ЛОГИЧЕСКИЕ ВОПРОСЫ ТЕОРИИ ТИПОВ ДАННЫХ
(Вычислительные системы)
1986 год Выпуск 114

УДК 519.685

# АЛГОРИТМ ИМИТАЦИИ ПЛАНИРОВАНИЯ

В.Э.Малышкин, С.А.Панкратов

В работах [1-3] описывается метод синтеза параллельных программ на вычислительных моделях с массивами. Один из этапов создания параллельных программ в нем - конструирование алгоритма (плана вычислений), решающего сформулированную на вычислительных моделях задачу. Центральное место в конструировании плана вычислений занимает алгоритм имитации планирования, идея которого впервые описана в [3]. В настоящей работе этот алгоритм описывается более детально и приводится к форме, пригодной для реализации.

## Основные понятия

Рассмотрим ряд понятий и алгоритмов, необходимых для изложения алгоритма имитации планирования.

Определение упрощенной вычислительной модели. Пусть заданы:

1) $X = Y \cup Z$, где $Y$ - счетное либо пустое множество переменных. Если $Y$ - непустое множество, то оно разбивается на конечное число непересекающихся, счетных, линейно упорядоченных подмножеств, которые называются массивами. Множество всех массивов обозначим $\overline{Y} = \{\overline{x}, \overline{y}, \ldots, \overline{z}\}$. Элементы массива $\overline{x} = \{x_1, x_2, \ldots, x_n, \ldots\}$ называются компонентами $\overline{x}$, компонент $x_n$ обозначается $\overline{x}[n]$. Множество $Z = \{x, y, \ldots, z\}$ - конечное множество переменных, $Z \cap Y = \emptyset$, элементы $Z$ называются простыми переменными.

2) $F \subseteq F1 \times N$, где $N$ - множество натуральных чисел, $F1 = \{a, b, \ldots, c\}$ - конечное множество функциональных символов (операций).

Элемент $(a,i) \in F$ обозначается $a^i$ и называется $i$-м вхождением $a$ в $F$. Операция $a$ называется простой, если $N_a = \{n \in N \mid (a,n) \in F\}$ - одноэлементное множество, и массовой, если $N_a$ содер-

жит более одного элемента. Функция In сопоставляет каждому вхождению $a^i$ единственное конечное множество входных $in(a^i)=In(a,i)$, а функция Out – единственное конечное множество выходных $out(a^i)=Out(a,i)$ переменных $in(a^i) \neq \emptyset, out(a^i) \neq \emptyset$. Индекс i в обозначениях далее будет опускаться в случаях, когда номер вхождения несуществен. Введенные определения формализуют обычный в модульном программировании прием, когда отлаженный программный модуль включается в новую программу и применяется либо к некоторому набору входных переменных для вычисления набора выходных, либо в цикле применяется к каждому набору из множества наборов входных переменных для вычисления множества наборов выходных переменных. Элемент $a^i \in F$ выражает возможность вычисления переменных $out(a^i)$ при выполнении операции $a \in F1$ с входными переменными $in(a^i)$, а операция a в конкретной интерпретации реализуется программным модулем.

Набор $C = (X, F, In, Out)$ называется упрощенной вычислительной моделью с массивами. На C определяется множество термов $T_0$, а также множество var(t) используемых переменных в терме t и вершина терма top(t):

а) если $x \in X$, то $x \in T_0$, $in(x) = out(x) = \{x\}$, $var(x) = \{x\}$, $top(t) = x$;

б) если $a \in F$, $in(a) = \{x_1, \ldots, x_n\}$, $t_1, t_2, \ldots, t_n$ – такие термы из $T_0$, что $x_i \in out(t_i)$, $i = 1, 2, \ldots, n$, то $a(t_1, t_2, \ldots, t_n) \in T_0$,

$$in(a(t_1, \ldots, t_n)) = \bigcup_{i=1}^{n} in(t_i), \quad out(a(t_1, \ldots, t_n)) = out(a),$$

$$var(t) = \bigcup_{i=1}^{n} (var(t_i) \setminus out(t_i)) \cup (in(a) \cup out(a)).$$

Запись $t = a(t_1, \ldots, t_n)$ означает, что t есть терм $a(t_1, \ldots, t_n)$, множеством подтермов терма t называется множество $st(t) = \{t\} \cup (\bigcup_{i=1}^{n} st(t_i))$, если $t = x$, $x \in X$, то $st(t) = \{x\}$. Глубиной терма t называется величина $d(t) = \max_i(d(t_i)) + 1$, $d(x) = 0$, $x \in X$, а вершиной top(t) терма t – вхождение операции a.

