МРНТИ 27.41.19
УДК 519.632.4

https://doi.org/10.51889/2020-3.1728-7901.05

Д.Ж. Ахмед-Заки[2], О.Н. Турар [1], Д.В. Лебедев [2]

[1]Казахский Национальный Университет имени аль-Фараби, г. Алматы, Казахстан
[2]Astana IT University, г. Нур-Султан, Казахстан

ФРАГМЕНТИРОВАННЫЙ АЛГОРИТМ ПОСТРОЕНИЯ АДАПТИВНОЙ СЕТКИ

Аннотация

В статье описывается использование системы автоматизации управления фрагментами вычислений, называемой LuNA (Language for Numerical Algorithms), для фрагментированного решения задачи построения адаптивной структурированной сетки. Основная идея LuNA состоит в том, чтобы использовать высокоуровневое представление алгоритма приложения, чтобы обеспечить его автоматическое выполнение на мультикомпьютерах с гибридными узлами без необходимости программирования специалистом на низком уровне. Построенная расчетная сеточная структура адаптируется к значениям заданной функции путем решения системы дифференциальных уравнений методом конечных разностей. В качестве дифференциального метода построения сетки использовано обращенное уравнение Бельтрами. Проведен ряд тестов на суперкомпьютере с использованием описываемого фрагментированного алгоритма, в том числе на большом количестве потоков для сетки размером в 500 миллионов узлов.

Ключевые слова: обращенное уравнение Бельтрами, структурированная сетка, адаптивная сетка, фрагментированное программирование, LuNA, высокопроизводительные вычисления, трехмерная декомпозиция.

Аңдатпа

Д.Ж. Ахмед-Заки[2], О.Н. Турар [1], Д.В. Лебедев [2]
[1]Әл-Фараби атындағы Қазақ Ұлттық Университеті, Алматы қ., Қазақстан
[2]Astana IT University, Нұр-Султан қ., Қазақстан

БЕЙІМДЕЛГЕН ТОРЛАРДЫ ҚҰРУДЫҢ ФРАГМЕНТТЕЛГЕН АЛГОРИТМІ

Мақалада адаптивті құрылымдық тор құру есебін фрагменттелген шығару үшін LuNA (сандық алгоритмдер тілі) деп аталатын есептеуді басқаруды автоматтандыру жүйесін қолдану сипатталған. LuNA-ның негізгі идеясы бағдарлама алгоритмінің жоғарғы деңгейдегі көрінісін қолдану, оны зерттеуші төменгі деңгейлі бағдарламалауынсыз гибридті түйінді мультикомпьютерлерде автоматты түрде орындай алуына мүмкіндік беру. Құрылған есептеу торы құрылымы берілген функцияның мәндеріне дифференциалдық теңдеулер жүйесін ақырлы айырым әдісімен шешу арқылы бейімделеді. Кері Белтрами теңдеуі дифференциалды тораптау әдісі ретінде қолданылады. Суперкомпьютерде сипатталған фрагменттелген алгоритмді қолдана отырып, бірнеше сынақтар өткізілді, оның ішінде 500 миллион түйінге арналған торлар саны көп болды.

Түйін сөздер: аударылған Бельтрами теңдеуі, құрылымдық тор, адаптивті тор, фрагменттелген бағдарламалау, LuNA, жоғары өнімді есептеу, 3D декомпозиция.

Abstract

FRAGMENTED ALGORITHM FOR ADAPTED GRID CONSTRUCTION

Akhmed-Zaki D.Zh.[2], Turar O.N. [1], Lebedev D.V. [2]
[1]Al-Farabi Kazakh National University, Almaty, Kazakhstan
[2]Astana IT University, Nur-Sultan, Kazakhstan

The paper describes the use of a computational fragments management automation system called LuNA (Language for Numerical Algorithms) for a fragmented solution for the problem of constructing an adaptive structured grid. The main idea behind LuNA is to use a high-level representation of an application's algorithm to enable it to automatically execute on hybrid node multicomputers without low-level programming by the researcher. The constructed computational grid structure is adapted to the values of the given function by solving the system of differential equations by the finite difference method. The inverse Beltrami equation is used as a differential meshing method. Several tests were carried out on a supercomputer using the described fragmented algorithm, including on a large number of threads for a mesh of 500 million nodes.

Keywords: inverted Beltrami equation, structured mesh, adaptive mesh, fragmented programming, LuNA, high performance computing, 3D decomposition.

