АРХИТЕКТУРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ С ПРОГРАММИРУЕМОЙ СТРУКТУРОЙ
(Вычислительные системы)
1980 год Выпуск 82

УДК 681.322.06:681.3.323

О ДЕЦЕНТРАЛИЗОВАННОМ РАСПРЕДЕЛЕНИИ ЗАДАНИЙ В ОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ С ПРОГРАММИРУЕМОЙ СТРУКТУРОЙ

В.В.Корнеев, О.Г.Монахов

Однородные вычислительные системы с программируемой структурой представляют собой совокупность N, $N \in \{1,2,\dots\}$, элементарных машин $ЭМ_1,\dots, ЭМ_N$, объединенных сетью связи с децентрализованным распределенным по ЭМ программируемым управлением [1–4]. Поступление заданий в систему возможно как с обобществленных внешних устройств [3–5], распределенных по системе, так и из самих ЭМ. В последнем случае задания порождаются в ходе исполнения ранее поступивших заданий. Задание s, требующее для своей реализации $k \le N$ элементарных машин, может реализовываться на любых k машинах, образующих связную адресную подсистему Q_s. Каждая ЭМ, принадлежащая Q_s, имеет поставленный ей в одно-однозначное соответствие адрес ЭМ в указанной подсистеме. В случае вхождения $ЭМ_j$ в несколько $Q_1,\dots, Q_q$, $R \in \{1,2,\dots\}$, $q \in \{1,\dots,R\}$, подсистем $ЭМ_j$, $j \in \{1,\dots, N\}$, имеет адреса $A_{j1},\dots, A_{jq}$, служащие для выделения рассматриваемой $ЭМ_j$ среди машин соответствующих виртуальных адресных подсистем $Q_1,\dots, Q_q$. Адреса A_{jr}, $r \in \{1,\dots, q\}$, $ЭМ_j$, $j \in \{1,\dots, N\}$, используются для определения (посредством путевой процедуры [1,3]) трасс передачи данных между машинами адресной подсистемы Q_r. В [3,6,7] рассмотрена $D(z,m)$-адресация машин подсистемы, обеспечивающая эффективную реализацию путевых процедур и, соответственно, эффективную организацию параллельных вычислений в выделенных для исполнения заданий адресных подсистемах. Настоящая работа посвящена децентрализованным алгоритмам построения адресных подсистем, реализующим выделение подсистем и задание адресов машин в них. Интерес к указанному классу алгорит-

мов обусловлен двумя обстоятельствами. Во-первых, децентрализованные алгоритмы, в отличие от централизованных [8], не требуют сбора информации о состоянии системы и поступивших в нее заданиях. И во-вторых, предлагаемые в настоящей работе децентрализованные алгоритмы могут быть использованы в комбинации с централизованными [8]. Последние составляют план разбиения системы на подсистемы, предназначенные для исполнения поступивших заданий. Само разбиение и доставка заданий в подсистемы с требуемым числом машин осуществляются децентрализованным алгоритмом выделения подсистем.

I. Выделение подсистем в вычислительных системах с программируемой структурой

I.I. Децентрализованные алгоритмы распределения заданий реализуются вычислительной системой с программируемой структурой как совокупность одинаковых алгоритмов, исполняемых в каждой ЭМ над своими данными и данными соседних с ней ЭМ. При этом вычислительная система функционирует как клеточный автомат [9], в роли клеток которого выступают ЭМ системы. Клетка такого автомата характеризуется состоянием $x \in X$, где X - множество допустимых состояний клетки. При функционировании клеточного автомата каждая клетка преобразует свое состояние в соответствии с одной и той же для всех клеток функцией от состояния самой клетки и состояний соседних с ней клеток. Преобразования заканчиваются при достижении каждой клеткой состояния, являющегося неподвижной точкой преобразования.

Применительно к вычислительным системам с программируемой структурой соседство клеток (элементарных машин) задается графом межмашинных связей. Элементарная машина $ЭМ_i$, $i = \overline{1,N}$, имеет v_i, $v_i \in \{1,2,\dots\}$, входных и v_i выходных полюсов для связи с другими машинами системы. Каждому входному (выходному) полюсу поставлен в одно-однозначное соответствие его номер p, $p \in \{1,\dots, v_i\}$. Линия межмашинного обмена (i_u, j_w), $u = \overline{1,v_i}$, $w = \overline{1, v_j}$, $i,j \in \{1,\dots,N\}$, $i \neq j$, обеспечивает передачу данных с выходного полюса u $ЭМ_i$ на входной полюс w $ЭМ_j$. Множество E ребер графа межмашинных связей обладает следующим свойством: если $(i_u, j_w) \in E$, то $(j_w, i_u) \in E$. Указанное свойство обеспечивает возможность каждой $ЭМ_i$ запросить состояние p-й соседней $ЭМ_j$, связанной с $ЭМ_i$ линиями (i_p, j_r) и (j_r, i_p), $r \in \{1,\dots, v_j\}$, $p \in \{1,\dots,v_i\}$. На уровне программно-аппаратных средств ЭМ запросы организуются как обмен ЭМ сообщениями [1-8].

I.2. Выделение подсистемы Q_s, предназначенной для реализации задания s, производится посредством двух операторов: оператора РАСПРЕДЕЛЕНИЕ ЗАДАНИЙ и оператора ЗАГРУЗКА ЗАДАНИЙ. Каждый из указанных операторов реализуется по своему децентрализованному алгоритму.

