ВОПРОСЫ ТЕОРИИ И ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ
(Вычислительные системы)
1978 год Выпуск 73

УДК 518:681.3

ОБ АЛГОРИТМАХ РАСПАРАЛЛЕЛИВАНИЯ СИСТЕМЫ ЦИКЛОВ НАД МАССИВАМИ

Р.М. Нуриев

В работе рассматриваются вопросы выявления параллелизма и организации параллельного выполнения системы циклов в схемах программ над массивами, подчиняющихся методологии структурированного программирования. А именно исследуется класс R-схем, структурированных в смысле Дийкстры, в телах циклов которых синтаксически разделены части, осуществляющие следующие функции: каждая часть (организатор) определяет текущие элементы массивов, обрабатываемые последующей частью. Может быть доказано, что для всякой схемы существует эффективная процедура построения эквивалентной схемы из R .

Формализуется понятие информационной связи. Показывается, что информационная связь между повторениями элементов системы циклов определяется решением системы уравнений, связывающих организаторы и индексные выражения одноименных переменных с индексами. Дается общий алгоритм распараллеливания схем класса R в предположении существования алгоритма τ , указывающего те итерации, на которых могут использоваться значения переменных, созданные на исходной итерации. Определяются условия, при которых такой алгоритм существует. Рассматривается класс схем программ, для которых алгоритм τ определяется совокупностью линейных функций. В рамках этого класса схем строится пример, показывающий, что существование множеств информационно не связанных повторений итераций не всегда влечет существование такой организации вычисления, при которой параллельно реализуются элементы этих множеств. Дается простой способ определения множеств информационно независимых повторений и организации их параллельного выполнения.

I. Основные определения

Схемами программ (в дальнейшем просто схемами), с каждой из которых связано имя (буква П), возможно, с нижними индексами, называются конечные множества, состоящие из инструкций алголоподобного вида (здесь и далее определения следуют работам [I-2]):

I) l_1: $x := f(x_1,\dots,x_n)$ then l_2 – присваивание;

2) l_1: if $p(x_1,\dots,x_n)$ then l_2 else l_3 – условный переход по предикату $p(x_1,\dots,x_n)$;

3) l_1: do $П_1$ while $p(x_1,\dots,x_n)$ then l_2 – цикл с телом $П_1$ и условием повторного выполнения $p(x_1,\dots,x_n)$;

4) l_1: do $П_1$ then l_2 – обращение к подсхеме $П_1$, среди которых выделена стартовая.

Здесь l_1, l_2, l_3 – метки; f и p – символы функций и предикатов соответственно; $П_1$ – непосредственная подсхема схемы, которой принадлежат инструкции вида 3 или 4; переменные $x, x_1,\dots,x_n$ могут быть как простыми, так и переменными с индексами вида $a[\varkappa_1(i_1^1,\dots,i_{n_1}^1),\dots,\varkappa_k(i_1^k,\dots,i_{n_k}^k)]$ ж), где индексы $i_1^l,\dots,i_{n_l}^l$ являются простыми переменными из некоторого выделенного множества I; конечные метки (или стоп-метки) для данной схемы есть те из выходных меток (l_2, l_3), которые не являются входными метками никакой инструкции схемы.

Пусть $P(П)$ – множество непосредственных подсхем схемы $П$, $M(П)$ – минимальное множество, определяемое условиями: $P(П) \subset M(П)$, $П_i \in M(П) \to P(П_i) \subset M(П)$.

Рассматриваются только детерминированные схемы, в $\bigcup_{П_i \in M(П) \cup \{П\}} П_i$ которых нет двух инструкций с одинаковыми входными метками.

Схема П называется циклической (с – схемой), если на множестве меток $L(П_i)$ всякой схемы $П_i \in M(П) \cup \{П\}$ существует частичный порядок, при котором входная метка каждой инструкции меньше выходных меток этой инструкции и $\forall П_i [П_i \in M(П) \cup \{П\} \to П_i \notin M(П_i)]$.

Через $\Omega_П$ и $\Delta_П$ соответственно обозначаются множество ячеек, значения которых используются, и множество ячеек, значения которых создаются в процессе вычисления П при некоторой интерпретации I.

ж) Функции $\varkappa_l$ – аналоги индексных выражений, содержащих переменные $i_1^l,\dots,i_{n_l}^l$ $(l = 1,\dots,k)$.

