ЭКСПЕРТНЫЕ СИСТЕМЫ И РАСПОЗНАВАНИЕ ОБРАЗОВ
(Вычислительные системы)
1988 год Выпуск 126

УДК 681.324

ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ И СРЕДЫ НА ПРИНЦИПЕ ЛОКАЛЬНОСТИ

Ю.С.Завьялов, А.И.Мишин

В в е д е н и е

Для всякой научной теории существуют границы применимости, которые определяются принципами, положенными в ее основу. Классическая физика исходит из принципа дальнодействия (глобальности, мгновенности взаимовлияний элементов), и ее законы применимы лишь до тех пор, пока время распространения воздействий в системе мало по сравнению с собственным временем протекающих процессов.

Основой современных ЭВМ являются теория алгоритмов, теория автоматов и теория электрических цепей с сосредоточенными параметрами. Теория алгоритмов описывает вычислительный процесс в виде последовательности арифметико-логических операций, не интересуясь размещением вычислителя и данных во времени и пространстве. Машинизированное определение алгоритма, машина Тьюринга, хотя и имеет дело с размещением данных на ленте, но лишь с целью указать последовательность выполнения операций.

Логическая сеть-электрическая схема, моделирующая вычислительный процесс, представляется совокупностью логических элементов, работающих с задержками, но мгновенно взаимодейст -

вующих посредством соединений, так что разница в расстояниях снова игнорируется. Эта модель применима лишь до тактовой частоты, при которой длина волны превышает линейный размер системы. При фиксированной тактовой частоте число элементов в системе ограничено, а значит, ограничены размеры решаемых задач (емкость оперативной памяти) и производительность ЭВМ (количество вычислителей-процессоров).

Предельное число элементов может быть достигнуто при изготовлении ЭВМ в виде суперкристалла, содержащего $\sim 10^8$ сверхбыстродействующих вентилей с временем переключения $\sim 10^{-11}$ с. Теперь логическую сеть приходится рассматривать как совокупность "мгновеннодействующих" логических элементов и медленнодействующих соединений с длинами порядка размера кристалла. Распространение сигналов по таким соединениям описывается теорией электрических цепей с распределенными параметрами. Принцип дальнодействия (глобальности) здесь утрачивает адекватность. Это обстоятельство требует пересмотра системотехнических принципов построения ЭВМ.

Перспективным путем является построение вычислительных систем на принципе локальности, предложенным А.И.Мишиным в 1977 г. [1,2]. Он заключается в том, что каждый элемент системы (процессор или модуль памяти) работает в своем темпе, слабо (логарифмически) зависящем от линейных размеров системы. Межэлементные взаимодействия осуществляются только посредством парных взаимодействий между соседними элементами. Для их упорядочения во времени используются не привязки к тактирующим сигналам, а непосредственно причинно-следственные связи между ними. Иначе говоря, квантование времени осуществляется событиями. Это позволяет неограниченно наращивать число элементов в системе без существенного понижения их быстродействия. В результате на практически важных параллельных алгоритмах достигается производительность, пропорциональная степени параллелиз-

ма задачи. Если специфика задачи требует увеличения разрядности слова с ростом размера задачи, то рабочая частота элемента локальной ВС обратно пропорциональна логарифму от размера задачи, а не размеру системы,как в случае глобальной ВС.

В данной статье рассматриваются основные свойства вычислительных систем и сред на принципе локальности в сопоставлении с глобальными вычислительными моделями. Приводимые оценки носят асимптотический характер относительно размера задачи (системы) с точностью до времени исполнения операции.

1. Производительность глобальной ЭВМ

Классическая ЭВМ состоит из процессора и памяти произвольного доступа для хранения программы и данных. Упорядочение событий в ЭВМ производится посредством привязки их к тактирующим сигналам. При линейном размере ЭВМ L и скорости распростра - нения сигналов v тактовая частота ν выбирается,исходя из условия $\tau_1 = 1/\nu \geq L/v$.

Все события, происходящие в пределах времени τ_1 ,рас - сматриваются как одновременные, а воздействия, распространяющиеся за время $\Delta t < \tau_1$, мгновенными. Поэтому вместо термина "принцип дальнодействия" мы будем употреблять термин "принцип глобальности", подчеркивая этим выполнение условия $\tau_1 \geq L/v$.

Предельная производительность ЭВМ есть $\Pi_1 = 1/\tau_1 \leq v/L$. Она достигается при однотипной (векторной) обработке больших массивов данных в конвейерной ЭВМ за счет распараллеливания памяти (разделения ее на ряд независимо работающих модулей),конвейеризации доставки данных к процессору и конвейеризации вычислительных операций.

Скорость распространения сигналов у машин первых поколений составляла 10^8 м/с, а $\tau_1 \gg L/v$, т.е. линейные размеры ЭВМ можно было бы увеличить в десятки раз без нарушения принци-

па глобальности. Но уже машины третьего поколения производительностью 10^8 опер./с, например "Крей", достигают своих допустимых размеров порядка 1 м. В силу особенностей микроминиатюризации схем скорость V снижается и уже сегодня составляет $\sim 10^7$ м/с. Поэтому ЭВМ производительностью 10^9 опер./с имеет размеры порядка 1 см. На 1 $см^2$ можно разместить процессор с емкостью памяти $\sim 10^6$ бит. На такой ЭВМ можно решать лишь задачи малого размера. Процессор с указанной производительностью близок к теоретическому пределу для электронных устройств.

