АРХИТЕКТУРА И МАТЕМАТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ
(Вычислительные системы)
1984 год Выпуск 104

УДК 681.322

РАСПОЗНАВАНИЕ НЕЗАВЕРШАЕМОСТИ АСИНХРОННОЙ ИНТЕРПРЕТАЦИИ ПАРАЛЛЕЛЬНОЙ МИКРОПРОГРАММЫ

С.М.Ачасова

Важными условиями корректности параллельных микропрограмм являются детерминированность и конечная продолжительность работы микропрограммы для любых исходных данных. Детерминированность асинхронной интерпретации параллельных микропрограмм изучалась в [1, 2]. В данной работе исследуются условия возникновения бесконечных (бесконечно повторяющихся) вычислений в асинхронных интерпретациях параллельных микропрограмм. Формулируются необходимые и достаточные условия того, что для любого массива данных, размера не более, чем N, микропрограмма не породит бесконечные вычисления. Предлагается алгоритм проверки на завершаемость асинхронной интерпретации параллельной микропропрограммы.

I. Используемые понятия

Приведем кратко перечень понятий, связанных с параллельными микропрограммами и их интерпретациями; более подробно эти понятия даны в [1,2,3,гл.2].

Массив обрабатываемых данных представляется в виде множества пар $\langle a,m \rangle$, где a – данное из алфавита A, m – место данного в массиве $M = \{m\}$. Здесь место представляется вектором координат в целочисленной n-мерной решетке, $m = x_1,\ldots, x_n$. Пара $\langle a,m\rangle$ называется клеткой, имеющей имя m и состояние a. Множество $\{\langle a,m \rangle\} \subseteq A \times M$ называется клеточным, если в нем нет клеток с одинаковыми именами. Конечное клеточное множество $W = \{\langle a_i,m_i\rangle\}$, $i = \overline{0,p}$, называется словом. Слово W можно представить двумя проекциями $Pr_1(W) = \{a_0,\ldots,a_p\}$ и $Pr_2(W) = \{m_0,\ldots, m_p\}$. По проекциям, элементы которых упорядочены таким образом, что имя клетки

и ее состояние имеют одинаковые номера в своих проекциях, можно восстановить слово W.

Множество слов $S = \{W_j\}$, имеющих одинаковые первые проекции и вторые вида $Pr_2(W_j) = \{\varphi_0(m_j), \ldots, \varphi_p(m_j)\}$, называется конфигурацией и записывается в виде $S(m) = \{\langle a_0, \varphi_0(m)\rangle, \ldots, \langle a_p, \varphi_p(m)\rangle\}$. Придавая конкретное значение аргументу, $m=m_t$, получаем слово $W_t = \{\langle a_0, \varphi_0(m_t)\rangle, \ldots, \langle a_p, \varphi_p(m_t)\rangle\}$ - элемент конфигурации. Рассматриваем только такие конфигурации, в которых используются функции сдвига на константу, $\varphi_i(m) = m + C$, C - n-мерный вектор с целочисленными компонентами.

Выражение θ: $S_1(m) * S_2(m) \to S_3(m)$ называется микрокомандой. Левая часть микрокоманды составлена из двух конфигураций: базы $S_1 = \{\langle a_0, m\rangle, \ldots, \langle a_p, \varphi_p(m)\rangle\}$ и контекста $S_2 = \{\langle b_0, \psi_0(m)\rangle, \ldots, \langle b_q, \psi_q(m)\rangle\}$, все функции φ_i, ψ_j попарно различны, $S_1 * S_2 = \{\langle a_0, m\rangle, \ldots, \langle a_p, \varphi_p(m)\rangle, \langle b_0, \psi_0(m)\rangle, \ldots, \langle b_q, \psi_q(m)\rangle\}$. Правая часть микрокоманды $S_3 = \{\langle c_0, m\rangle, \ldots, \langle c_p, \varphi_p(m)\rangle\}$ имеет ту же самую вторую проекцию, что и база S_1.

Микрокоманда θ: $S_1 * S_2 \to S_3$ применима к слову W, если в нем есть клетка $\langle a_t, m_t\rangle$ такая, что $S_1(m_t) * S_2(m_t) \subseteq W$. В слове W могут оказаться несколько клеток, относительно которых применима микрокоманда θ. Применение микрокоманды θ для конкретной клетки m_t называется микрооперацией и обозначается $\theta(m_t)$. Выполнение микрооперации состоит в изъятии из W слова $W_t = S_1(m_t)$ и присоединении к W слова $W'_t = S_3(m_t)$.

