ПАРАЛЛЕЛЬНОЕ УМНОЖЕНИЕ РАЗРЕЖЕННОЙ МАТРИЦЫ НА ВЕКТОР

Штейнберг Б.Я.

В данной работе предлагается алгоритм умножения разреженной матрицы на вектор для параллельных компьютеров с разделяемой памятью. Рассматриваемый алгоритм может быть эффективен при многократном умножении матрицы на вектор. Такая задача встречается в итерационных численных методах решения дифференциальных уравнений, например, в задачах моделирования слухового восприятия человека [1]. Для параллельных компьютеров с общей памятью методы автоматического распараллеливания таких задач исследовались в [12].

Разреженная матрица – это матрица, в которой подавляющее большинство элементов равно нулю. Такие матрицы можно хранить в памяти компьютера особым компактным образом. Идея компактного хранения состоит в том. чтобы в памяти имелись только ненулевые элементы и индексы этих элементов в матрице [2]. С одной стороны, это экономит память компьютера, а, с другой стороны, поскольку считывание матрицы происходит быстрее, экономится и время обработки.

Умножение разреженной матрицы на вектор может быть записано следующим фрагментом программы

```
      DO  99  I = 1, N
      Y(I) = 0                                          (*)
      DO  99  J = 1, M
99    Y(I) = Y(I) + A(I,J)*X(U(I,J))
```

Здесь X – входной вектор, Y – выходной вектор, A – матрица, строки которой состоят из ненулевых элементов исходной матрицы, U – матрица индексов, элемент которой U(I,J) равен номеру столбца J-го ненулевого элемента в I-ой строке исходной разреженной матрицы.

Параллельный компьютер с разделяемой памятью.

Будем рассматривать компьютер с N вычисляющими устройствами (процессорами) и N модулями памяти. Будем считать, что для любой перестановки из N элементов s коммуникационное устройство (коммутатор) компьютера позволяет одновременно организовать N соединений: каждый процессор с номером i соединяется с модулем памяти, имеющим номер s_i .

Одним из наиболее интересных американских суперкомпьютеров с разделяемой памятью был компьютер RP-3 фирмы IBM [5]. В России разделяемую память используют в суперкомпьютерах со структурно-процедурной организацией вычислений, которые делают в ТРТУ в городе Таганроге. Такой компьютер имеет сегментированную совместно используемую память (разделяемую) и поле процессоров, из которых с помощью коммутатора может собираться конвейер любой конфигурации (в частности, несколько конвейеров) [3, 4].

Вычислительная система с разделяемой памятью состоит из множества секторов памяти, множества элементарных процессоров и коммутатора. Процессоры не имеют своей локальной памяти. Допускается в один момент времени обращаться к разным секторам памяти (для чтения или записи).

Из каждого сектора памяти и из каждого процессора есть выход, ведущий на вход коммутатора (см. рис. 1). К каждому сектору памяти и к каждому входу каждого процессора ведет какой-нибудь выход коммутатора. Будем считать, что коммутатор обладает свойством полнодоступности, т.е. позволяет соединять входы и выходы в любых необходимых комбинациях. Более строго, если у коммутатора N входов и N выходов, то для любой

перестановки s из N чисел коммутатор позволяет одновременно соединить для каждого i = 1, ..., N каждый вход с номером i с выходом с номером s_i. Такие коммутаторы описаны, например, в [6], [7].

Рис. 1. Схема коммутации процессоров и модулей памяти.

Графовая модель задачи умножения разреженной матрицы на вектор.

Напомним, что граф называется двудольным, если множество всех его вершин состоит из вершин двух типов, и ребрами могут соединяться только вершины разных типов. Рассмотрим двудольный граф G, вершинами которого являются элементы векторов (массивов) X и Y. Две вершины

графа соединяются дугой, если соответствующие им элементы X(k) и Y(i) таковы, что при вычислении Y(i) используется X(k) (другими словами, если существует такое j , что U(i,j) = k) .

Паросочетанием в графе называется такое множество дуг, что никакие две дуги из этого множества не инцидентны одной вершине. Паросочетание в графе называется наибольшим, если не существует другого паросочетания, содержащего большее количество вершин (так что наибольших паросочетаний может быть несколько).

Известно [8, стр. 169] , что множество дуг двудольного графа можно разбить на такое количество паросочетаний, какова наибольшая степень вершин данного графа (и меньшее количество паросочетаний невозможно).

Теорема. Пусть максимальная степень вершин двудольного графа, соответствующего задаче умножения разреженной матрицы на вектор, равна числу d. Тогда за d шагов можно выполнить умножение данной матрицы на вектор. На каждом шаге производится одно параллельное чтение из сегментированной памяти, одна параллельная запись в сегментированную память, одно параллельное умножение и одно параллельное сложение. Меньшее количество шагов невозможно.

