

ПАРАЛЛЕЛЬНОЕ И РАСПРЕДЕЛЕННОЕ ПРОГРАММИРОВАНИЕ

УДК 004.272

ПРИМЕНЕНИЕ МЕТОДОВ ПАРАЛЛЕЛЬНОГО ПРОГРАММИРОВАНИЯ ДЛЯ МОДЕЛИРОВАНИЯ ПОТОКООТКЛОНЯЮЩИХ ТЕХНОЛОГИЙ НА КОМПЬЮТЕРАХ С ГИБРИДНОЙ АРХИТЕКТУРОЙ

© 2019 г. А. И. Никифоров^{а,*}, Р. В. Садовников^{а,**}

^аИнститут механики и машиностроения — обособленное структурное подразделение Федерального государственного бюджетного учреждения науки “Федеральный исследовательский центр

“Казанский научный центр Российской академии наук”

420131 Казань, ул. Лобачевского, д. 2/31, Россия

* E-mail: nikiforov@imm.knc.ru

** E-mail: sadovnikov@imm.knc.ru

Поступила в редакцию 30.05.2018 г.

После доработки 17.07.2018 г.

Принята к публикации 03.09.2018 г.

Обсуждаются вопросы применения методов параллельного программирования для моделирования процесса воздействия полимердисперсными системами на нефтяные пласты на гибридной вычислительной системе, позволяющей использовать ядра центрального и графического процессоров. Показана эффективность разработанного подхода для решения практических задач моделирования заводнения нефтяных пластов с применением полимердисперсных систем на компьютерах с гибридной архитектурой.

DOI: 10.1134/S0132347419010059

1. ВВЕДЕНИЕ

Методы воздействия на пласт полимердисперсными системами (ПДС) принадлежат к группе физико-химических методов повышения нефтеотдачи пластов. Их применение является эффективным на поздней стадии разработки. Методы воздействия ПДС основаны на повышении фильтрационного сопротивления обводненных зон и вовлечения в активную разработку неподвижных и малоподвижных запасов нефти. Прирост нефтеотдачи достигается за счет блокирования основных водопроводящих путей (высокопроницаемых промытых зон), по которым фильтруется основная масса закачиваемой воды, не оказывая существенного влияния на выработку менее проницаемых участков и пропластков. Сложность моделирования такого воздействия обусловлена пространственным характером и нестационарностью протекающих процессов, а также с необходимостью расчета полей концентрации полимера, полей давления и насыщенности, структурных изменений пористости и проницаемости на каждом временном шаге в каждой точке пространства. Все это требует значительных затрат времени и ресурсов компьютера.

Современные технологии параллельного и распределенного программирования позволяют перейти к более детальному моделированию про-

текающих процессов, повысить точность расчетов, значительно сократить время вычислительного эксперимента, осуществить за разумное время многовариантные расчеты для предсказания поведения моделируемого процесса в будущем. Все большей популярностью пользуются гибридные и гетерогенные вычислительные системы, в которых неоднородные по архитектуре и производительности вычислительные узлы оснащены графическими ускорителями (Nvidia, AMD) или сопроцессорами (Intel Xeon Phi). В последнее время компьютерные системы с такой архитектурой занимают доминирующие позиции списка TOP500 [1].

Вопросы динамического распределения нагрузки и возможности оптимального выполнения параллельных программ на гетерогенных вычислительных системах с распределенной памятью обсуждаются в [2]. Возможности автоматизации построения алгоритмов и программ в суперкомпьютерных технологиях рассматриваются в [3]. Поскольку встроенные средства для поддержки параллелизма популярных в научных вычислениях языков программирования: Fortran, C++, Python и др. на уровне стандарта языков еще в стадии развития [4, 5], то обычно используются средства, которые предоставляются в качестве расширений языка в виде специальных библио-

тек (MPI, OpenMP, CUDA, OpenCL, OpenACC и др. [6–10]). Библиотеки реализуют различные модели параллелизма, основаны на открытых стандартах и архитектурах, общедоступны и поддерживаются сообществом специалистов в области параллельного программирования и компьютерных архитектур. Для эффективного использования библиотек необходимо адаптировать схему алгоритма под архитектуру используемой вычислительной системы. В [11] на примере задачи о просачивании показано, что надлежащий выбор модели и адаптация алгоритмов к архитектуре гибридной вычислительной системы позволяют получать ускорения, большие, чем при использовании классических многопроцессорных систем. В [12] на основе кинетического подхода для задач неизотермической трехфазной фильтрации (вода–нефть–газ) с учетом возможных источников тепловыделения, предложен вычислительный алгоритм явного типа, параллельная реализация которого ориентирована на высокопроизводительные системы с графическими ускорителями (GPU). В [13] рассматривается применение кинетических моделей для вычислительных систем с экстра-массивным параллелизмом. В [14] рассматривается решение задач математического моделирования процессов нефтедобычи на основе кинетического подхода с применением высокопроизводительных вычислительных систем. В [15] рассматривается реализация алгоритма BiCGStab с блочным ILU(0) предобуславливателем на графических картах AMD FirePro v9800 и Nvidia Tesla C2075, а также на вычислительном сопроцессоре Intel Xeon Phi 5110P и на 2-х процессорах Intel Xeon E5-2680. Результаты тестирования при моделировании реальных месторождений показали, что на использованном алгоритме обе видеокарты уступают в производительности последнему поколению процессоров Intel.

