Math-Net.Ru
Общероссийский математический портал

И. Н. Молчанов, В. Е. Рябцев, Прямые методы решения систем линейных алгебраических уравнений с параллельной организацией вычислений,
Ж. вычисл. матем. и матем. физ., 1986, том 26, номер 4, 574–585

https://www.mathnet.ru/zvmmf8214

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением
https://www.mathnet.ru/rus/agreement

Параметры загрузки:
IP: 18.97.14.84
19 апреля 2025 г., 23:31:42

ЖУРНАЛ
ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И МАТЕМАТИЧЕСКОЙ ФИЗИКИ

Том 26, 1986 № 4

УДК 519.612

ПРЯМЫЕ МЕТОДЫ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ С ПАРАЛЛЕЛЬНОЙ ОРГАНИЗАЦИЕЙ ВЫЧИСЛЕНИЙ

МОЛЧАНОВ И. Н., РЯБЦЕВ В. Е.

(Киев)

Рассматриваются вопросы эффективной организации процесса решения линейных алгебраических систем прямыми методами на многопроцессорной ЭВМ типа MIMD в условиях ограниченности вычислительных ресурсов и технических возможностей. Проводится качественный анализ вычислительных схем, и оценивается эффективность распараллеливания.

Сократить время решения систем линейных алгебраических уравнений (с.л.а.у.) высокого порядка можно, если использовать многопроцессорные вычислительные системы (м.в.с.) с параллельной организацией вычислений различных типов (pipeline, MISD, SIMD или MIMD, см. [1], [2]). Однако потребности практики растут быстрее, чем производительность ЭВМ. Так, анализ одной достаточно типичной с.л.а.у., возникающей при дискретизации пространственного тела методом конечных элементов (исследовалось статически напряженное состояние), показал, что ее порядок составляет 44 605. Количество ненулевых элементов, содержащихся в верхнем треугольнике матрицы, оказалось равным 1 109 875 (матрица симметрична). Если ее рассматривать как ленточную, то полуширина ленты достигает 515, что требует хранения 22 971 575 элементов. Если же матрицу рассматривать как профильную, то количество элементов в профиле составит 15 149 475. Учитывая необходимость многовариантного счета (до нескольких сот вариантов), можно заключить, что решение подобных задач даже для ЭВМ с параллельной организацией вычислений требует значительного машинного времени и в ряде случаев невыполнимо.

1. При решении широкого класса научно-технических и практических задач (см., например, [3], [4]) возникает необходимость в решении с.л.а.у.

$$AX=B, \tag{1.1}$$

где A — невырожденная матрица порядка n, B — матрица размера $n\times l$, составленная из l векторов правых частей, а X — матрица искомых решений. Одним из путей ускорения решения таких задач является организация вычислительного процесса с параллельной обработкой информации.

Проблеме распараллеливания методов линейной алгебры посвящено много работ (см. библиографию в обзорах [5]–[7]). Однако большинство из них посвящено проблеме распараллеливания в условиях неограниченности вычислительных ресурсов и технических возможностей (например, количество процессоров составляет n, n^2 или n^4, не учитывается

время обмена данными и т. д.). Проведенный анализ [8] показал, что для каждого класса машин с параллельной организацией вычислений требуется своя адаптация прямых и итерационных методов решения, с.л.а.у. и создание новых алгоритмов, учитывающих структуру и архитектуру конкретного типа ЭВМ. Более того, даже в одном и том же классе МВС программная реализация, помимо структуры, должна учитывать как особенности ее функциональных возможностей, так и специфику операционной системы. Только с учетом этих факторов может быть достигнуто рекламируемое быстродействие ЭВМ.

Настоящая статья посвящена адаптации прямых методов решения с.л.а.у. на МВС типа MIMD в условиях ограниченности вычислительных ресурсов и технических возможностей. В качестве показателей, характеризующих эффективность вычислительных схем с параллельной организацией вычислений, будут использоваться коэффициент ускорения $K_у$ и коэффициент эффективности $K_э$, определяемые как

$$K_у = T(1)/T(p), \qquad K_э = T(1)/pT(p), \tag{1.2}$$

где $T(1)$ — время решения задачи на последовательной однопроцессорной ЭВМ, а $T(p)$ — на p-процессорной МВС. Ниже приводится структура многопроцессорной системы, для которой изложены имеющиеся результаты.

2. Гипотетическая МВС состоит из периферийного процессора (ПП), управляющих процессоров (УП) и p идентичных арифметических процессоров (АП), образующих с помощью коммутационной сети единую вычислительную среду.

АП обладают определенным относительно большим объемом оперативной памяти и способны осуществлять численную и логическую обработку информации в автономном и управляемом режимах. Операторы программ, управляющие инструкции и обрабатываемая информация поступают в АП либо по мере необходимости, либо в случае их высвобождения из вычислительного процесса.

