

УДК 681.142.1.01.

АЛГОРИТМИЧЕСКИЙ СИНТЕЗ СТРУКТУР МНОГОПРОЦЕССОРНЫХ ЦВМ И СИСТЕМ

И.В. Иловайский

Работа посвящена алгоритмизации этапа структурного синтеза цифровых систем (мы опираемся на известное деление процесса синтеза цифровых устройств на этапы структурного, логического и конструкторского синтеза).

В настоящее время известен ряд решений, позволяющих проводить алгоритмически этап логического синтеза. В частности, неклассические методы (не опирающиеся на теорию автоматов) используют в качестве исходного описания алгоритм выполнения системы команд, понимаемый как совокупность алгоритмов исполнения команд. Элементарными объектами алгоритма служат элементы будущей функциональной схемы [6].

В работах, относящихся к этапу структурного синтеза, т.е. к этапу получения алгоритма функционирования, большей частью рассматриваются вопросы определения числовых характеристик проектируемых ЦВМ [1,2].

Мы рассмотрим элементы классификации алгоритмов, соотношения между алгоритмами функционирования ЦВМ (элементарными объектами которых являются операции-команды или более крупные составляющие) и программами; синтез устройств, описываемых такими алгоритмами, по классу задач и переход от алгоритма функционирования к алгоритму выполнения системы команд. Тем самым мы дадим, по-видимому, первый в литературе вариант связанной методики структурного синтеза.

Мы будем опираться на введенные здесь же строгие описания

класса исходных и результирующих объектов. В частности, введем формальные определения для УЦВМ и вычислительной системы.

§ 1. Основные объекты и соотношения между ними

Программы и алгоритмы функционирования описываются формализмами двух видов - граф-схемами по Калужнину [3] с расширением свойств распознавателей и обобщенными граф-схемами.

ОПРЕДЕЛЕНИЕ 1. Граф-схема Γ есть четверка $\langle G, \mathcal{U}, \mathcal{R}, Z \rangle$, где Z - множество переменных (ячеек), на котором определены операторы из \mathcal{U} и распознаватели из \mathcal{R} , а G - отношение порядка на множестве $\mathcal{U} \cup \mathcal{R}$. Вершинам q_i с полустепенью исхода $\rho^+(q_i) > 1$ (p - вершины) сопоставлены распознаватели, вершинам с $\rho^+(q_i) = 1$ (q - вершины) - операторы. Есть входная вершина q_0 с полустепенью захода $\rho^-(q_0) = 0$ и выходная q_* с $\rho^+(q_*) = 0$.

Назовем правильным путь в G , начинающийся в q_0 и заканчивающийся в q_* . В G любая вершина лежит на правильном пути [4]. Поскольку определение допускает p -вершины с $\rho^+(p) > 2$, две смежных p -вершины p_1 и p_2 могут быть заменены одной, p_3 .

ОПРЕДЕЛЕНИЕ 2. Пусть в G никакие две p -вершины не смежны. Устраним в G все p -вершины с $\rho^-(p) > 1$ следующим образом. Пусть (рис. 1) часть графа G с такой p -вершиной. Заменим эту часть графа такой, как показана на рис. 2. Эта замена допускается эквивалентными преобразованиями схем программ

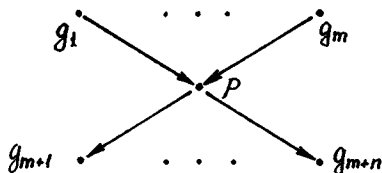


Рис. 1

[5]. В полученном графе сольем каждую пару $q_i \rightarrow p_i$ в одну вершину q_i^* . Процесс слияния будем продолжать, пока в графе не останется ни одной p -вершины. Оператор, соответствующий вершине (q, p_i) , будем именовать составным. Таким образом, мы полу-

чим граф, вершинам которого будут соответствовать операторы и составные операторы. Тройку $\langle G^*, \mathcal{U}^*, Z \rangle$ будем именовать обобщенной граф-схемой Γ^* . Смысл G^* и \mathcal{U}^* тот же, что в определении граф-схемы.

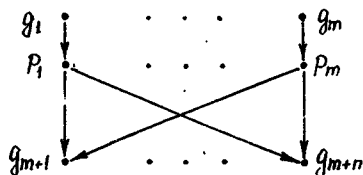


Рис. 2

ОПРЕДЕЛЕНИЕ 3. Вершинам графа J соотнесем операторы и распознаватели. В графе J дуги определятся следующим образом. Множество дуг $\{g_i \rightarrow g_a\}$ в J будет существовать тогда и только тогда, когда для того, чтобы оператор g_a мог выполняться, должны быть выполнены все операторы g_i и только они. Других дуг нет. (Например, g_a использует результаты работы операторов $\{g_i\}$, и только их). Четверку $\langle J, \mathcal{U}, \mathcal{R}, Z \rangle$ назовем граф-схемой J параллельной программы. Формализм вводится, поскольку параллельные программы (процессы) не могут быть однозначно заданы граф-схемами Γ .