В [16] для решения задачи фильтрации к системе скважин методом контрольных объемов на неструктурированной сетке, дуальной к сетке треугольников Делоне, применены методы параллельного программирования на графических ускорителях Nvidia. В [17, 18] для задач моделирования процессов фильтрации была предложена и программно реализована на шаблонах языка C++ схема вычислений на нескольких графических устройствах одновременно. Ускорение расчетов одновременно на трех графических устройствах Nvidia в 6 раз превысило расчеты на кластере с 8 процессорами Opteron 246. В [19] для решения задачи фильтрации была предложена схема распределения вычислений на гетерогенном кластере, узлы которого оснащены многоядерными центральными процессорами (CPU) и несколькими GPU. Расчеты показали высокую эффективность схемы для задачи фильтрации к скважинам со сложной траекторией. В [20] рассматриваются

вопросы применения методов параллельного программирования для решения обратной задачи идентификации тензоров коэффициентов проницаемостей трещин и блоков неоднородного анизотропного трещиновато–пористого пласта на многопроцессорном кластере с распределенной памятью. В [21] для решения задач заводнения нефтяных пластов с применением полимердисперсных систем на многопроцессорной вычислительной системе был предложен вычислительный алгоритм с помощью технологии MPI и дана оценка эффективности параллельных расчетов. В [22] рассматривается применение методов параллельного программирования для решения задач двухфазной фильтрации с большим количеством скважин на гибридной вычислительной системе, позволяющей использовать ядра CPU с помощью технологии OpenMP и ядра GPU с помощью технологии CUDA. Показана эффективность для задач прогнозирования разработки нефтяных месторождений на длительный срок на гибридной вычислительной системе.

В настоящей статье применены методы параллельного программирования для расчетов процесса воздействия ПДС на компьютере, в котором параллельные вычисления на CPU с помощью технологии MPI сочетаются с вычислениями на GPU с помощью технологии CUDA. Для реализации используются шаблоны языка C++ и объектно-ориентированные средства современного стандарта языка Fortran, а также возможности смешанного программирования. Показана эффективность разработанного подхода для современных компьютеров с гибридной архитектурой.

2. МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ПРОЦЕССА ВОЗДЕЙСТВИЯ ПДС

Механизм воздействия ПДС, а также математическая модель этого явления изложена в работах [23, 24]. Модель основана на известных законах сохранения массы и импульса, которые дополняются замыкающими соотношениями. При выводе замыкающих соотношений использована модель пористой среды в виде пучка капилляров, а также представление пористой среды в виде двух взаимопроникающих континуумов, что позволило описать процесс образования полимердисперсных агрегатов на микроуровне и их влияние на пористую среду в целом на макроуровне. Основные гипотезы и предположения, вывод уравнений и соотношений подробно изложены в работах, обзор которых приведен в [23, 24]. В этой статье мы приводим основные результирующие уравнения и соотношения. В крупномасштабном приближении модель представлена следующими уравнениями [23]:

$$m_1 + m_2 = m, \quad (2.1)$$

уравнения сохранения масс фаз и компонентов для первого континуума

$$\frac{\partial}{\partial t}(m_1 S_{1o}) + \nabla \mathbf{U}_o = -q_o, \quad (2.2)$$

$$\frac{\partial}{\partial t}(m_1 S_{1w}) + \nabla \mathbf{U}_w = -q_w, \quad (2.3)$$

$$\frac{\partial}{\partial t}(R_{11} m_1 S_{1w}) + \nabla (R_{11} \mathbf{U}_w) = -q_{11}, \quad (2.4)$$

$$\frac{\partial}{\partial t}(R_{12} m_1 S_{1w}) + \nabla (R_{12} \mathbf{U}_w) = -q_{12}, \quad (2.5)$$

уравнения движения фаз в виде обобщенного закона Дарси

$$\mathbf{U}_o = -\frac{k}{\mu_o} f_o \nabla P, \quad (2.6)$$

$$\mathbf{U}_w = -\frac{k}{\mu_w} f_w \nabla P, \quad (2.7)$$

уравнения сохранения масс фаз и компонентов для второго континуума

$$\frac{\partial}{\partial t}(m_2 S_{2o}) = q_o, \quad (2.8)$$

$$\frac{\partial}{\partial t}(m_2 S_{2w}) = q_w, \quad (2.9)$$

$$\frac{\partial}{\partial t}(R_{21} m_2 S_{2w}) = q_{21}, \quad (2.10)$$

$$\frac{\partial}{\partial t}(R_{22} m_2 S_{2w}) = q_{22}, \quad (2.11)$$

где $m_1 = m_1(x, y, z)$ — часть порового пространства, занятая подвижными нефтью и водой, $m_2 = m_2(x, y, z)$ — часть пористого пространства, занятая неподвижными жидкостями и компонентами, m — динамическая пористость, S_{ij} — насыщенность i -го континуума j -й жидкостью (i — номер континуума, $j = o, w$, где o — нефть, w — вода); R_{ij} — концентрация i -й примеси в j -м континууме; q_o, q_w — интенсивности перехода нефти и воды из первого континуума во второй, и q_{ij} — интенсивности перехода примесей из первого континуума во второй; $\mathbf{U}_o, \mathbf{U}_w$ — скорости фильтрации нефти, воды; μ_o, μ_w — динамические вязкости нефти и воды; f_o, f_w — фазовые проницаемости для нефти и воды.

