МАТЕМАТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ
(Вычислительные системы)
1979 год Выпуск 78

УДК 519.686.4:681.32.324

# О ПРИЕМЛЕМОСТИ БЛОК-СХЕМ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

Ю.И.Колосова

Под параллельной программой (р-программой) принято понимать [1,2] организованную совокупность последовательных программ (ветвей), выполняющихся на отдельных элементарных машинах однородной вычислительной системы и вступающих между собой во взаимодействия.

В работе исследуются р-программы, использующие групповые способы взаимодействий между ветвями. Такие взаимодействия задаются операторами двух типов - ОБМЕН и ОБОБЩЕННЫЙ УСЛОВНЫЙ ПЕРЕХОД (ОУП) - и заключается в том, что все ветви р-программы одновременно участвуют в обмене информацией или в передаче управления, которая осуществляется всеми ветвями в зависимости от значения некоторого признака по первому (а1) либо по второму (а2) адресу своего ОУП [3].

В р-программе могут быть ошибки, связанные как с последовательным характером выполнения операторов в ветвях, так и с неправильным заданием взаимодействий ветвей. Для выявления первых можно использовать методы отладки [4,6] или оптимизации [5]. Для выявления вторых в данной работе предлагается специальный метод, основанный на анализе управляющей граф-схемы р-программы и использующий вводимые понятия приемлемых блок-схем для ветви и р-программы, а также алгоритм, устанавливающий приемлемость блок-схемы ветви.

Управляющая граф-схема р-программы представляется совокупностью граф-схем ее l ветвей. Граф-схема ветви аналогична управляющей граф-схеме последовательной программы с одной входной и выходной вершинами [4]. Оператор ОБМЕН принадлежит множеству операторов-преобразователей, а ОУП - множеству операторов-распознавателей.

Таким образом, граф-схема р-программы - это $\bigcup_{i=1}^{l} G^i$, где $G^i =$ $= (V^i, \Gamma^i)$; $V^i = (v^i_j \mid j = 1,2,\ldots,k^i)$ - множество вершин $\Gamma^i = (\gamma^i_n, n \in N)$ - множество дуг графа $G^i$.

Множество $V^i = A^i \cup B^i$ $(A^i \cap B^i = \emptyset)$, где $A^i$ и $B^i$ - подмножества вершин, связанных соответственно с преобразователями и распознавателями.

П у т ь в граф-схеме ветви есть последовательность вершин $(v^i_{a_0}, v^i_{a_1}, \ldots, v^i_{a_r})$ $\forall j (j = 1,2,\ldots, r)$, $a_j \in \{1,2,\ldots,k^i\}$, для которой существует последовательность дуг $(\gamma^i_{a_1}, \gamma^i_{a_2}, \ldots, \gamma^i_{a_r})$ такая, что вершины $v^i_{a_{j-1}}, v^i_{a_j}$ $(j = 1,2,\ldots,r)$ соединяются дугой $\gamma^i_{a_j}$.

Входная и выходная вершины графа $G^i$ обозначаются соответственно через $b^i$ и $d^i$, а подмножество вершин, соответствующих операторам взаимодействий, - через $S^i = \{s^i_j \mid j=1,2,\ldots, m^i\}$.

Путь $(x, y_1, y_2, \ldots, y_q, z)$ называется у ч а с т к о м, если выполняются следующие условия:

1) $x \in (b^i) \cup S^i$, $z \in S^i \cup (d^i)$;

2) $\forall n (n = 1,2,\ldots,q)$ $y_n \in V^i \setminus M^i$, $M^i = \{b^i\} \cup S^i \cup \{d^i\}$; x и z называются соответственно начальной и конечной вершинами участка.

Г а м а к о м $U(x,z)$ называется множество таких участков $\{(x, y^j_1, y^j_2, \ldots, y^j_{q_j}, z) \mid j = 1,2,\ldots, h\}$, у которых начальная и конечная вершины совпадают, причем $\forall q_p (p=1,2,\ldots, j)$ путь из $y^j_{q_p}$ проходит через z.

К о н ф и г у р а ц и е й называется последовательность гамаков $U_0(x_0, z_0)$, $U_1(x_1, z_1), \ldots, U_k(x_k, z_k)$ такая, что $x_0 = b^i$, $z_k = d^i$, $z_0 = x_1$, $z_1 = x_2, \ldots, z_{k-1} = x_k$.

Блок-схема ветви называется п р и е м л е м о й, если соответствующая ей граф-схема $G^i$ есть объединение конфигураций.