Конфигурации $S_i(m)$ и $S_j(m)$ пересекаются, если найдутся слова $S_i(m_t)$ и $S_j(m_t+k)$, пересечение которых непусто $k = k_1 \ldots k_n$ - вектор с целочисленными компонентами. Микрокоманды θ_i и θ_j пересекаются, если пересекаются их левые части. Общими клетками слов $S_{i1}(m_t) * S_{i2}(m_t)$ и $S_{j1}(m_t+k) * S_{j2}(m_t+k)$ могут быть контекстные клетки той и другой микроопераций, или контекстные одной и базовые другой, или базовые той и другой. В соответствии с этим мы говорим, что микрокоманды пересекаются по контекстам, или контексту и базе, или по базам.

Множество микрокоманд $\{\theta_j\}$, $j = \overline{1,v}$, является микропрограммой Ψ. Микропрограмма Ψ применима к слову W, если к нему применима хотя бы одна микрокоманда $\theta_j \in \Psi$. Параллельная микропрограмма может интерпретироваться в синхронном и асинхронном режимах. Любая интерпретация является пошаговой процедурой. При этом шаг i синхронной интерпретации состоит в преобразовании слова W^{i-1} в

слово W^i путем выполнения одновременно всех применимых к W^{i-1} микрокоманд (всех микроопераций для каждой микрокоманды). На каждом шаге ординарной асинхронной интерпретации выполняется только одна микрооперация, любая из множества применимых к W^{i-1}, если это множество имеет более одного элемента, то результат шага i оказывается недетерминированным. В отличие от ординарной асинхронной можно рассматривать просто асинхронную интерпретацию, каждый шаг которой состоит в выполнении любого подмножества применимых микроопераций: от единственной, как при ординарной асинхронной, до всех применимых, как при синхронной интерпретации. В настоящей работе мы будем рассматривать только ординарную асинхронную интерпретацию и для краткости слово "ординарная" будем опускать.

Множество всех слов, которые могут быть получены из исходного W_0 путем применения микропрограммы Ψ, вместе с отношением следования слов, называется множеством вычислений, синхронных или асинхронных в зависимости от интерпретации. Любая последовательность слов называется вычислением. Граф отношения следования слов называется графом вычислений. Из-за возможной недетерминированности шага асинхронной интерпретации граф асинхронных вычислений при единственной начальной вершине, которой ставится в соответствие исходное слово W_0, может иметь несколько конечных вершин, соответствующих словам $W_{E_1}, W_{E_2}, \ldots, W_{ER}$, к каждому из которых неприменима ни одна микрокоманда.

2. Бесконечные вычисления. Незавершаемость

В асинхронных вычислениях по микропрограмме Ψ для исходного слова W_0 может возникнуть ситуация, в которой некоторое подмножество слов циклически повторяется и образует контур в графе вычислений.

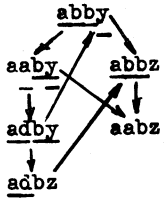

Рис.I

ПРИМЕР I. Граф асинхронных вычислений по микропрограмме

$$\Psi = \begin{cases} \theta_1: & abb \to -a- \\ \theta_2: & by \to --z \\ \theta_3: & ad \to -b \\ \theta_4: & a-y \to d-- \end{cases},$$

для исходного слова abby изображен на рис.I. (Прочерк в левой

части микрокоманды означает любое, безразличное для микрокоманды состояние клетки, прочерк в правой части означает, что соответствующая клетка не изменяет свое состояние в результате выполнения микрокоманды, т.е. является контекстной.) В графе имеется контур, соответствующий бесконечному вычислению (abby → aaby → adby)* ,
* - знак бесконечного повторения выражения в скобках.