Цикл, тело которого есть конечная последовательность с -схем $\Pi_1,\dots,\Pi_n$ таких, что $\forall i \forall k[(\Delta_{\Pi_i} \cap \Omega_{\Pi_{i+1}} \cap I=\emptyset) \& (\Delta_{\Pi_{i+1}} \cap \Omega_{\Pi_i} \cap I=\emptyset)]$, называется разделенным, $\Pi_1,\dots,\Pi_{n-1}$ - организаторами уровней $1,\dots,n-1$, Π_n - ядром цикла.

Схемы вида i: do Π_1 then i_2, i_2: do Π_2 then $i_3,\dots,i_k$: do Π_k then s и i_1: do Π_1 then i_2, i_2: if $p_1(x_1^1,\dots,x_{n_1}^1)$ then i_3 else s, i_3: do Π_2 then $i_4,\dots,i_{2d}$: if $p_{d-1}(x_1^d,\dots,x_{n_d}^d)$ then i_{2d+1} else s, i_{2d+1}: do Π_d then s называются разложенными и обозначаются через $\Pi_1\Pi_2\dots\Pi_k$ и $\Pi_1 p_1 \dots p_{d-1}\Pi_d$ соответственно. Схемы вида $\Pi_1\Pi_1\dots\Pi_k$ или $\Pi_1 p_1 \dots p_{k-1}\Pi_k$ обозначаются $\Pi_1\dots\Pi_k$ *).

Пусть $\Pi = \Pi_1\Pi_2\dots\Pi_k$. Элементы Π_i и Π_j схемы Π называются информационно связанными (обозначение $\Pi_i \rightleftarrows \Pi_j$), если существуют интерпретация I и последовательность $\Pi_{i_1},\dots,\Pi_{i_l}$ такая, что $\Pi_{i_1} = \Pi_i$, $\Pi_{i_l} = \Pi_j$, $i_1 < i_2 < \dots < i_l$, и между каждыми соседними элементами Π_{i_m} и $\Pi_{i_{m+1}}$ при выполнении Π на интерпретации I имеет место хотя бы одно из соотношений:

$$а)\quad \Delta_{\Pi_{i_m}} \cap \Omega_{\Pi_{i_{m+1}}} \neq \emptyset,$$

$$б)\quad \Delta_{\Pi_{i_m}} \cap \Delta_{\Pi_{i_{m+1}}} \neq \emptyset,$$

$$в)\quad \Delta_{\Pi_{i_{m+1}}} \cap \Omega_{\Pi_{i_m}} \neq \emptyset.$$

В системах вида $\Pi_1 p_1 \dots p_{k-1}\Pi_k$ элементы Π_i и Π_j называются информационно связанными, если либо $\Pi_i \rightleftarrows \Pi_j$, либо найдется такой $p_l (i \le l \le j)$, что $\Pi_i \rightleftarrows p_l$; кроме того, p_j и p_i, а также Π_j и p_i информационно связаны для $j \ge i$, т.е. они связаны информационной либо логикоинформационной зависимостью.

Далее со ссылкой "система циклов $\mathfrak{C}$" указывается система вложенных циклов $C_1,\dots,C_n$, где C_l для $l = 1,\dots,n$ есть цикл с телом $\Pi_{l,1}C_l\Pi_{l,2}$, а $C_{n+1}=\Pi$. В работе [3] доказано, что тела циклов всякой системы циклов можно представить в таком виде.

Реализацию системы циклов $\mathfrak{C}$ можно рассматривать как минимальное множество последовательностей, которому принадлежат после-

*) s - конечная метка, поэтому может в изображении не участвовать.

довательности вида $T_1^1 p_1^1 T_1^2 p_1^2 \ldots T_1^{k_1} p_1^{k_1}$, а также каждая последовательность, получающаяся из элементов этого множества заменой символов $T_i^{l_1,\ldots,l_i}$ на $П_{i,1}^{l_1,\ldots,l_i} T_{i+1}^{l_1,\ldots,l_i,1} p_{i+1}^{l_1,\ldots,l_i,1} T_{i+1}^{l_1,\ldots,l_i,2} p_{i+1}^{l_1,\ldots,l_i,2} \ldots T_{i+1}^{l_1,\ldots,l_{i+1}} p_{i+1}^{l_1,\ldots,l_i,l_{i+1}} П_{i,2}^{l_1,\ldots,l_i}$.

Таким образом, понятие информационной связи распространяется и на повторения элементов цикла.

