ИПМ им.М.В.Келдыша РАН

Абрау-2019 • Труды конференции

С.К. Григорьев, М.В. Якобовский

Особенности работы с данными динамического размера и сильно-связанными данными на примере задачи построения различного вида неструктурированных сеток

Рекомендуемая форма библиографической ссылки

Григорьев С.К., Якобовский М.В. Особенности работы с данными динамического размера и сильно-связанными данными на примере задачи построения различного вида неструктурированных сеток // Научный сервис в сети Интернет: труды XXI Всероссийской научной конференции (23-28 сентября 2019 г., г. Новороссийск). — М.: ИПМ им. М.В.Келдыша, 2019. — С. 290-294. — URL: http://keldysh.ru/abrau/2019/theses/07.pdf doi:10.20948/abrau-2019-07

Размещена также презентация к докладу

Особенности работы с данными динамического размера и сильно-связанными данными на примере задачи построения различного вида неструктурированных сеток

С.К. Григорьев[1], М.В. Якобовский[1]

[1] ИПМ им. М.В.Келдыша РАН

Аннотация. В работе рассматриваются вопросы обработки и передачи сложных и сильно связанных данных в задачах генерации неструктурированных сеток и работы с адаптивными к решению сетками в трёхмерных областях на многопроцессорных системах с распределённой памятью. На примере этих задач рассматриваются вопросы передачи данных, для которых невозможно априорно получить точные оценки количества памяти, резервируемой под буферы, в том числе и случай, когда объём передаваемых данных заведомо больше объёма буфера, который целесообразно зарезервировать, а так же задача передачи поддоменов адаптивных сеток при динамической балансировке нагрузки с сохранением корректных связей между отдельными элементами, а так же восстановление графа межпроцессорных взаимодействий. Работа с неструктурированными тетраэдральными сетками рассматривается на примере проекционного алгоритма гарантированной генерации тетраедральной сетки. Работа с адаптивными сетками рассматривается на примере реализации AMR на основе восьмеричных деревьев. В качестве языка программирования используется язык C++ со стандартной библиотекой, для параллельной обработки данных используется библиотека MPI.

Ключевые слова: MPI, неструктурированные сетки, адаптивные сетки.

Features of working with dynamic-length data and strongly connected complex data on the example of the task of generating various types of unstructured grids

S.K. Grigorjev[1], M.V. Yakobovskiy[1]

1 Keldysh Institute of Applied Mathematics (Russian Academy of Sciences)

Abstract. The discussion is about processing and transmitting of complex and strongly connected data in unstructured mesh generation and adaptive mesh

refinement in 3-dimensional regions for distributed calculations. These tasks are examples for transmitting large amounts of such data, that a priori assessment of necessary buffer space is impossible, including the case, when the number of transmitted data is obviously larger buffer size, which is advisable to reserve. Second example consists in transmitting an subdomain in adaptive mesh refinement while dynamically balancing load between processes with saving correct connections between separate elements and recovery processor interconnection graph. Projection-based tetrahedral mesh generation algorithm is used for generating unstructured tetrahedral grid. Adaptive mesh refinement implemented with using octree-tecnology is used for AMR part. All described software is implemented using C++ language with standard library, MPI library is used for implementing distributed calculations.

Keywords: MPI, unstructured grid, adaptive mesh refinement

При численном моделировании физических процессов методами механики сплошной среды широко используются геометрические расчетные сетки.

Во время выполнения всех этапов алгоритма построения тетраэдральной сетки проекционным методом все координаты узлов хранятся в рациональной арифметике. Для балансировки нагрузки [1] на распределённой системе необходимо передавать эти координаты между MPI-процессами.

Применение рациональных чисел произвольной разрядности для хранения координат создаёт сложности при передаче данных между процессорами из-за того, что для хранения числителя и знаменателя рациональной дроби используются динамические массивы, не поддерживаемые для регистрации в MPI.

В силу того, что общий объём оперативной памяти, требуемой для хранения координат, достаточно велик, использование статического буфера, заведомо вмещающего в себя все передаваемые элементы, представляется неэффективным, т.к. такой буфер потребляет слишком много памяти.

Для уменьшения затрат по памяти можно использовать буфер обмена меньшего размера, чем необходимый для приёма и передачи всех необходимых узлов, при передаче разбивая весь массив передаваемых данных на несколько массивов меньшего размера, которые можно по мере формирования передавать. Таким образом, при передаче данных большого объёма функция MPI_Send вызывается несколько раз, вместо одного.

Другой проблемой является невозможность регистрации типов данных с динамическими массивами как типов MPI. Под типами данных с динамическими массивами понимаются такие структуры, длины полей двух различных объектов которых не совпадают. При помощи таких структур в том числе реализуются числа произвольной разрядности в [2].

Чтобы передать рациональное число с произвольной разрядностью, необходимо создание механизма преобразования из рационального числа к целочисленному типу, поддерживаемому MPI. Для реализации такого

преобразования числитель и знаменатель каждого рационального числа записываются в буфер последовательно.

Для передачи целого числа произвольной разрядности реализуется алгоритм преобразования такого числа в несколько целых чисел (int) и обратно, что позволяет передавать числа произвольной разрядности с помощью MPI.

Таким образом, для приёма и передачи рациональных чисел используется целочисленный буфер, содержащий преобразованные в массивы целых числе числа произвольной разрядности.