***

В случае возникновения такой ситуации будем говорить, что микропрограмма Ψ способна порождать бесконечные вычисления и называть асинхронную интерпретацию микропрограммы Ψ не завершающейся для некоторого исходного слова или просто незавершающейся; вычисление такого рода также можно назвать незавершающимся. Это название связано с терминологией сетей Петри, а именно: граф асинхронных вычислений по микропрограмме Ψ для исходного слова W_0, в котором имеется контур, изоморфен графу достижимых маркирований сети Петри, моделирующей соответствующее множество вычислений и имеющей в себе в качестве фрагмента сильно незавершающуюся (strongly non-finishing) сеть [5]. В моделирующей множество асинхронных вычислений сети Петри каждой позиции соответствует клетка, переходу - микрооперация, начальное маркирование определяется исходным словом [2]. Сеть Петри N называется незавершающейся [5], если и только если существует начальное маркирование M_0 и последовательность срабатывания переходов v такие, что каждый переход сети оказывается в последовательности v бесконечно много раз. О сети, которая не является сильно незавершающейся, но содержит в себе такой фрагмент, говорят, что она обладает T-инвариантом (T от transition - переход). Определим это новое понятие. Если в сети N существует маркирование M такое, что после срабатывания из M переходов, образующих последовательность v , сеть вернется к тому же самому маркированию M , то вектор $\bar{v}$, образованный из v нижеописанным способом, называется T-инвариантом. Вектор $\bar{v}$ имеет число компонентов, равное числу переходов в сети N , каждый компонент есть целое число, равное числу вхождений соответствующего перехода в последовательность v .

ПРИМЕР 2. Сеть Петри, моделирующая множество вычислений из примера I, изображена на рис.2. Позиции сети имеют порядковые номера и содержательное обозначение как клетки. Граф достижимых маркирований этой сети дан на рис.3. Сеть имеет T-инвариант $\bar{v}$ = (I, 0,I,I), от маркирования (I,I,I,I,0,0,0) при срабатывании перехо -

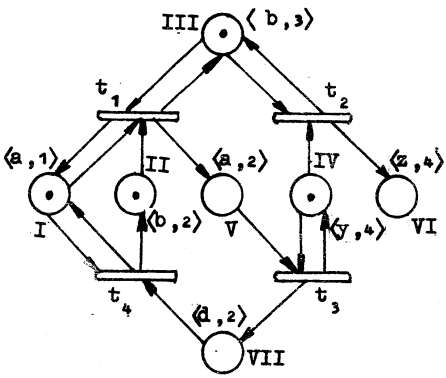

Рис.2

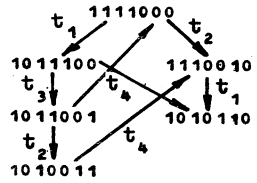

Рис.3

дов последовательности v = $= t_1, t_3, t_4$ сеть снова приходит к тому же самому маркированию.

***

Следуя [4], из незавершающихся вычислений будем выделять незаконные (invalid). Незаконным называется незавершающееся вычисление, в каждом слове которого применима одна и та же микрооперация, но ни на одном шаге вычисления эта микрооперация не выполняется. Иначе говоря, незаконным является незавершающееся вычисление, на каждом шаге которого применимы, по крайней мере, две микрооперации, которые пересекаются по контекстам или не пересекаются вовсе. При этом на каждом шаге одна микрооперация выполняется, не нарушая условия применимости другой, которая остается невыполненной, и эта невыполненная микрооперация - одна и та же на каждом шаге вычисления. В примере I вычисление $(abby \to aaby \to adby)^*$ является незаконным, так как в каждом его слове применима микрооперация $\theta_2(4)$: $\langle y,4 \rangle^* \langle b,3 \rangle \to \langle z,4 \rangle$, но ни на каком шаге вычисления она не выполняется.

В сети Петри, моделирующей множество асинхронных вычислений, среди которых есть хотя бы одно незаконное, имеется переход, возбужденный при каждом маркировании из некоторой замкнутой последовательности и никогда не срабатывающий. В сети Петри на рис.2 переход t_2 возбужден при любом маркировании из последовательности $(1111000 \to 1011100 \to 1011001)^*$, соответствующей незаконному вычислению $(abby \to aaby \to adby)^*$, этот переход соответствует микрооперации $\theta_2(4)$, из-за которой вычисление и является незаконным.

Незаконные вычисления не будем рассматривать как нарушающие корректность параллельной микропрограммы. Возникновение такого вы-

числения будем считать нереалистичным в том смысле, что приме - нимая микрооперация, которой ничто не мешает выполниться, обяза - тельно выполнится в конечное время. Другими словами,предполагаем, что при выполнении параллельной микропрограммы в асинхронном ре- жиме удовлетворяется условие конечной задержки для микроопераций.