Верхние индексы в выражении $П_1^{l_1,\ldots,l_i}$ называются номером повторения элемента $П_1 \in \bigcup_{i,j,k} \{П_{i,1}, П, П_{j,2}, p_k\}$ в системе циклов. Ясно, что по номерам повторений можно определить порядок их реализаций следующим образом. Дополним верхние индексы элементов $П_{i,1}$ до длины n нулями справа, а элементов $П_{j,2}$ – значками $\perp$ справа. Значения $\perp$ будем считать бо́льшими любого натурального числа и несравнимыми между собой. Тогда если номер повторения одного элемента лексикографически больше другого, то первый выполняется позже второго.

Для геометрической интерпретации выполнения системы циклов будем использовать n-мерное пространство, натуральные координаты которого интерпретируем как номера повторений элементов цикла.

2. Информационная связь между итерациями системы циклов

В соответствии с определением информационной связи, если две схемы $П_i$ и $П_j$ из последовательности $П = П_1 \ldots П_k$ содержат одну и ту же простую переменную d, входящую в левую часть оператора одной из этих схем, то $П_i$ и $П_j$ информационно связаны. Вообще между $П_i$ и $П_j$ имеется информационная связь, если в $П_i$ существует некоторая переменная d_1, которая используется или затирается некоторой схемой $П_{i_2}$; в той же ветви схемы $П_{i_2}$ есть некоторая переменная d_2, для которой существует $П_{i_3}$ с аналогичными переменными, и т.д. до $П_j$. Таким образом, имеет место

ЛЕММА. Если в системе циклов $\mathfrak{C}$ в схеме $П$ есть некоторая простая переменная, являющаяся левой частью некоторого оператора присваивания, то повторения схемы $П$ информационно связаны.

Можно видеть, что отношения вида "б" и "в" (см. стр.I40) являются несущественными и связанными лишь с экономией памяти. Докажем, что эти отношения являются устранимыми. Мы будем рассматривать систему циклов, оснащенных счетчиками числа повторений каждого цикла. Причем, отсчет числа повторений начинается с 0 при каждом обращении к циклу. Ясно, что в системе $\mathfrak{C}$ циклов со счетчиками положение этих счетчиков определяет номер повторения в каждый момент реализации системы. Имеют место очевидные замечания.

ЗАМЕЧАНИЕ I. Существует эффективная процедура, оснащающая любую систему циклов $\mathfrak{C}$ счетчиками.

ЗАМЕЧАНИЕ 2. В системе разделенных циклов счетчики входят в организаторы первого уровня.

Схему, для которой при всякой интерпретации в ходе реализации ни в одну ячейку памяти не происходит записи более одного раза, назовем с х е м о й с о д н о к р а т н о й з а п и с ь ю.

ТЕОРЕМА I. С у щ е с т в у е т э ф ф е к т и в н а я п р о ц е д у р а, п р е д с т а в л я ю щ а я в с я к у ю с и с т е м у ц и к л о в с о с ч е т ч и к а м и в в и д е с и с т е м ы с о д н о к р а т н о й з а п и с ь ю.

ДОКАЗАТЕЛЬСТВО. Искомую процедуру опишем неформально. Вместо всякой ячейки d или $a[w_1,\dots,w_n]$ ($w_1,\dots,w_n$ – значения индексных функций) введем массив $d[c]$ и $a[w_1,\dots,w_n,c]$. В эти ячейки массивов и осуществляется запись значений d и $a[w_1,\dots,w_n]$ на повторении c. Считывание значений осуществляется следующим образом. Если на повторении c нужно значение d или $a[w_1,\dots,w_n]$, то оно выбирается из ячеек $d[c_0]$ и $a[w_1,\dots,w_n,c_0]$ соответственно, где c_0 – наибольшее среди x меньших c, в лексикографическом порядке таких, что ячейки $d[x]$ или $a[w_1,\dots,w_n,x]$ непусты.

Таким образом, каждой ячейке (кроме счетчиков) значение присваивается не более одного раза.

Обозначим через $\tau(c,\Pi_1,\Pi_2,\gamma)$ алгоритм (если он существует) нахождения по номеру повторения c схемы Π_1 тех повторений c_1, на которых функцию, дающую по номеру повторения c те повторения, на которых результаты Π_2 могут быть использованы при подходящей интерпретации.