Пример управляющих связей в приемлемой блок-схеме ветви дан на рис.I. Здесь конфигурациями являются следующие последовательности гамаков:

$(b, a_1), (a_1, a_2), (a_2, a_3), (a_3, d)$;

$(b, a_1), (a_1, a_2), (a_2, a_3), (a_3, a_2), (a_2, a_3), (a_3, d)$;

$(b, a_1), (a_1, a_4), (a_4, d)$.

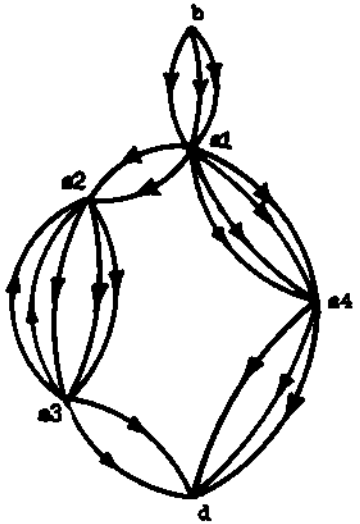

Рис. I

Таким образом, граф-схемы, соответствующие применимой блок-схеме ветви, не содержат недостижимых и тупиковых в смысле [4] вершин, а также вершин, принадлежащих множеству $M^i$ и в то же время не совпадающих ни с одной начальной или конечной вершиной какого-либо из гамаков. Такие вершины будем называть вершинами зависания.

В [7] было введено понятие допустимого взаимодействия, при котором выход хотя бы одной ветви на оператор взаимодействия, означает, что и остальные ветви р-программы должны выйти на оператор, задающий то же самое взаимодействие. Исходя из определения конфигурации, можно сказать, что множества $S^i$ тогда и только тогда состоят из вершин, соответствующих операторам, задающим допустимые взаимодействия, когда применимые блок-схемы ветвей можно представить одной и той же совокупностью конфигураций.

Блок-схема р-программы называется п р и м е н и м о й, если применима блок-схема каждой из ее ветвей и $\forall j (j = 1,2,\ldots,m)$ множество операторов, соответствующих множеству $S^i_j = \{a^i_j | i = 1,2,\ldots,l\}$, задает допустимые взаимодействия.

Опишем алгоритм определения применимости блок-схемы ветви. Аналогично [6] на основе моделирования процесса движения по графу строится матрица достижимости, по значениям строк и столбцов которой определяются недостижимые и тупиковые вершины. В отличие от [6] в процессе построения выделяются вершины из множества $S^i$, а из вершин разветвления, соответствующих операторам ОУП, процесс движения организуется поочередно по обоим направлениям до появления вершин из множества $M^i$. Такой подход позволяет выявить вершины зависания и построить множество вершин из $S^i$ (достижимых из начальной вершины), по которому определяются допустимые взаимодействия.

Для работы алгоритма считаются заданными матрица смежности $k^i$-го порядка и $k^i$-мерный вектор C, единичные значения компо-

ментов которого указывают на операторы взаимодействия. Такая информация может быть получена по описанию исходной р-программы, содержащему в явном виде информацию об управляющих связях и параметрах операторов взаимодействий, например, аналогично [6], в комментариях.

Предполагается, что все строки $v_i$ $(i = 1,2,\ldots,k^i)$ матрицы достижимости вначале процедуры заполнены нулями.

Пусть

$h$ – номер шага;

$E^\lambda$ – множество уже достигнутых вершин;

$E^h$ – множество вершин, исходных на шаге $h$;

$E^{h+1}$ – множество вершин, смежных с вершинами из $E^h$ и не содержащихся в $E^\lambda$;

$S^\lambda$ – множество уже достигнутых вершин из $S^i$;

$S^{h+1}$ – множество вершин из $E^{h+1}$, соответствующих операторам взаимодействия;

$S^6$ – множество вершин из $S^i$, достигнутых из начальной вершины очередного гамака;

СТ1 – память для запоминания одной вершины;

СТ2 – память для запоминания вершин из $S^6$, организованная по принципу очереди (первый пришел – первый ушел);

$i$ – номер очередной заполняемой строки $v_i$ матрицы достижимости.

Алгоритм состоит из выполнения следующих указаний.

1. Занести $1 \to i$, $\emptyset \to S^\lambda$.

2. Занести $\emptyset \to E^\lambda \to S^6$, очистить памяти СТ1 и СТ2 ($0 \to$ СТ1 $\to$ СТ2).