3. Граф стимуляций

Задача распознавания незавершаемости асинхронной интерпрета- ции параллельной микропрограммы Ψ для любого исходного слова,раз- мера не более, чем N , может иметь две постановки: I) выяснить, существует ли хотя бы одно слово размера не более N в алфавите $A \times M$, для которого микропрограмма Ψ порождает незавершающееся законное вычисление, 2) выявить все слова размера не более N в алфавите $A \times M$, для которых микропрограмма Ψ порождает незавер - шающиеся законные вычисления. Размер N слова определяется разме- ром n-мерной решетки, в которой размещен массив обрабатываемых данных, множество имен M определяется этой же решеткой. Задача распознавания незавершаемости и в той, и в другой постановках ре- шается с помощью графа стимуляций. Сначала неформально о построе- нии такого графа. Левую часть некоторой микрокоманды из Ψ берем в качестве первоначального фрагмента исходного слова. Взятому фраг- менту ставим в соответствие вершину графа. Микрокоманда выполня - ется, получившемуся слову ставится в соответствие другая вершина графа, вершины связываются дугой. Если к получившемуся слову при- менима какая-либо микрокоманда, то поступаем как в предыдущей си- туации (пока похоже на построение графа асинхронных вычислений). Если к слову не применима ни одна микрокоманда, то добавляем к не- му новые клетки так, чтобы оказалась применимой некоторая микро - команда из Ψ , иначе говоря, стимулируем микрокоманду (здесь по- явилась особенность построения графа стимуляций). Если возможны несколько вариантов расширения слова для стимуляции в каждом ва- рианте своей микрокоманды, то для каждого из этих вариантов заво- дится вершина графа. Так продолжается построение графа, при этом либо выполняется микрокоманда и для результата заводится вершина графа, либо слово расширяется с соответствующим введением новых вершин. Процесс построения для первой постановки задачи заканчи - вается,как только в графе появляется контур, соответствующий за- конному вычислению (как образуется контур,будет ясно из формаль -

ного описания алгоритма построения графа). Граф стимуляций считается построенным, если дальнейшие стимуляции невозможны вообще или невозможны без такого увеличения слова, которое выводит его за размеры n-мерной решетки, или, наконец, стимуляции бессмысленны (например, в случае, если слово растет за счет приписывания одних и тех же фрагментов, т.е. возникает повторение некоторой ситуации, и т.п.).

Перейдем к формальному определению. Граф стимуляций - ориентированный связный граф, имеющий вершины двух типов: подстановочные и неподстановочные. Каждой вершине соответствует слово, подстановочной - слово, к которому применима хотя бы одна микрокоманда, неподстановочной - слово, к которому ни одна микрокоманда не применима. Неподстановочная вершина является местом стимуляции микрокоманд. Неподстановочные вершины имеют смежными только подстановочные вершины. Подстановочные вершины могут иметь смежными вершины и того, и другого типов.

Структура графа стимуляций станет ясной из алгоритма его построения. Исходной для построения графа $G(\theta_i)$ служит некоторая микрокоманда θ_i: $S_{i1} * S_{i2} \to S_{i3}$. Слову $S_{i1}(m_k) * S_{i2}(m_k)$ ставится в соответствие вершина W_0 графа стимуляций. Клеткам, служащим для расширения слов, приписываются имена $m_k + I$, где I - набор из n целых положительных или отрицательных чисел. Знак числа в наборе зависит от того, по какую сторону на соответствующей оси координат от клетки с именем m_k - "зародышевой" клетки, помещается новая клетка. Модули чисел в I определяются расстоянием новой клетки от "зародышевой" по определенной оси координат.

Пусть в процессе построения графа стимуляций $G(\theta_i)$ последней получена вершина W_j, $j = 0,1,\ldots$. В зависимости от типа этой вершины и типа ее предшественницы выполняются те или иные пункты алгоритма.

а) Если вершина W_j является неподстановочной, то выполняется п.1.

б) Если вершина W_j является подстановочной и ей предшествует подстановочная вершина, то выполняется п.2. В том случае, если сравнения п.2 закончились безрезультатно, выполняется п.3.