Время работы этого алгоритма оказывается существенно меньше времени расчета: для поверхности, заполненной 20474 призмами, время передачи данных между процессорами составило от 0.238 до 0.289 секунды, при общем времени построения 657.159 секунд (10.9526 минут). Размер буфера составил 10 000 элементов типа int (32-х разрядных), при этом количество передаваемых пакетов составляло 30-40.

Ещё одним случаем работы с большим объёмом данных при работе с расчетными сетками является динамическая балансировка нагрузки, реализуемая для AMR. В этом случае основной проблемой является поддержание связности данных при передаче сеточных поддоменов между вычислительными узлами. В работе рассматривается реализация AMR на основе octree-технологии.

При динамической балансировке прием и передача поддоменов сетки сопряжена с необходимостью не только передать сеточные элементы, но и граф смежности между ними, а также сохранить корректные взаимосвязи между сеточными элементами на разных вычислительных узлах.

Эта проблема возникает из-за того, что в процессе работы количество сеточных элементов меняется независимо на различных процессорах, что приводит к необходимости создания дополнительных структур при приёме и передаче сеточных элементов.

Передача поддомена разделяется на два этапа: «упаковку» и передачу, а приём – на приём и «распаковку». Упаковка подразумевает копирование всех необходимых элементов в буфер приёма-передачи, и пересчет всех соответствующих индексов, а именно: графа смежности, отношений родительства и отношений инцидентности. Распаковка происходит в обратном порядке, но с небольшими отличиями: так, при распаковке некоторые сеточные элементы могут вообще не копироваться, так как они уже присутствуют на принимающем процессе.

Так же, следует рассмотреть вопрос перестроения фиктивных ячеек.

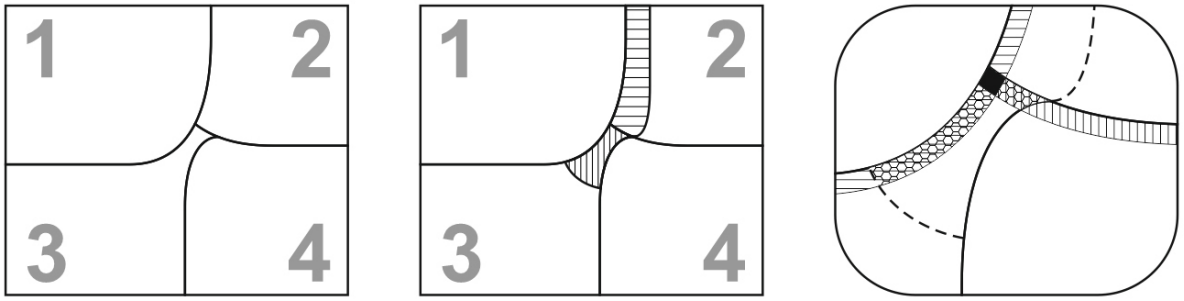

Рис. 1. Перестроение полей при балансировке нагрузки

На рис. 1 область, заштрихованная горизонтально, передаётся от процесса 2 к процессу 1, а область, заштрихованная вертикально – от процесса 3 к процессу 4. В правой части крупным планом изображена интересующая нас область. Горизонтальной штриховкой обозначена область виртуальных ячеек первого процесса. Вертикальной – часть поля виртуальных ячеек второго процесса (не показана та часть поля, которая принадлежит первому процессу). Пунктирной линией обозначены границы передаваемых областей. При передаче области от процесса 3 к процессу 4, возникает несоответствие нумерации между реальным обновлённым положением ячеек полей и теми данными, которые находятся на процессах 1,2 (на рисунке показана гексаэдральной штриховкой).

Решением этой проблемы является поэтапное удаление перемещённых ячеек: при упаковке ячеек передаваемые ячейки не удаляются, а помечаются как передаваемые. После завершения передачи данных, распаковки и присоединения переданной подобласти, новые номера ячеек возвращаются исходному процессу, и эти номера запоминаются.

Разработан и реализован комплекс программ, позволяющий осуществлять передачу сеточных элементов с координатами в рациональной арифметике между MPI-процессами. Так же, разработан и реализован программный комплекс передачи древовидных элементов адаптивных сеток. Таким образом, появился функционал работы с координатами произвольной длинны, а также с сильно связанными наборами данных для многопроцессорных систем.

Литература

1. Шакиров Р. Н. Применение методов обнаружения и компенсации ошибок в целочисленном классе cBigNumber // Программирование. – 2010. No 1. С. 50-65 (R.N.Shakirov C++ class for integers of unlimited range URL: http://www.imach.uran.ru/cbignum) [Электронный ресурс]. – Режим доступа: http://www.imach.uran.ru/cbignum
2. Grigorjev S.K., Yakobovskiy M.V. (2018) Static Balancing Methods in Projection-Based Mesh Generation Algorithm. In: Sokolinsky L., Zymbler M.

(eds) Parallel Computational Technologies. PCT 2018. Communications in Computer and Information Science, vol 910. Springer, Cham

References

1. Shakirov, R.N. Program Comput Soft (2010) 36: 36. https://doi.org/10.1134/S0361768810010068
2. Grigorjev S.K., Yakobovskiy M.V. (2018) Static Balancing Methods in Projection-Based Mesh Generation Algorithm. In: Sokolinsky L., Zymbler M. (eds) Parallel Computational Technologies. PCT 2018. Communications in Computer and Information Science, vol 910. Springer, Cham