Доказательство. Приведем алгоритм. Построим разбиение множества ребер графа на d паросочетаний. Каждому паросочетанию будет соответствовать шаг алгоритма. Все элементы Y(i) расположим в различных процессорах и присвоим им значение 0. Для произвольного паросочетания считываем из разделяемой памяти все элементы X(k), которым соответствуют вершины графа, покрываемые этим паросочетанием (все ребра паросочетания покрывают попарно разные вершины). Перешлем одновременно каждый считанный элемент X(k) в такой процессор, который содержит такой элемент Y(i) , что соответствующие этим элементам вершины графа в данном паросочетании соединяются дугой. Поскольку коммутатор вычислительной системы универсальный и все X(k) различны и

все Y(i) различны, то такая одновременная пересылка возможна. Далее в каждом процессоре выполняется оператор

$$Y(i) = Y(i) + A(i,j)*X(U(i,j))$$

Выполнив такие операторы для каждого паросочетания, получим вычисленными все величины Y(i).

Алгоритм нахождения наибольшего паросочетания в двудольных графах сложности $O(n^{5/2})$ можно найти в [8, стр. 390].

Обычно в разделяемой памяти размещаются все данные в каждом модуле памяти. Это позволяет из любого модуля памяти извлечь любое данное. Такой подход может быть уместен, если начальные данные не обновляются. Если алгоритм должен работать с обновляемыми данными, то такой подход будет отнимать много времени, поскольку на каждом шаге в память придется записывать только одно данное во все модули сразу. Алгоритм, приведенный в данной работе, на каждом шаге пишет во все модули памяти разные данные, что уменьшает число параллельных шагов записи в память в N раз. Для случая индексных выражений массивов, линейно зависящих от счетчика цикла, алгоритмы бесконфликтного размещения данных рассматривались в [9], [10].

Рассмотрим теперь случай, в котором количество процессоров p и количество модулей памяти m меньше длины N векторов X и Y. В этом случае следует в каждом модуле памяти разместить по N/m элементов массива X, а каждый процессор должен вычислять по N/p элементов массива Y. Будем для простоты считать, что оба числа N/m и N/p - целые. После такого распределения элементов массивов по модулям памяти и процессорам задача организации вычислений будет выглядеть следующим образом.

Рассмотрим двудольный граф, вершинами которого являются m модулей памяти и p процессоров. Для каждой пары элементов массивов

X(k) и Y(i) такой, что при вычислении Y(i) используется X(k), строится дуга графа между вершинами, соответствующими модулю памяти, содержащему X(k) и процессору, содержащему Y(i).

Теперь задача расписания параллельных вычислений сводится к уже рассмотренному случаю. Следует только оговорить, что в случае m=p=N элемент Y(i) все время находится в соответствующем процессоре. В общем случае один и тот же процессор в разные моменты времени может вычислять разные Y(i) , что приводит к необходимости дополнительных обменов данными между процессорами и модулями памяти.

Поскольку количество параллельных шагов вычислений равно наибольшей степени вершин полученного двудольного графа, возникает задача такой группировки элементов X(k) в модулях памяти и элементов Y(i) в процессорах, чтобы у полученного двудольного графа максимальная степень вершин была наименьшей. Это известная NP-полная задача [11, стр. 66].

ЛИТЕРАТУРА

1. Цукерман В.Д., Чешков Г.Н. Основы нелинейной динамики сенсорного восприятия. 1. Фазовое координирование в осцилляторных сетях с четным циклическим торможением.// Нейрокомпьютеры. Разработка. Применение. № 7-8, 2002.

2. Писсанецки С. Технология разреженных матриц. М., Мир, 1988, 411 с.

3. Каляев А.В., Арцатбанов А.Ю., Итенберг И.И. Принципы построения семейства высокопроизводительных ортогональных многопроцессорных вычислительных систем с программируемой архитектурой. В кн.

Многопроцессорные вычислительные структуры - Таганрог, Вып. 13/XXII, - 1991 г., с. 4-9.

4. Каляев А.В., Левин И.И. Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений //М., «Янус-К», 2003, 380 с.

5. Almasi G.S., Gottlib A. Highly Parallel Computing.- 1989, The Benjamin/Cummings Publishing Company, inc.

6. Архангельская А.А., Ершов В.А., Нейман В.И. Автоматическая коммутация каналов связи. – М.: «Связь», 1970, 192 с..

7. Бенеш В.Э. Математические основы теории телефонных сообщений.- М.: «Связь», 1968, 292 с.

8. Свами М., Тхуласираман К. Графы, сети, алгоритмы. М., Мир, 1984, 454 с.

9. Штейнберг Б.Я. Бесконфликтные размещения массивов при параллельных вычислениях// Кибернетика и системный анализ/ 1999, N 1, с. 166-178.

10. Штейнберг Б.Я. Математические методы распараллеливания рекуррентных программных циклов на суперкомпьютеры с параллельной памятью.// Ростов-на-Дону, Издательство Ростовского университета, 2004 г., 192 с.

11. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. - М.: Мир, 1982, 416 с.

12. Yuan Lin. Compiler Analysis of Sparse and Irregular Computations. Ph.D Thesis, Department of Computer Sciences University of Illinois Urbana-Champaign, 2000. www.polaris.org