в) Если вершина W_j является подстановочной, а ее предшественница нет, то выполняется п.3.

Алгоритм построения графа стимуляций содержит следующие пункты.

I. Выполняются все возможные расширения слова W_j так, чтобы при каждом варианте расширения оказывалась применимой некоторая микрооперация и новое слово не вышло за пределы n-мерной решетки. При этом должно быть непусто пересечение слова W_j и левой части стимулируемой микрокоманды, иначе говоря, по меньшей мере одна клетка левой части стимулируемой микрооперации должна содержаться в слове W_j, а остальные клетки присоединяются к W_j. Для каждого варианта расширения W_{j+k}, $k = 1,2,\ldots$, слова W_j в графе заводится вершина W_{j+k}, в которую из вершины W_j проводится дуга. Если из слова W_j невозможно стимулировать ни одной микрокоманды, то вершина W_j остается без последователей.

2. Все слова W_r, $r = 0,1,\ldots, j-1$, соответствующие подстановочным вершинам, сравниваются с W_j. Если $W_r \subset W_j$, то W_r дополняется клетками, послужившими для расширения слов в неподстановочных вершинах, лежащих на пути, исходящем из W_r. При этом будет столько вариантов дополнений, сколько путей в графе, исходящих из W_r, содержат подстановочные вершины, следующие непосредственно за неподстановочными. Если слово W_j оказалось равным хотя бы одному из слов W_r или получившихся из W_r в результате дополнений, то из W_j в W_r проводится дуга. Если есть путь из W_r в W_j, то в графе стимуляций образуется контур.

3. Все применимые к слову W_j микрооперации выполняются в ординарном асинхронном режиме. В графе заводится столько новых вершин $W_{j+1}, W_{j+2},\ldots$, сколько слов получилось при этом из W_j. Во вновь заведенные вершины проводятся дуги из W_j.

ПРИМЕР 3. Построим граф стимуляций $G(\theta_1)$ для первой микрокоманды из микропрограммы

$$\Psi = \begin{cases} \theta_1: & ab \to c- , \\ \theta_2: & ec \to -e , \\ \theta_3: & cbh \to e--, \\ \theta_4: & ec \to -a . \end{cases}$$

У вершины W_0 графа $G(\theta_1)$, соответствующей слову $W_0 = \{\langle a,x\rangle, \langle b,x+1\rangle\}$, предшественницы нет, поэтому выполняем п.3 алгоритма. Выполняем микрокоманду θ_1, получаем вершину $W_1 = \{\langle c,x\rangle, \langle b,x+1\rangle\}$. Эта вершина является неподстановочной, выполняем п.I. Есть два варианта расширения слова W_1, первый вариант $W_2 =$

$= \{\langle e,x-1\rangle, \langle c,x\rangle, \langle b,x+1\rangle\}$ стимулирует микрокоманду θ_2, второй вариант $W_3 = \{\langle c,x\rangle, \langle b,x+1\rangle, \langle h,x+2\rangle\}$ стимулирует микрокоманду θ_3. Вершины W_2 и W_3 являются подстановочными, им предшествует неподстановочная вершина, значит, в том и другом случаях выполняется п.3. Получаем вершины $W_4 = \{\langle e,x-1\rangle, \langle e,x\rangle, \langle b,x+1\rangle\}$ и $W_5 = \{\langle e,x\rangle, \langle b,x+1\rangle, \langle h,x+2\rangle\}$. Далее для W_4 выполняем п.2, для W_5 – п.1. Пункт 2 для W_4 выполнился без положительного результата, выполненный далее п.3 дал вершину $W_6 = \{\langle e,x-1\rangle, \langle a,x\rangle, \langle b,x+1\rangle\}$. После расширения слова W_5 для стимуляции микрокоманды θ_4 получили вершину $W_7 = \{\langle e,x-1\rangle, \langle e,x\rangle, \langle b,x+1\rangle, \langle h,x+2\rangle\}$, для нее выполняем п.3, дающий новую вершину $W_8 = \{\langle e,x-1\rangle, \langle a,x\rangle, \langle b,x+1\rangle, \langle h,x+2\rangle\}$. Для W_6 и W_8 выполняем п.2, который дает соединение W_6 и W_8 с W_0. Граф стимуляций $G(\theta_1)$ изображен на рис.4. В нем имеются два контура. Оба незавершающихся вычисления, соответствующие этим контурам, являются законными.