ОПРЕДЕЛЕНИЕ 4. Множество всех попарно неэквивалентных операторов и распознавателей любой граф-схемы образует базу B . Каждый элемент базы имеет свое имя. Все эквивалентные операторы (распознаватели) имеют одно и то же имя с добавлением порядкового номера. При установлении эквивалентности у составных операторов принимается во внимание только q - часть имени. Так, q_a и q_{aP_i} эквивалентны.

ОПРЕДЕЛЕНИЕ 5. В алгоритме функционирования ЦВМ совместим операторы, которые выполняются на одном оборудовании. В граф-схемах Γ и Γ^* это поведет к совмещению соответствующих вершин и дуг графов G и G^* (мы не прибегаем к понятию мультиграфа). Вершина s_i , полученная слиянием вершин g_1, \dots, g_n , будет носить имя, представляющее собой набор имен слитых вершин (взятых в произвольном порядке).

Структурной схемой \mathcal{S} будем именовать тройку $\langle S, \mathcal{U}^s, Z \rangle$, где граф S определяется по графу G (либо G^*) следующим

образом. Множество X вершин графа G (либо G^*) разбивается на подмножества X_1, \dots, X_m , так что в каждое из X_i входят те и только те вершины g , операторы которых выполняются на одном оборудовании. Множеству X_i сопоставляется вершина S_i в графе S . В графе S существует дуга $S_i \rightarrow S_j$ тогда и только тогда, когда в графе G (либо G^*) существует хотя бы одна дуга $\alpha \rightarrow \beta$ ($\alpha \in X_i, \beta \in X_j$). S назовем структурой ЦВМ. \mathcal{U}^S имеет тот же смысл, что и \mathcal{U} , а его компоненты суть соответствующие объединения компонент из $\mathcal{U} \cup \mathcal{R}$ (либо \mathcal{U}^*).

Поскольку формально любой алгоритм можно рассматривать как описание работы некоторого устройства [6], граф-схемы J также могут быть преобразованы в соответствующие им структурные схемы.

ОПРЕДЕЛЕНИЕ 6. Структурную схему, у которой вершины S_i соответствуют множествам M_i граф-схемы таким, что в каждое M_i объединены те и только те операторы, которые эквивалентны, будем именовать свернутой граф-схемой. Множество вершин свернутой граф-схемы есть $B \cup \{g_0, g_*\}$.

Одной структурной схеме может соответствовать несколько граф-схем, но не наоборот.

ОПРЕДЕЛЕНИЕ 7. Универсальная цифровая вычислительная машина (УЦВМ) есть устройство, алгоритм функционирования которого описывается такой граф-схемой Γ_0^* , что какова бы ни была граф-схема Γ_i программы (причем $B_i \subseteq B_0^*$), для любого пути с корнем g_0 в G_i , в G_0^* существует эквивалентный ему путь (с корнем в g_0^*). В этом определении эквивалентные операторы не различаются. Предполагается, что база B_0^* функционально полна.

Мы могли бы ограничиться правильными путями, это гарантировало бы выполнение всех интуитивно правильных (конечных) программ, однако практика программирования имеет дело и с такими программами, которые могут иметь реализации, не являющиеся правильными путями.

Данное нами определение фиксирует в терминах теории схем программ известное обстоятельство, что любая программа (в кодах команд машины) реализуема на этой машине.

Сформулируем необходимые и достаточные условия, при которых обобщенная граф-схема Γ_0^* задает УЦВМ.

ТЕОРЕМА 1. Для того, чтобы граф-схема Γ_0^* описывала УЦВМ, необходимо, чтобы (1) в графе S_0^* свернутой граф-схемы существовали для всех $b_i \in B_0^*$ дуги $g_0 b_i$ и $b_i g_*$; (2) подграф графа S_0^* с множеством вершин B_0^* был полным.

ДОКАЗАТЕЛЬСТВО (1). В множестве всех правильных путей, реализуемых в G_0^* , будут пути, у которых вторым (соответственно, предпоследним) может быть любой элемент из B_0^* .

(2). Уберем из полного подграфа с множеством вершин B_0^* дугу $s_i s_j$. Тогда любой путь, в котором встретится пара ab ($a \in X_i$, $b \in X_j$), невозможен в G_0^* .