При воздействии ПДС изменяется массообмен между двумя континуумами и структура порового пространства (динамическая пористость и абсолютная проницаемость), поэтому необходимо дополнить систему замыкающими соотношениями, которые строятся с использованием функций распределения пор по размерам и частиц по объемам для идеализированной модели пористой среды в виде пучка капилляров различных

радиусов и формулируются в виде краевых задач Коши. Задача Коши для функции распределения пор по размерам имеет вид [23]:

$$\frac{\partial \varphi}{\partial t} - \frac{\partial(u_r \varphi)}{\partial r} + u_\eta = 0, \quad (2.12)$$

$$\varphi(r, 0) = \varphi^0(r), \quad (2.13)$$

где u_r — скорость сужения поровых каналов, u_η — интенсивность блокирования поровых каналов [23].

Задача Коши для функции распределения частиц по объемам имеет вид [23]:

$$\frac{\partial \psi}{\partial t} + \frac{\partial(u_v \psi)}{\partial v} - u_\xi = 0, \quad (2.14)$$

$$\psi(v, 0) = \psi^0(v), \quad (2.15)$$

где u_v — скорость роста частиц, u_ξ — интенсивность сокращения количества частиц в подвижной жидкости [23].

Интенсивность перехода воды из подвижного состояния в неподвижное, вызванное блокированием поровых каналов рассчитывается по формуле [23]:

$$q_w = S_{1w} m_1 \frac{\int_0^\infty u_\eta r^2 dr}{\int_0^\infty \varphi(r) r^2 dr}. \quad (2.16)$$

Интенсивность перехода нефти из подвижного состояния в неподвижное [23]:

$$q_o = (1 - S_{1w}) m_1 \frac{\int_0^\infty u_\eta r^2 dr}{\int_0^\infty \varphi(r) r^2 dr}. \quad (2.17)$$

Интенсивность перехода полимера в неподвижное состояние [23]:

$$q_{11} = R_{11} q_w + q_{21}, \quad (2.18)$$

где $q_{21} = \frac{\partial a}{\partial t}$, a — масса адсорбированного полимера, которая определяется изотермой сорбции [23].

Интенсивность перехода частиц в неподвижное состояние [23]:

$$q_{12} = R_{12} q_w + q_{22}, \quad (2.19)$$

где $q_{22} = q_R^r$ — интенсивность осаждения агрегатов.

Изменение коэффициента динамической пористости описывается формулой [23]:

$$m_1 = \bar{m} \cdot m_1^0, \quad \bar{m} = \frac{\int_0^\infty r^2 \varphi dr}{\int_0^\infty r^2 \varphi^0 dr}. \quad (2.20)$$

Изменение коэффициента абсолютной проницаемости описывается формулой [23]:

$$k = \bar{k} \cdot k^0, \quad \bar{k} = \frac{\int_0^\infty r^4 \varphi dr}{\int_0^\infty r^4 \varphi^0 dr}. \quad (2.21)$$

Неизвестными в модели (2.1)–(2.11) являются давление P , насыщенности порового пространства пласта нефтью и водой в первом и во втором континуумах S_{ij} , концентрации полимера и частиц в первом и во втором континуумах R_{ij} , скорости фильтрации фаз U_o, U_w . Система уравнений (2.1)–(2.21), описывающая процесс воздействия ПДС, дополняется начальными и граничными условиями, которые формулируются при моделировании конкретной ситуации.

3. МЕТОД РЕШЕНИЯ

Численная аппроксимация уравнений (2.1)–(2.11) по пространству строится методом контрольных объемов. Область расчета представляется в виде набора из N слоев, в каждом слое область течения разбивается на подобласти ячейками диаграммы Вороного, построенной на основе триангуляции Делоне. Искомые поля давления, водонасыщенности и концентрации примесей связываются с узлами триангуляции Делоне и с серединой высоты ячейки в каждом из слоев, а сама ячейка служит контрольным объемом. Интегрирование уравнений по сеточным блокам (контрольным объемам) производится в предположении постоянства значений характеристик пласта по блоку. При вычислении поверхностных интегралов в уравнениях для водонасыщенности и концентрации примесей учитывается направление потоков. Для аппроксимации производных по времени используется схема IMPES метода [25], что приводит к системе разностных уравнений относительно давления P , водонасыщенности порового пространства пласта в первом S_{1w} и во втором S_{2w} континуумах, концентрации примесей в первом R_{11}, R_{12} и во втором R_{21}, R_{22} континуумах.

Задачи Коши для функций распределения пор по размерам (2.12), (2.13) и частиц по объемам (2.14), (2.15) решаются методом конечных элементов на сетке одномерных конечных элементов

с использованием линейного представления искомых функций на каждом элементе. Для аппроксимации по времени применяется неявная схема. При интегрировании конвективных членов учитываются характеристические направления. Значения функций распределения пор по размерам и частиц по объемам рассчитываются в той области, для которых эта величина имеет физический смысл, т.е. сеткой покрывалась конечная часть интервала $[0, \infty)$, за пределы которого решение не распространялось за время решения задачи (2.1)–(2.11) [21].

Метод решения задачи подробно изложен в [23, 24], здесь кратко приведем его основные этапы. На каждом временном слое необходимо вычислить: 1) поле давления во всех узлах пространственной сетки, 2) поле водонасыщенности в первом континууме, 3) поля концентрации полимера и частиц в первом континууме, 4) суммарную скорость фильтрации, 5) функции распределения пор по размерам и частиц по объемам для всех узлов сетки, 6) структурные изменения пористости и проницаемости пласта.

4. ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ НА ГИБРИДНОЙ СИСТЕМЕ

Метод численного решения задачи довольно сложен ввиду того, что кроме учета трехмерности и нестационарности протекающих процессов для расчета полей давления, насыщенности, концентрации примесей, необходимо рассчитывать структурные изменения пористости и проницаемости на каждом временном шаге в каждом узле сетки расчетной области. Это связано с решением задач Коши для функций распределения частиц по размерам и пор по объемам и требует значительных затрат времени и ресурсов компьютера. В [21] был разработан метод для вычислительного кластера с распределенной памятью, который позволил значительно повысить производительность расчетов. Адаптация метода под архитектуру кластера с распределенной памятью была основана на независимости расчета структурных изменений пористости и проницаемости в каждой ячейке расчетной сетки области. Это позволило провести декомпозицию области на непересекающиеся подобласти, в каждой из которых пористость и проницаемость рассчитывались независимо. Для реализации алгоритма расчетов на кластере была использована технология MPI. Программная реализация расчетов была осуществлена на языке Fortran [5].

Воздействие ПДС может существенно изменить пористость и проницаемость пласта вплоть до блокирования некоторых водопроводящих путей, изменяет направление потоков, подвижности и динамическое перераспределение фаз в пласте, что вносит существенные возмущения в распределение давления и, как следствие, приво-

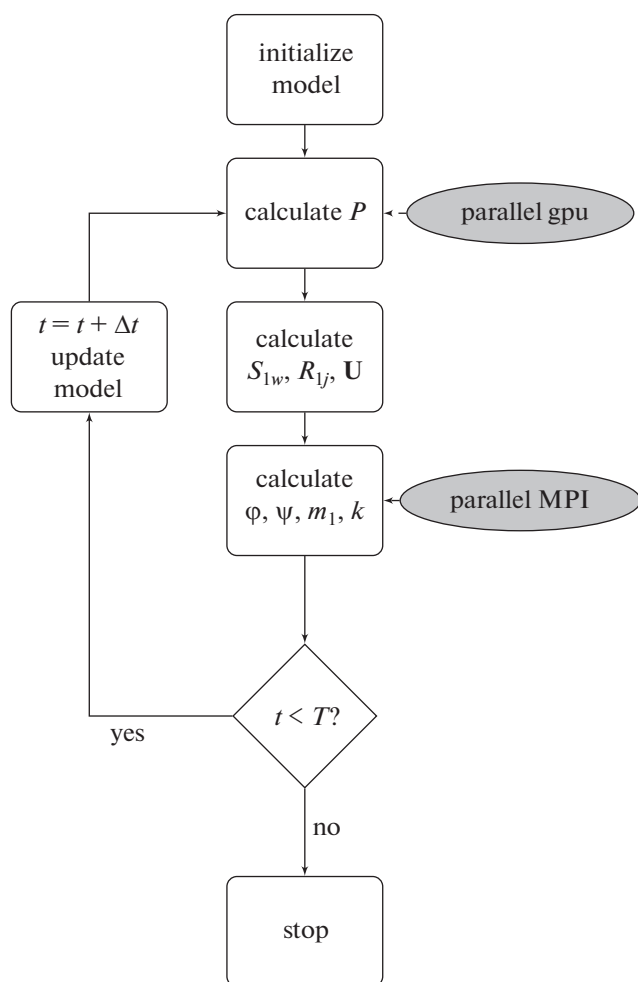


Рис. 1. Блок-схема алгоритма.

дит к увеличению количества итераций на каждом временном шаге при решении системы уравнений для давления. Решение системы линейных алгебраических уравнений может составлять более половины времени решения всей задачи, поэтому необходимо использовать методы ускорения этих вычислений.

Для снижения вычислительных затрат на решение системы линейных алгебраических уравнений при расчете воздействия ПДС мы применили разработанную нами и программно реализованную на шаблонах языка C++ библиотеку итерационных методов подпространств Крылова с предобуславливанием для решения разреженных систем линейных алгебраических уравнений большой размерности на графических ускорителях и гибридных вычислительных системах. Основные моменты реализации алгоритмов библиотеки подробно изложены в работах [16–19, 21, 22]. Библиотека была апробирована на решении ряда задач теории фильтрации. В [16, 18] для решения задачи фильтрации к системе скважин ме-

тодом контрольных объемов на неструктурированной сетке применены методы параллельного программирования на графических ускорителях Nvidia. В [17, 18] была предложена и программно реализована на шаблонах языка C++ схема вычислений на нескольких графических устройствах одновременно. Для распределения матрицы на графические устройства был адаптирован формат DMSR, а также была предложена схема матрично-векторного и векторно-векторного произведений. Использование нескольких GPU одновременно и обмен данными между ними через общую память CPU был осуществлен средствами технологии OpenMP [7]. Операции с матрицами и векторами локально на каждом графическом устройстве реализованы с помощью технологии CUDA [8]. Для балансировки загрузки нескольких GPU использовался метод декомпозиции области на подобласти с помощью методов, основанных на теории графов [26, 27]. В [19] было предложено несколько схем параллелизма, способы распределения матрицы задачи на нескольких вычислительных узлах и графических устройствах этих узлов, а также реализация операций матрично-векторного и векторно-векторного произведений для каждой схемы. В [19] был предложен метод двухуровневой декомпозиции области для использования на гетерогенном кластере, который позволил получить дополнительную декомпозицию области без увеличения нагрузки на коммуникационную сеть.