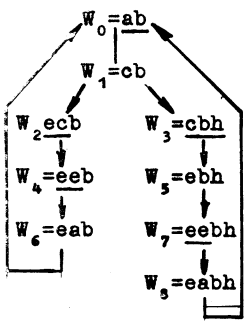

Рис.4

***

По графу стимуляций, имеющему контуры, восстанавливаются исходные слова, для которых микропрограмма Ψ порождает незавершающиеся законные вычисления. Таковыми будут все те слова, которые составляют контуры, соответствующие законным вычислениям, или имеют путь к любому слову контура. При необходимости слово дополняется клетками, послужившими для расширения слов в неподстановочных вершинах; это делается как описано в п.2 алгоритма. Так, из графа стимуляций на рис.4 можно выписать 6 таких слов: $\{\langle e,x-1\rangle, \langle a,x\rangle, \langle b,x+1\rangle\}$, $\{\langle e,x-1\rangle, \langle c,x\rangle, \langle b,x+1\rangle\}$, $\{\langle e,x-1\rangle, \langle e,x\rangle, \langle b,x+1\rangle\}$, $\{\langle e,x-1\rangle, \langle a,x\rangle, \langle b,x+1\rangle, \langle h,x+2\rangle\}$, $\{\langle e,x-1\rangle, \langle c,x\rangle, \langle h,x+1\rangle, \langle h,x+2\rangle\}$, $\{\langle e,x-1\rangle, \langle e,x\rangle, \langle b,x+1\rangle, \langle h,x+2\rangle\}$.

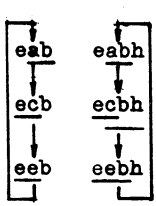

Рис.5.

Заметим, что граф стимуляций, по существу, объединяет в себе графы вычислений для нескольких исходных слов; такими являются слова, образованные на W_0 путем дополнения его клетками, расширившими слова в неподстановочных вершинах. Так, из графа сти –

муляций $G(\theta_1)$, данного на рис.4, можно выделить два графа вычислений для исходных слов $\{\langle e,x-1\rangle, \langle a,x\rangle, \langle b,x+1\rangle\}$ и $\{\langle e,x-1\rangle, \langle a,x\rangle, \langle b,x+1\rangle, \langle h,x+2\rangle\}$; эти графы изображены на рис.5.

4. Условия завершаемости асинхронной интерпретации параллельной микропрограммы

Необходимое и достаточное условие для распознавания незавершаемости асинхронной интерпретации параллельной микропрограммы в клеточном множестве размера не более N устанавливает

ТЕОРЕМА I. Для того чтобы асинхронная интерпретация параллельной микропрограммы Ψ была завершающейся в любом клеточном множестве размера не более N, необходимо и достаточно, чтобы ни в одном графе стимуляции $G(\theta_i)$, $\theta_i \in \Psi$, $i=\overline{1,|\Psi|}$, не было ни одного контура, соответствующего законному незавершающемуся вычислению.

ДОКАЗАТЕЛЬСТВО. Необходимость очевидна. Достаточность следует из способа построения графа стимуляций, при котором всевозможные расширения слов, начиная с левой части некоторой микрокоманды, рассматриваются для стимуляций микрокоманд из Ψ. Это значит, что нет слов, не представленных в графах стимуляций $G(\theta_i)$, $i=\overline{1,|\Psi|}$, таких, для которых выполнение некоторой последовательности микрокоманд образовывало бы незавершающееся вычисление.

Роль графа стимуляций при распознавании незавершаемости асинхронной интерпретации параллельной микропрограммы может исполнить сеть Петри, построенная также по принципу стимуляций микрокоманд. Сеть Петри, построенную по этому принципу, начиная с левой части микрокоманды θ_i, будем обозначать через $N(\theta_i)$. Позиции, соответствующие клеткам левой части микрокоманды θ_i, а также тем клеткам, которые добавляются в строящуюся сеть в согласии с расширением слов в графе стимуляций, имеют метки в начальном маркировании сети $N(\theta_i)$. Начальное маркирование такой сети не будет соответствовать одному исходному слову, а будет представлять множество слов. И сама сеть $N(\theta_i)$ будет моделировать сразу несколько множеств вычислений в соответствии с тем, что граф стимуляций объединяет в себе несколько графов вычислений для разных исходных