Пусть $a_1 = a[\nu_1(i_1^1,\dots,i_{n_1}^1),\dots,\nu_k(i_1^k,\dots,i_{n_k}^k)]$ и $a_2 = a[\varkappa_1(j_1^1,\dots,j_{m_1}^1),\dots,\varkappa_k(j_1^k,\dots,j_{m_k}^k)]$ – пара одноименных переменных схем Π_1 и Π_2 соответственно, первая из которых является левой частью некоторого оператора схем из $\{\Pi_1\} \cup M(\Pi_1)$.

Отметим, что $\tau(p,\Pi_1,\Pi_2,\gamma) = q$ тогда и только тогда, когда p и q являются решениями следующей системы уравнений (для некоторых одноименных переменных a_1 и a_2) относительно неизвестных $p = p_1, p_2, \dots, p_n$ и $q = q_1, q_2, \dots, q_n$:

$$\nu_1(\tilde{i}_1^1(q), \dots, \tilde{i}_{n_1}^1(q)) = \varkappa_1(\tilde{j}_1^1(p), \dots, \tilde{j}_{m_1}^1(p)),$$

$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$

$$\nu_k(\tilde{i}_1^k(q), \dots, \tilde{i}_{n_k}^k(q)) = \varkappa_k(\tilde{j}_1^k(p), \dots, \tilde{j}_{m_k}^k(p)),$$

где p и q – номера повторений системы циклов $\mathfrak{C}$, $\tilde{j}_1^k$ и $\tilde{i}_1^k$ – суперпозиция индексных выражений соответствующих переменных и функций, предъявляющих значения организаторов на повторениях $p_1, \dots, p_n$ и $q_1, \dots, q_n$ соответственно.

Для алголоподобных языков, в которых имеется синтаксическая форма лишь для организаторов первого уровня (оператор FOR...WHILE ...DO) i_s^r и j_d^r имеют вид $i_s^{r,0} + p_u h_s^{r,0}$, $j_d^{r,0} + q_v h_d^{r,1}$ $(r \in \{1, \dots, k\}$, $s \in \{1, \dots, n_r\}$, $d \in \{1, \dots, m_r\})$, $i_s^{r,0}$, $j_d^{r,0}$ – нижние границы, а $h_s^{r,0}$, $h_d^{r,1}$ – шаги изменения параметра цикла.

ЗАМЕЧАНИЕ 3. В циклах с однократной записью повторение c_1, на котором вычисляется значение некоторой переменной $a[v_1, \dots, v_n]$, используемое на повторении c, определимо в силу сохранения значений всех предыдущих повторений организаторов.

Действительно, для определения c_1 необходимо вычислить значение индексных выражений переменой $a[v_1, \dots, v_n]$ на повторении c. Затем для каждой переменной $a[\varkappa_1, \dots, \varkappa_n]$ следует вычислить набор $\varkappa_1, \dots, \varkappa_n$ на повторениях от $0,0,\dots 0$ до c. Наибольшее в лексикографическом порядке среди повторений, на которых произошло совпадение значений $\varkappa_1, \dots, \varkappa_n$ с набором значений $v_1, \dots, v_n$ на повторении c, и будет искомым повторением c_1.

Определим понятие п р о с т о г о п у т и в с-схеме Π по индукции. Всякую последовательность инструкций $i_1, \dots, i_n$ такую, что I) i_1 – стартовая инструкция, 2) входная метка каждой следующей инструкции является выходной меткой предыдущей инструкции и 3) одна из выходных меток инструкции i_n конечная, назовем д о п у с т и м о й.

Допустимую последовательность, не содержащую инструкций вида 3 или 4, назовем **простым путем** с-схемы П. Замена в допустимой последовательности инструкции вида 3 или 4 на простой путь в подсхеме или теле цикла, в конце которого приписана инструкция перехода с предикатом повторного выполнения этого цикла, также дает **простой путь** с-схемы П.

Пусть циклы $c_1,\ldots,c_m$ системы $\mathfrak{C}$ свободны *); $p_{i_1},\ldots,p_{i_m}$ – последовательность (возможно, с повторениями) простых путей системы $\mathfrak{C}$; $c_1,\ldots,c_m$ – возрастающая последовательность номеров итераций таких, что в любых соседних простых путях p_{i_k} и $p_{i_{k+1}}$ $(k = 1,\ldots,l-1)$ имеются одноименные либо простые переменные u_{i_k} и v_{i_k} соответственно, либо переменные с индексами, для которых решение $\tau(c,\gamma)$ системы (I) таково, что $\exists\gamma[c_{i_{k+1}} = \tau(c_{i_k},\gamma)]$, причем u_{i_k} является левой частью присваивания.