Введение

В настоящее время большая часть численных вычислений выполняется в больших распределенных системах с сотнями ядер и потоков. С развитием таких суперкомпьютерных систем усложняется алгоритмический аспект распределения задач по ядрам системы. Программисту необходимо управлять не только алгоритмом решения, но и низкоуровневыми аспектами программирования, такими как распределение задач по узлам или потокам распределенной системы, топология узлов для оптимизации влияния передачи данных на время работы системы. приложение. Это и является причиной изобретения специальных систем для автоматизации таких задач, когда программист описывает только объекты данных и параллельные части алгоритма, когда всю передачу данных, балансировку, распределение задач выполняет сама система.

В статье описывается использование такой системы, называемой LuNA (Language for Numerical Algorithms), для параллельного решения задачи построения структурированной сетки. Основная идея LuNA состоит в том, чтобы использовать высокоуровневое представление алгоритма приложения, чтобы обеспечить его автоматическое выполнение на мультикомпьютерах с гибридными узлами без необходимости программирования программистом на низком уровне. Описываемая задача запускалась только на ядрах процессора системы в автоматическом режиме.

Ранее несколько различных задач были распараллелены с помощью системы LuNA [1,2,3]. На этих системах есть примеры запуска параллельного трехфазного нефтегазового потока в задачах пористых сред [4]. Для этого типа распараллеливания авторы используют термин фрагментированное программирование [5,6].

Существует большое количество похожих систем с разным уровнем распараллеливания. Некоторые из них предназначены для использования на однопоточных машинах, например, с использованием технологии OpenMP [7-10], а некоторые предназначены для использования на машинах с распределенной памятью [11,12]. Система LuNA создана для распараллеливания систем с распределенной памятью с упором на автоматическую балансировку этих распределенных ядер.

Статья посвящена построению адаптированных числовых сеток и запуску этого алгоритма в системе LuNA. Сетка структурирована и вначале может быть представлена в виде декартовой сетки. Чтобы добиться адаптации сетки, мы решаем конкретную систему дифференциальных уравнений в той области, где требуемые функции представляют координаты конечных узлов сетки. Уравнения решаются конечно-дифференциальными методами за счет структурированного представления доменной сетки.

Дифференциальное уравнение и постановка задачи полностью описаны в следующем разделе. Также существуют методики, основанные на неявном конечно-дифференциальном методе переменных направлений (ADI) решения параболических уравнений. Уравнение в нашем случае фактически нелинейное и считается квазилинейным из-за ретроспективного учета компонентов метрического тензора. Это означает, что эти компоненты считаются постоянными на основе значений предыдущей итерации. Само уравнение в основном эллиптическое и превратилось в параболическое путем добавления искусственного временного параметра, итерация которого указана выше.

Постановка задачи

Обратное уравнение Бельтрами используется для построения адаптивной структурированной сетки в следующей форме [13]

$$\frac{\partial}{\partial s^j}\left(\sqrt{g^s}\,g_s^{jl}\right) = \sqrt{g^s}\,g_s^{im}\frac{\partial^2 s^l}{\partial \xi^i \partial \xi^m} \qquad (1)$$

в этом уравнении g^s - метрика искривленного пространства, которая отличается от метрики декартова пространства (заданная эталонная область), g_s^{im}- компоненты контравариантного метрического тензора, s^j - искомые криволинейные координаты. В этом уравнении есть суммирование по повторяющимся индексам с одной стороны уравнения, т.е. для всех индексов, кроме l. В трехмерном случае l, j, i может быть 1, 2 или 3, и, соответственно, мы имеем систему трех уравнений для каждой из координатных осей. Это уравнение является нелинейным, поскольку компоненты метрического тензора всегда зависят от распределения координатных осей.

Тем не менее, при численном решении этого уравнения многие элементы этого сложного уравнения можно рассматривать ретроспективно, т. е. рассматривать как некоторое скалярное

распределение, не зависящее от s_j в текущей итерации, но зависящее только от ξ_i. Понятно, что в этом случае шаг по времени должен быть небольшим, а также что значения метрического тензора должны обновляться в соответствии с уже рассчитанными значениями координат сетки s_j. В этом случае задача рассматривается и решается как квазилинейная.