2. Принцип локальности

Новый принцип организации вычислительных машин - *принцип локальности*. Основные свойства таких машин следующие [3]:

а) каждый элемент (процессор, модуль памяти) работает в своем темпе (асинхронность). При этом внутренне он может быть организован по принципу глобальности;

б) элементы расположены на минимальном (единичном) расстоянии друг от друга и взаимодействуют с соседями по мере готовности их к взаимодействию (локализация сообщений и локализованная синхронизация);

в) для упорядочения событий (смены состояний) во времени используются не привязки их к тактирующим сигналам, а непосредственно причинно-следственные связи между событиями, посредством которых и осуществляется квантование времени;

г) задержка при выполнении вычислительной операции процессором-вычислителем есть τ, а задержка при трансляции любым элементом - δ;

д) параллельность и конвейерность обработки информации; для практически важных задач большую часть операций составляют операции локальные, так что локальная ВС наиболее адекватна структурам задач.

Некоторые задачи, связанные с коллективным поведением (синхронизацией) автоматов, для локальных ∂C утрачивают смысл (события, одновременные в одной системе отсчета, неодновременны в другой).

Проиллюстрируем свойства "а"-"д" вычислительных машин с локальными взаимодействиями элементов на примере выполнения тьюринговых вычислений.

Машина Тьюринга состоит из логического блока-вычислителя (конечного автомата) и неограниченно наращиваемой ленты, разделенной на ячейки. На k-м шаге вычислитель, находясь во внутреннем состоянии $q_\alpha^{(k)} \in \{q_0, \ldots, q_A\}$, считывает знак $a_\beta^{(k)} \in \{a_0, \ldots, a_B\}$ из обозреваемой ячейки и вырабатывает новый знак $a_\beta^{(k+1)}$, записываемый на место прежнего, новое состояние $q_\alpha^{(k+1)}$ и значение символа $c^{(k+1)} \in \{-1, +1, 0\}$, в соответствии с которым осуществляется сдвиг ленты влево, вправо или она остается на месте, после чего вычислитель переходит в новое состояние $q_\alpha^{(k+1)}$.

Теория алгоритмов не интересуется структурой команды сдвига $c^{(k)}$. Однако ее можно представить в виде совокупности более простых (элементарных) сдвигов. Введем систему координат, принимая за начало отсчета положение вычислителя. Конкретная машина Тьюринга работает с лентой конечной длины N (за единицу длины принят линейный размер ячейки). На k-м шаге рабочие ячейки занимают зону длиной $2N$ на промежутке $[-N+1, N-1]$. Обозначим через X $(2N-1)$-мерное векторное пространство, где каждый вектор $x^{(k)} = x_i^{(k)}$, $i = -N+1, \ldots, N-1$, имеет компонентами содержимое $2N-1$ ячеек. Тогда сдвигу всей ленты соответствует отображение $X \times C^{(k+1)} \to X$, называемое глобальным сдвигом. Компоненты глобального сдвига $(x^{(k)}, c^{(k+1)}) \to x_i^{(k+1)}$ назовем локальными сдвигами.

Тьюринговы вычисления рассмотрим на цепочке конечных автоматов (рис.1), содержащей вычислитель P и 2N+2 элементов памяти M . Вычислитель и элемент памяти выполняют функции приема, хранения и передачи информации, а вычислитель, кроме того, и логические операции.

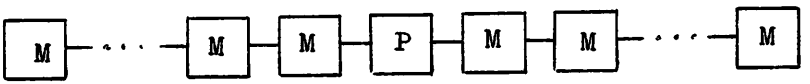

Рис. 1

Опишем взаимодействие элементов памяти M . Для этого введем следующие обозначения их состояний:

$\overset{\rightarrow}{\rightarrow}$ $\overset{\rightarrow}{\leftarrow}$

? и ? означают, что элемент "готов передать направо (верхняя стрелка) команду сдвига-приема информации слева или справа

$\overset{\leftarrow}{\leftarrow}$ $\overset{\leftarrow}{\rightarrow}$ $\overset{\rightarrow}{\rightarrow}$ $\overset{\rightarrow}{\leftarrow}$

(нижняя стрелка)"; ? и ? симметричны символам ? и ?;

! означает, что элемент "готов принять информацию слева или справа";

$\overset{\leftarrow}{}$

! означает, что правый граничный элемент "готов принять

$\overset{\rightarrow}{}$ $\overset{\leftarrow}{}$

информацию слева"; ! симметричен символу !.