УП обеспечивают организацию вычислительного процесса по общей программе, согласование работы АП и управление потоками данных. Один УП может управлять ходом вычислительного процесса в группе АП как в случае решения ими самостоятельных задач, так и при выполнении заданий, являющихся составной частью некоторой общей программы. Повысить эффективность работы АП за счет более гибкого управления можно, построив некоторую иерархическую структуру уровней УП.

ПП управляет всеми периферийными устройствами (дисками, лентами, дисплеями, АЦПУ и т. д.) и выполняет роль своеобразного буфера при обменах информацией между АП и внешней памятью (ВП). Он имеет $\varkappa$ каналов связи с коммутационной сетью, которые могут быть задействованы одновременно.

Коммутационная сеть обеспечивает передачу информации между ПП, УП и АП. В системе УП—АП возможно попарное соединение любого количества процессоров. Передача информации производится в полудуплексном режиме.

С точки зрения пользователя, МВС данной структуры можно представлять как систему из p независимых процессоров, связанных с внешней памятью посредством периферийного процессора. Коммутационная сеть способна обеспечить одновременный обмен информацией между внешней па-

мятью и несколькими (вплоть до $\varkappa$) процессорами наряду с независимой обработкой данных в остальных процессорах.

При такой архитектуре МВС для машинной реализации определенного фрагмента задания могут быть использованы различные процедуры обмена и обработки данных. В этом случае ставится задача выбора оптимальной процедуры, например минимизирующей время выполнения фрагмента. Пусть в каждый процессор группы из k АП требуется передать набор данных, содержащийся в одном из них. Для осуществления данного фрагмента можно применить процедуру цепной передачи данных [9], [10], основанную на организации обменов между АП по принципу сдваивания. С другой стороны, используя возможности ПП, можно передать в него указанный набор данных, а затем рассылать в АП сразу по $\varkappa$ каналам (принцип централизованной передачи). Очевидно, что если

$$\rceil \log_2 k \lceil > 1 + \rceil (k-1)/\varkappa \lceil,$$

где $\rceil x \lceil$ — минимальное целое, не меньшее x, то меньшее количество обменов и, следовательно, времени требуется для процедуры централизованной передачи.

В дальнейшем при исследовании различных вопросов, связанных с реализацией вычислительных алгоритмов, используются следующие обозначения: M — объем оперативной памяти АП (в словах), t_1 — среднее время поиска определенного набора данных в ВП, t_2 — время передачи одного слова из ВП в ПП, t_3 — среднее время коммутационных задержек, t_4 — время передачи одного слова в куммутационной сети, t_5 — среднее время выполнения одной арифметической операции.

Рассмотрим более сложный пример. Пусть в группе из k АП находятся наборы данных D_i объема V_i, $i=1, 2, \ldots, k$, которые необходимо последовательно преобразовать, согласно алгоритмам Q_{ij}, с помощью наборов данных C_j объема W_j, $j=1, 2, \ldots, n$, находящихся в ВП. Выбор процедуры реализации данного фрагмента определяется соотношением между W_j, V_i и M. Пусть

$$\max_{i,j}(V_i+W_j) \leqslant M. \tag{2.1}$$

Осуществить указанное задание можно, во-первых, путем последовательного выполнения алгоритмов Q_{ij} для всех j одновременно во всех АП. Процедура передачи C_j определяется в зависимости от k и $\varkappa$. Во-вторых, преобразование D_i с помощью C_j можно реализовать по следующей схеме (магистральная обработка):

$$\text{ВП} \xrightarrow{C_{r+k}} \text{ПП} \xrightarrow{C_{r+k-1}} \text{АП}_1 \xrightarrow{C_{r+k-2}} \text{АП}_2 \rightarrow \ldots \xrightarrow{C_r} \text{АП}_k.$$

Очевидно, что при использовании принципа централизованной и цепной передачи данных число обменов составит $n\rceil k\varkappa^{-1}\lceil$ и $n(1+\rceil \log_2 k \lceil)$. Применение процедуры магистральной обработки требует np обменов, поскольку, в силу (2.1), АП_s, $s=1, 2, \ldots, k-1$, должен сначала передать C_m соседу справа, а затем принять C_{m+1} от соседа слева. Если же справедливо соотношение

$$\max_{i,j}(V_i+2W_j) \leqslant M, \tag{2.2}$$

то прием и передача данных между АП выполняется только за два обмена. Тем самым процедура магистральной обработки в условиях (2.2) по-