После приведения всех производных в уравнении (1) к эталонной области, уравнение приводится к форме (2). В большинстве случаев за граничные значения координат s_j принимаются решения аналогичных уравнений для меньших размеров. Эти решения представляют собой адаптацию двумерных и одномерных сеток к одной и той же метрике управления [13].

$$a^{pq}\frac{\partial^2 s^i}{\partial\xi^p\partial\xi^q} = P^i, \quad i,p,q = 1,2,3$$
$$s(\xi)|_{\partial\Xi^3} = \varphi(\xi) \tag{2}$$

$$a^{pq} = J^2 g_\xi^{pq} = g_s^{kl} \times \left(\frac{\partial s^{k+1}}{\partial\xi^{p+1}}\frac{\partial s^{k+2}}{\partial\xi^{p+2}} - \frac{\partial s^{k+1}}{\partial\xi^{p+2}}\frac{\partial s^{k+2}}{\partial\xi^{p+1}}\right) \times \left(\frac{\partial s^{l+1}}{\partial\xi^{q+1}}\frac{\partial s^{l+2}}{\partial\xi^{q+2}} - \frac{\partial s^{l+1}}{\partial\xi^{q+2}}\frac{\partial s^{l+2}}{\partial\xi^{q+1}}\right) \tag{3}$$

$$P^i = \frac{J}{\sqrt{g^s}}\frac{\partial}{\partial\xi^m}\left(\sqrt{g^s}g_s^{j_i}\right)\left(\frac{\partial s^{j+1}}{\partial\xi^{m+1}}\frac{\partial s^{j+2}}{\partial\xi^{m+2}} - \frac{\partial s^{j+1}}{\partial\xi^{m+2}}\frac{\partial s^{j+2}}{\partial\xi^{m+1}}\right) \tag{4}$$

В тестовой задаче на декомпозицию решение задач меньшей размерности не является критичным, поэтому в качестве значений границ выбраны координаты равномерного распределения. Это не мешает сходимости запущенной задачи, поскольку, по сути, уравнение является квазилинейным эллиптическим уравнением и дает равномерное распределение узлов согласно заданной криволинейной метрике из заданных граничных значений.

Область решения задачи представляет собой отдельный куб (на всех рисунках сечение этого куба будет показано на уровне сетки z = 0,5) на отрезке [0,1] по всем трем координатам. Этот куб является эталонной площадкой, на которой выполняется решение задачи.

Численный алгоритм решения основан на итеративном решении параболической задачи, возникающей при добавлении временной составляющей. Проблема решается методом стабилизирующей поправки:

$$\frac{s^{k+\frac{1}{3}} - s^k}{\frac{\tau}{3}} = a^{11}[s^k]L_{11}^h\left[s^{k+\frac{1}{3}}\right] + a^{22}[s^k]L_{22}^h[s^k] + a^{33}[s^k]L_{33}^h[s^k] + 2a^{12}[s^k]L_{12}^h[s^k] + 2a^{13}[s^k]L_{13}^h[s^k] + 2a^{23}[s^k]L_{23}^h[s^k] + g[s^k] \tag{5}$$

$$\frac{s^{k+\frac{2}{3}} - s^{k+\frac{1}{3}}}{\frac{\tau}{3}} = a^{22}[s^k]L_{22}^h\left[s^{k+\frac{2}{3}}\right] - a^{22}[s^k]L_{22}^h[s^k] \tag{6}$$

$$\frac{s^{k+1} - s^{k+\frac{2}{3}}}{\frac{\tau}{3}} = a^{33}[s^k]L_{33}^h[s^{k+1}] - a^{33}[s^k]L_{33}^h[s^k] \tag{7}$$

Где,

$$L_{mp}[v] = \frac{\partial^2 v}{\partial\xi^m\partial\xi^p}, \quad m,p = 1,2,3$$

$$g^i = -P^i, \quad i = 1,2,3$$

В этом случае используется 7-точечная схема. Хотя уравнение также содержит смешанные производные, они будут рассматриваться ретроспективно, исходя из значений на предыдущей итерации. Каждая из дробных ступеней выполняется в соответствующем направлении. В качестве управляющей метрики использована функция (Рисунок 1)

$$w(s) = \frac{1}{0.1 + (s^1 - 0.7)^2 + (s^2 - 0.5)^2 + (s^3 - 0.5)^2}$$

Узлы, находящиеся в непосредственной близости от точки (0,7,0,5,0,5), будут иметь высокие значения, что приведет к сгущению сетки в этой точке. Алгоритм сходится на определенном количестве итераций, однако для проведения экспериментов и проверки производительности параллельного алгоритма замерялось одинаковое количество итераций.