Из определения информационной связи и алгоритма τ вытекает

ТЕОРЕМА 2. *Две итерации i и j подсхем $П_1$ и $П_2$ системы $\mathfrak{C}$ являются информационно независимыми, если для всяких вышеуказанных последовательностей путей и итераций таких, что $u_{i_1} \in D(П_1)$, $v_{i_m} \in D(П_2) \cup O(П_2)$ и $c_1 = i$, имеет место $c_n \neq j$.*

3. Описание алгоритма распараллеливания

Ниже мы опишем алгоритм распараллеливания при наличии алгоритма τ (т.е. алгоритма, отвечающего на вопрос: какие повторения зависят от данного повторения?).

Следует отметить, что алгоритма τ может и не быть, так как в общем случае алгоритма решения уравнений (I) может не существовать **).

Естественным выходом из данной ситуации является создание языков высокого уровня, которые основывались бы на записи в ка-

*) Свободность схем определяется обычным образом (см., например, [I]).

**) В АЛГОЛе допустимы такие индексные выражения, что система (I) содержит диофантовое уравнение.

честве индексных выражений некоторой информации о том, от каких повторений и каких созданных переменных зависит данная переменная на данном повторении. Тогда язык остается привычно последовательным, но имеется возможность распараллеливания общим алгоритмом R1. Для существующих языков программирования алгоритм R1 пригоден только для того класса программ, для которых эффективно находятся решения систем типа (I).

Пусть циклы системы $\mathfrak{C}$ являются свободными и обладают свойством однократной записи. Уровнем элементов $\Pi_{j,1}, \Pi_{j,2}, \Pi, p_l$ назовем числа i, j, n, l соответственно. Через $\tilde{\Pi}_m(k)$ $(k=1,\ldots,n)$ обозначим произвольный элемент из $\bigcup\limits_{i,j,k} \{\Pi_{i,1}, \Pi_{i,2}, \Pi, p_l\}$ уровня k. Через c' для повторения c обозначим ближайший номер (в лексикографическом порядке), меньший c.

Алгоритм R1 заключается в следующем. На первом шаге параллельно вычисляются те повторения c элемента $\tilde{\Pi}(k)$, которые удовлетворяют условиям:

$$1)\ \forall\gamma,\tilde{\Pi}_1(k_1)[(\forall c_1(c_1=(c_1^1,\ldots,c_n^1)\,\&\,\tau(c_1,\tilde{\Pi}_1(k_1),\tilde{\Pi}(k),\gamma)=c \rightarrow$$

$$(\exists i(c_i^1<0)\vee(\tau(c_1,\Pi_1(k_1),\Pi(k),\gamma)\ -\ \text{не определено}\ ))];$$

$$2)\ \forall\tilde{\Pi}_1(k_1),\ c_1,\ l[(l>k\,\&\,\tau(c_1,\tilde{\Pi}_1(k_1),\ p_l,\gamma)=c)\rightarrow(c_1=c_1^1,\ldots$$

$$\ldots,c_n^1\,\&\,\exists i(c_i^1<0))]\ .$$

Иначе говоря, вначале параллельно вычисляются те повторения элементов $\mathfrak{C}$, которые не зависят от предыдущих итераций.

Множество таких повторений $\tilde{\Pi}(k)^c$ включается в первоначально пустое множество F.

На каждом следующем шаге параллельно вычисляются все те повторения c схемы $\tilde{\Pi}_2(k_2)$, для которых

$$1)\ \forall\gamma,\tilde{\Pi}_3(k_3)\,\exists c_1[\tau(c_1,\tilde{\Pi}_3(k_3),\tilde{\Pi}_2(k_2),\gamma)=c\,\&\,c_1\in F\vee\exists i\ (c_1=$$

$$=c_1^1,\ldots,c_n^1\,\&\,(c_i^1<0))],$$

$$2)\ \forall\tilde{\Pi}_3(k_3),\ c_1,\ l[l>k\,\&\,\tau(c_1,\tilde{\Pi}_3(k_3),\ p_l,\gamma)=c\rightarrow\tilde{\Pi}_3(k_3)^{c'}\in$$

$$\in F\vee(\tau(c_1,\tilde{\Pi}_3(k_3),p_l,\gamma)\ -\ \text{не определено}\ )];$$

3) предикаты p_1 для $1 > k$ на повторении c' имеют значение истина.