требует $2(k+n-1)$ обменов. Для окончательного ответа на вопрос о целесообразности применения указанных процедур предположим существование двух разбиений набора данных C на сегменты C_j объема W_j и C_s' объема W_s', удовлетворяющих условиям (2.1) и (2.2). Количества сегментов разбиения будем считать связанными соотношением $2m=m'$. Полагая, что объем вычислений при обработке D_i не зависит от размерности C_j, получаем следующие соотношения для времени коммутационных задержек и передачи данных при выполнении данного фрагмента с применением процедуры магистральной обработки и с использованием централизованной передачи данных:

$$T' \approx 2(m'+k-1)(t_3+W't_4), \qquad T \approx m\left(\frac{k}{\varkappa}\right)(t_3+Wt_4),$$

где W и W' — размерность наибольшего сегмента разбиения. Если количество сегментов C_j велико, то

$$\frac{T'}{T} \approx \frac{4\varkappa}{k}\left[1-\frac{Wt_4}{2(t_3+Wt_4)}\right].$$

Таким образом, процедура магистральной обработки оказывается эффективнее, чем процедура, использующая принцип централизованной передачи, всегда, когда количество каналов связи с ПП не превосходит четвертой части количества задействованных АП. Аналогично, сравнение с процедурой цепной передачи показывает целесообразность применения последней лишь для $k \leqslant 8$.

В приведенных рассуждениях неявно предполагалось, что все данные, необходимые для работы АП, находятся в ПП. На самом деле это будет справедливо, если процесс поиска и считывания требуемого набора данных в ПП из ВП происходит на фоне вычислений в АП. Если на процесс обработки информации в системе АП затрачивается время τ_{ij}, то данное условие можно записать в виде

$$\max(t_1+W_jt_2) \leqslant \min_{i,j} \tau_{ij}. \tag{2.3}$$

Последнее соотношение будем называть условием согласованности, поскольку его выполнение гарантирует отсутствие прерываний вычислительного процесса в АП, связанных с ожиданием завершения обменов между ПП и ВП. Если обработка информации во всех АП производится одновременно с помощью одного и того же набора данных C_j, то под τ_{ij} в (2.3) подразумевается только время выполнения алгоритма Q_{ij} в АП. В этом случае τ_{ij} не зависит от i.

При построении алгоритмов с параллельной организацией вычислений, удовлетворяющих условию согласованности (2.3), в ряде случаев целесообразно менять структуру наборов данных. Наличие ПП позволяет преимущества работы с большими по объему сегментами данных при обменах между ВП и ПП сочетать с возможностью их разбиения в ПП и оперирования сегментами малого объема при обменах между АП. В связи с этим все алгоритмы можно разделить на два класса: 1) оперирующие на всех этапах вычислительного процесса сегментами фиксированной размерности; 2) изменяющие структуру и размерность сегментов в процессе выполнения отдельных фрагментов.

Наконец, в дальнейшем будем предполагать, что для МВС данной структуры время коммутационных задержек t_3 и время передачи одного слова по каналам коммутационной сети t_4 — величины одного порядка.

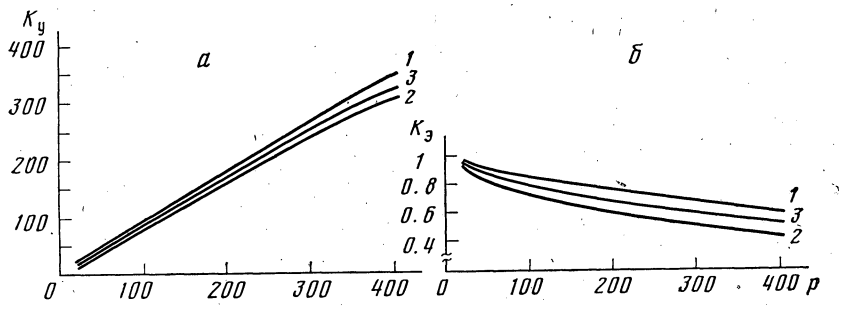
3. Пусть задана с. л. а. у. (1.1). При решении систем общего вида прямым методом пользователь, как правило, выбирает один из методов последовательного исключения неизвестных [16], [17], суть которых — в сведении исходной системы (1.1) к эквивалентной системе с треугольной матрицей (прямой ход) и в последовательном вычислении компонент решения (обратный ход). Указанные методы позволяют запоминать всю последовательность выполняемых преобразований, практически не изменяя объема хранимой информации. Это обстоятельство оказывается особенно важным при многовариантном счете с различными правыми частями.