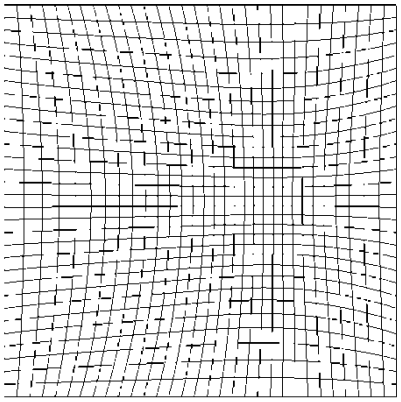

Рисунок 1. Сечение трехмерной сетки с адаптацией полученной путем решения задачи (2)-(4)

Фрагментированный алгоритм

Для распараллеливания решения описанной задачи использовалась трехмерная декомпозиция данных. В случае трехмерной декомпозиции вычисление выполняется для каждой части узлов, указанных блоками в каждом потоке. Если рассматривать все узлы сетки как один большой трехмерный блок, то при трехмерной декомпозиции данный блок разбивается на более мелкие блоки, равные друг другу во всех трех направлениях. Здесь блоки являются кубиками в ячейках эталонной сетки, и после адаптации эти кубики могут деформироваться.

Последовательный алгоритм представляет собой прогонку во всех трех направлениях по очереди. Каждая из этих прогонок проходит через все узлы сетки в одном направлении, затем каждый из них пересекает все подблоки декомпозиции и границы между ними. Прогонку можно распараллелить несколькими способами, например, с помощью алгоритма конвейерной развертки [14] или алгоритма распараллеливания из работы Яненко Н.Н., Коновалова А.Н., Бугрова А.Н., Шустова Г.В. [15-16].

Чтобы решить задачу (2-4), использовался другой способ распараллеливания. Суть этого алгоритма заключается в выполнении прогонок внутри каждого подблока с использованием текущих значений между блоками в качестве граничных условий. После этого на каждой итерации эти граничные значения должны изменяться из-за передачи координат приграничных уровней текущего подблока на каждый соседний подблок. В таком представлении меньшие прогонки работают параллельно на всех потоках, т.к. все они независимы друг от друга.

В таком алгоритме результат не такой, как в последовательном алгоритме, когда прогонка проходит от одного конца области к другому. В случае любой другой задачи такой метод нельзя использовать без надлежащего доказательства того, что он сходится к тем же значениям. Но в случае построения адаптивной сетки результат наиболее отчетливо виден. Конечно, существуют специфические характеристики и способы оценки качества сетки, такие как ортогональность криволинейных осей во всех узлах, прямоугольность ячеек, гладкость осей и т.д. Большинство этих показателей качества можно визуально наблюдать на самой сетке.

На рисунке 2 показан результат аналогичного алгоритма без передачи данных. При добавлении переносов граничных значений результат алгоритма показан на рисунке 2. Здесь и далее на рисунках использовалась сетка 32x32x32, разделенная на потоки 3x3x3, а центральный участок по оси Oz был визуализирован с помощью OpenGL. Изображение каждого фрагмента отображалось в отдельном окне, поскольку оно отображалось из одного потока.

В правой части рисунка 2 показаны все центральные срезы разложения вдоль плоскости z = 0,5. Там можно увидеть всю декомпозицию области куба, и все подблоки представлены в виде кубов только для простоты. Как видно с правой стороны при пересылке данных кубические блоки деформируются и принимают форму близкую к кускам сетки, полученной решением задачи последовательным алгоритмом.

Правильность такого подхода определяется тем, что здесь мы используем подход, аналогичный задачам с периодическими граничными условиями или построению структурированных сеток на прилегающих участках. Суть метода заключается в итерационном построении сетки на каждой из смежных областей и последовательном пересчете координат узлов на границе замыкания этих областей [1]. Но в то же время необходимо создать некоторую фиктивную область на границе стыка двух областей, содержащую узлы из обеих областей, близких к границам.