СЛЕДСТВИЕ. Условие (2) тем более должно выполняться для произвольной структурной схемы.

ТЕОРЕМА 2. Для того, чтобы граф-схема Γ_0^* описывала УЦВМ, достаточно, чтобы или (1) в графе G_0^* обобщенной граф-схемы Γ_0^* можно было бы достигнуть такой подграф G_i^* с множеством вершин $B_0^* \cup \{g_0, g_*\}$, что

(а) подграф с множеством вершин B_0^* полный; (б) для любой вершины $b \in B_0^*$ существуют дуги $g_0 b$ и $b g_*$; или

(2) граф G_0^* обобщенной граф-схемы был таков, что (а) для любой вершины g (кроме g_0 и g_*) существуют дуги $g_0 g$ и $g g_*$; (б) если дано разбиение множества вершин G_0^* на классы эквивалентности A_i по базе B_0^* (вершины g_0 и g_* не рассматриваются), то для любых двух (не обязательно разных) классов A_i и A_j существуют дуги $\alpha \rightarrow \beta$ и $\beta \rightarrow \alpha$ (где $\alpha \in A_i$, $\beta \in A_j$).

ДОКАЗАТЕЛЬСТВО (1). Какова бы ни была граф-схема Γ_i про-

граммы с базой $B_i \subseteq B_0^*$, любой путь в G_i имеет ему эквивалентный в выделенном нами подграфе G_i^* .

СЛЕДСТВИЕ. В простейшем случае граф-схема алгоритма функционирования УЦВМ выглядит следующим образом. В графе G_i^* (обобщенной граф-схемы) расщепим вершины, выделив из них p -часть. Мы получим граф-схему, граф G которой состоит из n частей (где $n = |B_i^*|$) вида (рис. 3) и отдельной части (рис. 4). Пользуясь эквивалентными

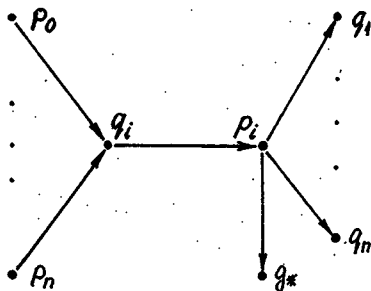


Рис. 3

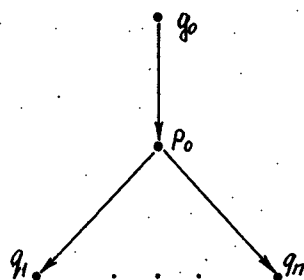


Рис. 4

преобразованиями схем программ [5], мы можем привести этот граф к виду (рис. 5), где распознаватель p осуществляет анализ команды, а оператор q_i - ее исполнение.

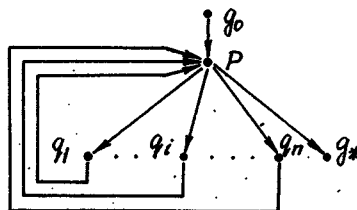


Рис. 5

Заметим, что граф-схема Γ_0 УЦВМ и граф-схема Γ_i программы — разные объекты, и тот факт, что среди операторов g_i граф-схемы программы есть команды условного перехода (ρ — вершины графа G_i), не делает их распознавателями в Γ_0 .

ДОКАЗАТЕЛЬСТВО (2). Действительно, пусть в некотором пути μ встретился оператор α . Тогда, какой бы оператор ни следовал в μ за α , условие (б) гарантирует существование соответствующей дуги $\alpha \rightarrow \beta$.

Условие (2) можно обобщить, введя в рассмотрение подпути длины 2, 3, 4 и т.д.

ОПРЕДЕЛЕНИЕ 8. Многопроцессорной будем считать УЦВМ, граф обобщенной граф-схемы которой содержит несобственный подграф, отвечающий одному из достаточных признаков, а структура структурной схемы которой не вырождается в одновершинный граф. Предполагается, что память общая и все устройства могут иметь к ней независимый доступ.

ОПРЕДЕЛЕНИЕ 9. УЦВМ будем именовать вычислительной системой (из отдельных машин), если в графе ее обобщенной граф-схемы можно выделить хотя бы два таких непересекающихся по вершинам подграфа, каждый из которых отвечает одному из достаточных признаков универсальности, причем соответствующие этим подграфам подструктуры структуры S также не имеют общих вершин.

ОПРЕДЕЛЕНИЕ 10. Однородной будем именовать систему, у которой как упомянутые подграфы, отвечающие признаку достаточности, так и соответствующие им подструктуры изоморфны (точнее, равны с точностью до нумерации эквивалентных вершин).