На множества $in(a^i)$ и $out(a^i)$ накладываются обычные для программных модулей ограничения, а именно:

1) $in(a^i) \neq in(a^j)$;

2) $|in(a^i)| = |in(a^j)| \wedge |out(a^i)| = |out(a^j)|$;

3) множества $in(a^i)$ и $in(a^j)$, а также $out(a^i)$ и $out(a^j)$ могут различаться только компонентами одноименных массивов, т.е. не допускается, к примеру, $in(a^1)=\{\bar{x}[1],\bar{x}[3]\}$, а $in(a^2)=\{\bar{x}[2],\bar{y}[3]\}$. Здесь $i\in N$, $j\in N$, $i\neq j$, запись $|in(a^i)|$ обозначает число элементов в множестве $in(a^i)$. При описании алгоритма имитации планирования в качестве In и Out будут взяты функции, сопоставляющие $a^i$ такие $in(a^i)$ и $out(a^i)$, что множество $in(a^i)\cup out(a^i)$ может содержать компоненты массивов только вида $\bar{x}[\alpha_a\cdot i+\beta_a]$. Здесь i – общее для компонентов массивов из $in(a^i)\cup out(a^i)$. Например, $in(a^i)=$ $=\{\bar{x}[2i+3],\bar{y}[3i-2]\}$, $out(a^i)=\{\bar{z}[2i-1]\}$, $in(b^i)=\{\bar{v}[5i-1]\}$.

Введем необходимую терминологию. Пусть $t\in T_0$, $d(t)\neq 0$. Если $z\in out(t)$, то будем говорить, что терм t вычисляет переменную z. Если терм $t_0=a(t_1,t_2,\ldots,t_n)$ принадлежит $st(t)$, то будем говорить, что операция (вхождение) a используется в терме t. Пусть для $t_0$ выполнено: $in(a)=\{x_1,\ldots,x_n\}$, $t_j=b_j(t_{j1},\ldots,t_{jm_j})$, $j=1,\ldots,n$. Тогда говорим, что вхождение $b_j$ используется в терме t для вычисления переменной $x_j$. Терм t будет часто изображаться графически, чтобы показать все используемые в нем переменные. В $T_0$ выделяется множество $T_1$ бесповторных термов. Терм t входит в $T_1$ тогда и только тогда, когда в графическом представлении t на пути от любого листа к корню каждое вхождение операции встречается только один раз для вычисления любой своей выходной переменной. Пусть $V\subset X$, $W\subset X$, $V\cap W=\emptyset$. Определим множества $T_V=\{t\in T_1\,|\,in(t)\subseteq V\}$, $T_V^W=\{t\in T_V\,|\,out(t)\cap W\neq\emptyset\}$. Всякое подмножество $T\subseteq T_V^W$ называется (V,W)-планом, если $\forall x\in W\ \exists t\in T\ (x\in out(t))$. Интерпретация I вычислительной модели C в области D задается обычным образом: каждой операции $a\in F1$ сопоставляется функция $f_a=I(a)$, каждой переменной $x\in X$ – единственный элемент $d_x\in D$, значение x. Впредь будут рассматриваться лишь те интерпретации, которые удовлетворяют условию $I(out(a^i))=f_a(I(in(a^i)))$. Здесь $I(in(a^i))$ обозначает множество $\{d_{x_1},d_{x_2},\ldots,d_{x_n}\}$ значений, сопоставленных в интерпретации I переменным $in(a^i)=\{x1,x2,\ldots,xn\}$. Такой же смысл обозначения $I(out(a^i))$. Таким образом, переменным $out(a^i)=\{y1,y2,\ldots,ym\}$ присваивается значение $\{d_{y_1},d_{y_2},\ldots,d_{y_m}\}$

терма $t = a(t_1, t_2, \ldots, t_n)$, а множество термов $T_V^W$ определяет множество всех определенных на C алгоритмов вычисления значений переменных W из значений переменных V [4]. На рис. I,а приведен пример упрощенной вычислительной модели, эта же модель показана на рис I,б, на котором массивы для наглядности представлены целостными объектами. Возможна следующая интерпретация такой модели: $I(e)$ – считывание первой записи файла x в память ЭВМ, $I(a^i)$ – считывание $(i+1)$-й записи файла x, $I(b^i)$ – вычисление i-й записи файла y из i-й записи файла x.

Понятно, что для ограничения вычислений нужны дополнительные средства (например, счетчики или предикаты, с помощью которых можно было бы завершить вычисления, когда все записи файла будут считаны), однако для описания алгоритма имитации планирования они не нужны и потому не вошли в определение упрощенной вычислительной модели, чтобы не усложнять текст несущественными деталями.

Алгоритм планирования. Простые вычислительные модели Э.Х.Тыугу [5] получаются из упрощенных вычислительных моделей, если $Y = \emptyset$, а F содержит только простые операции. Для построения $T_V^W$ на простой вычислительной модели используется алгоритм планирования [6], предполагается, что такая модель представлена двумя таблицами ТХ и ОП. Каждая строка таблицы ТХ имеет вид: $(x, A(x), comp(x))$, а таблицы ОП – $(a, in(a), out(a))$, $x \in X$, $a \in F$, $A(x) = \{a \in F \mid x \in in(a)\}$, $comp(x) = \{a \in F \mid x \in out(a)\}$. Так как множество $T_V^W$ может быть очень велико, то при планировании строятся множества переменных и операций, используемых в термах из $T_V^W$, и из них при необходимости конструируются нужные термы. Алгоритм планирования состоит из двух частей: восходящей и нисходящей. В восходящей части строятся множества переменных и операций, используемых в термах из $T_V$. Обозначим $V_0 = V = \{x_1, x_2, \ldots, x_n\}$ множество входных переменных задачи. Тогда множество $F_0 = \{a \in F \mid in(a) \subseteq V_0\} = \bigcup_{x \in V_0} \{a \in A(x) \mid in(a) \subset V_0\}$ содержит все операции вычислительной модели C такие, что $in(a) \subseteq V_0$, далее формируется множество $V_1 = \bigcup_{a \in F_0} \{out(a) \cup V_0$, на основе $V_1$ строится

$$F_1 = \bigcup_{x \in V_1 \setminus V_0} \{a \in A(x) \mid in(a) \subseteq V_1\}$$