Множество таких повторений $\tilde{\Pi}_2(k_2)^c$ на каждом шаге добавляется в множество F. Таким образом, в каждый следующий момент вычисляются не являющиеся "лишними" (что гарантируется условием 3) повторения c, для которых на предыдущих шагах подготовлены данные (на итерациях из F).

Процесс обрывается, если для всех c' повторения p_1 имеют значение ложь.

Алгоритм R1 для реализации параллельных вычислений требует больших ресурсов: большого объема памяти для хранения множества F и значительного времени для нахождения новых элементов этого множества (нахождение может осуществляться параллельно).

Объем затрат на дополнительное управление для организации параллельных вычислений может быть уменьшен, если между некоторыми элементами системы циклов либо отсутствует связь, либо она односторонняя. Например, все повторения $\Pi_{1,1}$ не зависят от повторений $\Pi_{j,2}$. Тогда можно вначале осуществлять параллельное выполнение системы C', которая получается из C выбрасыванием $\Pi_{j,2}$, а вслед за ним – выполнение повторений $\Pi_{j,2}$.

ЗАМЕЧАНИЕ 4. Определение множества F в общем случае носит динамический характер. Ввиду этого номера повторений, которые будет выполнять данный модуль (процессор-память) вычислительной системы, неопределимы заранее. Следовательно, нельзя так распределить массивы по памятям модулей, чтобы дублирование памяти и обмены между модулями были минимальными.

Частично проблема априорного распределения памяти может быть решена, если организаторы не зависят от ядра. Тогда алгоритм τ, одним из аргументов которого является ядро, может быть вычислен, когда вычислены все значения организаторов. Ввиду этого оптимальное размещение массивов ядра может быть определено после выполнения организаторов.

Ясно, что если организаторы также разделены, т.е. зависимость между организаторами уровня, меньшего некоторого k, и уровня, большего k, односторонняя, то можно применить аналогичный способ размещения массивов организаторов уровней, больших k.

Дополнительные расходы времени на управление параллельным выполнением, задаваемым алгоритмом R1, во многом определяется сложностью алгоритма τ.

С учетом замечания 4 мы рассмотрим класс схем алголоподобных программ, в которых имеются организаторы только первого уровня, аналогичные оператору (FOR...WHILE...) , а индексные выражения переменных есть линейные функции от значения организатора.

Тогда τ есть комбинация решений системы (I) линейных уравнений.

Однако наличие простого алгоритма τ обеспечивает лишь простоту нахождения множеств информационно независимых повторений (из которых и состоит F).

Следующий пример показывает, что наличие таких множеств совсем не означает возможность их параллельного выполнения.

ПРИМЕР I.

```
For l:= 1 step 1 until n1 do
For i:= 1 step 1 until n2 do
For j:= 1 step 1 until n3 do
For k:= 1 step 1 until n4 do
For m:= 1 step 1 until n5 do
u[i,j]:= f(u[i-1,j-1],u[i,j],u[i+k,k],u[m,j+m])
    next m
        next k
            next j
                next i
                    next l
```

Геометрическая интерпретация зависимости u[i,j] от повторений, которые должны быть подсчитаны до выполнений повторения l_1, i_1, j_1, приведена на рис.I.

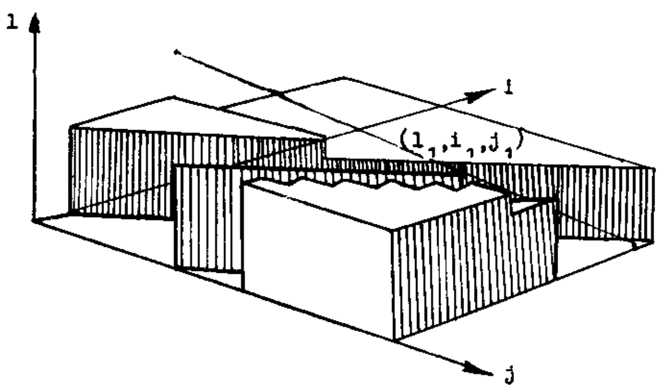

Рис. I

Можно видеть, что все точки всякой прямой $L_c(t)=(c-\alpha t, t, t)$ $(c>0, \alpha>0)$ являются номерами информационно не связанных повторений. Однако способа параллельного выполнения множеств повторений, лежащих на прямых, параллельных $L_c(t)$, при данных языковых средствах не существует.