Предполагается, что память Z системы разбита на подпамяти в соответствии с разбиением ее графов на подграфы-машин.

§ 2. Синтез структурной схемы ЦВМ по множеству программ выбранного класса задач

Класс задач задается множеством граф-схем параллельных программ. Требуется спроектировать многопроцессорную УЦВМ, которая (а) выполняла бы каждую параллельную программу за минимальное время; (б) имела бы минимальную цену (или объем) оборудования.

Задача имеет один критерий оптимума - минимум цены оборудования. Требование (а) в действительности является ограничением. Мы затрудняемся решать задачу минимизации оборудования при произвольно заданном времени, поскольку временные оценки программ устанавливаются в общем случае экспериментально и связь их с графом J программы неясна. Фиксированное время работы программы будет гарантироваться тем, что мы не нарушим тех свойств графа J граф-схемы J , которые ответственны за временные характеристики программы.

Если каждый оператор (вершина) графа будет выполняться на отдельном оборудовании, время работы программы будет минимальным, а цена оборудования будет равна сумме цен реализаций каждой вершины.

Таким образом, алгоритм синтеза должен (а) не нарушать тех свойств графа J , которые ответственны за минимальное время работы; (б) гарантировать получение наименьшей цены оборудования.

Пусть цена реализации i -того оператора есть c_i . Пусть при реализации m операторов на одном устройстве (слиянии соответствующих вершин графа J в вершину графа S) цена вершины графа S определится как

$$c_s = \max \{c_i\} \quad i = 1, \dots, m \quad (1)$$

Это правило выражает тот факт, что если есть дорогое устройство, переделка его для реализации еще одной (недорогой функции) не удорожит его. Строго говоря, это неверно, и есть случаи, когда объединение двух устройств в одно увеличивает цену оборудования. При построении алгоритма синтеза мы будем опираться на функцию подсчета цены более широкого вида, а (1) используем для получения оценок.

Объединим в граф J все графы J_i граф-схем параллельных программ входными вершинами q_0 и выходными q_* . Построим структуру S^0 , соответствующую графу J , назначив каждой вершину на отдельное устройство. Тогда общая цена оборудования будет равна

$$c = \sum_{i=1}^{|X|} c_i \quad (2)$$

где X - множество вершин графа J .

Пусть некоторое множество вершин A слито в одну. Тогда мы это множество можем охарактеризовать ценой u_A , вычислимой по формуле (1). Тогда новое значение функционала найдется как

$$u' = u - \sum_{c_j \in A} c_j + u_A. \quad (3)$$

Заметим, что в силу (1) $u' \leq u$, поскольку

$$u_A \leq \sum_{c_j \in A} c_j. \quad (4)$$

Свойство (4) имеет место не только в случае формулы (1), но и для других принципов подсчета. Именно на (4) мы и будем опираться в дальнейшем.

Ясно, что в силу (4) любой процесс объединения вершин будет обладать следующими свойствами: (а) он будет конечен; (б) на каждом шаге функционал (цена) u будет невозрастать.

Введем ограничения, выделяющие вершины, которые объединять нельзя. Множество попарно нессливаемых вершин будем именовать множеством несовместности N .

Рассмотрим конечный граф \mathcal{D} (без контуров) всех путей, начинающихся в g_0 [7]. Очевидно, операторы, попавшие в один и тот же ярус графа \mathcal{D} , могут выполняться одновременно, если они не принадлежат ветвям поддерева, у которого корнем является p -вершина. Тогда во множество несовместности N_z^i выделим вершины z -того яруса \mathcal{D} за исключением тех, которые принадлежат поддереву с p -корнем. Если таких вершин m , имеем m множеств несовместности $N_z^i \cup \{g_1^p\}, \dots, N_z^i \cup \{g_m^p\}$. Если таких поддеревьев n , получим $n \cdot m$ таких множеств N_z^i . Эти множества просто получаются в процессе вычисления транзитивного замыкания; именно, вершины z -того яруса найдутся из матрицы A^z , являющейся z -той степенью матрицы смежности графа J . Из этой же матрицы найдутся и все p -потомки среди вершин z -того яруса. Мы не описываем эту процедуру, поскольку она сводится к известной технике работы с матрицами смежности графа [8].

Естественно, такое определение множеств N_z гарантирует лишь достаточные условия несовместности вершин. Однако попытки выписать полные условия резко усложняют задачу.