При исполнении децентрализованного алгоритма РАСПРЕДЕЛЕНИЕ ЗАДАНИЙ внутреннее состояние $ЭМ_i$, $i \in \overline{1, N}$, характеризуется состоянием трех списков T_i, L_i, C_i, содержащих сообщения о нераспределенных, загружаемых и исполняемых заданиях соответственно.

Задание, поступившее в $ЭМ_i$, формирует сообщение H_0 {имя задания, требуемое число машин}, которое заносится в список нераспределенных заданий T_i, $i \in \overline{1, N}$. Каждая $ЭМ_i$, $i = \overline{1, N}$, имеет нагрузку $\Sigma_i = |T_i| + |L_i| + |C_i|$, равную сумме длин списков T_i, L_i, C_i.

При ненулевой длине списка нераспределенных заданий каждая $ЭМ_i$, $i = \overline{1, N}$, определяет нагрузки Σ_j, $j \in J_i$, множества J_i соседних с $ЭМ_i$ машин, $J_i = \{j \mid (i_p, j_k) \in E,\ p = \overline{1, v_i},\ k \in \{1, \ldots, v_j\}\}$. Если для всех $ЭМ_j$, $j \in J_i$, выполняется $\Sigma_j - \Sigma_i > \delta$, то $ЭМ_i$ оставляет свое внутреннее состояние без изменения (здесь δ – наперед заданная величина допустимого рассогласования). Иначе, находится $ЭМ_t$ такая, что $t \in J_i$, $\Sigma_t - \Sigma_i = \min\limits_{j \in J_i} (\Sigma_j - \Sigma_i)$. В указанную $ЭМ_t$ пересылается одно сообщение из списка T_i $ЭМ_i$ в совокупности с текстом соответствующего задания. Пересланное сообщение помещается в список T_t.

Алгоритм РАСПРЕДЕЛЕНИЕ ЗАДАНИЙ при наличии в $ЭМ_i$, $i = \overline{1, N}$, списка нераспределенных заданий T_i, выполнении условия $\Sigma_j - \Sigma_i > \delta$ для всех $j \in J_i$ и сумме Π_i длин списков L_i и C_i, меньшей или равной некоторой константе M, задающей желаемую степень мультипрограммирования ЭМ, $\Pi_i = |L_i| + |C_i| \le M$, удаляет произвольное сообщение T_{ig} из списка T_i и формирует на его основе сообщение, заносимое в список L_i. Здесь T_{ig} – g-й элемент списка T_i. Сообщения, являющиеся элементами списка L_i загружаемых заданий $ЭМ_i$, $i = \overline{1, N}$, представляют семикомпонентный вектор {имя задания, уровень отказа, направление отхода, использованные направления, число ненайденных машин, искомый уровень, число найденных машин}. В дальнейшем будем обозначать через L_{ik}[идентификатор] значение соответствующего идентификатора k-го элемента списка L_i загружаемых заданий $ЭМ_i$, если L_{ik}[идентификатор] находится в правой части оператора присваивания. Если L_{ik}[идентификатор] стоит в

левой части оператора присваивания, то соответствующая компонента элемента списка заданий получает значение правой части оператора присваивания. Например, a:= L_{ik}[уровень отказа] означает присвоение a значения, равного содержимому второй компоненты элемента L_{ik} списка загружаемых заданий.

Формирование в $ЭМ_i$ на основе T_{ig} сообщения, заносимого в список загружаемых заданий L_i, начинается с выделения свободного элемента L_{ik} списка L_i. Компоненты указанного элемента L_{ik} получают следующие значения:

L_{ik}[имя задания] := T_{ig}[имя задания];
L_{ik}[уровень отказа] := M;
L_{ik}[направление отхода] := 0;
L_{ik}[использованные направления] := $\emptyset$;
L_{ik}[число ненайденных машин]:= T_{ig}[требуемое число машин]-1;
L_{ik}[искомый уровень] := $П_i$;
L_{ik}[число найденных машин] := 1.

$ЭМ_i$ становится корневой ЭМ формируемой подсистемы Q_s, s = = T_{ig}[имя задания] = L_{ik}[имя задания].

Таким образом, децентрализованный алгоритм РАСПРЕДЕЛЕНИЕ ЗАДАНИЙ обеспечивает выравнивание нагрузки всех ЭМ и реагирует на изменение нагрузки ЭМ перераспределением заданий между списками T_i, $i = \overline{1,N}$, всех машин системы. Переход задания s из стадии распределения в стадию загрузки осуществляется как удаление сообщения из списка T_i и формирование соответствующего сообщения в списке L_i, $i = \overline{1,N}$.

I.3. Машина $ЭМ_i$, в которой произошел перевод задания s из списка T_i в список L_i, называется корневой машиной подсистемы Q_s, предназначенной для реализации задания s. Указанная подсистема строится оператором ЗАГРУЗКА ЗАДАНИЙ, который формирует связную подсистему Q_s с требуемым числом n элементарных машин. Пусть $R_j(x)$ - подмножество машин, находящихся на расстоянии j от корневой машины и имеющих нагрузку, не большую чем x. Множество ЭМ, входящих в Q_s, совпадает с множеством $\bar{Q}_s$, построенным по следующему алгоритму.

I. Задание уровня нагрузки x, равного $П_i$, x:= $П_i$, $П_i$ - нагрузка корневой ЭМ; создание подсистемы $\bar{Q}_s$ из одной $ЭМ_i$.

2. Задание значения расстояния j, равного единице, j:= 1.