Для вызова алгоритмов библиотеки итерационных методов на шаблонах C++ в процедурах на Fortran при расчете ПДС воздействия применялись два способа, основанных на методах смешанного (mixed language programming) и объектно-ориентированного программирования [28]. Первый способ заключался в написании C функции-обертки кода на C++, в которую через указатели передаются данные разреженной матрицы, построенной процедурами языка Fortran. Этот способ, хотя и более простой в использовании, но не сохраняет состояния матрицы между вызовами функции на каждом временном слое и требует многократного выделения, копирования и удаления всей разреженной матрицы на GPU на каждом шаге по времени. Разреженная матрица для формата CSR представляет набор из трех массивов:

rowptr[$N + 1$] – указатель индексов строк,
colid[Nz] – указатель индексов столбцов,
values[Nz] – значения элементов матрицы,

где N – количество неизвестных, Nz – количество ненулевых элементов матрицы. Первые два массива целочисленного типа, а последний – массив вещественного типа. Поскольку многократное копирование данных в память GPU не-

эффективно [8], был применен другой подход. Для использования возможностей объектно-ориентированного программирования современного стандарта языка Fortran [28] был написан модуль, состоящий из классов-обертки на Fortran над объектами классов библиотеки на C++. Это позволило сохранять состояние объектов библиотеки между вызовами процедур на каждом шаге по времени и реализовать однократное выделение памяти под структуры матрицы непосредственно на GPU. Использование модуля избавило от многократного выделения, освобождения и копирования на GPU векторов, реализующих формат разреженной матрицы, кроме вектора *values*, копирование которого в память GPU необходимо для обновления значений матрицы на каждом шаге по времени. Интерфейс модуля приведен в листинге 1.

```
module lib_gpu_solver
  use iso_binding

  include "gpu_solver_def.f90"

  !Fortran type to represent a C++ class
  type gpu_solver_type
    private
    !pointer to the class
    type (c_ptr)::ptr
  contains
    !class members
    procedure::init => init_gpu_solver
    procedure::solve => gpu_solve
  end type

  !Constructor
  interface gpu_solver_type
    module procedure create_gpu_solver
  end interface

  !C-functions wrappers
  contains
    function create_gpu_solver ( &
      precondition_type, &
      solver_type, &
      print_status)
    implicit none
    type (gpu_solver_type)::create_gpu_solver
    integer (c_int), value::precondition_type
    integer (c_int), value::solver_type
    integer (c_int), value::print_status
    create_gpu_solver%ptr=
    create_gpu_solver_c(
      precondition_type,
      solver_type,
      print_status
    )
  end function

  subroutine delete_gpu_solve r (this)
    implicit none
```

```
    type (gpu_solver_type)::this
    call delete_gpu_solver_c(this%ptr r)
  end subroutine

  integer function init_gpu_solver ( &
    this, ierror, N, Nz, &
    rowptr, colind, val)
    implicit none
    class (gpu_solver_type), intent (in)::this
    integer (c_int), intent (inout)::ierror
    integer (c_int), intent (in), value::N, Nz
    integer (c_int), dimension (N+1), intent (in)::rowptr
    integer (c_int), dimension (Nz), intent (in)::colind
    double precision, dimension (Nz), intent (in)::val
    init_gpu_solver=ini t_gpu_solver_c( &
      this%ptr r, ierror, N, Nz, &
      rowptr, colind, va l) &
  end function

  integer function gpu_solve (
    this ierror, N, Nz,
    val, x, x0, b, max_iter, tol, time)
    implicit none
    class (gpu_solver_type), intent (in)::this
    integer (c_int), intent (inout)::ierror
    integer (c_int), intent (in), value :: N, Nz
    double precision, dimension (Nz), intent (in)::val
    double precision, dimension (N), intent (inout)::x
    double precision, dimension (N), intent (inout)::x0
    double precision, dimension (N), intent (in)::b
    integer (c_int), intent (inout)::max_iter
    double precision, intent (inout)::tol
    double precision, intent (inout)::time
    gpu_solve = gpu_solve_c( &
      this%ptr r, ierror, N, Nz, val, &
      x, x0, b, max_iter, tol, time)
  end function
end module@@
```

Листинг 1: Интерфейс модуля библиотеки

```
module gpu_solver_object
  use lib_gpu_solver
  integer (c_int)::gpu_solver = 0
  integer (c_int)::gpu_precond = 1
  integer (c_int)::print_status = 0
  integer (c_int)::result_flag = 0
  integer (c_int)::max_iter = 2000
  integer (c_int)::ierror = 0
  double precision::tol = 1.e-6
  double precision::time = 0
  type (gpu_solver_type)::gpu_solver_obj
end module
```

Листинг 2: Интерфейс модуля объекта

Код модуля, представленный в листинге 2, содержит интерфейс объекта библиотеки. В листинге 3 представлено использование объекта в программном коде на Fortran: объявление модуля, инициализация и вызов метода объекта.

```
!accessing module
use gpu_solver_object
```

```

. . .
!initialization
gpu_solver_obj%init ( &
    ierror, n, nz+1, &
    rowptr, colind, values )
. . .
!call solver method
gpu_solver_obj% solve ( &
    ierror, n, nz+1, values, &
    x, x0, b, &
    max_iter, tol, time)
    
```