Для построения схем, реализующих вычислительный процесс на МВС, используется параллелизм, заложенный в последовательных алгоритмах. Предположим, что размерность системы (1.1) такова, что позволяет разместить исходную информацию в памяти p АП, т. е. $n(n+l)\leqslant pM$. Основное внимание уделим рассмотрению процесса факторизации матрицы A. Организация обратного хода описана в [10]. Пусть каждый АП содержит q столбцов матрицы A и $n=pq$. Для равномерного распределения вычислительной работы между АП целесообразно производить разбиение данных таким образом, чтобы в $АП_j$, $j=1, 2, \ldots, p$, содержались $(j+kp)$-е вектор-столбцы матрицы A, где $k=0, 1, \ldots, q-1$. Рассмотрим организацию вычислительного процесса на примере метода Гаусса, реализуемого по схеме единого деления с выбором ведущего элемента по столбцу [11]. При этом увеличим размерность столбцов на единицу для запоминания номера строки, элемент которой выбран в качестве ведущего.

Вычислительный процесс строится следующим образом. На i-м шаге, $i=1, 2, \ldots, n-1$, АП, содержащий i-й столбец исходной матрицы, производит выбор ведущего элемента и вычисляет соответствующие множители, с помощью которых преобразуется активная часть матрицы. (Активной частью матрицы принято называть подматрицу исходной матрицы, элементы которой подлежат дальнейшему преобразованию.) Полученный набор данных, представляющий собой $(n-i+2)$-мерный вектор, передается во все АП для выполнения соответствующих преобразований. Для передачи данных можно применять различные процедуры обмена. Если используется принцип цепной передачи данных, то время вычисления и передачи обрабатывающей информации на i-м шаге составит

$$T_i \approx \rceil \log_2 p \lceil [t_3+(n-i+2)t_4]+(n-i+1)t_5.$$

В течение этого времени остальные АП находятся в состоянии ожидания. Можно использовать иную схему обменов. На каждом шаге процессор, вычисливший соответствующие множители, передает их в другой АП, например соседу справа, и приступает к обработке собственной информации. Аналогично, каждый АП, получив соответствующие множители, сначала передает его другому АП, а затем производит преобразование собственных данных. Такую организацию обменов будем называть процедурой привилегированной передачи данных. При такой организации вычислительного процесса $АП_{j+1}$ начинает обработку данных на i-м шаге с запозданием T_L относительно момента начала работы $АП_j$. Поэтому если время вычисления соответствующих множителей τ_i не превосходит T_L, то данные, необходимые для работы $АП_j$ на $(i+1)$-м шаге, будут подготовлены к моменту завершения им i-го шага. Расписывая τ_i и T_L, имеем следующее



условие согласованной работы АП на i-м шаге:

$$t_3+(n-i+2)t_4\geqslant(n-i)t_5. \tag{3.1}$$

Если условие (3.1) справедливо для всех $i=1, 2, \ldots, n-1$, то общее время факторизации матрицы A составит

$$T_p\approx n\left(t_3+\frac{n}{2}t_4\right)+\frac{4n^3+9n^2}{6p}t_5, \tag{3.2}$$

в то время как схема с применением цепной передачи требует

$$T_c\approx n\lceil\log_2 p\rceil\left(t_3+\frac{n}{2}t_4\right)+\left(\frac{4n^3+9n^2}{6p}+\frac{n^2}{2}\right)t_5. \tag{3.3}$$

В рамках описанных схем вписываются также методы отражений, ортогонализации, квадратных корней, метод Гаусса с выбором ведущего элемента по активной части матрицы и др. Для метода отражений, например, условие (3.1) запишется в виде

$$t_3+(n-i+2)t_4\geqslant 3(n-i)t_5. \tag{3.4}$$

Для анализа эффективности предложенных вычислительных схем воспользуемся формулами (1.2). Время выполнения алгоритма на последовательной ЭВМ определяется количеством арифметических операций. Ограничиваясь старшими членами в (3.2) и (3.3), пренебрегая величиной t_3 и обозначая $\tau=t_4t_5^{-1}$, получаем следующие оценки для коэффициентов ускорения вычислительных схем, реализующих факторизацию матрицы A методом Гаусса:

$$K_у^P\approx\frac{p}{1+3p\tau/(4n)},\qquad K_у^C\approx\frac{p}{1+3p\tau\log_2 p/(4n)}.$$

Заметим, что оценка для $K_у^P$ справедлива только при выполнении (3.1). Для метода отражений в условиях (3.4) имеем

$$K_у^P\approx\frac{p}{1+3p\tau/(8n)},\qquad K_у^C\approx\frac{p}{1+3p\tau\log_2 p/(4n)}.$$

Для нахождения нормального решения систем с прямоугольными матрицами неполного ранга можно применять процесс нормализованного разложения [12]. По сути, он сводится к преобразованию исходной матрицы сначала с помощью правосторонних, а затем левосторонних умножений на цепочку матриц отражения (вращений). С точки зрения сложности машинной организации вычислительного процесса это эквивалентно двукратной факторизации матрицы методом отражений и ее транспонированию. Следовательно, эффективность схем с параллельной организацией вычислений во многом будет определяться тем, насколько