3. Построение $R_j(x)$; <u>если</u> $R_j(x) = \emptyset$, <u>то</u> переход на 4, <u>иначе</u> переход на 5.

4. Выбор среди машин, находящихся на расстоянии, меньшем $(j+1)$, от корневой и не вошедших в $\bar{Q}_s$, машины $ЭМ_f$ с минимальной нагрузкой $П_f$; присвоение уровню нагрузки x значения $П_f$, $x := П_f$; переход на 2.

5. Построение подмножества $R = R_j(x) \setminus (R_j(x) \cap \bar{Q}_s)$, включающего не входящие в $\bar{Q}_s$ машины $R_j(x)$; если $|R| + |\bar{Q}_s| < n$, то переход на 6; иначе переход на 7.

6. Включение R в подсистему $\bar{Q}_s$, $\bar{Q}_s := R \cup \bar{Q}_s$, $j := j+1$; переход на 3.

7. Если $|R| + |\bar{Q}_s| > n$, то переход на 8; иначе переход на 9.

8. Исключение из R подмножества, состоящего из $|\bar{Q}_s| + |R| - n$ элементарных машин; включение R в $\bar{Q}_s$, $\bar{Q}_s := \bar{Q}_s \cup R$; переход на 9.

9. Подмножество $\bar{Q}_s$ построено; конец.

Децентрализованный алгоритм ЗАГРУЗКА ЗАДАНИЙ выполняется в каждой $ЭМ_i$, $i = \overline{1,N}$, как совокупность процессов $ALLOCATION_{ik}$, $k \in \{1,\dots,|L_i|\}$, $|L_i|$ – число элементов в списке L_i. Процесс $ALLOCATION_{ik}$ порождается [I-4] в $ЭМ_i$ при помещении элемента L_{ik} в список L_i. Процесс $ALLOCATION_{ik}$ осуществляет в $ЭМ_i$ действия по формированию подсистемы Q_s, $s = L_{ik}$[имя задания].

Если процесс $ALLOCATION_{ik}$ обнаруживает, что L_{ik}[число ненайденных машин] $\neq 0$, то указанный элемент L_{ik} поступает в обработку. В результате обработки L_{ik}[число ненайденных машин] принимает нулевое значение, а в одной или нескольких соседних с $ЭМ_i$ машинах в списках L_j, $j \in J_i$, появляются элементы L_{jp} с L_{jp}[число ненайденных машин], не равным нулю.

В зависимости от того, равно число использованных направлений степени вершины v_i графа межмашинных связей ($|L_{ik}$[использованные направления]$| = v_i$) или нет, обработка ведется либо процедурой РАСПРОСТРАНЕНИЕ, либо процедурой ОТХОД.

Процедура РАСПРОСТРАНЕНИЕ выполняется при L_{ik}[число ненайденных машин] $\neq 0$ и $|L_{ik}$[использованные направления]$| \neq v_i$. Суть указанной процедуры – равномерное распределение ненайденных машин между неиспользованными направлениями (с которых не получен отказ) к соседним ЭМ.

Пусть $a = L_{ik}$[число ненайденных машин] и $b = v_i - |L_{ik}$[использованные направления]$|$, тогда на долю p_t-го неиспользованного направления достанется D_t ненайденных машин, $t = \overline{1,b}$. Значения D_t вычисляются по следующему алгоритму.

I. c:= b; t:= 1.

2. Если c = 0, то переход на 5; иначе переход на 3.

3. Присвоение D_t минимального целого, не меньшего (a : c), D_t:=]a : c[.

4. a:= a - D_t; c:= c - 1; t:= t + 1; переход на 2.

5. Конец.

По вычислении значений D_t, $t=\overline{1,b}$, в $ЭМ_i$ начинается формирование сообщений H_1{ L_{ik}[имя задания], L_{ik}[искомый уровень], D_t} для передачи их через выходные полюса p_t соседним ЭМ.

По выдаче на полюса соответствующих сообщений L_{ik}[число ненайденных машин] присваивается нулевое значение (L_{ik}[число ненайденных машин] := 0), и процесс $ALLOCATION_{ik}$ переводится в ждущее состояние [I-4]. Из этого состояния он будет выведен приходом одного из сообщений: H_2(L_{ik}[имя задания]), H_3(L_{ik}[имя задания]), H_4(L_{ik}[имя задания], уровень отказа, число ненайденных машин) .

Выше была представлена работа процесса $ALLOCATION_{ik}$ при выдаче сообщений в соседние ЭМ. Рассмотрим действия $ALLOCATION_{ik}$ при приеме сообщений, в которых значение идентификатора имя задания равно L_{ik}[имя задания].

I.3.I. По получении сообщения H_1(L_{ik}[имя задания], L_{ik} [искомый уровень] , D_p), сформированного процедурой РАСПРОСТРАНЕНИЕ в $ЭМ_i$ и переданного в $ЭМ_j$ по линии межмашинного обмена (i_p, j_q), в $ЭМ_j$ производится проверке того, входит $ЭМ_j$ в строящуюся подсистему Q_s или нет. При вхождении $ЭМ_j$ в Q_s в списке L_j существует элемент g такой, что L_{jg}[имя задания] = L_{ik}[имя задания]. Элемент L_{jg} претерпевает следующие трансформации: L_{jg} [искомый уровень] := L_{ik}[искомый уровень] , L_{jg}[число ненайденных машин] := L_{jg}[число ненайденных машин] + D_t, L_{jg} [использованные направления] := L_{jg} [использованные направления] $\cup$ $\cup$ q. Далее в $ЭМ_j$ формируется сообщение H_2(L_{ik}[имя задания]), которое передается на выходной полюс q линии (j_q, i_p) . Указанное сообщение выводит $ALLOCATION_{ik}$ из состояния ожидания ответа на переданное сообщение H_1(L_{ik}[имя задания], L_{ik} [искомый уровень] , D_p).