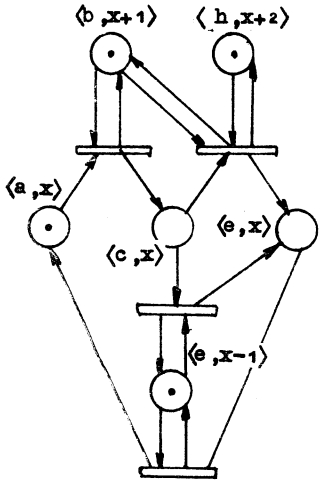

Рис.6

слов. Сеть Петри, моделирующая процесс стимуляций, изображенный графом на рис.4, дана на рис.6.

Сеть Петри $N(\theta_i)$ по объему информации, ее представляющей, может оказаться меньше графа стимуляций $G(\theta_i)$. Действительно, в графе стимуляций каждой вершине соответствует слово, многие клетки от слова к слову повторяются и многократно фиксируются, в то время как в сети Петри каждой клетке соответствует только одна вершина графа – позиция.

При использовании сети Петри вместо графа стимуляций для решения задачи распознавания незавершаемости асинхронной интерпретации параллельной микропрограммы Ψ теорема I должна быть перефразирована следующим образом.

ТЕОРЕМА 2. Для того, чтобы асинхронная интерпретация параллельной микропрограммы Ψ была завершающейся в любом клеточном множестве размера не более N, необходимо и достаточно чтобы ни одна сеть Петри $N(\theta_i)$, $\theta_i \in \Psi$, $i = \overline{1,|\Psi|}$, не обладала Т-инвариантом, соответствующим законному незавершающемуся вычислению.

5. Предварительный анализ микропрограммы

Прежде чем проверять, является ли завершающейся асинхронная интерпретация некоторой микропрограммы Ψ, используя для этого теорему I или 2, можно провести несложный предварительный анализ микропрограммы с тем, чтобы выяснить, выполняется ли необходимое условие незавершаемости асинхронной интерпретации микропрограммы. Это условие связано с отношением возможного следования микрокоманд при выполнении микропрограммы.

Микрокоманда Θ_j может следовать за Θ_i, $\Theta_i \to \Theta_j$, если правая часть Θ_i вместе с контекстом пересекается с левой частью Θ_j, т.е. $S_{i3}(m_t) * S_{i2}(m_t) \cap S_{j1}(m_t+k) * S_{j2}(m_t+k) \neq \emptyset$. Очевидно, что введенное отношение не является транзитивным.

Отношение следования делит множество микрокоманд на некоторое число подмножеств, для каждого из которых граф отношения следования является связным. Каждое подмножество является замкнутым для провокаций, т.е. если начать стимуляции от некоторой микрокоманды из подмножества, то можно будет стимулировать только те микрокоманды, которые принадлежат тому же подмножеству. На рис. 7 изображен граф отношения следования микрокоманд микропрограммы из примера 3. Здесь все микрокоманды оказались связанными отношением следования. Сформулируем необходимое условие незавершаемости асинхронной интерпретации параллельной микропрограммы Ψ, справедливость его ясна без доказательства.

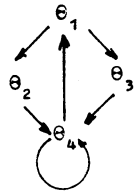

Рис.7

Необходимое условие незавершаемости. Для того чтобы асинхронная интерпретация параллельной микропрограммы Ψ могла быть незавершающейся, необходимо, чтобы в графе отношения следования микрокоманд из Ψ был хотя бы один контур.

После того, как построен граф отношения следования микрокоманд и он оказался с контуром, что означает, что нужно продолжить исследование микропрограммы с целью обнаружить ее незавершаемость, можно все же не торопиться строить графы стимуляций или сети Петри, а продолжить анализ микропрограммы в несколько ином направлении, мы назовем его "количественным". Целью "количественного" анализа является попытка исключить из множества микрокоманд те из них, которые нецелесообразно стимулировать, так как применение их не может привести к незавершающимся вычислениям. Вот несколько таких типов микрокоманд.

I. В правой части микрокоманды есть некоторая буква алфавита состояний, которой нет в левых частях остальных микрокоманд.

