика». Диплом содержал перечень пройденных в летнем курсе тем, список индивидуальных успехов и достижений конкретного участника, включая олимпиадные достижения, а также рекомендацию к продолжению изучения информатики, программирования и робототехники.

Выводы

Авторами разработано программное и методическое обеспечение 10-дневнего каникулярного цикла занятий алгоритмикой для разновозрастных групп. Цикл из 10 занятий алгоритмикой может быть проведен в разновозрастной группе из 10-12 детей одним педагогом, имеющим опыт работы с системой ПиктоМир.

Для проведения данного цикла занятий необходим комплект планшетов с предустановленной системой ПиктоМир. Желательно также использование реального робота, работающего по командам с пульта или под управлением программы на планшете, на котором установлен ПиктоМир.Данный компактный курс в игровой форме знакомит детей с набором фундаментальных понятий, необходимым для освоения программирования.

Успешность освоения этого набора понятий была объективно подтверждена успешным участием в заключительной олимпиаде, предусмотренной программой курса.

Приложение. Примеры первого, шестого и девятого задания олимпиады

Далее приводятся формулировки соответствующих заданий и иллюстрирующие таблицы.

Задание 1. Роботы Тягун и Двигун должны переместить ящик в клетку, отмеченную зеленым крестиком.

Таблица 1. Шаблоны программ Тягуна и Двигуна.

Шаблон программы робота Тягун.

Шаблон с подсказкой для робота Двигун - он должен ждать 8 тактов.

Задание 6. Двигуны должны переместить бочки и ящик в отмеченные клетки.

Таблица 2. Шаблоны программ двух Двигунов.

Шаблон программы для первого Двигуна.

Шаблон программы для второго Двигуна.

Задание 9. Роботы Вертуны должны "починить" (закрасить) испорченные (потрескавшиеся) клетки.

Таблица 3. Шаблоны программ двух Ветртунов

Шаблон программы для первого Вертуна.

Шаблон программы для второго Вертуна

Conducting a course of lectures “Algorithmics” in a summer camp for preschoolers and elementary school students

A. G. Kushnirenko, A. G. Leonov, I. N. Gribanova, M. V. Raiko

Abstract. This article describes the methodology and software that allows to conduct a short course "Algorithmics" in a mixed-age group of a summer camp. The software and methodology were obtained by adapting the materials of the annual course "Algorithmics for Preschoolers" [1].

Keywords: informatics, algorithmic, programming, PiktoMir, school olympiads.

Литература

1. А.Г.Кушниренко, А.Г.Леонов, М.В.Райко, И.Б.Рогожкина. Методические указания по проведению цикла занятий «Алгоритмика» в подготовительных группах дошкольных образовательных учреждений с использованием свободно распространяемой учебной среды ПиктоМир // Тр. НИИСИ РАН. 2015. с. 1–125. Пиктомир. [Электронный ресурс] / Режим доступа: https://www.niisi.ru/piktomir/m2016.pdf свободный.
2. А.Г.Кушниренко, А.Г.Леонов, И.Н.Грибанова,М.В. Райко. Годовой цикл занятий «Алгоритмика для дошкольников» в подготовительных группах ДОУ. М., Мозаика-Синтез. 2018. №7, 174-175.
3. И.Н.Грибанова, М.В. Райко. Методика использования олимпиад в курсе «Алгоритмика для дошкольников». СПб., Культ-Информ-Пресс. 2017, 122-126.
4. Инфомир. [Электронный ресурс] / Режим доступа: https://infomir.ru/ свободный.