Листинг 3: Использование объекта

Таким образом, предложенный подход расширяет, представленный нами в работе [21] метод и сочетает возможности использования нескольких ядер CPU с помощью технологии MPI [6] для подсчета структурных изменений пористости и проницаемости и GPU с помощью технологии CUDA [8] для подсчета распределения давления. Алгоритм параллельных расчетов, адаптированный для использования на компьютерах с гибридной архитектурой формулируется следующим образом: 1) процессор p_0 считывает основные начальные данные задачи из файлов и вычисляет для всех узлов сетки: 1.1) поле давления на GPU; 1.2) поле водонасыщенности в первом континууме с учетом нового распределения давления в пласте; 1.3) поле концентрации полимера и частиц в первом континууме R_{11}, R_{12} ; 1.4) суммарную скорость фильтрации $U = U_o + U_w$ через скорости фильтрации фаз; 2) процессор p_0 рассылает другим процессорам данные для расчетов пористости m_i и проницаемости k пласта; 3) каждый из процессоров p_i получает данные от процессора p_0 и вычисляет: 3.1) значения функции распределения пор по размерам $\phi(r, t)$; 3.2) значения функции распределения частиц по объемам $\psi(v, t)$; 3.3) значения пористости m_i и проницаемости k пласта; 3.4) высылает подсчитанные значения пористости m_i и проницаемости k пласта процессору p_0 , который записывает их в соответствующие ячейки памяти; 4) увеличивается расчетное время и осуществляется переход к пункту 1.1. Вычисления продолжаются до тех пор пока не будет достигнуто конечное время исследования. Блок-схема алгоритма представлена на рис. 1.

5. ЧИСЛЕННЫЕ РЕЗУЛЬТАТЫ

Численные результаты получены на участке реального месторождения, детальная геологическая характеристика которого была построена на основе трехмерной геометризации залежей и месторождения в целом и проводилась по специальной программе. Для каждого пласта и слоя строились: поверхности кровли и подошвы в соответ-

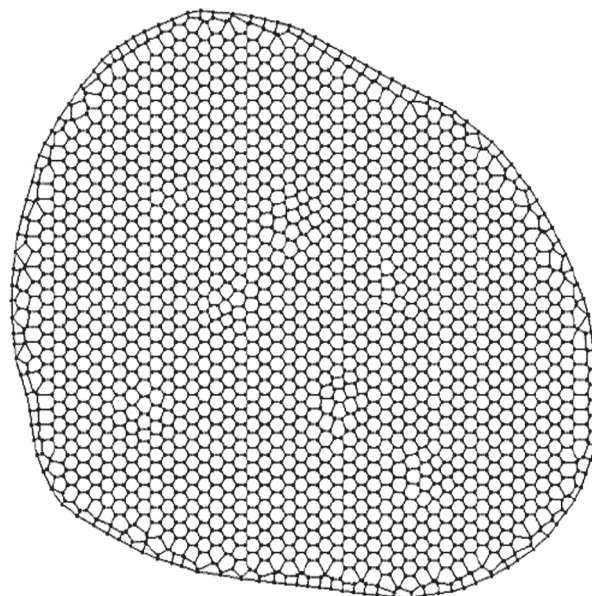


Рис. 2. Расчетная сетка области

ствии с их гипсометрическим положением в пространстве; общие, эффективные и нефтенасыщенные толщины; начальные распределения пористости, проницаемости и начальных запасов нефти. Модель резервуара для гидродинамических расчетов была представлена пятью слоями. Расчетная сетка участка месторождения в каждом слое имеет 1487 ячеек и содержит 2972 узла (рис. 2). Сетка одномерных конечных элементов для расчета функций распределения в каждой ячейке расчетной области состояла из 29 конечных элементов. Залежь на начальном этапе разрабатывалась при естественном водонапорном режиме. Настройка гидродинамической модели проводилась по фактическим данным обводненности

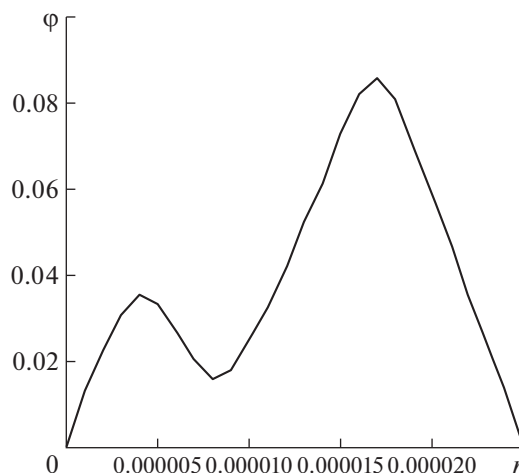


Рис. 3. Функция распределения пор по размерам.

