Параллельная композиция в поэлементной схеме метода конечных элементов*

С.П. Копысов, А.К. Новиков, Н.К. Пиминова

Институт механики УрО РАН

Предлагается подход к параллельной композиции в поэлементной схеме метода конечных элементов, состоящий из двух этапов: разделения сетки с заданными свойствами (локализация несвязанных сеточных данных) и сборки (суммирования) распределенных векторов. Таким образом, обеспечивается более регулярный доступ к компонентам конечно-элементных векторов и уменьшение числа конфликтов при обращении к подсистеме памяти. Используя отношение соседства, формируются подмножества несвязанных конечных элементов, исключающие состояние гонки при параллельном суммировании поэлементных векторов в поузловые. Рассматриваются несколько вариантов композиции.

Ключевые слова: параллельная композиция, упорядочение неизвестных, разделение неструктурированной сетки, поэлементная конечно-элементная схема.

1. Введение

Вычислительные схемы решения конечно-элементных систем, исключающие явное формирование матрицы коэффициентов (глобальной матрицы жесткости), за счет переноса вычислений на уровень отдельных конечных элементов принято называть поэлементными (element-by-element) [1, 2]. Такие схемы позволяют не только обходиться без хранения матрицы коэффициентов системы, но и более эффективно использовать кэш-память за счет локализации данных [2]. Основным отличием поэлементной схемы является способ вычисления матрично-векторного произведения, при котором используются два варианта организации вычислений: с извлечением данных в вектор меньшего размера и разнесением данных в вектор большего размера [3].

В большинстве случаев конечно-элементные аппроксимации строятся на неструктурированных расчетных сетках, что приводит к нерегулярному доступу к сеточным данным существенно уменьшающим эффективность параллельных вычислений. Узким местом поэлементных конечно-элементных схем является суммирование компонент из поэлементных векторов. При параллельном суммировании компонент, соответствующих общим узлам сетки возникает, так называемое состояние гонки, когда несколько параллельных вычислительных процессов обращаются к одной ячейке памяти. Данная операция суммирования включает в себя: чтение текущего значения компоненты, чтение компоненты из поэлементного вектора, сложение текущего значения с поэлементной компонентой и запись результата. Состояние гонки приводит к вычислительным ошибкам, которые могут быть связаны, как с чтением текущего значения компоненты вектора, так и с записью результата.

Для того, чтобы исключить такие ошибки предлагается упорядочение конечно-элементных неизвестных, при котором одновременно в суммировании не участвуют конечные элементы, содержащие общие вершины. Используя отношения соседства, выделяются слои конечных элементов, при помощи которых сетка разделяется на подмножества конечных элементов для каждого параллельного вычислительного процесса. Поэлементные векторы наследуют полученное упорядочение степеней свободы. Таким образом суммирование (сборка) векторов рассматривается во взаимосвязи с упорядочением, как часть поэлементной операции композиции.

*Работа выполнена при поддержке РФФИ (проекты 16-01-00129-а, 14-01-00055-а).

2. Композиция в поэлементной схеме метода конечных элементов

Конечно-элементные вычисления предполагают применение алгоритмов, в которых кроме параллельных и последовательных вычислений существуют операции, объединяющие (суммирующие) данные из параллельных ветвей алгоритма, на основе некоторого разделения. Эти операции будем называть композицией или операциями композиции. Будем полагать, что композиция состоит из двух этапов, взаимосвязанных, но разнесенных по времени выполнения: 1) разделение с заданными свойствами (локализация данных и сбалансированность вычислительной нагрузки); 2) объединение / суммирование (сборка) распределенных данных. Этап разделения предшествует соответствующему шагу вычислительной схемы метода конечных элементов, а этап сборки является частью этого шага.

Как правило, в методе конечных элементов под сборкой (ансамблированием) $\mathcal{A}$ подразумевается суммирование локальных матриц жесткости конечных элементов в глобальную матрицу жесткости системы уравнений $K = \mathcal{A}^{\mathcal{T}}\tilde{K}\mathcal{A}$ [4, 5], где $K \in \mathbb{R}^{N\times N}$ — глобальная матрица жесткости; $\tilde{K} \in \mathbb{R}^{\tilde{N}\times\tilde{N}}$ — блочно-дигональная матрица, составленная из локальных матриц жесткости конечных элементов; $\tilde{N} = \sum_{e=1}^{m} N_e$, здесь m — число конечных элементов, N_e – число степеней свободы в конечном элементе e. Данная операция осуществляется при помощи отображения локального пространства номеров неизвестных (степеней свободы) $[1, 2, \ldots, N_e]$ в глобальное $[1, 2, \ldots, N]$. Операция реализуется или в виде косвенной индексации неизвестных, или в виде арифметических операций — умножения на матрицу инцидентности, которое в поэлементных схемах эффективно выполняется на графических ускорителях [6]. В поэлементных схемах сборка переносится на этап решения конечной элементной системы и применяется уже не к матрицам, а к векторам в виде $q = Kp = \mathcal{A}^T\tilde{K}\mathcal{A}p = \mathcal{A}^T\tilde{q}$. Это позволяет разделить произведение $q = Kp$ на две операции: поэлементное произведение $\tilde{q} = \tilde{K}\mathcal{A}p$ и сборку $q = \mathcal{A}^T\tilde{q}$, существенно отличающиеся, как по степени параллелизма, так и, в случае неструктурированных сеток, по локализации данных.