Сдвиг информации осуществляется посредством элементарных (парных) взаимодействий элементов. Парное взаимодействие выполняется по принципу "запрос-ответ" и состоит в передаче знака $a_{\beta}^{(k)}$ и команды сдвига одним элементом и приеме этой информации другим элементом пары. Состояния элементов пары до и после взаимодействия можно записать в виде подстановок:

а) $\overset{\rightarrow}{\rightarrow}$? $\overset{\rightarrow}{\leftarrow}$! $\rightarrow$! $\overset{\rightarrow}{\rightarrow}$?, б) $\overset{\rightarrow}{\rightarrow}$? $\overset{\rightarrow}{\leftarrow}$! $\rightarrow$! $\overset{\rightarrow}{\leftarrow}$?, в) $\overset{\leftarrow}{\leftarrow}$! $\overset{\rightarrow}{\rightarrow}$ $\rightarrow$? $\overset{\leftarrow}{\leftarrow}$!,

г) $\overset{\leftarrow}{\rightarrow}$! $\overset{\leftarrow}{\leftarrow}$ $\rightarrow$? $\overset{\rightarrow}{\rightarrow}$!, д) $\overset{\rightarrow}{\rightarrow}$? $\overset{\rightarrow}{\leftarrow}$! $\rightarrow$! $\overset{\leftarrow}{\leftarrow}$!, е) $\overset{\leftarrow}{\leftarrow}$! $\overset{\rightarrow}{\rightarrow}$ $\rightarrow$! $\overset{\leftarrow}{\leftarrow}$!.

Подстановки "а" и "б" задают сдвиги информации в правой от вычислителя P цепочке элементов, "в" и "г" - в левой цепочке, "д" и "е" описывают работу правого и левого граничных элементов.

Взаимодействие вычислителя P с соседними элементами памяти M осуществляется особым образом. Не вдаваясь в подробности, укажем, что вычислитель, выполнив логические функции по определению $a_{\beta}^{(k+1)}, q_{\alpha}^{(k+1)}, c^{(k+1)}$, переходит к взаимодействию с соседями, например, при значении $c^{(k+1)} = -1$ он посылает левому соседу новый знак $a_{\beta}^{(k+1)}$ и команду сдвига влево, затем считывает знак из правого элемента памяти и посылает ему команду сдвига влево. Аналогичные действия выполняются при $c^{(k+1)} = +1$.

Перед началом работы все элементы памяти содержат знаки $a_{\beta}^{(0)}$ и находятся в состоянии приема информации. Вычислитель, находясь в состоянии $q_{\alpha}^{(0)}$, выполняет логические функции, т.е. находит $a_{\beta}^{(1)}$, $q_{\alpha}^{(1)}, c^{(1)}$, и посылает соседним элементам команды сдвига. Каждую последующую команду он передает по мере готовности соседей к взаимодействию, не дожидаясь выполнения предыдущих команд всеми элементами. Следовательно,в цепочке параллельно исполняется множество различных команд сдвига, в результате чего тьюрингова операция исполняется как совокупность N локальных сдвигов и логической операции.

Понятие машины Тьюринга может быть обобщено на r-мерную решетку ($r = 1,2,3$), в узлах которой расположены конечные автоматы-вычислитель и элементы памяти, соединенные с $2r$ соседями. Такую структуру, реализуемую в r-мерной среде с локальными взаимодействиями,можно рассматривать как неограниченно наращиваемую логическую сеть. При ее работе появляется особая операция - трансляция (прием-передача слова элементом), обусловленная размещением элементов и данных во времени и про-

странстве. В классическом алгоритме трансляций нет, они появляются только при вложении алгоритма в вычислительную модель и вносят дополнительный вклад во временную сложность алгоритма.

Если логическая сеть организуется на принципе глобальности, то длительность одного рабочего такта машины τ_1 ограничена снизу величиной $L/v = N^{1/r}/v$. Временная сложность алгоритма есть $T_1 = s\tau_1 = sN^{1/r}/v$, где s - число рабочих тактов. Если логическая сеть организуется на принципе локальности, то в виду конвейеризуемости глобального сдвига операция исполняется на локальной системе в темпе работы вычислителя. Поэтому временная сложность алгоритма определяется числом локальных тактов $T = s\tau$, где τ - длительность такта вычислителя. Ускорение процесса вычислений по сравнению с глобальной реализацией машины Тьюринга составляет $T_1/T = \tau_1/\tau \sim N^{1/r}$.

3. Глобальные машины параллельного действия

Исторически первой появилась концепция клеточных автоматов. Как и машина Тьюринга, они используют идею r-мерной решетки, в узлах которой помещаются элементы-конечные автоматы. Клеточный автомат допускает параллельную работу всех своих элементов и при вычислениях, и при трансляциях, причем и те и другие выполняются за одинаковое время τ_1 .Несмотря на солидный возраст клеточные автоматы не оказали существенного влияния на вычислительную технику.Дело в том,что элемент-процессор коллектива вычислителей должен быть достаточно сложным, чтобы интер - претировать программу, записанную в его памяти, а теория автоматов не располагает языком для компактного задания такого вычислителя. Поэтому клеточные автоматы реализуются в основном в виде разного рода специализированных процессоров, получивших название систолических матриц (массивов).

Для повышения эффективности клеточных автоматов в составляющие их конечные автоматы были введены соединительные элементы "мгновенного" действия для выполнения трансляций. Такие однородные системы были предложены за рубежом в 1960 г. и получили название итеративных систем Холланда [5], а в отечественной литературе однородных вычислительных систем и сред [6]. Подобные системы и среды создаются и сегодня, в частности, в виде матричных процессоров.