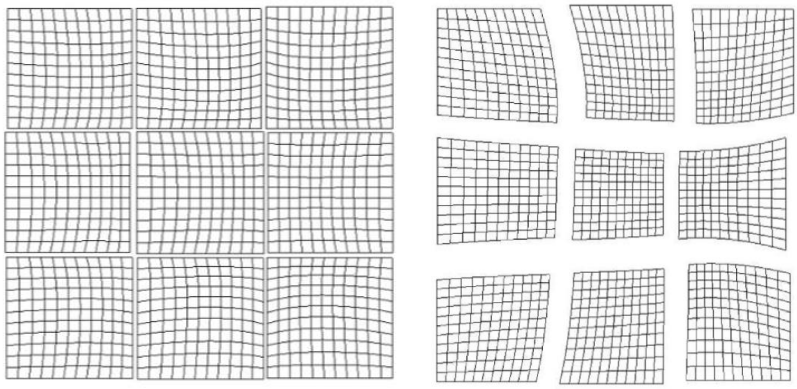

Рисунок 2. Результаты построения сетки с помощью параллельного алгоритма: слева – без передачи данных между блоками; справа – с передачей данных

При работе с декомпозицией такой необходимости не возникает из-за частичного перекрытия подблоками друг друга, и, как следствие, узлы приграничного уровня одного блока могут использоваться как граничные узлы другого. Именно из-за этого эффекта эти подблоки перекрывают друг друга и нет необходимости создавать искусственную область для вычисления значений на границах.

В реализации с фрагментированным программированием данные координат результирующей криволинейной сетки являются фрагментами расчета для каждого подблока. Их граничные данные также являются фрагментами. Как в параллельном, так и в фрагментированном алгоритме координаты узлов копируются в отдельные массивы (по 6 массивов для каждой грани кубического блока), которые возвращаются процедурами как фрагменты данных. Кроме того, координаты на каждой итерации представляют собой отдельные фрагменты данных; поэтому необходимо очистить использованные фрагменты данных.

Фрагментированное программирование в системе LuNA выполняется после описания всех фрагментов и их использования в функциях. Каждый вычислительный блок и каждый фрагмент данных автоматически распределяются в своей памяти. Вычислительный блок определяется как некоторая функция на языке C, и импортируется напрямую в исполняющую программу. Ее атрибутами являются фрагменты данных, и, в зависимости от того, какие фрагменты являются вводными и по их готовности система решает запускать тот или иной фрагмент вычислений или нет. При этом каждый фрагмент данных создается один раз. Вычислительные фрагменты могут исполняться не в заданном порядке, а по готовности входных фрагментов данных, при этом балансировка узлов и возможное копирование данных происходит автоматически.

Полученные результаты

Работа описанного параллельного алгоритма была протестирована на кластере на сетках разного размера, до 842x842x842. Тестовые расчеты проводились на суперкомпьютере «МВС» Объединенного суперкомпьютерного центра Российской академии наук, который состоит из узлов с двумя процессорами Xeon E5-2690 и 64 ГБ оперативной памяти на узел. Результаты проведенных тестов представлены в таблице 1.

Таблица 1 Тестирование параллельного алгоритма на кластере для сеток разного размера

Число потоков	*Время работы алгоритма для разного размера сетки*			
	82x82x82	*122x122x122*	*282x282x282*	*842x842x842*
8	*59,97*	*205,59*	**	**
27	———	*94,62*	———	**
64	*30,85*	*104,44*	*674,26*	*12155,2*
125	*22,58*	*69,17*	*499,7*	*7354,31*
216	———	*45,62*	———	*4533,77*
343	———	———	*315,93*	*3529,01*
512	*14,5*	*26,91*	*224,48*	*2622,24*

Для параллельного алгоритма необходимо было запускать только то количество процессов, которое является кубом целого числа, поэтому было выполнено 512 потоков, и по той же причине в некоторых полях есть «———». Поля со знаком ** означают, что для такого запуска памяти каждого узла было недостаточно. В связи с этим сложно говорить об эффективности алгоритма по сравнению с последовательным кодом. Однако можно повысить эффективность, взяв запуск на 64 потоках в качестве эталонного запуска. Например, тогда эффективность запуска, то есть отношение выигрыша в скорости к увеличению количества потоков, для 512 потоков будет 0,58, общий график показан на рисунке 5. Как правило, рисунок 5 показывает, что эффективность распараллеливание резко не снижается.

Заключение

В статье описывается работа над алгоритмом, направленным на построение адаптированных числовых сеток путем решения обратного уравнения Бельтрами с использованием конечно-дифференциальных методов. Этот алгоритм был распараллелен и запущен в системе LuNA для автоматической балансировки узлов. Алгоритм распараллеливания специфичен для задачи построения сетки. Алгоритм был запущен на кластере с несколькими настройками. Поскольку для представленного количества узлов последовательной сетки не удалось запустить последовательный алгоритм, здесь приводится график эффективности параллелизации по сравнению с запуском на 64 потоках (рисунок 3).