Следующая теорема устанавливает критерий существования гиперплоскостей размерности n-1 (в фазовом пространстве), все точки которых являются номерами информационно не связанных повторений.

Пусть в системе циклов **C** в схеме П нет переменных, одноименных с переменными, создаваемыми в $П_{j,2}$, а в $П_{i,1}$ – создаваемых в П и $П_{j,2}$; и пусть $\tau_i(c,\gamma)$ – решения системы (I) для пар одноименных переменных схемы П. Тогда имеет место

ТЕОРЕМА 3. Существует такая гиперплоскость G размерности n-1, что все ее целочисленные точки являются номерами информационно не связанных повторений схемы П тогда и только тогда, когда существует целочисленный вектор m и для всякого i либо $\tau_i(c, \gamma)$ не определено, либо скалярное произведение $((c-\tau_i(c, \gamma)), m)$ положительно при $c_n > \tau_i(c, \gamma)$.

Причем в качестве G можно взять гиперплоскость с нормалью m.

ДОКАЗАТЕЛЬСТВО. Пусть $c = c_1, \ldots, c_n$ – некоторое повторение системы циклов, G_c – гиперплоскость с нормалью m, проходящая через точку c. Тогда $x=\tau(c,\gamma_0)$ находится в положительном относительно m полупространстве, так как $(c-x,m)>0$, по условию теоремы. Причем расстояние между x и гиперплоскостью G_c равно $\frac{(c-x,m)}{|m|}$.

Покажем теперь, что всякое повторение y, от которого зависит повторение x, также находится в положительном полупространстве на расстоянии от G_c бо́льшем, чем $\frac{(c-x,\ m)}{|m|}$. Действительно, пусть $y=\tau(x,\gamma_1)$, тогда $(c-y,m) = (c-x+x-y,m) = (c-x,m)+(x-y,m)$. Но так как $x=\tau(c,\gamma_0)$, а $y=\tau(x,\gamma_1)$, то, по условию теоремы, $(c-x,m)>0$ & $(x-y,m)>0$. Отсюда $(c-y,m)>0$, так как расстояние от y до G_c есть $\frac{(c-y,m)}{|m|} = \frac{(c-x,m)}{|m|} + \frac{(x-y,m)}{|m|} > \frac{(c-x,m)}{|m|}$.

Следовательно, всякое повторение, от которого зависит повторение c, не лежит на гиперплоскости G_c. Что и требовалось доказать.

Существование гиперплоскости G размерности $n-1$ всегда обеспечивает и существование некоторого порядка параллельного выполнения повторений, номера которых лежат на гиперплоскостях, параллельных G.

ТЕОРЕМА 4. *В условиях теоремы 3 существует схема параллельных вычислений повторений схемы П, номера которых лежат на гиперплоскостях, параллельных G.*

ДОКАЗАТЕЛЬСТВО. Из доказательства теоремы 3 следует, что для каждого повторения множество номеров повторений, от которых оно зависит, лежит по одну сторону плоскости G, проходящей через номер данного повторения. Следовательно, при обходе гиперплоскостей, начинающемся с гиперплоскости первой в направлении $(-m)$ и далее в порядке удаления от нее в направлении m, информационные связи не нарушаются. А в силу предположений, сделанных относительно схемы П, все повторения $П_{i,1}$ для любого i могут быть вычислены до начала вычислений повторений П, а повторения схем $П_{j,2}$ для любого j могут быть вычислены после.

Следующая теорема указывает достаточные условия существования параллельного счета по гиперплоскостям размерности, меньшей $n-1$. Теорема обобщает результат [4] об исчезающих индексах.

ТЕОРЕМА 5. *Если для системы $\mathfrak{C}$ существуют вектора $m_1, m_2, \ldots, m_r$ такие, что*

$$\forall l \forall c[((c,m_1)=0) \& ((c,m_2)=0) \& \ldots \& ((c,m_l)=0) \& (c_r > \tau(c,\gamma)) \rightarrow$$

$$\rightarrow ((c-\tau(c,\gamma),m_l) \geq 0 \& (l=r-1 \rightarrow (c-\tau(c,\gamma),m_r) > 0)],$$

то существует схема параллельных вычислений системы $\mathfrak{C}$, при которой параллельно вычисляются повторения схемы П, номера которых лежат на плоскостях, параллельных $G_r = \{c \mid ((c,m_1)=0) \& \ldots \& ((c,m_r)=0)\}$.