По множествам N_z образуем отношение N несовместности и его отрицание — отношение N^* совместности (возможности выполнения операторов на одном устройстве). Оба отношения симметричны и нетранзитивны. N^* , кроме того, рефлексивно. Поскольку слить m вершин можно только, если они все попарно совместимы, множество сливаемых вершин всегда есть множество вершин полного подграфа графа N^* . В силу изложенного, сформулированная нами задача оптимизации сводится к нахождению такого разбиения графа N^* на полные подграфы, что функционал

$$u = \sum_{j=1}^m u_j \quad (5)$$

будет минимален. Здесь u_j — цена j -го элемента разбиения (подсчитывается по (1)), m — количество элементов разбиения.

Нам не удалось свести эту задачу к известным или найти ее точное решение. Поэтому мы дадим эвристические методы направленного перебора, опирающиеся на соотношение (4).

Заметим, что исходным множеством вершин служит множество Y вершин графа \mathcal{D} , а не графа J . $|Y| > |X|$, соответственно $\sum_{j=1}^{|Y|} c_j > \sum_{j=1}^{|X|} c_j$. Дуги в графе S определяются по графу J либо \mathcal{D} путем дизъюнкции тех строк (и столбцов) матрицы смежности графа J (либо \mathcal{D}), которые соответствуют сливаемым вершинам.

Оценим минимальное значение функционала (5). В худшем случае, когда все N_z попарно не пересекаются, мы получим разбиение, состоящее не менее чем из $m = |N|_{\max}$ множеств, а цена не превысит величины $m \cdot \max_{c_i \in Y} \{c_i\}$. С другой стороны, цена не может быть меньше чем $\min_{c_i \in Y} u_z$, где u_z — цена множества N_z , $u_z = \sum c_i (c_i \in N_z)$. Таким образом,

$$|N|_{\max} \cdot \max_{c_i \in Y} \{c_i\} \geq u_{\min} \geq \min_z \left\{ \sum_{c_i \in N_z} c_i \right\}. \quad (6)$$

Мы допускаем нестрогость, отбрасывая имя вершин и ее цену.

1°. В графе N^* будем последовательно находить и изымать максимальные полные подграфы. Каждый такой подграф стягивается

в вершину графа S . Процедура нахождения максимального подграфа в N^* эквивалентна поиску максимального внутренне устойчивого множества в графе N . Алгоритм нахождения внутреннего устойчивого множества известен [8], [9].

Оценим сверху число шагов. Число внутренней устойчивости $\alpha(N)$ связано с хроматическим числом $\gamma(N)$ соотношением:

$$\alpha(N) \cdot \gamma(N) \geq |N_1| = |Y|.$$

Так как на каждом шаге находится и изымается хотя бы один подграф, $|N_{i+1}| = |N_i| - \alpha_i(N_i)$. На каждом i -м шаге $\alpha_i(N_i) \cdot \gamma_i(N_i) = |N_i| \cdot \alpha_i$, где $\alpha_i \geq 1$ — неизвестный коэффициент. Несложные выкладки дают

$$|N_n| = |N_1| \cdot \prod_{i=1}^{i=n-1} \left(1 - \frac{\alpha_i}{\gamma_i(N_i)}\right).$$

Предположив, что α и γ не зависят от номера шага i и выбраны так, что n максимально, получим

$$|N_n| = |N_1| \cdot \left(1 - \frac{\alpha}{\gamma}\right)^{n-1},$$

откуда, учитывая, что на последнем шаге могут остаться одновершинные максимально устойчивые множества $|N_n| = 1$, имеем

$$n = \left\lceil 1 + \frac{\lg |N_1|}{\lg \left(1 - \frac{\alpha}{\gamma}\right)} \right\rceil.$$

Очевидно, возрастание α поведет к уменьшению n , а возрастание γ — к возрастанию n , поэтому возьмем $\alpha = 1$, а γ заменим его оценкой сверху, $\rho_m + 1$, максимальной степенью вершины в графе [9].

Поскольку $\rho_m^1 \geq \rho_m^2 \geq \dots \geq \rho_m^n$, то γ на любом шаге не может превышать ρ_m^1 и предположение о независимости α и γ (при их надлежащем выборе) от номера шага оправдывается. Поэтому

$$n \leq \left\lceil 1 + \frac{\lg |N_1|}{\lg \frac{\rho_m^1}{\rho_m^1 + 1}} \right\rceil \quad (7)$$

Подсчет по (7) говорит о том, что процесс быстро сходится (n лежит в пределах нескольких десятков).

Поскольку нет гарантии, что изымаемый максимальный полный

подграф имеет \mathcal{U}_{max} , улучшить оценку (6) мы не можем.

2°. В исходном графе N^* заменим каждую вершину с ценой c_i полным графом с c_i вершинами. Мы получим новый граф с $\sum_{i=1}^{|N|} c_i$ вершинами.