удачно реализована процедура транспонирования. Зависимость $K_у$ от количества АП, задействованных в процессе факторизации матрицы максимального для получаемой конфигурации МВС порядка, показана на фигуре *а*. Фигура *б* иллюстрирует изменение $K_э$ при факторизации матрицы фиксированного порядка с увеличением количества АП, задействованных в вычислительном процессе. Графики даны для случая $M=5\cdot10^4$ и $\tau=4$; кривые *1* — метод отражений, *2* — метод Гаусса, *3* — метод квадратных корней.

Рассмотрим случай, когда порядок матрицы A системы (1.1) не позволяет расположить ее в памяти АП и для организации вычислительного процесса ее необходимо некоторым образом разбить на сегменты. Предположим, что матрица A разбита на сегменты, содержащие целиком несколько столбцов. Пусть в качестве метода факторизации выбран метод Гаусса. Его можно реализовать по схемам первого и второго классов. В первом случае разбиение данных производится с таким расчетом, чтобы в памяти одного АП можно было разместить одновременно два сегмента. Для схемы второго класса размерность выбирается так, чтобы памяти АП хватило для хранения одного сегмента и одного вектор-столбца исходной матрицы. Если через q_k обозначить количество столбцов в сегменте для каждой из схем, то

$$q_1=\left[\frac{M}{2(n+1)}\right],\qquad q_2=\left[\frac{M-n-1}{n+1}\right].$$

Предположим, что сегменты исходной и факторизованной матриц хранятся в ВП и $n=pq_kr_k$. Выполнение i-го шага схемы первого класса состоит из четырех этапов:

1) вызов в j-й АП, $j=1, 2, \ldots, p$, из ВП $[(i-1)p+J]$-го сегмента исходной матрицы;

2) преобразование их с помощью множителей, хранимых во всех уже факторизованных сегментах, вызываемых для этого во все АП из ВП;

3) факторизация сегментов, содержащихся в АП;

4) запись полученных сегментов в ВП.

Отличие при реализации схемы второго класса заключается в том, что сегменты, вызываемые из ВП, разбиваются в ПП на отдельные столбцы и уже с их помощью преобразуются данные в АП.

Для передачи данных могут применяться различные процедуры. Предположим, что на первом, втором и четвертом этапах используется принцип централизованной передачи. При выполнении третьего этапа по схеме первого класса обмен данными осуществляется по цепному принципу, а для схемы второго класса — по принципу привилегированной передачи. Время реализации второго этапа на i-м шаге для каждой схемы составит

$$T_1^2\approx\sum_{j=1}^{ip}\left\{\frac{p}{\varkappa}(t_3+nq_1t_4)+2[n-(j-1)q_1]q_1^2t_5\right\},$$

$$T_2^2\approx\sum_{j=1}^{ipq_2}\left[\frac{p}{\varkappa}(t_3+nt_4)+2(n-j+1)q_2t_5\right].$$

В каждом случае предполагаются выполненными условия согласованности (2.3), принимающие, соответственно, вид

$$t_1+nq_1t_2\leqslant t_3+nq_1t_4+2(n-ipq_1)q_1^2t_5,$$

$$t_1+nq_2t_2\leqslant q_2(t_3+nt_4)+2(n-ipq_2)q_2^2t_5.$$

На выполнение третьего этапа на $(i+1)$-м шаге затрачивается время

$$T_1^3\approx\sum_{j=1}^{p}\{]\log_2(p-j+1)\lceil(t_3+nq_1t_4)+$$

$$+2[n-ipq_1-(j-1)q_1]q_1^2t_5\},$$

$$T_2^3\approx\sum_{j=1}^{p}\{q_2(t_3+nt_4)+2[n-ipq_2-(j-1)q_2]q_2^2t_5\}.$$

По мере высвобождения АП из вычислительного процесса на этом этапе производится обмен информацией. Тем самым выполнение третьего и четвертого этапов i-го шага и первого этапа $(i+1)$-го шага частично перекрываются. Если же для данного i и всех $j<p$ имеет место

$$t_1+t_3+nq_k(t_2+t_4)\leqslant 2q_k^2[n-(ip+j)q_k]t_5,$$

то в суммарное время выполнения $(i+1)$-го шага войдет время обменов с ВП только одного АП. Суммируя по всем i, выражая все в терминах M, n, p, $\varkappa$ и ограничиваясь лишь старшими членами, получаем следующие оценки времени факторизации матрицы A методом Гаусса по каждой из вычислительных схем:

$$T_1\approx\frac{4n^2}{pM}\left(t_1+\frac{M}{2}t_2\right)+\frac{2n^2}{M}\left(\frac{n^2}{\varkappa M}+\log_2 p\right)\times$$

$$\times\left(t_3+\frac{M}{2}t_4\right)+\left(\frac{2n^3}{3p}+\frac{n^2}{2}\right)t_5,$$

$$T_2\approx\frac{2n^2}{pM}(t_1+Mt_2)+\left(\frac{n^3}{2M\varkappa}+n\right)(t_3+nt_4)+\frac{2n^3}{3p}t_5.$$