При физической реализации ОВС элементы взаимодействуют не мгновенно, а с задержкой $\delta \ll \tau_1$. Синхронная работа возможна поэтому только в окрестности радиуса $R \leq \tau_1/\delta$ (за единицу длины принят линейный размер элемента), так что максимальное число N_1 процессорных элементов в системе ограничено величиной R^r. Следовательно, ограничены и размеры решаемых задач и производительность ОВС.

Потенциальную производительность ОВС определяют в предположении параллельной работы всех элементов как $\mu_0 = N_1/\tau_1$. Поэтому, казалось бы, увеличив длительность такта в μ раз, можно увеличить число элементов до $\mu^r N_1$ и поднять производительность до $\Pi_0' = \mu^{r-1}\Pi_0$ для двух- и трехмерных ОВС. В этом и лежат истоки концепции "неограниченного параллелизма".

Однако при определении реальной производительности ОВС следует учитывать потери времени на обмены данными между процессорами и возможные простои некоторой их части в силу особенностей алгоритма. Обозначим усредненное количество вычисляющих процессоров через m. Под размером задачи n будем понимать размер массива данных, например, размер r-мерной матрицы. Очевидно, должно быть $N_1 \geq n^r$. Количество вычисляющих процессоров характеризуется функцией $m = f(n)$, называемой *глубиной параллелизма* задачи.

Типичным является степенной закон $m = O(n^l)$ ($l = 0,1,2,3$). Примерами таких алгоритмов являются алгоритмы вычисления l-мерных интегралов и l-мерные явные разностные схемы в граничных задачах для уравнений в частных производных. Глубина параллелизма неявных разностных схем, реализуемых методом дробных шагов, на порядок меньше и равна $O(n^{l-1})$. Широкий класс составляют алгоритмы, где $m = O(n)$. Сюда относятся различные методы решения линейных уравнений, в частности, метод Гаусса с выбором главного элемента, итерационные методы решения уравнений, численный анализ изображений, многие информационно-логические задачи и т.д. Для последовательных алгоритмов $m = O(1)$; среди них выделим алгоритмы с регулярной обработкой данных, например, вычисления многочленов по схеме Горнера. Для максимального распараллеливания алгоритма размерность ВС должна быть не меньше показателя степени в выражении функции $m(n)$. В противном случае глубина параллелизма ограничивается размерностью ОВС.

Параллелизм является характеристикой алгоритма, но не задачи. Если одну и ту же задачу решать разными способами, то может статься, что алгоритм с большей глубиной параллелизма потребует выполнения большого числа операций и не приведет к снижению временной сложности задачи. Например, явные разностные схемы в сравнении с неявными (с расщеплением) имеют на порядок большую глубину параллелизма, но временная сложность тех и других на уравнениях параболического типа одного порядка.

Реальная производительность ОВС будет $\Pi = m/\mu\tau_1$, т.е. пропорциональна глубине параллелизма алгоритма. И хотя с увеличением размера задачи до $n = \mu M = \mu\tau_1/\delta$ количество вычисляющих процессоров также возрастает, этого недостаточно для роста производительности во всех случаях параллельных алгоритмов.

При $m = n^l$ имеем $\Pi = n^l/\mu\tau = (\mu\tau)^{l-1}/\delta^l$. Отсюда видно, что увеличение производительности с ростом μ для степен-

ного закона параллелизма будет наблюдаться только при $l \geq 2$. На алгоритмах с глубиной параллелизма $m = n$, $l = 1$ имеем $\Pi = 1/\delta$, т.е. производительность обратно пропорциональна задержке элемента при выполнении соединительной функции. На последовательных алгоритмах $(l = 0)$ она асимптотически стремится к нулю как $1/\mu$.

Заблуждения, связанные с непониманием указанных фактов, проникли уже в вузовские учебники (см., например, К.Г.Самофалов, Г.М.Луцкий. Структуры и организация функционирования ЭВМ и систем. - Киев, 1978. - С. 16).

Рассмотрим часто фигурирующий в литературе пример распараллеливания при вычислениях выражений вида $F = f_1 \otimes f_2 \otimes \ldots \otimes f_n$, где $\otimes$ - знак бинарной операции. Если операции ассоциативные, то на глобальной n-процессорной линейной системе можно вычислить F по схеме сдваивания за $\log_2 n$ шагов и временная сложность алгоритма есть $T = \mu\tau_1 \log_2 n$. Реальная производительность будет $\Pi = n/T = 1/\delta \log_2 n$, т.е. стремится к нулю с ростом n , хотя и не так быстро, как для последовательных алгоритмов. Вычисление выражения за наименьшее число шагов не является решением задачи распараллеливания, как это ошибочно утверждается в статье В.Е.Котова (см. Котов В.Е. Программирование параллельное //Математическая энциклопедия. Т. 4. -М., 1984. - С. 647-649). Необходимо учитывать затраты времени на операции суперпозиции и подстановки.

В последнее время возлагаются надежды на достижение производительности, пропорциональной числу процессоров в системах потоковой, конвейерной и мультипроцессорной архитектуры, например, по проекту МАРС. Но поскольку эти системы основываются на принципе глобальности, то для них справедливы все указанные выше ограничения.