К полученному графу применим предыдущую процедуру. Процесс займет

$$n \leq \left[1 + \frac{\lg \sum_{i=1}^{|N|} c_i}{\lg \frac{\rho_m^{*1}}{\rho_m^{*1} + 1}} \right] \quad (8)$$

шагов (ρ_m^{*1} - максимальная степень вершины в преобразованном графе).

Поскольку на каждом шаге выбирается максимальный полный подграф (а подграф полного графа - полный), любой граф, получившийся расщеплением вершины, либо целиком войдет в этот максимальный полный подграф, либо целиком не войдет.

Так как на каждом шаге изымается граф с максимально возможным числом вершин, т.е., в терминах исходного графа, с \mathcal{U}_{max} , а число шагов, а значит, и компонент разбиения известно, из (8), вычислим верхнюю грань для функционала (5), используя соотношение (1), и учитывая (6), для данного алгоритма будем иметь:

$$\left[1 + \frac{\sum_{i=1}^{|N|} c_i}{\lg \frac{\rho_m^{*1}}{\rho_m^{*1} + 1}} \right] \cdot \max_i \{c_i\} \geq \mathcal{U}_{min} \geq \min_{c_i \in N_z} \{ \sum c_i \} \quad (9)$$

3°. Оба предыдущих процесса требуют предварительного нахождения всех множеств несовместности. От этого недостатка свободен следующий метод. Будем производить допустимые объединения по мере нахождения множеств несовместности N_z . При этом на каждом шаге в процесс вовлекается количество вершин, не превышающее $2 \cdot |N|_{max}$. Число шагов равно числу этих множеств. Справедлива оценка (6) для функционала. При прочих равных условиях качество процесса зависит от способа организации каждого шага, но мы этот вопрос рассматривать не будем.

4°. Мы отправлялись при построении алгоритма от объединенного графа J . Можно расчленить процесс на этапы, на каждом из которых по графу J_i строится структура S_i каким-либо

из описанных способов. Затем производится объединение структур S_i . Последняя задача сводится к предыдущей. В самом деле, множество вершин структуры S_i можем считать множеством несовместности. Разница лишь в том, что в последнем случае множества несовместности не пересекаются.

5°. Использование результатов статистического моделирования. Здесь мы ограничимся (не претендуя на полноту) замечаниями, показывающими, как ввести эти данные в алгоритм синтеза.

Пусть известны частоты появления отдельных операторов в граф-схемах J_j . Разделим операторы на две группы - часто используемых A_1 и редко используемых A_2 . Изяем из графов J_j вершины A_2 и используем основной алгоритм синтеза. Вершины из A_2 либо назначим на устройства полученной структурной схемы S , либо отведем для них отдельную вершину, либо для вершин из A_2 проведем допустимые назначения на вершины S , а к остатку применим основной алгоритм синтеза. Последний метод нетрудно распространить на k групп операторов.

6°. Полученный граф S структурной схемы S проверяется на выполнение следствия из теоремы I.

7°. Некоторые частные случаи синтеза. Нет нужды применять основной алгоритм для синтеза простых (например, однопроцессорных) УЦВМ. Решение дает достаточные условия, в частности, следствие теоремы 2 (рис. 5). К использованию достаточных условий сводится и случай, когда требуется УЦВМ с заданными количествами процессоров, реализующих определенные элементы базы.

В свете изложенного становится ясным, почему алгоритмы функционирования многих существующих УЦВМ сводится к виду (рис. 5) и его вариантам.

§ 3. Синтез вычислительных систем по классу задач, заданных параллельными программами

1°. Основные определения.

ОПРЕДЕЛЕНИЕ II. Будем именовать граф-схему (программы) граф-схемой I-го ранга, если она не является подпрограммой какой-либо программы (аналогично и для алгоритмов функционирования). Граф-схемы операторов граф-схемы I-го ранга имеют ранг 2 и т.д.

Вообще, пусть дана граф-схема i - I-го ранга $J^{i-1} = \langle J^{i-1}, \mathcal{N}^{i-1}, \mathcal{R}^{i-1}, Z^{i-1} \rangle$ и пусть даны граф-схемы $J_1, \dots, J_j, \dots, J_m$, описывающие операторы из \mathcal{N}^{i-1} и распознаватели из \mathcal{R}^{i-1} . Тогда эти граф-схемы $J_1, \dots, J_j, \dots, J_m$ будут граф-схемами i -того ранга $J^i = \langle J^i, \mathcal{N}^i, \mathcal{R}^i, Z^i \rangle$.