Заметим, что если процедуры, реализующие этапы каждого шага, для обеих схем совпадают, то схема первого класса будет более быстродействующей лишь в случае $nMt_4\leqslant(M-4n)t_3$, причем она накладывает более жесткие условия согласованности. В [9] рассматриваются подобные схемы реализации метода отражений.

Помимо непосредственного вычисления $K_у$ и $K_э$ (по 1.2), для качественного анализа схем с параллельной организацией вычислений можно воспользоваться методикой, предложенной в [13]. В рамках несколько упрощенной модели она позволяет получить оценку снизу для коэффициента эффективности $\beta_p(n)$ в терминах, характеризующих алгоритм, разбиение данных и технические параметры МВС:

$$\beta_p(n)\geqslant\frac{\sigma}{u}\left(1+\frac{\lambda m\tau}{u}\right)^{-1}, \tag{3.5}$$

где σ и u — средний и максимальный объем вычислений в одном АП на одном шаге, m — максимальный объем обмена между двумя процессорами на одном шаге, а $\lambda=\max\lambda_k$ (λ_k — хроматический класс графа, соответ-

ствующего k-му шагу вычислений). Величины σu^{-1} и mu^{-1} называются коэффициентами равномерности и макроконвейерности, а λ — коэффициентом локальности данного разбиения.

Не составляет труда определить все необходимые величины для вычислительных схем, предложенных выше, причем наиболее точные оценки для коэффициентов равномерности, макроконвейерности и локальности можно получить с учетом продолжительности работы МВС на каждом шаге. Непосредственно посчитанные по (1.2) и полученные по (3.5) оценки коэффициентов эффективности для схем первого и второго классов имеют вид

$$K_э^1 \approx \left(1-\frac{3p}{4n}\right)\left[1+3\tau p\left(\frac{n}{2M\varkappa}+\frac{\log_2 p}{2n}\right)\right]^{-1},$$

$$K_э^2 \approx \left(1+\frac{3np\tau}{4M\varkappa}\right)^{-1},$$

$$\beta_p^1(n) \geqslant \left(1-\frac{pM}{2n^2}\right)\left[1+\tau\left(\frac{2np}{M\varkappa}+\frac{\log_2 p}{n}\right)\right]^{-1},$$

$$\beta_p^2(n) \geqslant \left(1+\frac{np\tau}{M\varkappa}\right)^{-1}.$$

Как видно из этих соотношений, полученные результаты хорошо согласуются между собой.

Наивысшей точностью в классе прямых методов обладает метод ортогонализации, реализуемый как процесс переортогонализации [12]. Однако данный процесс практически непригоден для решения больших с.л.а.у. из-за необходимости частого обращения к ВП. Для устранения этого недостатка изменим вычислительные формулы. Пусть вычислены векторстроки c_j, $j=1, 2, \ldots, m$, ортогонализованной матрицы C. Построение c_r для $r=m+1, \ldots, m+k$ осуществляется следующим образом:

$$\tilde{v}_r^{(i)} = v_r^{(i-1)} - \sum_{j=1}^{m} (v_r^{(i-1)}, c_j)\, c_j - \sum_{j=m+1}^{r-1} (v_r^{(i-1)}, v_j^{(i)})\, v_j^{(i)}, \tag{3.6}$$

где $v_r^{(i)}$ — нормированный вектор $\tilde{v}_r^{(i)}$ для $i=1, 2, \ldots, s$, $v_r^{(0)}=a_r$, а s — количество выполняемых итераций, т. е. $c_r=v_r^{(s)}$. Если для вычисления скалярных произведений используется режим накопления, то процесс (3.6) будет устойчив как в смысле малости эквивалентных возмущений, так и в смысле сохранения ортогональности получаемых векторов c_r для всех k таких, что

$$\gamma^k \geqslant (2^k \varepsilon n)^{1/2} + 2^k \varepsilon n, \tag{3.7}$$

где

$$\gamma = \min_{1 \leqslant l \leqslant n} \min_{\alpha_i} \left(\|a_l\|^{-1} \left\| a_l - \sum_{i \neq l} \alpha_i a_i \right\| \right),$$

а ε — относительная машинная точность. Из (3.6) вытекает целесообразность разбиения данных на сегменты, содержащие целиком несколько строк расширенной матрицы $(A|B)$ системы (1.1). Формулы преобразования строк матрицы B аналогичны (3.6). При таком разбиении становится возможным вычислять составляющие компонент матрицы X в процессе ортогонализации [10].