Следующий пример показывает, что при отсутствии гиперплоскостей размерности $n-1$ могут существовать и гиперплоскости размерности меньшей чем $n-1$, которые не могут быть найдены с помощью теоремы 5. Попутно этот пример показывает и возможность применения теоремы 5 для нахождения гиперплоскости размерности $n-2$ $(n=3)$.

ПРИМЕР 2.

```
For x:= 1 step 1 until n1 do
For y:= 1 step 1 until n2 do
For z:= 1 step 1 until n3 do
For k:= 1 step 1 until n4 do
For p:= 1 step 1 until n5 do
u[x,y,z]:= f(u[x,y-1,z],u[x,y,z-1],u[x-1,y,z],u[x-p,p,q],
u[x-k,y,k])
        next p
            next k
                next z
                    next y
                        next x
```

Геометрическая интерпретация зависимости точки x_1,y_1,z_1 от повторений, которые необходимы для вычислений $u[x_1,y_1,z_1]$, приведена на рис.2.

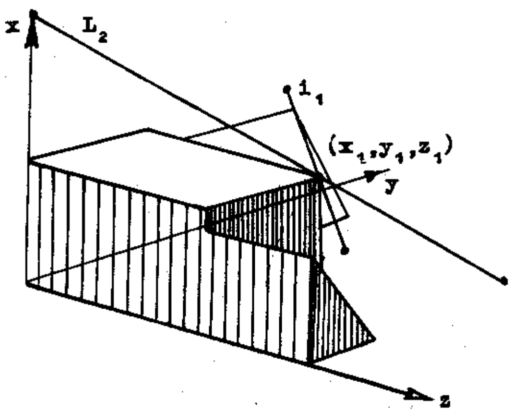

Рис. 2

Здесь существуют два класса прямых, не пересекающих множество номеров повторений, которые должны быть подсчитаны к моменту вычисления точки x_1,y_1,z_1. Прямая L_1 (рис.2) лежит в плоскости $x=\text{const}$; прямая L_1 не лежит в плоскости $x = \text{const}$.

Первая прямая может быть найдена применением теоремы 5 ($m_1 = (I,0,0)$, $m_2 = (0,-I,I)$). Порядок вычисления по прямым параллельным L_1 следующий. Вначале считаются точки прямых в плоскости $z\equiv1$, начиная от прямой, ближайшей к (0,0,0) и параллельной L_1, затем точки прямой, ближайшей к этой прямой, в сторону удаления от (0,0,0) и т.д. После этого аналогично считаются точки в плоскости $z \equiv 2$ и т.д.

Вторая прямая не обнаруживается с помощью теоремы 5. Порядок вычисления по прямым, параллельным L_2, сложнее. Вначале считаются точки прямых, ближайших к оси z, начиная от ближайшей к точке (0,0,0), затем – ближайших к оси y, также начиная с ближайших (0,0,0). Затем снова вдоль оси z (ближайших к просчитанным), затем вдоль оси y и т.д.

При наличии зависимости между П и $П_{j,1}$, П и $П_{i,2}$, П и P_i критерии теорем 3 и 5 могут быть использованы в алгоритме R1.

Можно видеть, что теоремы 3 и 5 являются обобщением результатов работы [4].

Автор выражает благодарность Котову В.Е., Цейтлину Г.Е., Халилову А.И., Миренкову Н.Н., Вальковскому В.А. за полезные обсуждения.

Литература

1. LUCKHAM D.C., PARK D.M., PATERSON M.S. On formulized computer programs.-"J.Comput.and System Sci.", 1970, v.4, p.220-240.

2. ТРАХТЕНБРОТ Б.А. Об универсальных классах схем программ. -В кн.: Теория программирования. Труды симпозиума. Ч.I, Новосибирск, 1972, с. 239-249.

3. НУРИЕВ Р.М. Необходимые и достаточные условия существенной распараллеливаемости программ. -"Изв. АН СССР. Серия техническая кибернетика", 1976, № 2, с.105-111.

4. LAMPORT L. The parallel execution of do loops.-"Communs ACM", 1974, v.17, N 2, p.83-93.

Поступила в ред.-изд.отд.
27 января 1977 года