Для поэлементных вычислительных схем, в качестве основного варианта разделения будем рассматривать упорядочение с выделением несвязанных подмножеств конечных элементов. Каждому такому подмножеству ставится в соответствие параллельный процесс и/или множество нитей, а этап сборки применяется к несвязанным конечным элементам.

3. Разделение сетки на подмножества ячеек.

Будем считать, что два подмножества ячеек сетки не связаны в данный момент времени, если одновременно извлекаемые из этих подмножеств ячейки сетки не содержат общих вершин. Таким образом, ограничение на связанность ячеек накладывается только на момент обращения к подмножеству. Это означает, что ячейки внутри подмножества и сами подмножества могут быть топологически связаны.

Соответствующие разделения сетки вычисляются при помощи разделения сетки на неперекрывающиеся слои ячеек и последующего объединения слоев (ячеек) в подмножества ячеек (блоки). Разделение сетки на слои осуществляется, исходя из отношения соседства ячеек по следующему алгоритму: задается начальный слой s_1; последующие слои определяются из выражения $s_i = Adj(s_{i-1}) \setminus s_{i-1}$. Здесь $s_i = \{e_j^{(i)}\}$ — множество ячеек (конечных элементов) сетки в слое i; $e_j^{(i)}$ — ячейка с номером j в слое i; $Adj(s_i) = \bigcup_{j=1}^{m_i} Adj(e_j^{(i)})$ — множество ячеек сетки соседних (смежных) со слоем i; m_i — число ячеек в слое. При формировании слои ячеек сетки нумеруются и определяется их число — n_s. Отношение соседства для ячеек сформулируем следующим образом: две ячейки сетки будем считать соседними, если они содержат хотя бы один общий узел.

На рис. 1 представлены 64 слоя, полученные для неструктурированной сетки, состоящей

из 31744 шестигранников и 36873 узлов. Цветами выделены слои с четными и нечетными номерами. По построению ячейки из разных слоев одного цвета не являются соседними.

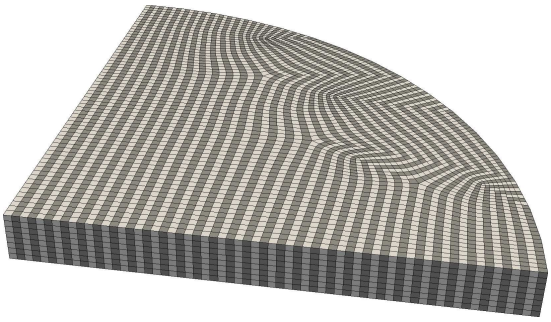

Рис. 1. Разделение сетки на слои.

На основе полученных слоев сетки рассматривалось несколько вариантов построения подмножеств ячеек:

1. *Блочный.* Объединение слоев сетки $S = \bigcup_{i=1}^{n_s} s_i$ в соответствии с порядковыми номерами слоев, далее разделение полученного объединения на подмножества S_i, $i = \overline{1, n_p}$ размера m/n_p, где n_p — число параллельных вычислительных процессов (рис. 2);
2. *Нечетно-четный.* Объединение в подмножества $S_i, i = \overline{1, n_p}$ сначала всех слоев с нечетными номерами, а затем с четными (рис. 3а). В качестве примера на рис. 3б показано подмножество S_1.

Рис. 2. Подмножества ячеек сетки, полученные при блочном варианте построения.

а) б)

Рис. 3. Нечетно-четный вариант построения подмножеств ячеек: а) группы из восьми слоев, ячейки из которых участвуют в параллельных вычислениях, выполняемых $n_p = 8$ процессами; б)последовательность слоев сетки, образующих подмножество S_1, над ячейками которого процесс с номером ноль выполняет сборку $q = \mathcal{A}_1^T \tilde{q}$, здесь $\mathcal{A}_1^T \subset \mathcal{A}^T$.