2. Выполнение микрокоманды уменьшает количество вхождений некоторой буквы алфавита в клеточное множество, и нет микрокоманд, увеличивающих количество вхождений той же буквы.

3. Микрокоманда производит сдвиг букв в массиве данных по некоторым координатам, и нет микрокоманд, которые бы сдвигали те же буквы в противоположном направлении, и т.п.

Предварительный "количественный" анализ продемонстрируем на примере микрокоманды сложения многих двоичных чисел.

ПРИМЕР 4. Граф отношения следования микрокоманд в программе

$$\Psi = \left\{ \begin{array}{l} \theta_1 : \begin{array}{cc} I & 0 \\ 0I & \rightarrow I0 \\ 00 & \end{array} \\ \theta_2 : \begin{array}{cc} 0 & I \\ I & \rightarrow 0 \\ 0 & \end{array} \end{array} \right.$$

имеет вид контура $\theta_1 \rightleftarrows \theta_2$. Однако из количественных соображений, а именно: θ_1 две единицы заменяет одной, а θ_2 не производит новых единиц, θ_1 следует исключить из множества стимулируемых микрокоманд. Выполнение микрокоманды θ_2 также не приведет к незавершающемуся вычислению, так как она осуществляет только сдвиг единицы в верхнюю часть массива данных. Таким образом, для этой простой микропрограммы сразу делаем вывод: ни для каких исходных данных она не может породить незавершающиеся вычисления.

6. Алгоритм распознавания незавершаемости асинхронной интерпретации параллельной микропрограммы

Беря за основу все изложенное выше, предлагаем алгоритм распознавания незавершаемости асинхронной интерпретации параллельной микропрограммы.

I. Проводится предварительный анализ микропрограммы. Если в графе отношения следования микрокоманд нет ни одного контура, то асинхронная интерпретация микропрограммы не будет незавершающейся ни для какого исходного слова. Если в результате "количественного" анализа микропрограммы после удаления некоторых микрокоманд граф отношения следования оставшихся микрокоманд не имеет контуров, то асинхронная интерпретация микропрограммы является завершающейся для любого исходного слова. Выполнение алгоритма заканчивается. Если граф отношения следования микрокоманд имеет хотя бы один контур, то продолжается выполнение алгоритма.

2. При распознавании незавершаемости в узкой постановке - может или не может быть асинхронная интерпретация незавершающейся, для каждой микрокоманды строится граф стимуляций. Если в графе появляется контур, соответствующий законному незавершающемуся вычислению, то выполнение алгоритма прекращается и делается вывод: асинхронная интерпретация микропрограммы может быть незавершающей-

ся для некоторых исходных слов. Если ни один граф стимуляций не имеет контуров или имеющиеся контуры соответствуют незаконным вычислениям, то выполнение алгоритма заканчивается с выводом: асинхронная интерпретация микропрограммы не порождает незавершающихся вычислений для любых слов размера не более заданного N .

3. При необходимости выявления всех слов, для которых асинхронные вычисления по микропрограмме являются законными незавершающимися, для каждой микрокоманды строится граф стимуляций до своего завершения. По графам, в которых имеются контуры, восстанавливаются с учетом расширения слов для стимуляций все слова, принадлежащие контурам, соответствующим законным незавершающимся вычислениям.

Литература

1. БАНДМАН О.Л. Асинхронная интерпретация параллельной микропрограммы. - Кибернетика, 1984, № 2, с.14-20.

2. АЧАСОВА С.М. Анализ асинхронной интерпретации параллельных микропрограмм. - В кн.: Однородные вычислительные системы из микро-ЭВМ (Вычислительные системы, вып.97). Новосибирск, 1983, с.28-52.

3. Методы параллельного микропрограммирования/Под ред. Бандман О.Л. - Новосибирск: Наука, 1981. - 181 с.

4. KWONG Y.S. On the Absence of Livelocks in Parallel Programs. - In: Lect.Notes in Comp.Sci., v.70, Springer-Verlag, 1979, p.172-190.

5. MEMMI G., ROUCAIROL G. Linear Algebra in Net Theory. -In: Lect.Notes in Comp.Sci., v.84, Springer-Verlag, 1979, p.213-223.

Поступила в ред.-изд.отд.
5 сентября 1984 года