Активизировались также работы в области оптических вычислительных средств, якобы обещающих снять ограничения, прису -

щие электронным системам, за счет обеспечения оптическим путем произвольных глобальных связей между элементами [4]. Однако, как явствует из вышесказанного, именно наличие глобальных связей независимо от их природы (электрических, оптических и т.д.) есть основной фактор, ограничивающий производительность вычислительных машин.

В самом деле, для оптической ВС размер логического вентиля близок к величине $3\lambda\sqrt{N_в}$, где λ - длина световой волны, $N_в$ - общее число вентилей в системе. Время распространения сигнала по линейному размеру ВС есть $\tau_1 = 3\lambda\sqrt{N_в}\sqrt{N_в}/c$ (c - скорость света), а тактовая частота $\nu = 1/\tau_1 = c/3\lambda N_в$. Если γ - число вентилей в одном процессоре, то число последних в системе будет $m = N_в/\gamma$. Потенциальная производительность такой ВС есть $П_0 = \nu m = c/3\lambda\gamma$. В отличие от электронных систем она не зависит от числа элементов (размера задачи) и является физической константой. Если принять $\lambda = 10^{-6}$ м, $\gamma = 10^5$, то производительность ВС будет $П_0 = \nu = 10^9$ опер./с. Таким образом, оптический подход не может решить проблему создания ВС со сверхвысокой производительностью.

4. Параллельные ВС на принципе локальности

Система параллельного действия с локальными взаимодействиями элементов обладает всеми достоинствами глобальных ВС (благодаря возможности выполнения элементом соединительной функции - трансляции информации - за минимально возможное время δ, определяемое линейным размером элемента) плюс свойством локализованной синхронизации (самосинхронизации). Это свойство, с одной стороны, позволяет существенно ослабить (до логарифма от размера задачи) зависимость τ от линейного размера системы и тем самым неограниченно наращивать число элементов в ней без существенного увеличения τ. В результате производи-

Таблица

Метрические характеристики различных ВС

Типы вычислительных машин	Структура машины	Метрика-расстояние между соседними элементами
Машина Тьюринга (МТ)	Решетка	$\rho_{МТ} \sim \sqrt[r]{N}$
Машина Неймана (ЭВМ)	Звезда	$\rho_{ЭВМ} \sim L$
Синхронный клеточный автомат (СКА)	Решетка	$\rho_{СКА} \sim \sqrt[r]{N}$
Асинхронный клеточный автомат (АКА)	Решетка	$\rho_{АКА} = \tau$
Итеративные системы Холланда и однородные вычислительные среды (ОВС)	Решетка	$\rho_{ОВС} \in \{\rho_{МТ}, \rho_{ЭВМ}, \rho_{СКА}\}$
Локальные асинхронные вычислительные среды (ЛВС)	Решетка	$\rho_{ЛВС} \in \{\delta, \tau\}$

ПРИМЕЧАНИЕ: ρ - расстояние между смежными элементами,

N - число элементов в машине,

L - линейный размер машины,

τ - задержка процессорного элемента,

δ - задержка соединительного элемента ($\delta \ll \tau$),

r - размерность решетки.

тельность локальной ВС целиком определяется алгоритмом. С другой стороны, указанное свойство обеспечивает максимальное значение *плотности событий* - произведения плотности размещения элементов на рабочую частоту, которая является главной характеристикой качества ВС. Отношение плотностей событий локальных и глобальных систем пропорционально отношению линейных размеров системы и элемента и равно $n\log n/\log n = n$.

На практике локальные системы реализованы в виде асинхронных клеточных автоматов, названных *волновыми процессорами* и предназначенных, как и систолические структуры (синхронные клеточные автоматы), для решения задач с регулярной граф-схемой алгоритма [4].

Метрические характеристики различных ВС представлены в таблице.

5. Организация взаимодействий элементов в локальной системе

Элемент локальной ВС является программируемым автоматом, для которого время вычислений между двумя последовательными взаимодействиями варьируется в зависимости от выполняемых операций. Поэтому локальную ВС можно рассматривать как универсальную вычислительную среду, позволяющую реализовать любые вычислительные схемы. В частности, на r-мерной среде можно моделировать без растяжения во времени глобальную однородную ВС с числом элементов $(\mu\tau/\delta)^r$, где коэффициент $\mu \geq 1$, как и выше, отражает зависимость частоты работы элемента $1/\mu\tau$ от их общего числа в системе.

