УДК 681.142.1.01

ОБ ОПТИМАЛЬНОМ СОВМЕЩЕНИИ ОПЕРАЦИЙ В ОПЕРАТОРНЫХ СХЕМАХ ПРИ РАСПРЕДЕЛЕНИИ ПАМЯТИ И УСТРОЙСТВ

А.С. Никитин
(Киев)

Дается алгоритм минимизации числа тактов, за которое выполняется операторная схема, в классе следующих эквивалентных преобразований операторных схем: П1) замена адресов в адресных частях операций, П2) перестановка операций из одного оператора в другой и П3) замена устройств, на которых выполняются операции. Операторные схемы описывают первый уровень параллельной работы устройств – совмещение.

Пусть заданы множества адресов $B = \{b\}$ и устройств $K = \{k\}$.

Оператором A_i называется множество операций $A_i = \{x\}$. В операцию $x = \langle k_x ; B_x \rangle$ входят устройство $k_x \in K$, на котором эта операция выполняется, и адресная часть $B_x \subset B$, представляющая последовательность адресов, участвующих в работе k_x. При этом $B_x = B_{x1} * B_{x2}$ и $b_{x2s} \neq b_{x2j}$ при $s \neq j$, где B_{i1} – используемые, B_{i2} – результативные адреса.

С каждой операцией x связана продолжительность $t(k_x, x)$ ее выполнения на устройстве k_x, а с каждым используемым или результативным местом $\langle x; s \rangle$ в B_x – временные про-

межутки обращений к адресу b_{xs} при работе k_x. $d(x) \subset K$ – допустимое множество устройств для выполнения x. Для пары устройств задается совместимость, т.е. возможность их одновременной работы.

Линейной операторной схемой над памятью B и устройствами K называется конечная последовательность операторов $S(B,K) = A_0, A_1, \dots, A_g$ с выделенными начальными и результативными адресами схемы, причем для каждой операции x схемы выполняются следующие условия:

I) в промежутках обращения устройства k_x к своему любому результативному адресу b_{xs} другие устройства не могут обращаться к этому же адресу;

2) одновременно могут работать только совместимые устройства.

Пусть τ – длительность такта, т.е. время между началами выполнения соседних операторов в $S(B,K)$. Тогда схема выполняется за $g+1$ такта или за $\tau(g+1)$ единиц времени. На i-ом такте одновременно начинают выполняться все операции из A_i. Выполнение операции x длится непрерывно $t(k_x, x)$ единиц времени, в течение которого преобразуется содержимое результативных адресов в Bx в соответствии с временными промежутками обращения к ним.

Результирующим местом в $S(B,K)$ для используемого места $\langle x;s\rangle$ ($x \in A_i$) называется такое результативное место $\langle y;l\rangle$ ($y \in A_j$, $i > j$) что $b_{xs} \equiv b_{yl}$ и в промежутке между концом вырабатывания содержимого адреса b_{yl} в y и началом его использования в x другие операции не изменяют это содержимое

Строится канонический граф $G_S(X, U)$ схемы $S(B,K)$, содержащий необходимую информацию для оптимизации схемы, каждой вершине $x \in X$ которого взаимно однозначно сопоставлена операция $x \in \bigcup\limits_{i=0}^{g} A_i$ и из y в x идет дуга $u = \overrightarrow{yx}$, если $\langle y;l\rangle$ является результирующим местом для $\langle x;s\rangle$. При этом дуга u отмечается множеством последовательностей

$$F(u) = \bigcup_{l,s} \langle b_{yl} ; l; s\rangle ,$$

$$b_{yl} \in B_{y2}, \quad b_{xs} \in B_{x1} .$$

Рассматриваются три типа эквивалентных преобразований линейных операторных схем.

П1. Замена адресов b_{xs} на новые адреса $b^1_{xs} = W(x,s)$ в адресных частях операций схемы, при этом для новой последовательности должны выполняться следующие условия:

У1) последовательность является операторной схемой,

У2) для каждого используемого места в операциях схемы сохраняется результирующее для него место.

П2. Преобразование перестановки операций, при котором операция $x \in A_i$ удаляется из A_i и присоединяется к A_j, при этом для полученной последовательности должны выполняться условия У1 и У2.

П3. Преобразование замены устройств, при котором для выполнения операции x вместо k_x назначается новое допустимое устройство, при этом для полученной последовательности выполняются условия У1 и У2.

Эквивалентной реализацией схемы $S(B,K)$ на (B', Q) называется схема $S^*(B', Q)$, получаемая из $S(B,K)$ с помощью эквивалентных преобразований П1-П3 с использованием новых памяти B' и устройств Q.

ПОСТАНОВКА ЗАДАЧИ. В классе $\Sigma = \{S^*(B',Q)\}$ всех эквивалентных реализаций схемы $S(B,K)$ на (B',Q) найти такую схему $S^{**}(B',Q) \in \Sigma$, которая выполняется за минимальное число тактов.