I.3.2. Если $ЭМ_j$ не входит в Q_s и ($П_j$ - L_{ik} [искомый уровень]) $\le$ 0, в $ЭМ_j$ определяется свободный элемент списка L_j. Пусть это будет L_{jf}. Составляющие элемента L_{jf} получают следующие значения: L_{jf} [имя задания] := L_{ik} [имя задания] ; L_{jf}[уро-

вень отказа] := M ; L_{jf} [направление отхода] := q ; L_{jf} [использованные направления] := q ; L_{jf}[число ненайденных машин] := D_p - 1; L_{jf} [искомый уровень] := L_{ik} [искомый уровень] ; L_{jf} [число найденных машин] := 0.

По окончании формирования L_{jf} в $ЭМ_j$ создается сообщение H_3\{ L_{ik} [имя задания]\} , передаваемое на выходной полюс q линии $\{ j_q, i_p\}$.

Принедшее в $ЭМ_i$ сообщение H_3\{имя задания\} выводит процесс $ALLOCATION_{ik}$ из состояния ожидания ответа на сообщение H_1 \{ L_{ik} [имя задания] , L_{ik} [искомый уровень] , D_p \}, если значение идентификатора имя задания в сообщении H_3 равно L_{ik} [имя значения] и $ALLOCATION_{ik}$ находился в ожидании ответа. Независимо от того, находился $ALLOCATION_{ik}$ в ожидании или нет, в списке L_i находится элемент L_{ie} такой, что значение L_{ie}[имя задания] равно значению идентификатора имя задания в сообщении H_3. Если $ЭМ_i$ не корневая в строящейся подсистеме, то она пересылает сообщение H_3 [имя задания] в направлении L_{ie} [направление отхода].

I.3.3. Если $ЭМ_i$ является корневой в подсистеме, то сообщение H_3\{ имя задания\} вызывает присчет единицы к счетчику найденных машин L_{ie} [число найденных машин] . Если значение указанного счетчика равно n , то построение подсистемы Q_m закончено.

I.3.4. Если ($П_j$ - L_{ik}[искомый уровень])> 0 и $ЭМ_j$ не входит в Q_m , в ЭМ формируется сообщение H_4\{L_{ik} [имя задания],$П_j$, D_p\}, которое передается на выходной полюс q линии $\{ j_q, i_p\}$.

$ЭМ_i$, получив сообщение H_4 \{имя задания,уровень отказа,число ненайденных машин\} , производит следующие трансформации элемента L_{ik} , L_{ik} [имя задания] равно значению идентификатора имя задания сообщения H_4 , списка L_i загружаемых заданий: L_{ik} [уровень отказа] := min \{L_{ik} [уровень отказа] , $П_j$\}; L_{ik} [использованные направления] := L_{ik} [использованные направления]Up; L_{ik} [число ненайденных машин] := L_{ik} [число ненайденных машин] + + D_p . По выполнении вышеперечисленных действий $ALLOCATION_{ik}$ выходит из состояния ожидания ответа на посланное сообщение H_1\{L_{ik} [имя задания] , L_{ik} [искомый уровень] , D_p\} , если он был в ожидании такового.

I.3.5. При | L_{ik} [использованные направления]| = v_i процедура ОТХОД анализирует значение L_{ik} [направление отхода] .Если L_{ik} [направление отхода] равно нулю, $ЭМ_i$ является корневой в строящейся подсистеме Q_m . Поэтому L_{ik} [искомый уровень] := L_{ik}

[уровень отказа] , L_{ik} [уровень отказа] := M , L_{ik} [использованные направления] := $\emptyset$, что обеспечивает продолжение построения Q_n (см.шаг 4 алгоритма п.I.3) путем перехода к процедуре РАСПРОСТРАНЕНИЕ. Если L_{ik} [направление отхода] $\neq 0$ ($ЭМ_i$ не является корневой), то в $ЭМ_i$ формируется сообщение $H_4\{ L_{ik}$ [имя задания], L_{ik} [уровень отказа] , L_{ik} [число ненайденных машин] $\}$, которое поступает на выходной полюс $p = L_{ik}$[направление отхода] .Линия межмашинного обмена (i_p, j_q) обеспечит передачу этого сообщения в соседнюю $ЭМ_j$ (см.[I-4]). $ЭМ_j$ интерпретирует принятое сообщение как четырехкомпонентный вектор, первая компонента которого задает вид сообщения (а именно H_4), следующие компоненты, соответственно - имя задания, уровень отказа, число ненайденных машин.

После передачи указанного сообщения в $ЭМ_i$ производятся действия по подготовке к дальнейшей работе: L_{ik} [уровень отказа] := M , L_{ik} [направление отхода] := 0, L_{ik} [использованные направления] := $\emptyset$, L_{ik} [число ненайденных машин] := 0, L_{ik} [искомый уровень] := 0.

Таким образом,алгоритм ЗАГРУЗКА ЗАДАНИЙ обеспечивает построение связной подсистемы Q_n с заданным числом n элементарных машин. Сигнал окончания построения подсистемы Q_n формируется в корневой ЭМ указанной подсистемы. После назначения адресов машин и рассылки по машинам подсистемы предназначенной каждой из ЭМ программы освобождаются элементы списка загружаемых заданий, соответствующие заданию s . Последнее переходит в список исполняемых заданий.