Как видно из рис. 2, в блочном варианте некоторые слои неструктурированной сетки разделены на части, принадлежащие соседним подмножествам. Благодаря разделению слоев получаются равные по числу ячеек подмножества и соответственно достигаются равные вычислительные нагрузки на параллельные процессы в поэлементной схеме.

Следует отметить, что в нечетно-четном варианте топологически связанные нечетные и четные слои разнесены в блоках по времени выполнения вычислений (рис.3б). Одновременно в вычислениях задействованы ячейки (рис.3а) из восьми несвязанных слоев.

4. Результаты вычислительных экспериментов.

Остановимся на предварительных результатах вычислительных экспериментов. Предложенные варианты распараллеливания операции композиции апробировались на сетках шестигранных конечных элементов, применяемых при решении динамической задачи теории упругости в трехмерной постановке, и сопряженных задач деформирования и газовой динамики [3]. Вычислительные эксперименты осуществлялись на восьмиядерном узле кластера X4 (Институт механики УрО РАН), в котором установлены: два CPU Intel Xeon E5-2609 Quad Core 2.4GHz, 64 ГБ оперативной памяти. Программное обеспечение: операционная система — Linux 3.2.68, компилятор — g++ 4.7.2, при компиляции приложения применялась оптимизация третьего уровня.

Распараллеливание операции сборки $q = \mathcal{A}^T\tilde{q}$ выполнялось в рамках модели общей памяти средствами OpenMP: рассматривались варианты распараллеливания цикла по конечным элементам при помощи директивы OpenMP `for` и явное распараллеливание, исходя из номера нити.

Следует отметить, что на неструктурированной сетке с упорядочением ячеек (по построению, операция сборки вызывала состояние гонки, что приводило к вычислительной погрешности. До применения упорядочения, состояние гонки устранялось при помощи директивы `critical`. В таком случае время сборки было существенно больше, чем в последовательном варианте. Предложенное упорядочение ячеек, одновременно участвующих в сборке, позволило исключить синхронизацию при записи в вектор q из параллельных вычислительных процессов (нитей OpenMP).

На неструктурированных сетках для четвертой части мембраны (подобных сетке, показанной на рис.1) с числом ячеек $m = 107136, 253952, 507904$ получено ускорение 4.2 – 5.1 раза на восьми ядрах и 3.1 – 3.2 раза на четырех ядрах центральных процессоров одного вычислительного узла. В ряде случаев, за счет введенного упорядочения, наблюдалось полутора-трехкратное ускорение в последовательном варианте сборки. Данные результаты были получены без учета архитектурных особенностей вычислительного узла, связанных с неоднородным доступом к памяти (NUMA).

Сочетание предложенного упорядочения с изменением доступа к памяти при помощи утилиты numactl с опцией interleave=all позволило получить ускорение в 6.3 раза при сборке на сетке из 507904 шестигранников на восьми ядрах двух процессоров, за счет уменьшения доступа в память ассоциированную с другим процессором вычислительного узла. Ускорение на поэлементном произведении $\tilde{q} = \tilde{K}\mathcal{A}p$ составило примерно 7.7 раза при отсутствии зависимости по данным (каждая нить вычисляла произведение локальной матрицы жесткости на локальный вектор). Как было отмечено ранее, произведении $q = Kp$, включает поэлементное произведение и сборку вектора q. В этом случае получено ускорение в 7.6 раза.

5. Заключение.

Благодаря исключению синхронизации, при применении поэлементной композиции получено примерно шестикратное ускорение в операции сборки векторов на восьмиядерном

узле кластера относительно наилучшего последовательного варианта алгоритма сборки векторов для поэлементной схемы метода конечных элементов.

Дальнейшие исследования поэлементой композиции будут направлены на выявление факторов наиболее влияющих на ускорение вычислений, таких как нумерация компонент результирующего вектора, упорядочение ячеек в слоях сетки, размещение данных на разных уровнях подсистемы памяти вычислительного узла.

Предложенные варианты разделения сетки обобщаются на большее число процессорных ядер (MIC-архитектура, GPU) и вычислительных узлов. Ограничение на алгоритмы разделения, связанное с числом слоев сетки, которое зависит от геометрии области и ячейки сетки, а также от общего числа ячеек сетки, ослабляется введением нескольких уровней разделения.

Литература

1. Carey G.F., Barragy E., Mclay R. and Sharma M. Element-by-element vector and parallel computations // Communications in Appl. Numer. Meth. 1988. Vol. 4. pp. 299–307.