При реализации параллельных вычислений на локальной ВС процессорные элементы многократно взаимодействуют между собой. Рассмотрим, как это выполняется на кольцевой ВС из m процессоров (рис.2). Пусть каждый из процессоров должен передать

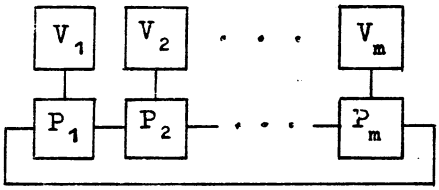

Рис. 2

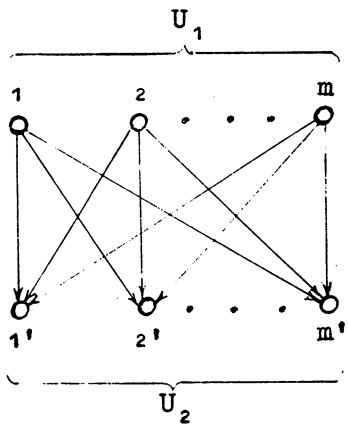

Рис. 3

всем остальным одно слово и принять от них их слова. Необходимость в таком обмене возникает, например, при решении системы m уравнений методом простых итераций. Эта схема взаимодействий может быть представлена полным двудольным графом (рис.3), одному подмножеству вершин которого $U_1 = (1,2,\ldots,m)$ поставлены в соответствие процессоры, работающие в режиме передачи информации, а вершинам другого подмножества $U_2=(1',2',\ldots,m')$ - процессоры, принимающие информацию. Схема обмена может быть описана циклической подстановкой $(1,2,\ldots, m)$ порядка m. Однократное выполнение подстановки означает прием слова i-м элементом от $(i-1)$-го элемента и передачу слова $(i+1)$-му элементу; m-кратное выполнение подстановки реализуется на кольцевой ВС за время $m\tau$ одного оборота m слов по кольцу. В результате выполняется m^2 элементарных взаимодействий "прием-передача". Это *циклический* способ обмена.

В случае, когда обмен информацией между процессорами осуществляется массивами данных, более эффективен *трансляционно-циклический* способ реализации полной схемы взаимодействий. При этом способе процессоры передают свои массивы,

например, вправо по кольцу по очереди. Передача осуществляется так: один "ведущий" процессор передает, остальные "ведомые" принимают и транслируют данные, а левый сосед ведущего только принимает данные, но не передает, он "поглотитель". Перед началом следующей передачи функции процессоров могут меняться. Отметим, что в рассмотренной в п.2 линейной ВС поглотителями являются левый и правый граничные элементы.

Если количество слов в массивах меньше числа процессоров в кольце, то передающий процессор может выполнять и функции поглотителя. Для этого после передачи массива он передает функции ведущего соседу справа, а сам становится на прием (поглощение данных) переданного им ранее массива. Выполнив эту операцию, он переходит в режим ведомого.

Из вышесказанного следует, что в памяти каждого процессора должны храниться, по крайней мере, три подпрограммы его работы: в режиме ведущего, в режиме ведомого и в режиме поглотителя. Разумеется, кроме названных, процессор содержит и другие служебные подпрограммы, например, интерпретатор операторов языка, а "главный" процессор, посредством которого осуществляется общее управление вычислительной средой, также и программу ввода-вывода данных, интерпретируемую программу и полный интерпретатор.

Рассмотрим моделирование однопроцессорной машины с адресуемой памятью произвольного доступа на кольцевой системе (см. рис. 2), процессоры которой имеют кольцевые памяти из V запоминающих элементов [7].

Исходные данные представим в виде двумерной таблицы $A = [a_{ij}]$. Значениям первых индексов $i \in (1,\ldots,m')$, $m' \le m$, соответствуют номера процессоров. Значения вторых индексов $j \in (1,\ldots,V'), V' \le V$, записываются в адресные разряды запоминающих элементов, в информационные разряды кото-

рых записываются значения a_{ij} . Процессор P_1 выделяется в качестве главного для выполнения функций вычислителя, остальные процессоры выполняют функции поиска и доставки данных к нему. Предполагается, что процессор P_1 имеет память программ, в которую записана программа его работы. Предполагается также, что передача информации по процессорному кольцу и запоминающим кольцам осуществляется в одном направлении, например, по часовой стрелке.

Система функционирует следующим образом. Процессор P_1 из памяти программ считывает очередную команду, например одноадресную, и начинает поиск слова, записанного по указанному двуместному адресу (i,j) . Адрес посылается по процессорному кольцу. Получая его, очередной процессор сравнивает первую компоненту этого адреса со своим номером. Если они совпадают, то процессор выполняет поиск слова в своем запоминающем кольце, если нет, то передает адрес дальше.

При поиске слова в запоминающем кольце процессор поочередно считывает слова из него и сравнивает их по номеру j . В случае совпадения он посылает слово по процессорному кольцу к главному процессору P_1 . Последний, получив его, выполняет вычисления, после чего считывает очередную команду.

Отметим, что в рассмотренном режиме работы системы парное взаимодействие соседних элементов памяти может быть задано одной подстановкой ? ! → ! ?, где, как и ранее, символ ? означает состояние передачи, а символ ! состояние приема информации. Взаимодействие процессора с соседними элементами (процессорными или запоминающими) задается управляющими сигналами $y_p \in \in \{0,1\}$ и может быть описано обобщенной подстановкой. Для этого введем символы:

$?(y_1 P \vee M y_2)$ означает программируемое состояние передачи информации процессору $P(y_1 = 1, y_2 = 0)$ или элементу памяти $M(y_1 = 0, y_2 = 1)$;

$!(y_3 P \vee M y_4)$ означает программируемое состояние приема информации от процессора $P(y_3 = 1, y_4 = 0)$ или элемента памяти $M(y_3 = 0, y_4 = 1)$.