Рассмотрим вычислительную схему первого класса, основанную на применении процедуры магистральной обработки. Согласно полученным

результатам для разбиения данных по (2.1) и (2.2), количество строк в сегменте следует выбирать таким, чтобы в памяти АП можно было разместить четыре сегмента $(A|B)$.

Пусть $n=pqr$. На m-м шаге, $m=1, 2, \ldots, r$, $АП_j$, $j=1, 2, \ldots, p$, ортогонализует строки в сегменте $(m-1)p+j$. Для реализации процесса (3.6) вызов из ВП всех сегментов, вычисленных на предыдущих шагах, осуществляется с использованием процедуры магистральной обработки данных. Построение векторов $v_k^{(i)}$, где $k=(m-1)pq, \ldots, mpq$, на каждой итерации i можно рассматривать как продолжение выполнения магистральной обработки в системе АП без притока извне новой обрабатывающей информации. Количество арифметических операций в одном АП при этом несколько увеличивается. Поэтому можно задерживать начало следующей итерации до момента построения $v_h^{(i)}$ в $\rceil p/2\lceil$-м АП на текущей итерации. Тем самым реализацию m-го шага можно рассматривать как процедуру магистральной обработки информации в p процессорах с помощью $s[p(m-1)+\rceil p/2\lceil]$-го набора данных, находящихся в ВП. Непосредственный подсчет и оценка коэффициента эффективности дают следующие результаты:

$$K_э \approx \left(1+\frac{5n\tau}{3n\tau+4M}\right)^{-1}, \qquad \beta_p(n) \geqslant \left(1+\frac{2n\tau}{M}\right)^{-1}.$$

Следует заметить, что, в силу (3.7), данная схема может применяться только для с. л. а. у. с хорошо обусловленными матрицами A. В [10] рассматривались и другие вычислительные схемы, в том числе для систем с худшей обусловленностью.

4. Предположим, что несимметричная матрица A большого порядка имеет ленточную структуру с шириной ленты m, причем в столбце количество элементов, лежащих ниже главной диагонали, составляет d. Как и для систем с плотными матрицами, будем считать, что разбиение данных производится по столбцам. Количество столбцов в сегменте выбирается с учетом того, чтобы, помимо хранения ненулевых элементов матрицы, в АП резервировалось место для запоминания дополнительных элементов, возникающих в процессе факторизации. Тем самым для схем первого и второго классов имеем

$$q_1 = \left\lceil \frac{M}{2(d+1)+m} \right\rceil, \qquad q_2 = \left\lceil \frac{M-d}{m+d+1} \right\rceil.$$

Ограничимся случаем, когда ширина ленты удовлетворяет условию

(4.1) $\qquad m+q_k \leqslant (p-1)q_k, \qquad k=1, 2.$

Здесь организовать вычислительный процесс можно таким образом, чтобы факторизация матрицы непрерывно осуществлялась на $(p-1)$-м АП. Тем временем последний АП производит обмен данными, т. е. записывает в ВП факторизованный сегмент и считывает сегмент, преобразование которого начнется со следующего шага. Если для факторизации используется метод отражений, а для обмена данными в процессе работы $(p-1)$-го АП применяется принцип цепной передачи, то условие согласованности (2.3) для схемы первого класса приобретает вид

$$2[t_1+t_3+(m+d)q_1(t_2+t_4)] \leqslant \rceil \log_2(p-1)\lceil (t_3+ \\ +dq_1t_4)+6dq_1^2t_5.$$

Аналогично, для схемы второго класса с использованием процедуры привилегированной передачи имеем

$$2[t_1+t_3+(m+d)q_2(t_2+t_4)]\leqslant(q_2+p-1)(t_3+dt_4)+4dq_2^2t_5. \tag{4.2}$$

Кроме того, в этом случае необходимо, чтобы

$$t_3+dt_4\geqslant 3dt_5, \tag{4.3}$$

что гарантирует от возникновения ожиданий, связанных с подготовкой обрабатывающей информации.