2. Копысов С.П., Новиков А.К. Метод декомпозиции для параллельного адаптивного конечно-элементного метода // Вестник Удмуртского университета. Математика. Механика. Компьютерные науки. 2010. № 3. С. 141–154.

3. Копысов С.П., Кузьмин И.М., Недожогин Н.С., Новиков А.К., Рычков В.Н., Сагдеева Ю.А., Тонков Л.Е. Параллельная реализация конечно-элементных алгоритмов на графических ускорителях в программном комплексе FEStudio // Компьютерные исследования и моделирование. 2014. Т. 6. № 1. С. 79–97.

4. Cecka C., Lew A., Darve E. Assembly of Finite Element Methods on Graphics Processors // Int. J. Numer. Meth. Engng. 2011. Vol. 85. Issue 5. pp. 640–669.

5. Markall G.R., Slemmer A., Ham D.A., Kelly P.H.J., Cantwell C.D. and Sherwin S.J. Finite element assembly strategies on multi-core and many-core architectures // Int. J. Numer. Meth. Fluids. 2013. Vol. 71. Issue 1. pp. 80–97.

6. Новиков А.К., Копысов С.П., Кузьмин И.М., Недожогин Н.С. Конечно-элементные технологии для кластеров гибридной архитектуры / Параллельные вычислительные технологии (ПаВТ'2015): труды международной научной конференции (31 марта – 2 апреля 2015 г., г. Екатеринбург). Челябинск: Издательский центр ЮУрГУ, 2015. С. 442–447.

Parallel composition in element-by-element scheme of finite element method

S.P. Kopysov, A.K. Novikov, N.K. Piminova

Institute of Mechanics UB RAS

The approach to the parallel composition in element-by-element scheme of finite element method is offered. This approach consists of two phases: partition mesh with desired properties (localization uncoupled mesh data) and assembly (summation) distributed vectors. Thus, a regular access to the components of finite-element vectors is provided and the number of conflicts when accessing the memory subsystem is reduced. Subsets of uncoupled finite elements excluding a race condition with parallel summation of vectors in a unit element by element are formed by using the ratio of the adjacency. Several variants the compositions are considered.

Keywords: parallel composition, ordering unknowns, partitioning of unstructured mesh, element-by-element scheme, finite element method.

References

1. Carey G.F., Barragy E., Mclay R. and Sharma M. Element-by-element vector and parallel computations // Communications in Appl. Numer. Meth. 1988. Vol. 4. pp. 299–307.

2. Kopysov S.P., Novikov A.K. Metod dekompozitsii dlya parallel'nogo adaptivnogo konechno-elementnogo metoda [Domain decomposition for parallel adaptive finite element algorithm]. Vestnik Udmurtskogo universiteta. Matematika. Mekhanika. Komp'yuternye nauki [The Bulletin of Udmurt University. Mathematics. Mechanics. Computer Science]. 2010. No. 3. P. 141–154.

3. Kopysov S.P., Kuzmin I.M., Nedozhogin N.S., Novikov A.K., Rychkov V.N., Sagdeeva Y.A., Tonkov L.E. Parallelnaya realizaciya konechno-elementnyh algoritmov na graficheskih uskoritelyah v programmnom komplekse FEStudio [Parallel implementation of a finite-element algorithms on a graphics accelerator in the software package FEStudio]. Komp'yuternye issledovaniya i modelirovanie [Computer Research and Modeling]. 2014. Vol. 6. No. 1. P. 79–97.

4. Cecka C., Lew A., Darve E. Assembly of Finite Element Methods on Graphics Processors // Int. J. Numer. Meth. Engng. 2011. Vol. 85. Issue 5. P. 640–669.

5. Markall G.R., Slemmer A., Ham D.A., Kelly P.H.J., Cantwell C.D. and Sherwin S.J. Finite element assembly strategies on multi-core and many-core architectures // Int. J. Numer. Meth. Fluids. 2013. Vol. 71. Issue 1. P. 80–97.

6. Novikov A.K., Kopysov S.P., Kuzmin I.M., Nedozhogin N.S. Konechno-elementnye teknologii dlya klasterov gibridnoj arhitektury [Finite element technologies for hybrid aritecture clusters] Parallelnye vychislitelnye tekhnologii (PaVT'2015): Trudy mezhdunarodnoj nauchnoj konferentsii (Ekaterinburg, 31 marta – 2 aprelya 2015) [Parallel Computational Technologies (PCT'2015): Proceedings of the International Scientific Conference (Ekaterinburg, Russia, March, 31 – April, 2, 2015)]. Chelyabinsk, Publishing of the South Ural State University, 2015. P. 442–447.