2. Адресация машин подсистемы

2.I. Для определения посредством путевой процедуры [I-3,6,7] трасс передачи данных между машинами подсистемы Q_n каждой ЭМ , входящей в указанную подсистему, приписывается адрес.В основу алгоритма задания адресов положена D(z,m)-адресация [3,6,7], обеспечивающая эффективную реализацию путевых процедур. Суть предлагаемого алгоритма состоит в следующем. Каждой $ЭМ_i$, $i \in \{ 0,1,\ldots, |Q_n|-1\}$, подсистемы Q_n приписывается адрес A_i . Адрес $ЭМ_i$ задается последовательностью ярусных номеров $A_i = A_i^{m-1},\ldots, A_i^1, A_i^0$, где $A_i^j \in \{0,1,\ldots, v\}$, $j = \overline{0,m-1}$, v - степень вершин графа межмашинных связей, m - расстояние от корневой машины $ЭМ_0$ подсистемы Q_n

до наиболее удаленной от $ЭМ_0$ машины подсистемы*). Адрес корневой машины $ЭМ_0$ подсистемы Q_s полагаем равным $A_0 = 0,\ldots,0$. Адрес $ЭМ_i$ представляет собой последовательность номеров выходных полюсов ЭМ, лежащих на кратчайшем пути от корневой $ЭМ_0$ до $ЭМ_i$, $i \in \{1,\ldots, |Q_s| - 1\}$. Если между $ЭМ_0$ и $ЭМ_i$ существует несколько путей одинаковой длины, то в качестве адреса A_i берется последовательность, соответствующая одному из указанных путей. При этом если расстояние $ЭМ_i$ от корневой $ЭМ_0$ равно $l (l < m)$, то в адресе $ЭМ_i$ ярусные номера $A_i^j = 0$ при $j \in \{m-1, \ldots, 1, 0\}$. Машины $ЭМ_i$ и $ЭМ_k$ входят в адресную подсистему $R_r(A_i^{m-1}, \ldots, A_i^r)$ уровня r, если $A_i^j = A_k^j$ для всех $j \geq r$, $r \in \{0, 1, \ldots, m-1\}$. Очевидно, что машины адресной подсистемы $R_r(A_i^{m-1}, \ldots, A_i^r)$, $r \in \{0, 1, \ldots, m-1\}$ образуют связный подграф структуры системы, так как для $ЭМ_i$ с адресом $A_i = A_i^{m-1}, \ldots, A_i^r, 0, \ldots, 0$ существует путь до любой $ЭМ_k$ с адресом $A_k = A_i^{m-1}, \ldots, A_i^r, A_k^{r-1}, \ldots, A_k^0$, определяемый $A_k^{r-1}, \ldots, A_k^0$. При этом ярусный номер A_k^{r-1} равен номеру выходного полюса $ЭМ_i$, лежащего на кратчайшем пути в $ЭМ_k$. Назовем $ЭМ_i$ с адресом $A_i = A_i^{m-1}, \ldots, A_i^r, 0, \ldots, 0$ корневой ЭМ уровня $r \in \{m-1, \ldots, 1, 0\}$ адресной подсистемы $R_r(A_i^{m-1}, \ldots, A_i^r)$. Таким образом, адрес любой ЭМ задает кратчайший путь (последовательность номеров выходных полюсов), ведущий к ней от корневых ЭМ тех адресных подсистем, к которым данная ЭМ принадлежит.

Пусть требуется определить выходной полюс, принадлежащий кратчайшему пути из $ЭМ_i$ в $ЭМ_k$. Пусть также $ЭМ_i$ находится на расстоянии l от корневой $ЭМ_0$. Определяем $r = \max\{j \mid A_i^j \neq A_k^j,\ j = \overline{0, m-1}\}$. Если $r < m-1$, то номер требуемого выходного полюса равен A_k^r. Если $r \geq m-1$, номер выходного полюса равен значению ярусной функции $G_i^r(A_i^r \oplus A_k^r)$ $(m-1 \leq r \leq m-1)$, $+$ - операция сложения по модулю z, $z =]\log_2(v+1)[$, $]x[$ - наименьшее целое, не меньшее x. Значение $G_i^r(A_i^r \oplus A_k^r)$ $(m-1 \leq r \leq m-1,\ r = \max\{j \mid A_i^j \neq A_k^j,\ j = m-1, \ldots, m-1\})$ равно номеру выходного полюса, лежащего на кратчайшем пути от $ЭМ_i$ до адресного подмножества $R_r(A_i^{m-1}, \ldots, A_i^{r+1}, A_i^r)$. При использовании табличного метода задания ярусных

*) Длины адресов могут быть уменьшены за счет перемещения корневой ЭМ подсистемы Q_s в центр подсистемы. Центром подсистемы называется ЭМ с минимально возможным расстоянием до наиболее удаленной от нее машины подсистемы.

функций в $ЭМ_i$, находящейся на расстоянии $0 \le l \le m-1$ от корневой $ЭМ_0$, необходимо $l \cdot v \cdot]\log_2(v+1)[$ битов, а для задания всей совокупности таблиц ярусных функций в подсистеме необходимо $V = \sum_{i=1}^{n} n_i \cdot v \cdot i \cdot] \log_2(v+1)[$, где n_i – число ЭМ, находящихся на расстоянии i от корневой $ЭМ_0$, в подсистеме. При этом длина адреса ЭМ в подсистеме составит $m \cdot]\log_2(v+1)[$ битов.