3. Если $v_i \in S^i$, т.е. $v_i$ соответствует некоторому оператору взаимодействия $s$, то $s \to E^h$, если $s$ – преобразователь, иначе $s(A1)^{*)} \to E^h$, $s(A2)^{*)}$ запомнить в СТ1, если $s$ – распознаватель. Если $v_i \notin S^i$, то $v^i \to E^h$.

4. Сформировать множество $E^{h+1}$:

а) выполнить процедуру (используя матрицу смежности) выбора вершин, смежных с вершинами из $E^h$ ($\Gamma E^h \to E$);

б) выполнить $E \setminus (E \cap E^\lambda) \to E^{h+1}$.

---

*) Через $s(A1)$ обозначаем переход распознавателя по одному направлению, а через $s(A2)$ – по другому, т.е. смежной вершиной для $E^h$ будет соответственно либо $A1$, либо $A2$.

5. Если $E^{h+1} = \emptyset$, то перейти к п. I3.

6. Выполнить $E^{\lambda} \cup E^{h+1} \to E^{\lambda}$ .

7. Заслать единицы в $\tau_i$-ю строку матрицы достижимости в позиции, соответствующие вершинам из $E^{h+1}$ .

8. Выполнить (используя вектор C ) процедуру выбора из множества $E^{h+1}$ вершин, соответствующих операторам взаимодействия ($\Gamma E^{h+1} \to S^{h+1}$ ).

9. Выполнить $S^{\delta} \cup S^{h+1} \to S^{\delta}$ . Если $i = I$, то выполнить $S^{\lambda} \cup (S^{\delta} \setminus S^{\delta} \cap S^{\lambda}) \to S^{\lambda}$, иначе перейти к п. I0.

I0. Выполнить $E^{h+1} \setminus S^{h+1} \to E^{h}$ .

II. Если $E^{h} = \emptyset$ , то перейти к п. I3.

I2. Перейти к п. 4.

I3. Вершины из $S^{\delta}$ занести в порядке очереди в СТ2. Если в $S^{\delta}$ содержится одна вершина или ни одной, то перейти к п. I4, иначе к п. I6.

I4. Выполнить $\emptyset \to S^{\delta}$. Если память СТI непуста, то выбрать ее содержимое и заслать в $E^{h}$ , перейти к п. 4, иначе к п.I5.

I5. Если память СТ2 непуста, то выбрать ее очередной элемент $a$ . Если $a$ – преобразователь, то $a \to E^{h}$ и перейти к п.4; если $a$ - распознаватель, то $a(A2)$ поместить в СТI, $a(AI) \to E^{h}$ и перейти к п.4. Если память СТ2 пуста, то к п.I7.

I6. Выдать сообщение о наличии вершин зависания. Для продолжения выполнения алгоритма перейти к п. I4.

I7. Заполнение очередной строки матрицы достижимости закончено. Если не все строки заполнены, то $(i+1) \to i$ и перейти к п.2; иначе к п.I8.

I8. Конец.

Аналогично [6], недостижимые и тупиковые вершины определяются нулевыми значениями компонентов начальной строки и конечного столбца матрицы достижимости. Информация о вершинах зависания получается в процессе выполнения алгоритма. При отсутствии этих трех типов вершин граф-схема представляет приемлемую блок-схему ветви, а для р-программы с идентичными ветвями и приемлемую блок-схему р-программы.

Если множества $S^{\lambda}$, построенные для каждой ветви, совпадают по порядку следования элементов и блок-схема каждой ветви р-программы с неидентичными ветвями приемлема, то и ее блок-схема будет приемлема.

ПРИМЕР. На рис.2 представлена блок-схема, а на рис.3 – вектор C. Матрица смежности не приведена, так как для определения смежных вершин достаточно рис.2. Для вершин-распознавателей из $S^1$ за a(A1) берем направление с (+), за a(A2) – направление с (–).

