Обращение матрицы методом Гаусса с выбором главного элемента по строке

Подкорытов Максим

26 апреля 2013 г.

1 Особенности MPI-реализации обращения матрицы с использованием алгоритма Гаусса с поиском главного элемента по всей матрице.

Далее, A — это обращаемая матрица, B — вспомогательная матрица, соответствующая правой части, G — массив индексов переставляемых столбцов. Также, пусть у нас есть p процессов, s — число блоков в одной блочной строке, n — длина стороны обращаемой матрицы, m — длина стороны блока. Матрицы между процессами распределяем по строкам (процессу с номером k достаются блочные строки с номерами k, $k+p$,..., $k+p(\frac{s+p-1}{p}-1)$). Для облегчения работы с обменом данными между процессами, выделим во всех процессах под матрицы одинаковое количество памяти ($\frac{s+p-1}{p}mn$), и ту часть выделенной памяти, в которой не будут лежать элементы матрицы, заполним нулями.

Основные этапы работы программы

Прямой ход алгоритма Гаусса. Имеется цикл длины s; будем считать, что в текущий момент мы находимся на шаге с номером i.

1. Вычисляем первую строчку рабочей области.
2. Ищем главный элемент в рабочей области. После поиска делаем Allreduce, после которого каждому процессу становится известен номер процесса $rank$, в котором лежит главный блок, и координаты блока в этом процессе.
3. Делаем перестановку столбцов в матрице A; поскольку она роздана по строкам, синхронизация не нужна. Делаем перестановку индексов в массиве G.
4. Процесс с номером $rank$ умножает строку матрицы A, в которой содержится главный блок, на этот блок. То же самое делается с матрицей B. Затем процесс с номером $rank$ делает BCast этих строк всем процессам.
5. Процесс, в котором содержится i-я строка матриц A и B, отсылает их процессу с номером $rank$, и записывает на их место строки, принятые при BCast на прошлом шаге. Процесс с номером $rank$ просто записывает на место отосланных строк принятый строки.

6. Все процессы вычитают из строк рабочей области (для определенности, с номером) принятую строчку, домноженную на i-й блок в -й строке слева. При этом процесс, в котором лежит строка с глобальным номером i, из этой строки ничего не вычитает.

Обратный ход алгоритма Гаусса. i — текущий шаг цикла, цикл проходится от $s-1$ до .

1. Вычисляем номер процесса $rank$, в котором лежит строчка с номером i и локальный номер a_i этой строчки. Назовём эту строчку опорной, для определенности. Затем делается Bcast строчки матрицы B с номером a_i и коренным процессом $rank$.

2. Вычисляем рабочую область — те строчки матрицы B, которые в неразбитой матрице лежат выше опорной строчки.

3. Вычитаем из строк рабочей области (для определенности, с номером , который принимает все возможные значения в рабочей области) опорную строку, домноженную на i-й блок -й строки матрицы A. При этом блоки опорной строки домножаются на соответствующий блок слева.

Подсчет невязки

Для экономии сил, переинициализируем матрицу так, что она будет роздана по столбцам. Тогда будет удобно посчитать каждый блок матрицы $(A^{-1}A-\ )$, и затем посчитать невязку.

Расчет пересылаемых данных и количества точек синхронизации

Точки синхронизации

После того, как найден главный блок, нужно сообщить его координаты всем процессам.

На каждом шаге, после окончания вычитания из своих строк, процессы начинают ждать BCast (начало нового шага).

Пересылка первой строки рабочей области процессу, нашедшему главный блок.

Пересылка строки матрицы B во время обратного хода.

Итого, порядка s синхронизаций.

Пересылаемые данные

Пересылка информации о том, что главный блок найден и координат главного блока: si $(in\ )$ на каждом шаге прямого хода алгоритма Гаусса.

Swap строчек после поиска главного элемента: $mn\,si$ ( ) на каждом шаге прямого хода алгоритма Гаусса.

Пересылка вычитаемой строчки при обратном ходе алгоритма Гаусса: $mn\,si$ ( ) на каждом шаге.

Получается всего $n + (n)$.