и т.д. до тех пор, пока при некотором целом положительном k не окажется $F_k = \emptyset$. На этом завершается восходящая часть алгоритма

планирования. Множества $F_i$ содержат те и только те операции, которые используются в термах из множества $T_V$. Если $W \not\subseteq V_k$, то планирование прекращается, так как не могут быть вычислены все переменные из $W$. В противном случае начинается нисходящая часть планирования. Задача нисходящей части алгоритма планирования заключается в построении тех переменных и операций, которые используются в термах из $T_V^W$. Обозначим $F^* = \bigcup_{i=0}^{k} F_i$ и определим множества

$G_1 = \bigcup_{x \in W} \{a \in F^* \mid a \in comp(x)\}$, $H_1 = \bigcup_{a \in G_1} in(a)$, далее, для $i=2,3,\ldots$

$G_i = \bigcup_{x \in H_{i-1}} \{a \in F^* \mid a \in comp(x) \wedge a \notin \bigcup_{m=1}^{i-1} G_m\}$, $H_i = \bigcup_{a \in G_i} in(a)$.

Построение $G_i$ и $H_i$ завершается, когда при некотором $r$ окажется $G_r = \emptyset$. Множества $G_i$ и $H_i$, $i = 1,2,\ldots,r$, содержат все те операции и переменные, которые используются в термах из $T_V^W$. По завершении планирования в ТХ и ОП остаются лишь переменные и операции из множеств $H_i$ и $G_i$, остальные удаляются. Очевидна линейная временная сложность алгоритма планирования относительно числа дуг в графическом представлении вычислительной модели. Необходимый $(V,W)$-план $T$ может быть теперь построен алгоритмом выбора [3], суть его в следующем.

I. Термы "опускаются" из $W$, т.е. для каждой переменной $x \in W$ выбирается из $comp(x)$ одна из операций, затем выбираются операции, которые вычисляют те входные переменные выбранной операции, которые не входят в $V$, и т.д., пока не будет построен бесповторный терм $t$, вычисляющий $x$ и $in(t) \subseteq V$. Это случится обязательно в силу свойств ТХ и ОП. Требование бесповторности $t$ может привести к backtraching, т.е. к отмене сделанного на некотором шаге выбора операции и выбора вместо нее другой.

2. Для выбора операций могут быть использованы различные эвристические стратегии, например, выбирается всегда операция $a$ для реализации которой есть ресурсы, т.е. есть ресурсы для выполнения модуля, вычисляющего функцию $f_a = I(a)$, либо выбирается $a \in F_i$ с минимальным номером $i$ и т.д. Использование той или иной стратегии определяется свойствами, которыми должен обладать $T$.

Формулировка задачи. Упрощенная вычислительная модель, представленная на рис.I,б, может быть записана с использованием средств языка ОПАЛ [2] в виде:

C: (n: перем; $\bar{x}$, $\bar{y}$ : массив перем;
операция e in n out $\bar{x}$[1]; M = {i $\in$ N•j = i + 1};

$(i,j) \in M$ (операция a in $\bar{x}[i]$ out $\bar{x}[j]$;
операция b in $\bar{x}[i]$ out $\bar{y}[i]$;););

Здесь $M = \{(i,j) \in N \times N \mid j = i+1\}$, запись $(i,j) \in M$ операция a in $\bar{x}[i]$ out $\bar{x}[j]$ определяет массовую операцию a, которая применима к компонентам $\bar{x}[i]$ для вычисления $\bar{x}[j]$, причем i и j взяты из некоторой пары $(i,j) \in M$, M называется областью применимости a, N – терминальный символ языка, обозначает множество натуральных чисел.

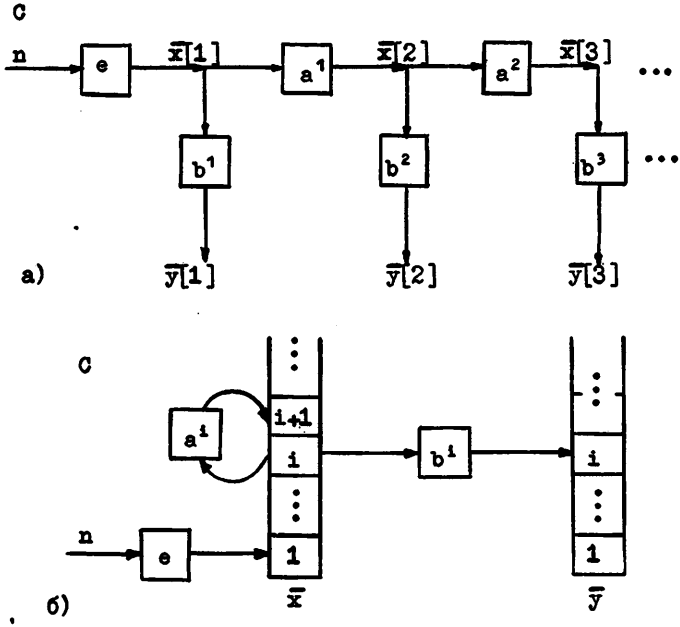


Рис. I

На упрощенной вычислительной модели C может быть сформулирована задача:

ПРИМ: на C вычислить $\bar{y}$ из n ;

Алгоритм решения задачи ПРИМ задается бесконечным множеством термов

$T_V^W$: $t_1 = b^1(e(n))$, $t_2 = b^2(a^1(e(n)))$,..., $t_n = b^n(a^{n-1}(\ldots$
$\ldots a^1(e(n))\ldots))\ldots$, $V = \{n\}$, $W = \{\bar{y}\}$,

запись $\{\bar{y}\}$ обозначает множество $\{\bar{y}[1], \bar{y}[2], \ldots\}$.

Для наших целей множество $T_V^W$ удобно представлять алгоритмом, перечисляющим термы из $T_V^W$ в некотором порядке. Для конструирования этого алгоритма на основе упрощенной вычислительной модели C формируется простая вычислительная модель C'(рис.2),которая оп-

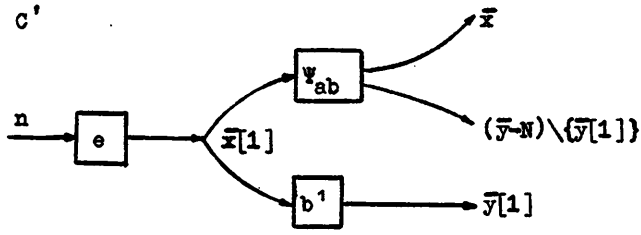


Рис. 2

ределяет вхождения всех операций C , используемых в термах из множества $T_V$. Как обычно, в C' операция $\Phi_{ab}$ показывает, что переменные C',обозначенные символами $\bar{x}$ и $(\bar{y}-N)\setminus\{\bar{y}[1]\}$, могут быть вычислены из переменной $\bar{x}[1]$. Здесь символ $(\bar{y}-N)\setminus\{\bar{y}[1]\}$ обозначает множество $\{\bar{y}[2], \bar{y}[3],\ldots\}$. Вообще,далее символом $\bar{x}-M, M\subseteq N$, будем обозначать множество $\{\bar{x}[i] \mid i\in M\}$. Операция $\Phi_{ab}$ задается параметрическим термом $\bar{x}[i]\to a\to\bar{x}[j]\to b\to\bar{y}[j]$ , в котором в качестве переменных используются символы вида $\bar{x}[i],\bar{y}[j]$,областью применимости $M = \{i\in N \mid j = i+1\}$ и отношением порядка $(<,)$ , при котором $(i_0,j_0) < (i_1,j_1)$ , если $i_0 < i_1$. В соответствии с этим порядком легко строится алгоритм, перечисляющий элементы M:(I,2), (2,3),... . Каждому элементу M соответствует конкретный терм, который получается из параметрического подстановкой вместо индексов i и j их значений, т.е. элементу (I,2) соответствует конкретный терм $\bar{x}[1]\to a\to\bar{x}[2]\to b\to\bar{y}[2]$ ,элементу (2,3) - конкретный терм $\bar{x}[2]\to a\to\bar{x}[3]\to b\to\bar{y}[3]$ и т.д. Множество всех конкретных термов определяет множество всех используемых в них вхождений операций a и b,и именно они необходимы в C (используются в термах из $T_V^W$) для вычисления переменных $\{\bar{y}[2],\bar{y}[3],\ldots,\bar{x}[1],\bar{x}[2],\ldots\}$ из переменной $\bar{x}[1]$ , а также порядок их применения. Заметим,что объединение множеств входных переменных всех конкретных термов есть $\{\bar{x}[1],\bar{x}[2],\ldots\}$ , однако наличие отношения порядка на M позволяет, как легко установить, вычислить $\{\bar{y}[2],\bar{y}[3],\ldots,\bar{x}[1],\bar{x}[2],\ldots\}$

только из $\{\bar{x}[1]\}$ . Поэтому $in(\phi_{ab}) = \{\bar{x}[1]\}$, а в $out(\phi_{ab})$ помещаются все используемые во всех конкретных термах переменные, т.е. $out(\phi_{ab}) = \{\bar{x},(\bar{y}-M)\setminus\{\bar{y}[1]\}\}$ . Если провести аналогию с языками программирования, то параметрический терм задает тело цикла, а область применимости с отношением порядка на ней - заголовок цикла. Для вычисления переменных $\bar{x}$ и $(\bar{y}-M)\setminus\{\bar{y}[1]\}$ из $\bar{x}[1]$ по описанию $\phi_{ab}$ может быть, к примеру, сконструирована программа

```
for i=1 step 1 do
        (x̄[i+1]:= a(x̄[i]);
        ȳ[i+1]:= b(x̄[i+1]););
```

Очевидно, что такая программа никогда не завершит работу, но вопросы остановки вычислений здесь не рассматриваются. Таким образом, операция $\phi_{ab}$ определяет алгоритм вычисления $out(\phi_{ab})$ из $in(\phi_{ab})$ . Операции такого вида называются $\phi$-операциями. Вместо бесконечного множества $T_V^W$, $V = \{n\}$, $W = \{\bar{y}\}$ , на $C'$ может быть построено конечное множество $T_{V'}^{W'}$, $V' = \{n\}$, $W' = \{\bar{y}[1], (\bar{y}-M)\setminus\{\bar{y}[1]\}\}$ , состоящее из двух термов:

1) $n \to e \to \bar{x}[1] \to b^1 \to \bar{y}[1]$;

2) $n \to e \to \bar{x}[1] \to \phi_{ab} \to (\bar{y}-M)\setminus\{\bar{y}[1]\}$.