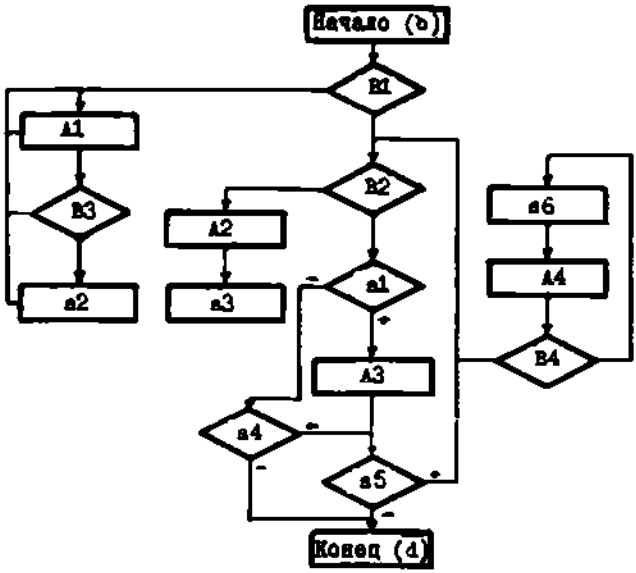


Рис. 2

| b | A1 | A2 | A3 | A4 | B1 | B2 | B3 | B4 | a1 | a2 | a3 | a4 | a5 | a6 | d |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | I | I | I | I | I | I | 0 |

Рис. 3

В результате выполнения пп. I–I8 получаем, что к вершинам зависания относятся a2, a3, a1. Из полученной матрицы достижимости (рис.4) имеем, что к недостижимым вершинам относятся a6, A4 и B4, а к тупиковым – A1, B3, a2, A2, a3 .

| | b | A1 | A2 | A3 | A4 | B1 | B2 | B3 | B4 | a1 | a2 | a3 | a4 | a5 | a6 | d |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| b | | I | I | I | | I | I | I | | I | I | I | I | I | | I |
| A1 | | I | | | | | | I | | | I | | | | | |
| A2 | | | | | | | | | | | | I | | | | |
| A3 | | | I | I | | | I | | | I | | I | I | I | | I |
| A4 | | | I | I | I | | I | | I | I | | I | I | I | I | I |
| B1 | | I | I | I | | | I | I | | I | I | I | I | I | | I |
| B2 | | | I | I | | | I | | | I | | I | I | I | | I |
| B3 | | I | | | | | | I | | | I | | | | | |
| B4 | | | I | I | I | | I | | I | I | | I | I | I | I | I |
| a1 | | | I | I | | | I | | | I | | I | I | I | | I |
| a2 | | I | | | | | | I | | | I | | | | | |
| a3 | | | | | | | | | | | | | | | | |
| a4 | | | I | I | | | I | | | I | | I | I | I | | I |
| a5 | | | I | I | | | I | | | I | | I | I | I | | I |
| a6 | | | I | I | I | | I | | I | I | | I | I | I | I | I |
| d | | | | | | | | | | | | | | | | |

Рис. 4

Современные оптимизирующие трансляторы, получающие для целей оптимизации информацию о динамике исполнения программы, способны выявлять недостижимые и тупиковые операторы [5]. Такие трансляторы можно использовать и для анализа р-программ, реализовав в них процедуры построения множества $\mathcal{A}^{\lambda}$ и выявления вершин зацикливания.

## Литература


1. КАРПИНОВ Э.А., КОСАРЕВ Ю.Г. Однородные универсальные вычислительные системы высокой производительности. -Новосибирск: Наука, 1966. -с.234-292.

2. МИРЕНКОВ Н.Н. Системное параллельное программирование. - Новосибирск, Б.и., 1978. - 36 с.-(Препринт/ОВС-05).

3. КЕРБЕЛЬ В.Г., КОЛОСОВА Ю.И., КОРНЕЕВ В.Д., МИРЕНКОВ Н.Н., ЩЕРБАКОВ Е.В. Средства программирования системы МИНИМАКС. - Вычислительные системы. Вопросы теории и построения вычислительных систем. 1974, вып.60. с.143-152.

4. КАРП Р.М. Заметка о приложении теории графов для ЦВМ. - Кибернетический сборник, 1962, вып. 4, с.123-134.

5. КАСЬЯНОВ В.Н. Практический подход к оптимизации программ. -Новосибирск. Б.м. 1978. - 43 с.-(Препринт/ВЦ СО АН СССР; 135).

6. ЛИПАЕВ В.В., ФИЛЛОНСКИЙ Л.А., ШНЕЙДЕР Б.Н. Отладка систем управляющих алгоритмов.-М.: Сов.радио, 1974. - с.142-151.

7. КОЛОСОВА О.И., ШИРЕНКОВ Н.Н. Исследование взаимодействий параллельных процессов. -Вычислительные системы. 1973, вып. 57. с. 115-124.

Поступила в ред.-изд.отд.
23 ноября 1978 года