Алгоритм оптимального совмещения операций, решающий поставленную задачу, использует канонический граф схемы и построен по схеме последовательного развития, анализа и отбора вариантов с использованием нижних границ числа тактов в схеме для предпочтительного развития вариантов.

Пусть уже построена подсхема $S_{i-1} = A_0, A_1, \ldots, A_{i-1}$ и пусть $E(S_{i-1})$ – выполнимое подмножество операций (оно находится по графу $G_s(X,U)$ и подсхема S_{i-1}), которые претендуют на вхождение в A_i, дающее подсхему $S_i = S_{i-1}, A_i$.

В схеме алгоритма оптимального совмещения операций используются в указанном порядке следующие правила: правила А1, А2 отсева выполнимых вершин (операций) из $E(S_{i-1})$ для получения $F_i \subset E(S_{i-1})$, не содержащего вершин, которые хотя и выполнимы, но не могут войти ни в одно A_i, не нарушая эквивалентности подсхемы S_i; правила В1, В2 развития из F_i вариантов для оператора A_i $(A_i \subset F_i)$; правила

CI, C2, C3, C4 для отсева подсхем $S_{ij} = S_{i-1}, A_{ij}$, полученных в результате развития вариантов; правило Д для предпочтения развития подсхем; правило Г для вычисления нижних границ числа тактов в схемах, развиваемых из данной подсхемы.

Эти правила имеют следующее содержание:

AI – отсев вершин из $E(S_{i-1})$ из-за нарушения условия УI в любом продолжении подсхемы S_{i-1}, если хотя бы одна из отсеиваемых операций включена в A_i;

A2 – отсев вершин из $E(S_{i-1})$ из-за занятости или несовместимости всех допустимых для нее устройств;

BI – развитие вариантов перебором подмножеств вершин в P_i;

B2 – развитие вариантов $\{A_{ij}\}$ оператора A_i заменой устройств из K на допустимые устройства из Q в операциях подмножеств, полученных правилом BI;

CI – отсев вариантов S_{ij} из-за нарушения условия УI;

C2 – отсев вариантов S_{ij} из-за нарушения условия У2;

C3 – отсев вариантов S_{ij} из-за ограниченности объема памяти B';

C4 – отсев вариантов S_{ij} из-за их неоптимальности;

Д – дальнейшее развитие того варианта, который с наибольшей вероятностью содержит оптимальный вариант (здесь выбирается вариант с минимальной нижней границей числа тактов).

В правиле C3 используется следующая оценка объема памяти B', при котором не существует эквивалентной замены адресов. Если в замкнутой линейной операторной схеме S(B,K) существует такой момент времени, для которого сумма числа активных и пассивных адресов в этот момент превосходит объем памяти B', то в S(B,K) нельзя сделать эквивалентную замену адресов из памяти B'

Это позволяет разделить преобразование ПI на два этапа: проверке существует ли эквивалентная замена адресов для данного B' (это делается правилом C3) и сама замена (это делается в конце работы алгоритма оптимального совмещения). Такое разделение ПI на два этапа существенно сокращает перебор вариантов.

Нижние границы числа тактов для подсхемы S_{ij} в правилах Г вычисляются по   и длине критического пути в подграфе графа $G_S(X,U)$.

Легко видеть, что алгоритм оптимального совмещения операций с помощью правил AI–Г осуществляет целенаправленные эквивалентные преобразования ПI – П3 исходной схемы $S(B,K)$, приводящие к схеме $S^{**}(B',Q)$, которая является решением поставленной задачи.

С помощью операторных схем можно описывать, например, микропрограммные автоматы и мультипрограммные системы с несколькими управляющими автоматами, разделяющими память и устройства операционного автомата и работающими параллельно. При этом нужно использовать следующее соответствие между их элементами:

регистры ЭЦВМ	–	память B,
цепи и блоки для выполнения операций	–	устройства K,
операции	–	операции X,
линии связи между регистрами и устройствами	–	допустимость адресов в адресных частях операций,
команды	–	операторы A_i,
алгоритм функционирования ЭЦВМ	–	операторная схема $S(B,K)$

С помощью алгоритма оптимального совмещения операций можно решать также следующие практически важные частные случаи поставленной задачи:

1) минимизация объема памяти B в $S(B,K)$ при преобразованиях ПI–П3;

2) рациональный выбор объема памяти (регистров) и устройств (сумматоров, сдвигателей и т.д.) с помощью критерия $F(B',Q,\grave{\tau}\ g)$;

3) минимизация времени работы мультипрограммной системы автоматов, содержащей несколько параллельно работающих управляющих автоматов, которые совместно используют память и устройства операционного автомата.