Очевидно, что если "раскрутить" $\phi$-операцию $\phi_{ab}$ во втором терме, то термы один и два определяют множество термов $\{t_1, t_2, \ldots, t_n, \ldots\}$, т.е. $T_V^W$ .

## Описание алгоритма

Конструирование $\phi$-операций выполняется алгоритмом имитации планирования, его описание будет сделано на ряде примеров. Пусть задан фрагмент упрощенной вычислительной модели (рис.3,а):

```
C: (n:перем; x̄,ȳ,z̄: массив перем;
операция e in n out x̄[1];  M = {i ∈ N×j = i+1 };
(i,j) ∈ M   операция f in x̄[i] out ȳ[i];
            операция g in ȳ[i] out x̄[j];
            операция a in ȳ[i] out z̄[i];));
```

Как и в алгоритме планирования, предполагается, что упрощенная вычислительная модель представлена таблицами, содержащими описания: ОП - всех простых операций упрощенной вычислительной

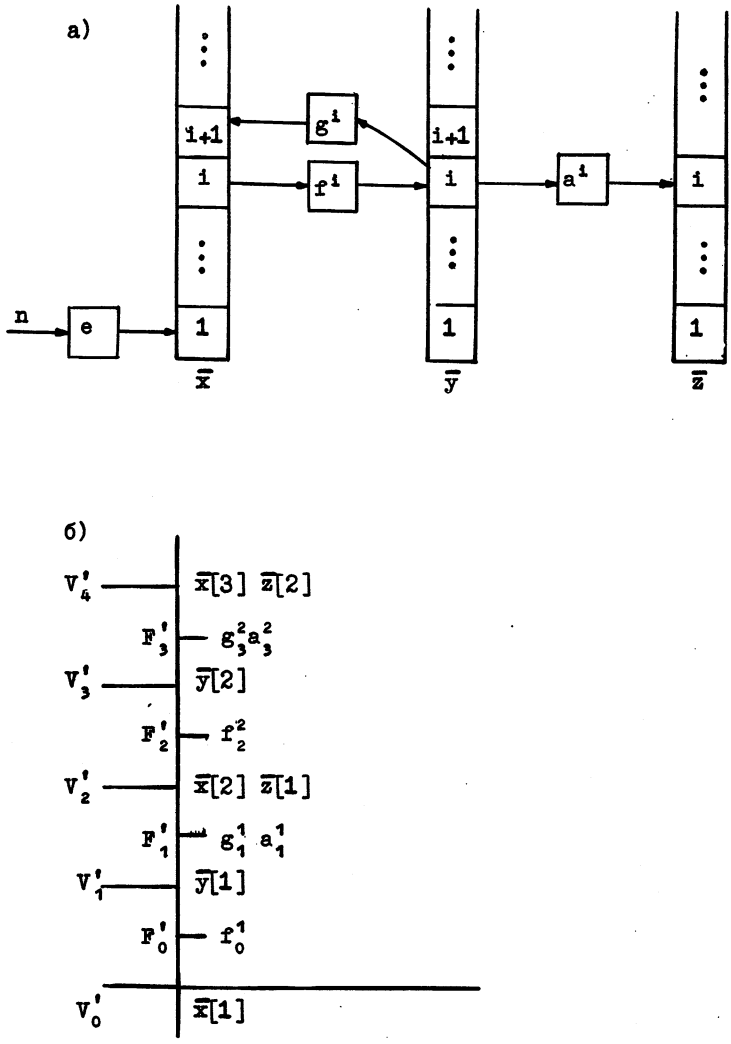


Рис. 3. Пример имитации планирования на фрагменте упрощенной вычислительной модели.

модели; ТХ – входных и выходных переменных всех операций из ОП; ТХМ – массивов, ОПМ – массовых операций.

Таблицы ТХМ и ОПМ имеют ту же структуру, что и таблицы ТХ и ОП соответственно. В ТХМ множество $A(\bar{x})$ содержит все массовые операции, входные наборы которых содержат компоненты массива $\bar{x}$, а $comp(\bar{x})$ – все массовые операции, чьи выходные наборы содержат компоненты $\bar{x}$. В ОПМ множество, к примеру, $out(g)$ содержит в параметрической форме список выходных переменных массовой операции $g$, $out(g) = \{\bar{x} - (N \setminus \{1\})\}$. Всюду ниже предполагается, что область применимости M массовой операции есть линейное множество, т.е. имеет вид:

$$M = \{i \in N \times i_1 = \alpha_1 \cdot i + \beta_1 \times \ldots \times i_n = \alpha_n \cdot i + \beta_n\},$$

где $\alpha_n$, $\beta_n$ – такие целые числа (константы), что $i \in N$ $(i_n > 0)$, $n = 1, \ldots, n$. Далее вхождения массовых операций будут нумероваться значением параметра $i$.