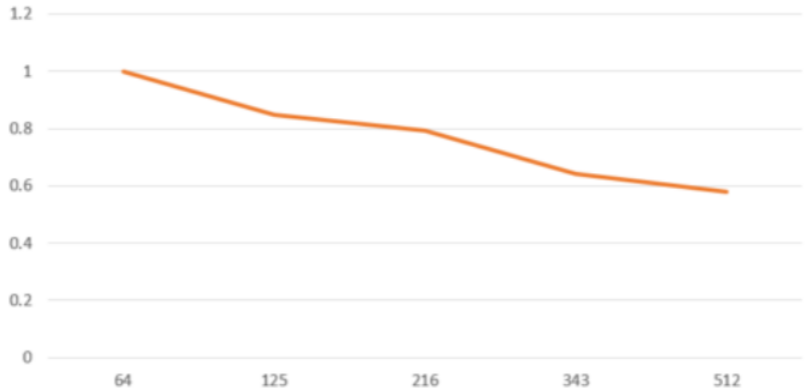

Рисунок 3. Эффективность параллельного алгоритма на большом количестве потоков по сравнению с запуском на 64 потоках

Дальнейшая работа будет в основном сосредоточена на сравнении времени работы алгоритма с чисто параллельной реализацией на машине с распределенной памятью. Ожидается, что фрагментированный алгоритм покажет меньшую эффективность, чем чистая параллельная реализация. Хотя с учетом того, что фрагментированный алгоритм балансируется автоматически, считается достаточным иметь 10-15% потери эффективности автоматизированной системы параллелизации.

Благодарности

Работа выполнена при финансовой поддержке Комитета науки Министерства образования и науки Республики Казахстан (грант № BR05236447).

Список использованной литературы:

1 Ахмед-Заки Д.Ж., Дарибаев Б.С., Иманкулов Т.С., Турар О.Н. Высокопроизводительные вычисления задачи добычи нефти на мобильной платформе с использованием технологии CUDA. Евразийский журнал математических и компьютерных приложений. - 2017. Том 5, Выпуск 2. - С. 4-13.

2 Иманкулов Т.S., Д.Ж. Ахмед-Заки, Б.С. Дарибаев и О.Н. Турар. Мобильная платформа HPC для решения проблемы добычи нефти. Труды 13-й Международной конференции по информатике в управлении, автоматизации и робототехнике (ICINCO 2016), Том 2 Лиссабон, Португалия. 29 - 31 июля 2016 г. - С. 595-598.

3 Ахмед-Заки Д., Данаев Н., Мухамбетжанов С., Иманкулов Т. Анализ и оценка процессов тепломассопереноса в пористых средах на основе модели Дарси-Стефана. ECMOR XIII - 13-я Европейская конференция по математике добычи нефти. 2012 г.

4 Ахмед-Заки Д.З., Лебедев Д.В., Перепелкин В.А. 2016. Реализация трехмерной численной модели трехфазного потока жидкости («нефть-вода-газ») в системе фрагментированного программирования LuNA. Журнал суперкомпьютеров, SI: технологии параллельных вычислений. Springer.

5 Перепелкин В.А., Беляев Н. .. 2017. Автоматизированная поддержка графических процессоров в системе фрагментированного программирования LuNA. Параллельные вычислительные технологии - 14-я Международная конференция, РаСТ 2017, Нижний Новгород, Россия, 4-8 сентября 2017 г., Труды.

6 Малышкин В.Е., Щукин Г.А. Распределенный алгоритм динамического отображения многомерных данных на многомерном мультикомпьютере в системе фрагментированного программирования LuNA. Параллельные вычислительные технологии - 14-я Международная конференция, РаСТ 2017, Россия, 2017.

7 Гроссер Т., Грёсслингер А. и Ленгауэр Ч. Polly - Выполнение многогранных оптимизаций на низкоуровневом промежуточном представлении. Письма о параллельной обработке, 22 (4): статья 1250010, 28 страниц, декабрь 2012 г.

8 Родригес Г., Мартин М. Дж., Гонсалес П., Турино Дж., Доалло Р. Проверка параллельных кодов с помощью компилятора: опыт Cetus и LLVM. Международный журнал параллельного программирования, 2013 г., стр. 782-805.