Смысл разнесения по рангам объектов J , \mathcal{N} , \mathcal{R} ясен из определения J^i по J^{i-1} . Что же касается памяти Z , будем считать, что $Z^i \cap Z^{i-1} = \emptyset$, связь же (по информации) граф-схем J^i и J^{i-1} происходит через память $Z^{i,i-1}$, причем $Z^{i,i-1} \cap Z^{i-1} \neq \emptyset$. В граф-схеме J^i переменные из $Z^{i,i-1}$ используются в операторах q_0 и q_* (как входные и выходные переменные).

ОПРЕДЕЛЕНИЕ 12. N -ранговой граф-схемой будем именовать совокупность $\langle J^1, \{J^2\}, \dots, \{J^{n-1}\}, \{J^n\} \rangle$. Мы не рассматриваем развернутую граф-схему J_0^n , которую можно получить из этой совокупности последовательным применением операции суперпозиции [3].

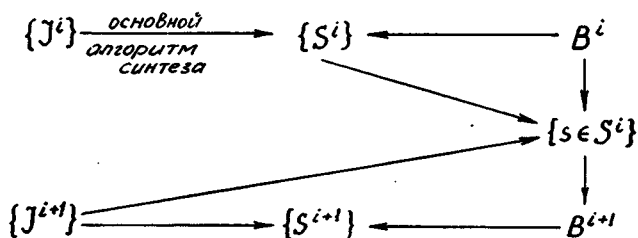


Рис. 6

Граф-схеме i -того ранга соответствует структурная схема i -того ранга, n -ранговой граф-схеме - n -ранговая структурная схема.

ОПРЕДЕЛЕНИЕ 13. Вычислительной системой (ВС) n -го ранга будем называть такую ВС, алгоритм функционирования и структурная схема которой имеют ранг n , а все операторы граф-схемы Γ^n суть УЦВМ. Тогда введенное нами в § I определение 9 есть результат применения суперпозиции к граф-схемам Γ^{n-1} и Γ^{n-2} . Аналогично можно определить и многопроцессорную УЦВМ n -го ранга.

ОПРЕДЕЛЕНИЕ 14. n -ранговой будем именовать такую ВС, у которой все составляющие вычислительной системы i - i -го ранга суть вычислительные системы i -того ранга ($i=2, \dots, n$). Для ВС i - i -го ранга ее составляющие ВС i -того ранга назовём подсистемами. Можно дать аналогичное определение и для многопроцессорной n -ранговой УЦВМ.

ОПРЕДЕЛЕНИЕ 15. Однородной будем именовать такую n -ранговую ВС, что для любого i ($i=2, \dots, n$) все составляющие (операторы) ВС i - i -го ранга изоморфны (т.е. как графы граф-схем, так и структуры i -того ранга). Отметим, что наш формализм не противоречит определению ОВС, данному в [10], а также приведенным там примерам 2-ранговых ОВС и n -ранговых ОВС (иерархий ОВС).

2°. Синтез n -рангового устройства либо вычислительной системы.

Синтез системы (или устройства) i -го ранга можно провести методами § 2. Мы получим структуру, вершинам которой соответствуют множества граф-схем $i+1$ -го ранга (подпрограммы), а множество дуг описывает коммутацию систем $i+1$ -го ранга между собой. Поступая подобным образом, мы можем получить как ВС, так и устройство, некоторые составляющие которого суть ВС и некоторые даже не отвечают необходимым условиям универсальности.

Как и в случае синтеза УЦВМ, процедура упрощается, если требуется построить некоторую структуру регулярного вида. Рассмотрим такую n -ранговую ОВС, что граф граф-схемы любого ранга полный. Тогда синтез сводится к определению того, сколько подсистем $i+1$ -го ранга должна иметь система i -того ранга и сколько должно быть рангов. В [10] рассматривается эта задача применительно к 2-ранговой ОВС.

Другие задачи синтеза, например, поиск вида графа G^1 для двухранговой системы [10], мы здесь рассматривать не будем.

Опишем общую процедуру синтеза n -рангового устройства. Ранг объекта указывается верхним индексом, элементы (составляющие) объекта ранга i имеют ранг $i+1$.

Пусть даны $B^1, \dots, B^i, \dots, B^n$ и граф-схема J^1 , т.е. известны алгоритмы элементов всех баз $\{J^2\}, \dots, \{J^i\}, \dots, \{J^{n+1}\}$.

По J^1 строим структуру S^1 . Множество X^1 вершин структуры S^1 таково, что его элементы суть наборы из компонент

базы B^i (слившиеся операторы, которые должны исполняться на одном устройстве). Тогда каждый $s \in X^i$ может быть получен следующим образом. Пусть $s_j = \{b'_1, \dots, b'_k\}$ ($s_j \in X^i$; $b'_1, \dots, b'_k \in B^i$). Известны граф-схемы J^1_1, \dots, J^2_k , описывающие элементы b'_1, \dots, b'_k базы B^i . Тогда $J_{s_j} = \bigcup_{i=1}^k J^i_{b'_i}$. Операция объединения осуществляется известными методами [4].