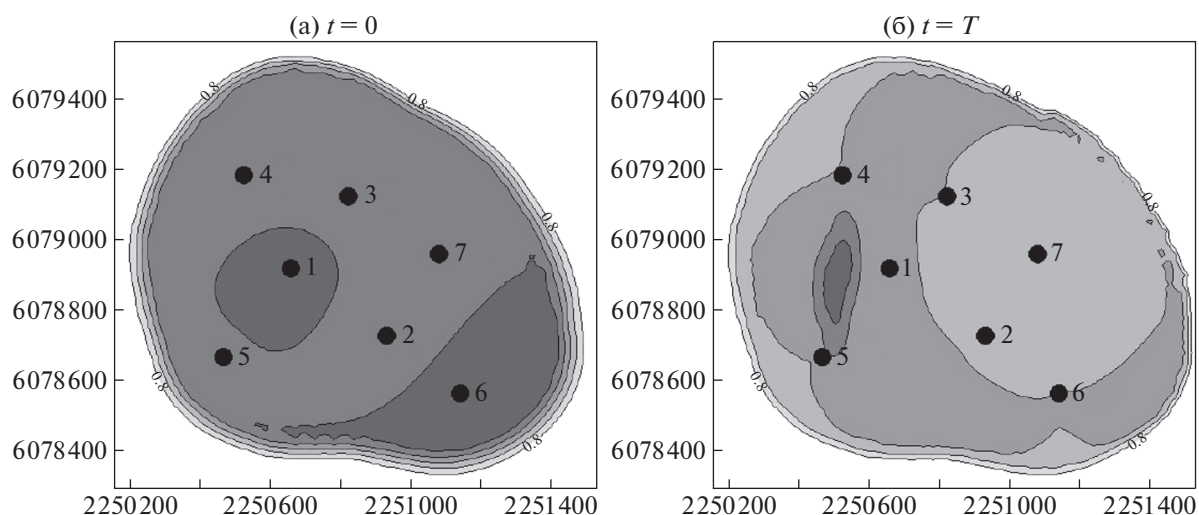


Рис. 4. Водонасыщенность верхнего проплатка.

продукции. При этом принимались следующие значения параметров жидкости: плотность нефти в пластовых условиях 817.0 кг/м^3 , плотность нефти в поверхностных условиях 862.0 кг/м^3 , динамическая вязкость нефти в пластовых условиях $20 \text{ мПа} \cdot \text{с}$, плотность воды в пластовых условиях 1182 кг/м^3 , вязкость воды в пластовых условиях $1.44 \text{ мПа} \cdot \text{с}$, объемный коэффициент 1.138. Расчетный участок разбурен 7 скважинами. Для моделирования изменения пористости и проницаемости в скважину 7 закачивали ПДС в течение четырех месяцев при постоянном давлении нагнетания 13 МПа. На рис. 3 приведена функция распределения пор по размерам для закачиваемой суспензии. Массовая концентрация полимера в растворе бралась равной 0.07%. Объемная концентрация частиц в нагнетаемой воде $0.0001 \text{ м}^3/\text{м}^3$. На рис. 4а, рис. 4б приведено распределение водонасыщенности верхнего проплатка на начальный и конечный моменты времени, соответственно.

Для расчетов использовался персональный компьютер следующей конфигурации: Intel(R) Core(TM) i7 CPU 920, 2.67 GHz, SODIMM DDR3 6 GB, GPU Nvidia GeForce GTX 590 (2xGF110, 512 kernels) и следующие программные средства: OS x64 Ubuntu 16.04 LTS, CUDA 8.0, GCC 5.4.0. Все вычисления на CPU и GPU проводились с двойной точностью. Для решения задачи ПДС воздействия на одном ядре процессора требуется 10,6 ч. Использование GPU для расчета поля давления на каждом шаге по времени позволяет сократить время решения задачи почти в 3 раза до 3,6 ч. Совместное использование технологии MPI для расчета структурных изменений пористости и проницаемости и GPU для расчета поля давления позволяет сократить общее время решения задачи до 42 мин. На рис. 5 представлены графики

ускорения вычислений на компьютере с гибридной архитектурой и на кластере с распределенной памятью [21]. Видно, что применение методов параллельного программирования для гибридных компьютеров позволяет значительно повысить производительность расчетов при невысоком энергопотреблении по сравнению с кластером. Большая масштабируемость задачи для гибридного компьютера по сравнению с кластером достигается за счет ускорения вычисления поля давления на GPU. Перспективным также является использование нескольких GPU одновременно, особенно при большем количестве узлов сетки и скважин.

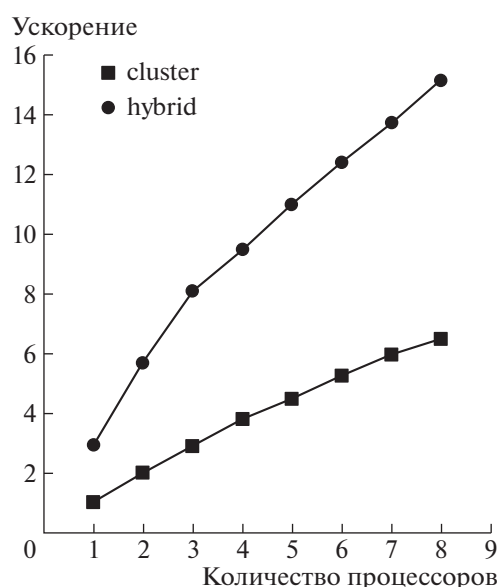


Рис. 5. Ускорение параллельных вычислений.

6. ЗАКЛЮЧЕНИЕ

Представлена реализация и результаты моделирования потокоотклоняющей технологии (ПДС) на компьютере с гибридной архитектурой. Предложена адаптация метода решения задачи под архитектуру компьютера с помощью методов параллельного программирования. Приведено сравнение ускорения расчетов на компьютере с гибридной архитектурой с расчетами на кластере. Дана оценка производительности расчетов.