9 Удай Бондхугула, Мутху Баскаран и Шрирам Кришнамурти, Дж. Рамануджам, А. Рунтев и П. Садаяппан Автоматические преобразования для минимизированной коммуникации распараллеливания и оптимизации локальности в многогранной модели. Международная конференция по строительству компиляторов (ETAPS CC) 2008 г.

10 Палковски М., Белецки В. TRACO: Распараллеливающий компилятор от исходного кода к исходному. Вычислительная техника и информатика 35 (6): 1277-1306 (2016).

11 Рамон-Кортес К., Рамон Амела, Хорхе Эжарк, Филипп Клаусс, Роза М. Бадиа. Автопараллель: модуль Python для автоматического распараллеливания и распределенного выполнения гнезд аффинных циклов. 8-й семинар по Python для высокопроизводительных и научных вычислений (PyHPC 2018)

12 Грибл М. Автоматическое распараллеливание программ цикла для архитектур с распределенной памятью. Университет Пассау, хабилитация (2004 г.)

13 Лисейкин В.Д., Шокин Ю.И., Васева И.А., Лиханова Ю.В. Технология построения разностных сетей. Новосибирск: Наука, 2009. - 414 с.

14 Ахмед-Заки Д.Ж., Лебедев Д.В., Перепелкин В.А. Сравнение эффективности параллельных реализаций метода развертки: метод параллельного конвейера, параллельная развертка. Вестник КазНУ, серия математика, механика, информатика 3 (91), Алматы, 2016. С. 75-85.

15 Яненко Н.Н., Коновалов А.Н., Бугров А.Н., Шустов Г.В. Об организации параллельных вычислений и «распараллеливания» прогонов. Вычислительные методы механики сплошных сред. 1978. Т. 9.№ 7. С. 139-146

16 Быков А.Н., Ерофеев А.М., Сизов Е.А., Федоров А.А. Метод прогона гибридного компьютерного распараллеливания. Вычислительные методы и программирование. 2013 г. 14. - С.43-47.

References

1 Ahmed-Zaki D.Zh., Daribaev B.S., Imankulov T.S., Turar O.N. (2017) Vysokoproizvoditel'nye vychislenija zadachi dobychi nefti na mobil'noj platforme s ispol'zovaniem tehnologii CUDA [High-performance computing of oil production tasks on a mobile platform using CUDA technology]. Evrazijskij zhurnal matematicheskih i komp'juternyh prilozhenij. Tom 5, Vypusk 2. 4-13. (In Russian)

2 Imankulov T.S., D.Zh. Ahmed-Zaki, B.S. Daribaev i O.N. Turar (2016). Mobil'naja platforma HPC dlja reshenija problemy dobychi nefti [Mobile PC platform for solving the problem of oil production]. Trudy 13-j Mezhdunarodnoj

konferencii po informatike v upravlenii, avtomatizacii i robototehnike (ICINCO 2016), Tom 2 Lissabon, Portugalij. 595-598. (In Russian)

3 Ahmed-Zaki D., Danaev N., Muhambetzhanov S., Imankulov T.(2012) Analiz i ocenka processov teplomassoperenosa v poristyh sredah na osnove modeli Darsi-Stefana [Analysis and evaluation of heat and mass transfer processes in porous media based on the Darcy-Stephan model]. ECMOR XIII - 13-ja Evropejskaja konferencija po matematike dobychi nefti. (In Russian)

4 Ahmed-Zaki D.Z., Lebedev D.V., Perepelkin V.A. 2016. Realizacija trehmernoj chislennoj modeli trehfaznogo potoka zhidkosti («neft'-voda-gaz») v sisteme fragmentirovannogo programmirovanija LuNA [Implementation of a three-dimensional numerical model of a three-phase liquid flow ("oil-water-gas") in the Luna fragmented programming system]. Zhurnal superkomp'juterov, SI: tehnologii parallel'nyh vychislenij. Springer. (In Russian)

5 Perepelkin V.A., Beljaev N. 2017. Avtomatizirovannaja podderzhka graficheskih processorov v sisteme fragmentirovannogo programmirovanija LuNA [Automated GPU support in the Luna Fragmented Programming system]. Parallel'nye vychislitel'nye tehnologii - 14-ja Mezhdunarodnaja konferencija, PaCT 2017, Nizhnij Novgorod, Rossija, Trudy. (In Russian)