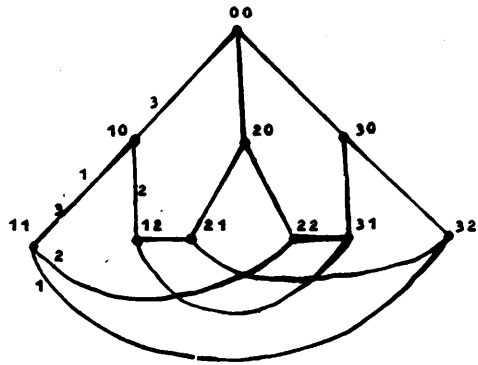

Рис. I

ПРИМЕР I. На рис. I показано задание адресов на графе Петерсена ($n = 2$, $v = 3$), а в табл.I и 2 приведены ярусные функции соответственно для вершины с адресами IO и II (на ребрах, инцидентных данным вершинам, показана нумерация направлений).

Отметим возможность двоичной интерпретации данной адресации, при которой A_i^j представляется двоичным числом с $]\log_2(v+1)[$ разрядами. Полный двоичный адрес формируется как последовательность из m групп битов, каждая из которых соответствует A_i^j, $i = \overline{0, |Q_n|-1}$, $j = \overline{0, n-1}$. Каждый разряд полного двоичного адреса рассматривается как ярусный номер. Такое представление уменьшает длину таблиц ярусных функций и упрощает реализацию алгоритма вычисления этих функций за счет некоторого увеличения длины путей между ЭМ.

Таблица I

$G^1(1)$	3
$G^1(2)$	3
$G^1(3)$	3

Таблица 2

$G^1(1)$	3
$G^1(2)$	I
$G^1(3)$	2
$G^0(1)$	3
$G^0(2)$	-
$G^0(3)$	3

ПРИМЕР 2. При двоичной интерпретации адресов их длина будет составлять $m' = m \cdot]\log_2(v+1)[$ битов, а размер таблиц ярусных функций $V' = \sum_{i=1}^{n} n_i \cdot i \cdot 2 \cdot]\log_2(v+1)[$.

Таблица 3

$G^3(1)$	3
$G^2(1)$	3

Таблица 4

$G^3(1)$	2
$G^2(1)$	3
$G^1(1)$	3
$G^0(1)$	3

Таблица 5

	l	V	d
I	4	90	I,93
2	4	60	2,I4
3	4	200	I,67

В табл.3 и 4 приведены ярусные функции при двоичной интерпретации соответственно для вершин с адресами IO и II (в двоичной интерпретации OIOO и OIOI).

Назовем средним диаметром графа с заданной адресацией математическое ожидание длины пути между двумя равновероятно выбранными вершинами на графе, подсчитанное при соблюдении алгоритмов передачи, задаваемых адресацией. В табл.5 приведены значения параметров: l - длина адреса (бит); V - объем памяти, необходимой для хранения всей совокупности таблиц (бит); d - средний диаметр для графа Петерсена в случае задания адресации: I) для примера I; 2) для примера 2; 3) при табличном задании кратчайших путей между машинами, требующем для размещения таблиц $N \cdot \lceil \log_2(v+1) \rceil$ битов памяти одной ЭМ, где N - число машин в подсистеме.

2.2. Алгоритм задания D(z,m)-адресации состоит из 2-х этапов. На первом этапе происходит назначение адресов для ЭМ, входящих в подсистему; на втором - выравнивание адресов и вычисление ярусных функций. Пусть F_j - множество машин, находящихся на расстоянии j от корневой ЭМ в подсистеме Q_m, а A - множество машин в подсистеме Q_m, имеющих адреса. Суть алгоритма НАЗНАЧЕНИЕ АДРЕСОВ состоит в следующем.

I. Присвоить адрес корневой ЭМ, $A_0 := 0$, $F_0 := A :=$ {корневая ЭМ}.

2. $j := 1$.

3. Определить F_j : если $F_j = \emptyset$, то переход на 6.

4. Для каждой $ЭМ_i \in F_j$ найти $ЭМ_k \in F_{j-1}$, соседнюю с $ЭМ_i$; присвоить адрес $ЭМ_i$, $A_i := A_k * p$, где p - номер направления связи от $ЭМ_k$ к $ЭМ_i$, * - знак операции конкатенации; $A := A \cup \{ЭМ_i\}$.

5. $j := j+1$; переход на 3.

6. Всем ЭМ подсистемы Q_m назначены адреса; конец.

Таким образом, каждая ЭМ $\in Q_s$ имеет адрес длиной $j \cdot]\log_2(v+1)[$, где j – расстояние данной ЭМ от корневой.

На втором этапе задания адресации длины всех адресов выравниваются (путем сдвига влево до величины $m \cdot]\log_2(v+1)[$, где m – расстояние от корневой ЭМ до наиболее удаленной ЭМ подсистемы), происходит заготовка таблиц ярусных функций и таблиц расстояний до подсистем. Под каждую из указанных таблиц в $ЭМ_i \in Q_s$ отводится $l \cdot v \cdot]\log_2(v+1)[$ битов, где l – расстояние $ЭМ_i$ от корневой ЭМ, и $l \cdot v \cdot k$ битов соответственно, $2^k > D$, D – диаметр графа межмашинных связей. Строение таблицы расстояний до подсистем аналогично строению таблицы, задающей значения ярусных функций. Разница состоит в том, что вместо номеров полюсов, лежащих на кратчайшем пути до адресных подсистем, в таблице расстояний содержатся величины расстояния до них. Первоначально элементы таблицы расстояний получают значения $2^k - 1$.