Непосредственный подсчет коэффициентов эффективности для схем каждого класса с применением процедуры цепной передачи дает

$$K_э^1\approx\left(\frac{2}{3}-\frac{3p}{4m}\right)\left(1+\frac{p\tau\log_2 p}{4m}\right)^{-1},$$

$$K_э^2\approx\left(1-\frac{3p}{4m}\right)\left(1+\frac{p\tau\log_2 p}{4m}\right)^{-1}.$$

При организации обменов по процедуре привилегированной передачи для схемы второго класса, если выполняются (4.2) и (4.3),

$$K_э^2\approx(1-1/p)\,[1+p\tau/(4m)]^{-1}.$$

Все указанные схемы могут быть применены для факторизации симметричных положительно-определенных матриц методом квадратных корней. При этом ширина ленты, удовлетворяющая (4.1), значительно увеличивается, поскольку в процессе факторизации структура матрицы не меняется. В [14] рассматривались основанные на клеточном разбиении данных вычислительные схемы, которые могут применяться для решения систем с профильными матрицами.

5. В каждом конкретном случае для решения больших с.л.а.у. необходим предварительный анализ не только особенностей и структуры имеющейся системы, но и исследование физических свойств задачи, к ней приводящей. Без этого нельзя выбирать соответствующий метод решения и длину машинного слова, гарантирующего достоверность получаемого результата.

Только после выполнения такого анализа можно приступать к построению вычислительной схемы с распараллеливанием процесса обработки информации, которую целесообразно применить в данном конкретном случае. Это, в свою очередь, требует знания структуры МВС, ее функциональных возможностей и технических характеристик. Так, например, для МВС, у которых время коммутационных задержек значительно превосходит время передачи единицы информации, более эффективными оказываются схемы, оперирующие на всех этапах сегментами данных большой размерности (схемы первого класса). Поэтому всегда целесообразно хотя бы грубо оценить эффективность распараллеливания той или иной вычислительной схемы. Особое внимание при этом следует уделить проверке условий согласованности, без которых неизбежны простои процессоров в ожидании данных и конфликты в коммутационной сети. Большое значение приобретает также вопрос о разбиении данных. Рассмотренное разбиение на прямоугольные сегменты позволяет обойти некоторые проблемы устойчивости, которые неизменно возникнут при других (например, при клеточном) разбиениях.

Исследования показывают, что, соблюдая все эти условия, можно даже путем «естественного» распараллеливания на МВС данной архитектуры получить практически линейный рост коэффициента ускорения с увеличением числа процессоров, задействованных в вычислительном процессе. Создание алгоритмов, специально нацеленных на параллельную организацию счета,— направление, за которым видится будущее методов вычислений, ориентированных на многопроцессорные системы.

Литература

1. *Flynn M. J.* Very high-speed computing systems.– Proc. IEEE, 1966, v. 54, p. 1901–1909.
2. *Головкин Б. А.* Параллельные вычислительные системы. М.: Наука, 1980.
3. *Глушков В. М., Молчанов И. Н.* О некоторых проблемах решения задач на ЭВМ с параллельной организацией вычислений. – Кибернетика, 1981, № 4, с. 82–88.
4. *Бате К., Вилсон Е.* Численные методы анализа и метод конечных элементов. М.: Стройиздат, 1982.
5. *Heller D.* A survey of parallel algorithms in numerical linear algebra. – SIAM Rev., 1978, v. 20, № 4, p. 741–777.
6. *Sameh A.* An overview of algorithms in numerical linear algebra. – Bull. Dir. etud. et rech., 1983, t. C, № 1, p. 129–134.
7. *Фаддеева В. Н., Фаддеев Д. К.* Параллельные вычисления в линейной алгебре. II. – Кибернетика, 1982, № 3, с. 13–31.
8. *Михалевич В. С., Молчанов И. Н.* Тенденции развития супер-ЭВМ и проблемы, ими порождаемые. – Препринт ИК АН УССР. Киев, 1984, № 84-4.
9. *Молчанов И. Н., Рябцев В. Е.* Об организации вычислений при решении больших систем линейных алгебраических уравнений методом отражений. – Кибернетика, 1983, № 2, с. 24–28.
10. *Молчанов И. Н., Рябцев В. Е.* Распараллеливание прямых методов решения систем линейных алгебраических уравнений на ЭВМ класса MIMD. – Деп. в ВИНИТИ, № 3863-84 ДЕП.
11. *Фаддеев Д. К., Фаддеева В. Н.* Вычислительные методы линейной алгебры. М. – Л.: Физматгиз, 1963.
12. *Воеводин В. В.* Вычислительные основы линейной алгебры. М.: Наука, 1977.
13. *Глушков В. М., Капитонова Ю. В., Летичевский А. А.* Эффективность параллельных вычислений в ограниченных ресурсах. – Докл. АН СССР, 1980, т. 254, № 3, с. 527–530.
14. *Глушков В. М. и др.* Макроконвейерные вычисления функций над структурами данных. – Кибернетика, 1981, № 4, с. 13–21.

Поступила в редакцию 13.VIII.1984
Переработанный вариант 2.X.1984