По формулировке задачи начинается восходящая часть планирования на вычислительной модели, описанной таблицами ТХ и ОП. После построения каждого множества $V_r$, $r = 0, 1, \ldots, k$, проверяется, содержит ли $V_r \setminus V_{r-1}$ какие-либо переменные из входных наборов некоторых вхождений массовых операций (предполагается, при $r = 0$ $V_{r-1} = \emptyset$). Если нет, то строятся множества $F_r$ и $V_{r+1}$, в противном случае планирование прерывается и делается попытка сформировать $\phi$-операцию. Пусть в примере на рис.3,а в множество $V_{r-1}$ попала переменная $n$, тогда $\bar{x}[1] \in V_r$. Так как существует массовая операция $f$, вхождение $f^1$ которой содержит $\bar{x}[1]$ в $in(f^1)$, то планирование на простой вычислительной модели прерывается, формируется множество $V_0' = V_r$. Среди операций $A(\bar{x})$ из таблицы ТХМ ищутся такие вхождения, чьи множества входных переменных содержатся в $V_0'$, и из них образуется множество $F_0'$, в примере $F_0' = \{f^1\}$. Тот факт, что $f^1 \in F_0$, будет обозначаться нижним индексом, который называется уровнем. Таким образом, запись $f_0^1$ означает, что $f^1 \in F_0'$ и имеет уровень нуль. Все элементы множества $F_0'$ заносятся во вновь образуемое множество $L$. Определим необходимое далее понятие идентичности термов $t_1$ и $t_2$:

I) если $t_1 = a_n^i(x_1, \ldots, x_n)$, $t_2 = a_k^j(y_1, \ldots, y_m)$, $i \neq j$, то $t_1$ и $t_2$ идентичны;

2) пусть $t_1 = a_n^i(t_1', \ldots, t_m')$, $t_2 = a_k^j(t_1'', \ldots, t_m'')$ и термы $t_1'$ и $t_1'', \ldots, t_m'$ и $t_m''$ попарно идентичны либо попарно глубины 0; термы $t_1$ и $t_2$ идентичны тогда и только тогда, когда при $n \geq k$ (либо $k \geq n$) для любого $i = 1, \ldots, m$ уровень вхождения $top(t_i')$ не меньше (соответственно не больше) уровня вхождения $top(t_i'')$.

Параметрический терм $t$, задающий множество $\{t_1, t_2, \ldots\}$ конкретных термов, называется самоподпитывающимся относительно множества переменных $V$, если для любого $k = 1, 2, \ldots$

$$in(t_k) \subseteq \bigcup_{m=1}^{k-1} var(t_m) \cup V.$$

Далее алгоритм имитации планирования работает следующим образом. Полагаем $n := 0$.

Шаг I. $n := n+1$; как и в алгоритме планирования, строятся множества $V_n'$, $F_n'$, $F_n^* := \bigcup_{m=0}^{n-1} F_m'$; если $F_n' = \emptyset$, то закончить работу.

Шаг 2. Выбрать произвольным образом новую массовую операцию $a$, такую, чтобы во множество $F_n'$ попали вхождения этой операции (под "новая операция" понимается здесь еще не выбранная на этом шаге операция при данном значении $n$); если больше нет новых операций, то перейти на шаг I.

Шаг 3. Пусть множество $F_n'$ содержит конечное число вхождений массовой операции $a$; если это не так, то перейти на шаг 4;

а) выбрать новое вхождение $a_n^i$ из $F_n'$; если все вхождения выбраны, то перейти на шаг 2;

б) если никакое вхождение массовой операции $a$ не содержится в $L$, то $a_n^i$ заносится в $L$; перейти на "а";

в) пусть $\{a_{m_1}^{i_1}, \ldots, a_{m_k}^{i_k}\} \subseteq L$; эти вхождения упорядочиваются по возрастанию $|i - i_l|$, $l = 1, \ldots, k$, и в таком порядке строятся все пары идентичных бесповторных термов максимально возможной глубины, один из которых $(t_2)$ имеет в вершине $a_n^i$, а другой $(t_1)$ – одно из вхождений $\{a_{m_1}^{i_1}, \ldots, a_{m_k}^{i_k}\}$. В $t_1$ и $t_2$ могут использоваться только вхождения операций из $F_n^*$. Термы $t_1$ и $t_2$ строятся одновременно алгоритмом выбора, максимальность глубины $t_1$ и $t_2$ означает, что алгоритм выбора заканчивает работу лишь тогда, когда на некотором шаге ни для каких переменных строящихся термов $t_1$ и $t_2$ не может быть сделан выбор операции. Идентичные термы $t_1$ и $t_2$ строятся максимально возможной глубины с тем, чтобы $\phi$-операция

определяла как можно больше вхождений операций. С этой же целью упорядочиваются вхождения $a^{i_1}_{m_1},\ldots,a^{i_k}_{m_k}$;