Обобщенная подстановка имеет вид:

$$?(y_1 P \vee M y_2)\,!(y_3 P \vee M y_4) \rightarrow !(y_3 P \vee M y_4)\,?(y_1 P \vee M y_2).$$

При $y_1 = y_2 = y_3 = y_4 = 0$ процессор находится в состоянии обработки информации. Значения управляющих сигналов y_p задаются программой работы процессора.

Время исполнения операции главным процессором P_1 определяется временем доставки к нему операнда, пропорциональным линейному размеру системы. Однако при исполнении векторных операций по процессорному кольцу перемещается уже не одно слово, как при скалярной операции, а поток слов. При этом производительность системы с одним вычисляющим процессором достигает максимального значения $1/\tau$, где время исполнения операции процессором $\tau \sim \log n$.

Организация параллельных вычислений на локальной ВС рассмотрена в работе [7]. Из сделанных там выводов отметим лишь некоторые. Во-первых, для записи программ могут быть использованы известные алгоритмические языки, например АЛГОЛ, расширенные процедурами по взаимодействию элементов (см., например, команды взаимодействия при моделировании машины Тьюринга и традиционной ЭВМ). Во-вторых, можно применять языки высокого уровня типа АПЛ, позволяющие записывать программы в терминах крупных операторов (векторных, матричных). В-третьих, потребовалась разработка нового метода ассоциативного (безадресного) управления вычислительным процессом, при котором значительная часть программного управления связывается с данными.

6. Производительность локальных ВС

Центральным понятием с точки зрения временной сложности алгоритма является понятие операции [8]. В общем случае операция определяется как пара "элементарная операция - доставка операндов".

Суперпозиция r трансляций $T_1 \circ T_2 \circ \ldots \circ T_r = T^r$ (трансляция T есть прием-передача слова элементом) называется *доставкой*. Если за дискретную единицу измерения расстояния в системе принять линейный размер d элемента, пропорциональный логарифму от размера задачи n, то длина доставки T^r равна $rd \sim r \log n$. Длина доставки называется *радиусом операции*. Операция, радиус которой растет не медленнее степенной функции от размера задачи, называется *глобальной*, в противном случае операция называется *локальной* [8].

В традиционных ЭВМ и ВС время доставки операндов включается в длительность выполнения операции процессором, т.е. здесь все операции глобальные.

Параллелизм алгоритма характеризуется теперь не просто числом независимых на каждом шаге операций, но также удельным весом среди них операций локальных. Чем он больше, тем большее число вычисляющих процессоров можно задействовать на каждом этапе вычислений [8].

Реальная производительность ВС растет пропорционально количеству вычисляющих процессоров: $\Pi = m/\tau$. Поэтому в случае степенного закона параллелизма алгоритма $m = O(n^1)$ производительность локальной системы в $n \cdot \log n / \log n = n$ раз выше, чем глобальной. Даже при $m = O(1)$ на регулярных алгоритмах она остается равной $1/\tau$, а не стремится к нулю, как $1/\mu$, в случае глобальной системы. Еще раз подчеркнем что это достигается за счет более высокого темпа выполнения локальных операций, как это было продемонстрировано на тьюрин-

говых вычислениях. И только тогда, когда основная часть операций в алгоритме глобальные, производительность локальной ВС того же порядка, что и глобальной.

7. Архитектура локальных ВС

Глубина параллелизма алгоритма зависит от того, насколько архитектура системы соответствует структуре данных алгоритма. Предлагается ВС с тремя уровнями иерархии, отражающими это требование: центральная ЭВМ, кольцевая ВС и двумерная вычислительная среда типа дискретного тора.

На *центральной* ЭВМ решаются нераспараллеливающиеся задачи, а также готовятся данные при использовании ВС и среды.

Кольцевая ВС - это кольцо из m процессоров с модулями адресуемой памяти емкостью V слов. Темп работы процессоров в кольце при двумерной схеме организации модулей памяти может в $\sqrt{m}$ раз превосходить темп работы одной ЭВМ с памятью емкости mV. Это эффект распределенной памяти. Для глобальных ВС он отсутствует. При реализации на кольцевой ВС параллельных алгоритмов с глубиной параллелизма не ниже $O(n)$, $n \sim m$, производительность системы в m раз выше производительности одного процессора. Повышение производительности по сравнению с одной ЭВМ с емкостью памяти mV может достигать $m\sqrt{m}$.

Если модули памяти у процессоров суть линейные, например кольцевые структуры [7], то можно использовать процессоры с предельным быстродействием, например 10^9 опер./с. В этом случае производительность системы дополнительно возрастает в $\sqrt{V}$ раз по сравнению с системой с модулями адресуемой памяти на задачах с большим объемом данных.