Идея алгоритма ВЫЧИСЛЕНИЕ ЯРУСНЫХ ФУНКЦИЙ сводится к следующему. Каждая $ЭМ_i \in Q_s$ посылает по всей подсистеме сообщение, содержащее $\{A_i, C\}$, где A_i – адрес $ЭМ_i$, а C – счетчик транзитных ЭМ, пройденных сообщением (исходное значение C равно единице). Получив такое сообщение, $ЭМ_k \in Q_s$ определяет по счетчику C величину расстояния до адресной подсистемы, к которой принадлежит $ЭМ_i$, и если оно меньше, чем имеющееся у нее в таблице расстояний до подсистем Y_k, то $ЭМ_k$ заносит в таблицу ярусной функции номер полюса, по которому пришло данное сообщение, а в таблицу Y_k – значение счетчика C, т.е. величину расстояния до адресной подсистемы.

Действия по изменению таблиц Y_k и G_k выполняются с помощью процедуры КОРРЕКЦИЯ, входными параметрами которой являются: A_k – адрес $ЭМ_k$, G_k – таблица ярусной функции в $ЭМ_k$, Y_k – таблица расстояний до адресных подсистем от $ЭМ_k$, m_k – расстояние от $ЭМ_k$ до корневой ЭМ, полученное сообщение $\{A_i, C\}$ и q – номер полюса, по которому получено данное сообщение. Суть процедуры КОРРЕКЦИЯ состоит в следующем.

1. Определяется $r = \max\{j \mid A_k^j \neq A_i^j,\ j = \overline{0, m-1}\}$, где m – длина адреса.

2. Если $r > m_k$, то переход на 4.

3. Если $Y_k^r(A_k^r \oplus A_i^r) > C$, то $Y_k^r(A_k^r \oplus A_i^r) := C$, $G_k^r(A_k^r \oplus A_i^r) := q$.

4. Конец.

2.3. Реализация децентрализованного алгоритма НАЗНАЧЕНИЕ АДРЕСОВ совмещается с реализацией алгоритма ЗАГРУЗКА ЗАДАНИЙ. При этом каждая ЭМ вначале включается в подсистему, а затем ей присваивается адрес. Для выполнения указанного совмещения элементы списка загружаемых заданий L_i, $i = \overline{0, |Q_s| - 1}$ (см.п.I.3), дополняются следующими компонентами: {адрес, расстояние}, где L_{ik} [адрес] – адрес $ЭМ_i$ в подсистеме Q_s, L_{ik} [имя задания] = s и L_{ik}[расстояние] – расстояние $ЭМ_i$ от корневой ЭМ, $k = \overline{1, |L_i|}$. У корневой ЭМ обе компоненты равны нулю.

Сообщение H_1 (см.п.I.4) при формировании в процедуре РАСПРОСТРАНЕНИЕ дополняется следующими элементами {адрес, счетчик}, где H_1[адрес] := L_{ik} [адрес]. p, p – номер выходного полюса, по которому выдается сообщение H_1, . – операция конкатенации, H_1 [счетчик] := L_{ik}[расстояние] + 1.

Рассмотрим, какие действия добавляются в процессе ALLOCATION при приеме сообщения H_1, переданного в $ЭМ_j$ по линии $\{i_p, j_q\}$. Если $ЭМ_j$ входит в строящуюся подсистему Q_s (см.п.I.3.1) и L_{jg} [расстояние] > H_1 [счетчик], то L_{jg} [адрес] := H_1[адрес] и L_{jg} [расстояние] := H_1 [счетчик]. Если $ЭМ_j$ не входит в подсистему Q_s и включается в нее только при получении данного сообщения H_1 (см.п.I.3.2), то L_{jf} [адрес] := H_1 [адрес], L_{jf}[расстояние] := := H_1[счетчик]. В этом случае в сообщении H_3 появляется новый элемент {расстояние}, H_3[расстояние] := L_{jf}[расстояние], значение которого равно расстоянию $ЭМ_j$ от корневой ЭМ.

При приеме сообщений H_3, корневая ЭМ (см.п.I.4.3) определяет расстояние до наиболее удаленной ЭМ подсистемы: L_{ik} [расстояние] := max (L_{ik}[расстояние], H_3[расстояние]).

Таким образом, в результате совместной работы алгоритмов ЗАГРУЗКА ЗАДАНИЙ и НАЗНАЧЕНИЯ АДРЕСОВ построена подсистема Q_s, в которой каждой $ЭМ_j$ заданы L_{jk}[расстояние] – расстояние от корневой ЭМ и L_{jk}[адрес] – адрес $ЭМ_j$ в данной подсистеме (длиной L_{jk}[расстояние]·]$\log_2(v + 1)$[ ), а корневой $ЭМ_i$ известно расстояние до наиболее удаленной ЭМ подсистемы – L_{ik}[расстояние].

2.4. Второй этап задания адресации – алгоритм ВЫРАВНИВАНИЕ АДРЕСОВ И ВЫЧИСЛЕНИЕ ЯРУСНЫХ ФУНКЦИЙ – начинается с того, что корневая $ЭМ_i$ формирует сообщение H_5 {имя задания, адрес, расстояние, счетчик}, где H_5[имя задания] := L_{ik}[имя задания], H_5[адрес] := L_{ik}[адрес], H_5[расстояние] := L_{ik}[расстояние], H_5 [счетчик] := 1, и рассылает по всем выходным полюсам.