6 Malyshkin V.E., Shhukin G.A.(2017) Raspredelennyj algoritm dinamicheskogo otobrazhenija mnogomernyh dannyh na mnogomernom mul'tikomp'jutere v sisteme fragmentirovannogo programmirovanija LuNA [Distributed algorithm for dynamic display of multidimensional data on a multidimensional multicomputer in the fragmented programming system LuNA]. Parallel'nye vychislitel'nye tehnologii - 14-ja Mezhdunarodnaja konferencija, PaCT 2017, Rossija. (In Russian)

7 Grosser T., Grjosslinger A. i Lengaujer Ch. Polly (2012) - Vypolnenie mnogogrannyh optimizacij na nizkourovnevom promezhutochnom predstavlenii [Performing polyhedral optimizations on a low-level intermediate representation]. Pis'ma o parallel'noj obrabotke, 22 (4): stat'ja 1250010, 28. (In Russian)

8 Rodriges G., Martin M. Dzh., Gonsales P., Turino Dzh., Doallo R.(2013) Proverka parallel'nyh kodov s pomoshh'ju kompiljatora: opyt Cetus i LLVM [Checking parallel codes with the compiler: the experience of Cetus and LLVM]. Mezhdunarodnyj zhurnal parallel'nogo programmirovanija, 782-805. (In Russian)

9 Udaj Bondhugula, Muthu Baskaran i Shriram Krishnamurti, Dzh. Ramanudzham, A. Runtev i P. Sadajappan (2008) Avtomaticheskie preobrazovanija dlja minimizirovannoj kommunikacii rasparallelivanija i optimizacii lokal'nosti v mnogogrannoj modeli [Automatic transformations for minimized communication parallelization and locality optimization in a polyhedral model]. Mezhdunarodnaja konferencija po stroitel'stvu kompiljatorov (ETAPS CC). (In Russian)

10 Palkovski M., Belecki V. (2016) TRACO: Rasparallelivajushhij kompiljator ot ishodnogo koda k ishodnomu [TRACO: Parallelizing compiler from source code to source code]. Vychislitel'naja tehnika i informatika 35: 1277-1306. (In Russian)

11 Ramon-Kortes K., Ramon Amela, Horhe Jezhark, Filipp Klauss, Roza M. Badia. (PyHPC 2018) Avtoparallel': modul' Python dlja avtomaticheskogo rasparallelivanija i raspredelennogo vypolnenija gnezd affinnyh ciklov [Auto Parallel: A Python module for automatic parallelization and distributed execution of affine loop sockets]. 8-j seminar po Python dlja vysokoproizvoditel'nyh i nauchnyh vychislenij. (In Russian)

12 Gribl M. (2004) Avtomaticheskoe rasparallelivanie programm cikla dlja arhitektur s raspredelennoj pamjat'ju [Automatic parallelization of loop programs for distributed memory architectures]. Universitet Passau, habilitacija. (In Russian)

13 Lisejkin V.D., Shokin Ju.I., Vaseva I.A., Lihanova Ju.V. (2009) Tehnologija postroenija raznostnyh setej [Technology for constructing difference networks]. Novosibirsk: Nauka. 414. (In Russian)

14 Ahmed-Zaki D.Zh., Lebedev D.V., Perepelkin V.A. (2016) Sravnenie jeffektivnosti parallel'nyh realizacij metoda razvertki: metod parallel'nogo konvejera, parallel'naja razvertka [Comparison of the efficiency of parallel implementations of the sweep method: parallel pipeline method, parallel sweep]. Vestnik KazNU, serija matematika, mehanika, informatika 3 (91), Almaty. 75-85. (In Russian)

15 Janenko N.N., Konovalov A.N., Bugrov A.N., Shustov G.V.(1978) Ob organizacii parallel'nyh vychislenij i «rasparallelivanija» progonov [About the organization of parallel computing and" parallelization " of runs]. Vychislitel'nye metody mehaniki sploshnyh sred. T. 9.№ 7. 139-146. (In Russian)

16 Bykov A.N., Erofeev A.M., Sizov E.A., Fedorov A.A. (2013) Metod progona gibridnogo komp'juternogo rasparallelivanija [A method for running hybrid computer parallelization. Computational methods and programming]. Vychislitel'nye metody i programmirovanie. 14. 43-47. (In Russian)