К каждой из J^2 применяем описанную процедуру и получаем $\{J^3\}$ и т.д. до $\{J^n\}$. Тогда набор $\langle J^1, \{J^2\}, \dots, \{J^3\}, \dots, \{J^n\} \rangle$ и есть структурная схема n -рангового цифрового устройства (системы). На рис. 6 изображен один шаг процесса порождения объектов синтеза.

Синтез алгоритма выполнения системы команд, который является исходной информацией для этапа логического синтеза [6], может быть выполнен по структурной схеме устройства и операторам, описывающим работу этих устройств, в соответствии с изложенной методикой. Другими словами, если $\{J^n\}$ суть алгоритмы элементов базы B^{n-1} (алгоритмы команд), то после проведения операции объединения получаем алгоритм выполнения системы команд.

§ 4. Заключительные замечания

Мы описали процесс синтеза в предположении, что известны базы B^1, \dots, B^n и запись элементов баз B^{i-1} через элементы базы B^i . Очевидно, нахождение опорного набора баз $\langle B^1, \dots, B^n \rangle$, пригодного для широкого класса устройств (или ВС), является нетривиальной задачей. Известны отдельные подходы [11], но общий метод отсутствует.

Выскажем ряд замечаний эвристического характера, которые могут быть полезны.

Предложенная методика синтеза требует отказа от записи алгоритмов на традиционных формальных языках, поскольку при этом исходным объектом синтеза окажется граф-схема Γ , описывающая существенно последовательный процесс. Преобразование ее в граф-схему J параллельной программы формально спасает положение, но применять такой подход к n -ранговым граф-схемам было бы неэффективно, ибо дробная запись через мелкие объекты обесценит метод синтеза. Задача должна быть описана в виде крупноблочной [12] параллельной программы 1-го ранга. Для этой

программы синтезируется структура I-го ранга. Устройства \mathcal{U}^1 , в свою очередь, должны быть описаны как крупноблочные параллельные программы 2-го ранга, и т.д.

В этой связи представляется перспективным новое направление в методах вычислений - параллельные алгоритмы, операторами в которых являются действия над многомерными массивами [13].

Предложенный метод синтеза переносит, по нашему мнению, центр тяжести исследовательской работы с задач синтеза систем и устройств на конструирование крупноблочных параллельных алгоритмов различных рангов.

Л и т е р а т у р а

1. Геллер С.И., Журавлев Ю.Г. Основы логического проектирования цифровых вычислительных машин, М., "Сов. радио", 1969.
2. Мультипроцессорные вычислительные системы. Под ред. Хетагурова Я.А., М., "Энергия", 1971.
3. Калужни Л.А. Об алгоритмизации математических задач - "Проблемы кибернетики", 1959, вып.2, с.51 - 68.
4. Карп Р.М. Заметка о приложении теории графов к программированию для вычислительных машин, - "Кибернетический сборник", 1962, вып.4, с.123-134.
5. Ershov A.P. Theory of program schemata. - In: IFIP congress 71. Amsterdam, 1971, vol. Invited papers, p.144-163.
6. Иловайский И.В. Формальный синтез схем устройств по алгоритмам их функционирования. - "Вычислительные системы", 1971, вып.47, с.56 - 72.
7. Поспелов Д.А. О некоторых математических проблемах, возникающих при совместной работе нескольких вычислительных машин. - "Труды МЭИ", 1964, 53, вып.2, сер.ВТ, с.97 - 110.
8. Берг К. Теория графов и ее применения. М., "ИЛ", 1962.
9. Зыков А.А. Теория конечных графов. Новосибирск, "Наука", 1969.
10. Евремов Э.В., Косарев Ю.Г., Однородные универсальные вычислительные системы высокой производительности. Новосибирск, "Наука", 1966.
11. Голышев Л.К. Структурная теория цифровых машин. М., "Энергия", 1971.
12. Канторович Л.В. Перспективы работ в области автоматизации программирования на базе крупноблочной системы. - "Труды Матем. ин-та АН СССР им. Стеклова", 1968, 96, с.5 - 15.

13. Jones P.D., Lincoln N.R., Thornton J.E. Whither computer architecture ? -In: IFIP congress 71. Amsterdam, 1971, booklet TA-4, p.162 - 168.

Поступила в ред.-изд.отдел
10 февраля 1972 г.