СПИСОК ЛИТЕРАТУРЫ

1. <https://top500.org>. Cited May 11, 2018
2. Аветисян А.И., Гайсарян С.С., Самоваров О.И. Возможности оптимального выполнения параллельных программ, содержащих простые и итерированные циклы на неоднородных параллельных вычислительных системах с распределенной памятью // Программирование. 2002. № 1. С. 38–54.
3. Ильин В.П., Скопин И.Н. О производительности и интеллектуальности суперкомпьютерного моделирования. // Программирование. 2016. № 1. С. 10–25.
4. Вьюкова Н.И., Галатенко В.А., Самборский С. Поддержка параллельного и конкурентного программирования в языке C++ // Программирование. 2017. № 5. С. 48–59.
5. Горелик А.М. Современные технологии разработки больших программ для решения сложных вычислительных задач на компьютерах различной архитектуры // Программирование. 2006. 6. С. 56–70.
6. MPI: A Message Passing Interface (MPI) Standard (<http://mpi-forum.org/>). Cited May 11, 2018.
7. OpenMP Architecture Review Board (<http://www.openmp.org>). Cited May 11, 2018.
8. NVIDIA Corporation. CUDA Toolkit Documentation v9.1.85. <http://docs.nvidia.com/cuda/index.html>. Cited May 11, 2018.
9. <https://www.khronos.org/opencl>. Cited May 11, 2018.
10. What is OpenACC? <https://www.openacc.org>. Cited May 11, 2018.
11. Морозов Д.Н., Трапезникова М.А., Четверушкин Б.Н., Чурбанова Н.Г. Моделирование задач фильтрации на гибридных вычислительных системах // Матем. моделиров. 2012. Т. 24. № 10. С. 33–39.
12. Люпа А.А., Морозов Д.Н., Трапезникова М.А., Четверушкин Б.Н., Чурбанова Н.Г. Моделирование трехфазной фильтрации явными методами на гибридных вычислительных системах // Матем. моделиров. 2014. Т. 26. № 4. С. 33–43.
13. Четверушкин Б.Н. Кинетические модели для решения задач механики сплошной среды на суперкомпьютерах // Матем. моделиров. 2015. Т. 27. № 5. С. 65–79.
14. Люпа А.А., Морозов Д.Н., Трапезникова М.А., Четверушкин Б.Н., Чурбанова Н.Г., Лемешевский С.В. Моделирование процессов нефтедобычи с применением высокопроизводительных вычислительных систем // Матем. моделиров. 2015. Т. 27. № 9. С. 73–80.
15. Богачев К.Ю., Богатый А.С., Лапин А.Р. Использование графических ускорителей и вычислительных сопроцессоров при решении задачи фильтрации // Вычислительные методы и программирование. 2013. Т. 14, выпуск 3. С. 357–361.
16. Губайдуллин Д.А., Никифоров А.И., Садовников Р.В. Использование графических процессоров для решения разреженных СЛАУ итерационными методами подпространств Крылова с предобуславливанием на примере задач теории фильтрации // Вест. Нижегородского ун-та им. Н.И. Лобачевского. Сер.: Информационные технологии. 2011. № 1. С. 205–212.
17. Губайдуллин Д.А., Никифоров А.И., Садовников Р.В. Библиотека шаблонов итерационных методов Крылова для численного решения задач механики сплошных сред на гибридной вычислительной системе // Вычислительные методы и программирование. 2010. Т. 11, № 2. С. 351–359.
18. Губайдуллин Д.А., Никифоров А.И., Садовников Р.В. Библиотека `gru_sparse` для численного решения задач механики сплошных сред на гибридной вычислительной системе // Вестник Нижегородского ун-та им. Н.И. Лобачевского. Сер.: Информационные технологии. 2011. № 2. С. 190–196.
19. Губайдуллин Д.А., Никифоров А.И., Садовников Р.В. Об особенностях использования архитектуры гетерогенного кластера для решения задач механики сплошных сред // Вычислительные методы и программирование. 2011. Т. 12. 450–460.
20. Губайдуллин Д.А., Никифоров А.И., Садовников Р.В. Идентификация тензоров коэффициентов проницаемости неоднородного анизотропного трещиновато-пористого пласта // Вычислительная механика сплошных сред. 2011. Т. 4. № 4. С. 11–19.
21. Никифоров А.И., Садовников Р.В. Решение задач заводнения нефтяных пластов с применением полимердисперсных систем на многопроцессорной вычислительной системе // Матем. моделирование. 2016. Т. 28. № 8. С. 112–126.
22. Никифоров А.И., Садовников Р.В. Параллельные вычисления на гибридной вычислительной системе в задачах двухфазной фильтрации // Вычислительные методы и программирование: новые вычислительные технологии. 2018. Т. 19. № 1. С. 9–16.
23. Газизов А.Ш., Никифоров А.И., Газизов А.А. Математическая модель вытеснения нефти водой с применением полимердисперсных систем // Инженерно-физический журнал. 2002. Т. 75. № 1. С. 91–94.
24. Никифоров А.И., Низаев Р.Х., Хисамов Р.С. Моделирование потокоотклоняющих технологий в нефтедобыче. Казань: Изд-во “ФЭН” АН РТ. 2011. 224 с.
25. Азиз Х., Сеттари Э. Математическое моделирование пластовых систем. Пер. с англ. Москва: Недра, 1982. 407 с.
26. Hendrickson B., Leland R. The Chaco user’s guide. Technical Report Sand93–2339. Albuquerque: Sandia National Laboratories, 1993.
27. Pastukhov R.K., Korshunov A.V., Turdakov D.Y. et al. Improving quality of graph partitioning using multi-level optimization Program Comput Soft (2015) 41: 302. <https://doi.org/doi.10.1134/S0361768815050096>
28. Горелик А.М. Объектно-ориентированное программирование на современном Фортране // Программирование. 2004. № 3. С. 71–80.