г) для каждой пары построенных идентичных термов в установленном порядке строится параметрический терм и его область применимости. В примере на рис.3,б при n=3 можно построить пару идентичных термов $\bar{x}[1] \to f^1_0 \to \bar{y}[1] \to g^1_1 \to \bar{x}[2]$ и $\bar{x}[2] \to f^2_2 \to \bar{y}[2] \to g^2_3 \to \bar{x}[3]$; заменив в любом из них конкретные номера компонентов индексами (одинаковые номера компонентов массивов заменяются одним и тем же индексом), получаем параметрический терм t вида $\bar{x}[i] \to f \to \bar{y}[i] \to g \to \bar{x}[j]$. Его область применимости $M = \{i \in N \mid j = i+1\}(<,)$ получается сравнением пары идентичных термов с учетом линейности областей применимости всех используемых в идентичных термах массовых операций. На M определен порядок, при котором при i= 1 из параметрического терма получается конкретный терм $t_1$, при i=2 - конкретный терм $t_2$ и т.д. Несложный арифметический подсчет показывает, что терм $\bar{x}[i] \to f \to \bar{y}[i] \to g \to \bar{x}[j]$ самоподпитывающийся относительно $V'_0$, и для того, чтобы можно было с помощью алгоритма имитации планирования построить все определенные t и M вхождения операций, достаточно, чтобы $\bar{x}[1] \in V'_0$. Таким образом, построена ф-операция $\phi_{fg}$, определенная t и M, $in(\phi_{fg}) = \{\bar{x}[1]\}$, $out(\phi_{fg}) = \{\bar{x}, \bar{y}\} = \bigcup_k var(t_k)$. На этом же уровне может быть построен параметрический терм $\bar{y}[i] \to a \to \bar{z}[i]$ с областью применимости N, однако он, очевидно, не самоподпитывающийся;

д) все вхождения операций, определенные построенной ф-операцией, удаляются из ОПМ, $\bigcup_{m=0}^{n} F'_m$ и L. Впредь не рассматриваются также пары идентичных термов, если хотя бы в одном из них используются удаленные вхождения. Построенная ф-операция помещается в $F_r$, а все выходные переменные ф-операции - в $V'_{n+1}$;

е) если вхождение $a^i_n$ не будет вычеркнуто из ОПМ (т.е. не построилась ни одна ф-операция с использованием $a^i_n$), то $a^i_n$ заносится в L.

В примере на рис.3 после построения $F'_3$ будет сконструирована ф-операция $\phi_{fg}$, и в $V'_4$ попадут все компоненты массивов $\bar{x}$ и $\bar{y}$. При этом в $F'_4$ будут входить все вхождения массовой операции a ; перейти к "а".

Шаг 4. Если $F'_n$ содержит бесконечное множество вхождений некоторой массовой операции, то они определяют ф-операцию с областью применимости, на которой не задано отношение порядка. В на-

шем примере все вхождения массовой операции a из $F_4'$ определяют ф-операцию $\phi_a$ с параметрическим термом $\overline{y}[i] \to a \to \overline{z}[i]$, самоподпитывающимся относительно $V_4'$, область применимости N, $in(\phi_a) = \{\overline{y}\}$, $out(\phi_a) = \{\overline{z}\}$. Такая ф-операция также помещается в $F_4$, а все определенные ею вхождения операций вычеркиваются из ОПМ, L и $F_{n+1}^*$; перейти на шаг 2.

Алгоритм завершает свою работу, если при некотором m будет $F_m' = \emptyset$, все оставшиеся в $F_m^*$ вхождения операций (их конечное число) помещаются в $F_r$ и вычеркиваются из ОПМ.

Не всегда, однако, существует m такое, что $F_m' = \emptyset$, а именно: если параметрический терм имеет вид $\overline{x}[i] \to a \to \overline{x}[j]$ с областью применимости $M = \{p \in N \times i = \alpha_1 \cdot p + \beta_1 \times j = \alpha_2 \cdot p + \beta_2\}$, где $\alpha_1 \neq \alpha_2$, то может быть, что $F_m' \neq \emptyset$ для любых $m \in M$. Возможны два варианта, которые иллюстрируются следующими примерами. Пусть в упрощенной вычислительной модели определена массовая операция $\overline{x}[i] \to a \to \overline{x}[j]$ с областью применимости $M = \{i \in N \times j = 2i\}$ и при некотором n $x[1] \in V_n'$. Тогда $a^1 \in F_n'$, $in(a^1) = \{\overline{x}[1]\}$, $out(a^1) = \{\overline{x}[2]\}$, $a^2 \in F_{n+1}'$, $in(a^2) = \{\overline{x}[2]\}$, $out(a^2) = \{\overline{x}[4]\}$. При этом алгоритм имитации планирования построит параметрический терм $\overline{x}[i] \to a \to \overline{x}[j]$ с областью применимости $M = \{i \in N \times j = 2i\}$, который неверно определяет множество всех вхождений операций a в $F_n^*$ при $n = 1, 2, \ldots$ В данном случае это будут вхождения вида $\overline{x}[2^{i-1}] \to a^k \to \overline{x}[2^i]$, $i = 1, 2, \ldots$, $k = 2^{i-1}$. Такие параметрические термы отвергаются, а в $F_m'$, $m = n, n+1, \ldots$, остаются вхождения операции a. Другой вариант будет в случае массовой операции $\overline{x}[2i] \to a \to \overline{x}[4i+2]$, $i = 1, 2, \ldots$ Если при некотором n в $F_n'$ входят все компоненты массива $\overline{x}$ вида $\overline{x}[4i]$, $i \in N$, то на каждом последующем уровне $k = n+1, n+2, \ldots$ могут быть построены ф-операции с параметрическими термами вида $\overline{x}[4i] \to a \to \overline{x}[8i+2]$, $\overline{x}[8i+2] \to a \to \overline{x}[16i+6], \ldots$, $i \in N$. Все такие операции помещаются в $F_r$. На практике это означает, что в этих случаях множества $F_n'$ строятся до тех пор, пока в памяти ЭВМ могут быть представлены номера компонентов массивов из входных и выходных множеств вхождений массовых операций, после чего алгоритм имитации планирования завершает свою работу. Число построенных операций $F_n'$ будет невелико, так как номера компонентов массивов с ростом n возрастают экспоненциально.