Элементарная машина $ЭМ_j$, получив сообщение H_5 по линии $\{i_p, j_q\}$, определяет элемент g в списке L_j с L_{jg}[имя задания] = H_5 [имя задания]. Если такого элемента нет, то сообщение H_5 игнорируется. В противном случае, если сообщение получено в первый раз, длина адреса в компоненте L_{jg} [адрес] устанавливается (путем сдвига влево) равной H_5 [расстояние]·]$\log_2(v+1)$[битов и резервируется место в памяти $ЭМ_j$ под таблицу ярусных функций G_j и таблицу расстояний до подмножеств Y_j. Содержимое таблицы G_j обнуляется, а элементы Y_j получают значения, равные H_5[расстояние] + 1.

Если $ЭМ_j$ получает сообщение H_5 не в первый раз, то формирования таблиц и выравнивания адресов не происходит.

Далее, по получении сообщения H_5 выполняется процедура КОРРЕКЦИЯ, и H_5 [счетчик] := H_6 [счетчик] + 1. После этого на все выходные полюса, за исключением полюса q, посылается преобразованное сообщение H_5. Кроме этого, всем выходным полюсам без исключения посылается сообщение H_6 {имя задания, адрес, счетчик}, сформированное в $ЭМ_j$, где H_6 [имя задания] := L_{jg} [имя задания], H_6 [адрес] := L_{jg}[адрес], H_6[счетчик]:= 1.

Рассмотрим действия, производимые в $ЭМ_j$ ($j = \overline{1,n}$) по получении сообщения H_6, по линии межмашинного обмена $\{i_p, j_q\}$. Определяется элемент g в списке L_j с L_{jg}[имя задания] = H_6[имя задания]. Если такого элемента не оказалось, то принятое сообщение игнорируется, иначе по принятому сообщению выполняется процедура КОРРЕКЦИЯ. Если в результате этой процедуры изменилось содержимое таблиц Y_j и G_j, то H_6[счетчик] := H_6 [счетчик] + 1 и измененное сообщение рассылается по всем выходным полюсам, кроме полюса q, в противном случае сообщение H_6 далее не рассылается.

После того как таблицы Y и G во всех $ЭМ \in Q_s$ заполнены, адресация на подсистеме Q_s считается заданной.

Заключение

Предложенное в работе децентрализованное распределение заданий предполагает прохождение каждым заданием, в ходе его загрузки, трех этапов. Первый этап начинается поступлением задания в систему и заканчивается определением корневой ЭМ подсистемы, на которой задание будет исполняться. Второй этап заключается в построении адресной подсистемы, содержащей требуемое заданием число ЭМ и включающей корневую ЭМ. Каждая ЭМ построенной на этом этапе

адресной подсистемы получает адрес, выделяющий указанную ЭМ из множества машин подсистемы. Этот адрес используется для определения трасс передачи данных между машинами подсистемы. Третий этап состоит в согласовании виртуальных адресов машин с адресами ЭМ, входящих в подсистему, и в собственно загрузке заданий в машины адресной подсистемы. По окончании третьего этапа задание поступает на выполнение.

В работе представлены алгоритмы, реализующие первые два этапа, а также вариант построения D(z,m)-адресации [6,7], позволяющий сократить длину требуемых таблиц.

Литература

1. КОРНЕЕВ В.В., ХОРОШЕВСКИЙ В.Г. Вычислительные системы с программируемой структурой. -Электронное моделирование,1979, № I, с.42-52.

2. КОРНЕЕВ В.В., ХОРОШЕВСКИЙ В.Г. Архитектура вычислительных систем с программируемой структурой. Новосибирск, Б.и.,1979.-48 с. (Препринт/ ИМ СО АН СССР; ОВС-10).

3. КОРНЕЕВ В.В., ХОРОШЕВСКИЙ В.Г. Структура и функциональная организация вычислительных систем с программируемой структурой.Новосибирск, Б.и., 1979.- 48 с. (Препринт/ИМ СО АН СССР; ОВС-11).

4. КОРНЕЕВ В.В. Элементарная машина однородной вычислительной системы с программируемой структурой.- Кибернетика,1980, № I, с.75-81.

5. КОРНЕЕВ В.В. О подключении внешних устройств к однородной вычислительной системе. - В кн.: Вычислительные системы. Вып. 63. Теория однородных вычислительных систем. Новосибирск,1975,с. 103-108.

6. Однородные вычислительные системы на базе микропроцессорных больших интегральных схем./ Бандман О.Л., Евреинов Э.В.,Корнеев В.В.,Хорошевский В.Г. - В кн.: Вопросы теории и построения вычислительных систем (Вычислительные системы, вып.70.) Новосибирск, 1977, с.3-28.

7. КОРНЕЕВ В.В., МОНАХОВ О.Г. Организация межмашинных взаимодействий в вычислительных системах с программируемой структурой. - Электронное моделирование, 1980, № 5, с.16-22.

8. ЕВРЕИНОВ Э.В., ХОРОШЕВСКИЙ В.Г. Однородные вычислительные системы.- Новосибирск: Наука, 1978. - 318 с.

9. Фон НЕЙМАН Дж. Теория самовоспроизводящихся автоматов.-М.: Мир,1971.- 380 с.

Поступила в ред.-изд.отд.
2 июня 1980 года