Вычислительная среда образуется процессорными элементами с малым набором команд, образующими дискретный тор

размера $m \times V$. Вычислительная система, содержащая двумерную среду из элементов c_{ij}, $i = 1,\ldots,m, j = 1,\ldots,V$, и кольцо процессоров $P_1,\ldots, P_m$, каждый из которых имеет блок памяти для хранения данных и программы, представлена на рис. 4 ($\square$ - процессор, • - элемент среды). На такой системе эффективно

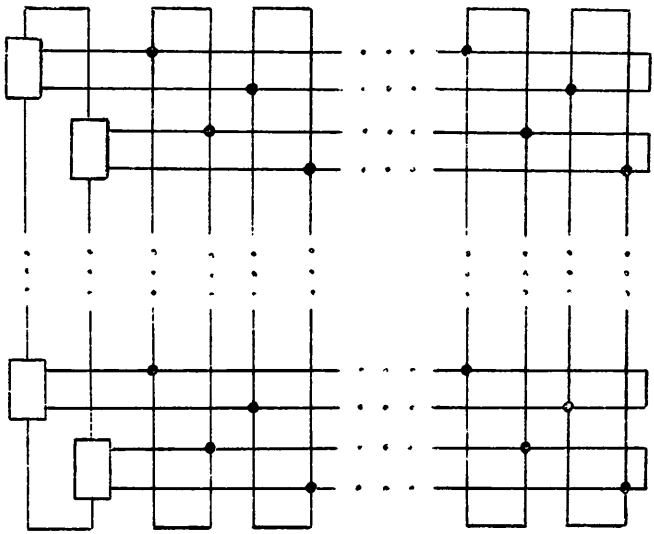

Рис. 4

реализуются алгоритмы с глубиной параллелизма $O(n^2)$ и объемом данных порядка mV . Производительность системы оказывается в $m^2\sqrt{mV}$ раз выше производительности одной ЭВМ с суммарной емкостью памяти mV . Особенности локальных вычислений (оценки временной сложности алгоритма и производительности) на кольцевой и двумерной ВС рассмотрены в работах [8,9] на примерах задач численного синтеза изображений и механики сплошной среды. Подобные системы эффективны при решении задач в систе - мах распознавания сложных образов и экспертных систем.

Теоретически можно представить двумерную ВС с линейными модулями памяти. На такой ВС эффективно реализуются алгоритмы снова с глубиной параллелизма $O(n^2)$, но с большим объемом данных, например, трехмерные неявные разностные схемы. При этом производительность системы в $m^3\sqrt{V}$ раз больше производительности одной ЭВМ с суммарной емкостью памяти m^2V.

Уже при современной технологии реально создать кольцевую ВС из $m = 10^3$ процессоров производительностью на параллельных алгоритмах до 10^{12} опер./с, а вычислительную среду до 10^{15} опер./с.

В США производительность супер-ЭВМ "Cyber 180" увеличена путем присоединения процессорного кольца. Кроме того, разрабатывается ВС на 10^{12} опер./с на базе процессоров производительностью 10^9 опер./с. В Англии созданы экспериментальные варианты квазилокальной вычислительной среды на 10^9 опер./с и 10^{10} опер./с стоимостью в 10 раз меньше стоимости супер-ЭВМ той же производительности. Англичане назвали свою среду "Computer Surface" ("вычислительная поверхность"), а ее элемент "Transputer". Термин "транспьютер" (транзистор + компьютер) подчеркивает, что по отношению к создаваемым из этих элементов большим системам он рассматривается как атомарный, аналогично транзистору в дискретных системах. Именно с этих позиций были более 25 лет назад предложены отечественные тремины "вычислительная среда" и "элемент среды".

Авторы благодарят В.А.Леуса за обсуждение работы, способствовавшее ее улучшению.

Литература

1. А.с. 742926 СССР. Вычислительная среда /А.И. Мишин - Опубл. в Б.И., 1980, № 23.

2. А.с. 767752 СССР. Ячейка вычислительной среды / А.И.Мишин. - Опубл. в Б.И., 1980, № 36.

3. МИШИН А.И. Параллельные вычислительные среды с локальными взаимодействиями элементов //Автоматика и телемеханика. - 1982. - №10. - С. 147-154.

4. Оптическая вычислительная техника //ТИИЭР. - 1984. - Т. 72, № 7.

5. ХОЛЛАНД Дж. Вычислительные машины, построенные по итеративной схеме //Микроэлектроника и большие системы. -М.,1967. - С. 145-153.

6. ЕВРЕИНОВ Э.В., КОСАРЕВ Ю.Г. Однородные вычислительные системы высокой производительности. - Новосибирск: Наука,1966. - 306 с.

7. ВАЛИЕВ М.К., МИШИН А.И. Организация параллельных вычислений на системе с локальными взаимодействиями элементов //Автометрия. - 1983. - №6. - С. 88-96.

8. ЛЕУС В.А., МИШИН А.И. Временная сложность задачи визуализации на различных вычислительных моделях. -Новосибирск,1986. - 40 с.- (Препринт/АН СССР. Сиб.отд-ние. Ин-т математики;№29).

9. ЗАВЬЯЛОВ Ю.С., МИШИН А.И. Временная сложность алгоритмов задач гидро-аэродинамики и производительность параллельных вычислительных систем. - Новосибирск, 1985. - 8 с. - (Препринт/АН СССР, Сиб. отд-ние. Ин-т математики;№20).

Поступила в ред.-изд.отд.

22 апреля 1987 года