По завершении работы алгоритма планирования продолжается планирование на простой вычислительной модели, которое вновь может быть прервано для построения новых ф-операций, и так до тех

пор, пока не завершится восходящая часть планирования на простой вычислительной модели. Далее следует нисходящая часть планирования, в процессе которого возможно удаление каких-либо ϕ-операций.

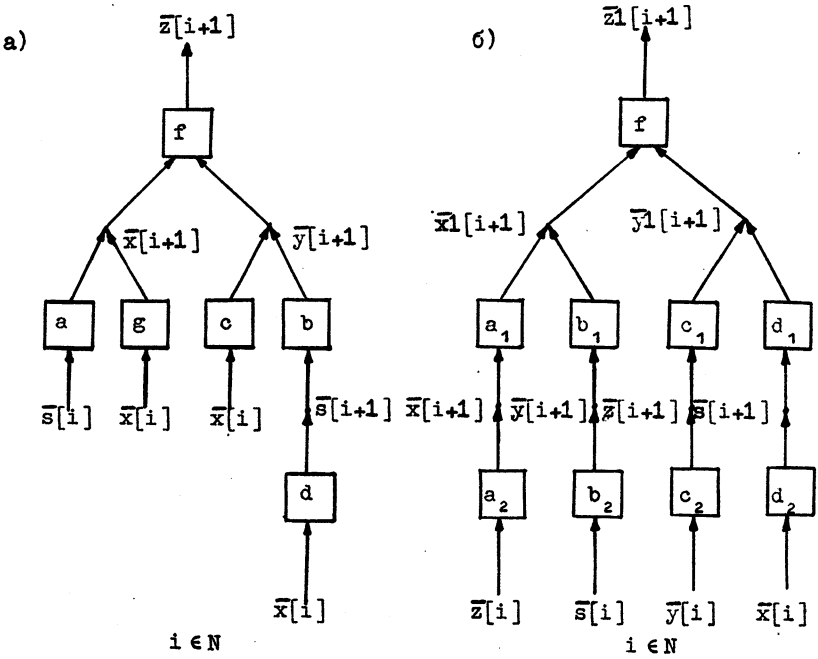


Рис. 4

Для уменьшения числа ϕ-операций в C' имеет смысл использовать в определении ϕ-операции вместо параметрического терма параметрическое дерево, в графическом представлении которого допускается, чтобы в переменную вело более одной дуги (рис.4,а). В этом смысле параметрическое дерево избыточно, поэтому из него алгоритмом выбора извлекается самоподпитывающееся параметрическое дерево, не содержащее избыточных вычислений (удаляются избыточные поддеревья). Как показывает пример на рис.4,б, требование самоподпитываемости приводит к тому, что извлекается именно параметрическое дерево, а не терм, и возможен случай, когда никакое поддерево не может быть удалено.

Описанный алгоритм планирования позволяет строить множество тех и только тех вхождений операций, которые используются в термах

из множества $T_V$. В различных модификациях он применяется для решения и многих других задач планирования.

## Литература


1. МАЛЫШКИН В.Э. Синтез параллельных программ на вычислительных моделях с массивами. I. Формальная модель.- Новосибирск, 1982, - 29 с. (Препринт/ВЦ СО АН СССР: № 379).

2. МАЛЫШКИН В.Э. ОПАЛ - язык описания параллельных алгоритмов.-В кн.: Системное и теоретическое программирование. Тез.докл. Всесоюз. сим., Кишинёв, 1983, с.260-261.

3. МАЛЫШКИН В.Э. Система синтеза параллельных программ на основе вычислительных моделей с массивами. - Новосибирск, 1984. - 44 с. (Препринт/ВЦ СО АН СССР: № 505).

4. ЕРШОВ А.П. Вычислимость в произвольных областях и базисах. - В к.: Семиотика и информатика. М., 1982, вып.19, с.3-58.

5. ТЫУГУ Э.Х. Решение задач на вычислительных моделях.-Журнал вычислит.математики и мат.физики. 1970,т.10,№3,с.38-46.

6. ЗАСЫПКИН А.В., МАЛЫШКИН В.Э. Параллельный алгоритм планирования вычисления на вычислительных моделях. -В кн.: Многопроцессорные вычислительные системы и их математическое обеспечение.Новосибирск, 1982, с.51-58.



Поступила в ред.-изд.отд.
1 ноября 1985 года