tanulmányok

21/1974

MTA Számítástechnikai és Automatizálási Kutató Intézet Budapest

ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
И АВТОМАТИЗАЦИИ ВЕНГЕРСКОЙ АКАДЕМИИ НАУК

Д О К Л А Д Ы С Е М И Н А Р А

проводимого 19-23-го февраля 1973г. в Венгрии во время совещания по теме 1-15.1. "Структура и методологические рекомендации по созданию систем автоматизированного проектирования дискретных устройств"

1973

Készült az OMKDK sokszorosító üzemében
F. v.: Jamoch Gyula

E L Ő S Z Ó

A KGST keretében müködő "Automaták általános elméletének kidolgozása" /I-15.1./ szakbizottság 1973.február 19-23 között Magyarországon tartotta ülését az MTA Számitástechnikai és Automatizálási Kutató Intézet szervezésében. A szakbizottsági ülések a résztvevő országok munkabeszámolóinak és a következő időszakra vonatkozó munkatervnek a megvitatása után mint kollokvium folytatódnak, ahol a témához kapcsolódó, az egyes országokban elért legujabb eredményekről és a jövőbeni elképzelésekről előadások hangzanak el. Az eddigi gyakorlat szerint az elhangzott előadásokról kiadvány nem készült. E hiányosság pótlását javasolta a magyar fél, vállalva az előadásokból összeállitott tanulmánykötet kiadását.

Reméljük, hogy kezdeményezésünk szokássá válik a további években, és igy alkalom nyilik a még egészen uj kutatási eredmények megismerésére.

Л И Д И - 72

Илона ФИДРИХ к.ф-м.н.

Исследовательский Институт Вычислительной
Техники и Автоматизации ВАН

Система ЛИДИ-72 является версией от 1972-го года системы обработки данных списковой структуры, разработанной для системы проектирования цифровых устройств с помощью вычислительных машин в отделе Цифровой Техники Исследовательского Института Вычислительной Техники и Автоматизации ВАН под руководством Ужоки Миклоша.

В данной системе проектирования цифровых устройств информация на каждой стадии проектирования представляется в виде направленного графа, любая вершина которого может быть снабжена последовательностью атрибутов любой длины. Так как во время проектирования меняются не только значения отдельных атрибутов, но и структура графа, внутримашинное представление информации должно быть достаточно гибким - это и было одному из важнейших требований к системе ЛИДИ-72. С целью ускорения функционирования системы проектирования внутреннее представление информации должно обеспечить возможность последовательной обработки атрибутов отдельных вершин. Кроме этих требований естетственным желанием было использование при представлении информации по возможности минимальное количество машинных слов. С целью обеспечения возможности простого переноса системы на любую вычислительную машину требованием к системе ЛИДИ-72 было также использование специальных свойств употребляемой машины в минимальной мере.

На основе вышеуказанных требований была разработана такая списковая структура информации, которая дает возможность для непосредственного представления направленного графа, вершины которого могут быть снабжены атрибутами. Сама система ЛИДИ-72 - набор подпрограмм, выполняющих определенные операции над информацией данной структуры, написанных на языке ФОРТРАН

и в настоящее время хранящихся в спомогательной библиотеке вычислительной машины ВАН типа CDC 3300.

Внутреннее представление информации

Во внутреннем представлении информации системы ЛИДИ-72 число машинных слов, занимаемых одним элементом списка, меняется в зависимости от типа и от расположения элемента, но каждый элемент обладает однословной головой. Ссылка на любой элемент списка означает ссылку на голову элемента, а на слово ссылаемся по индексу (адресу) данного слова.

В одном машинном слове расположено, как правило, несколько полей элемента, из которых с точки зрения выбранного представления информации важную роль играют поля, отведенные различным знакам, указывающим на некоторые свойства содержимых других полей данного элемента списка.

Во внутреннем представлении информации системы ЛИДИ-72 каждой вершине представляемого направленного графа соответствует один и только один вершинный элемент списка. Голова вершинного элемента содержит внутреннее название, отведенное названию вершины, которой соответствует данный элемент списка. (В системе ЛИДИ-72 названия вершин - следовательно и названия вершинных элементов - не являются обязательно идентификаторами, а какая-то вершина графа определяется последовательностью названий вершин, лежащих на каком-то, ведущем к данной вершине пути).

Если множество атрибутов, отведенное к какой-то вершине представляемого графа, является непустым множеством, то голова соответствующего вершинного элемента фигурирует со знаком A. В этом случае слово, предшествующее голове вершинного элемента, содержит число атрибутов N соответствующей вершины, и предшествующие этому слову N слова содержат сами значения атрибутов.

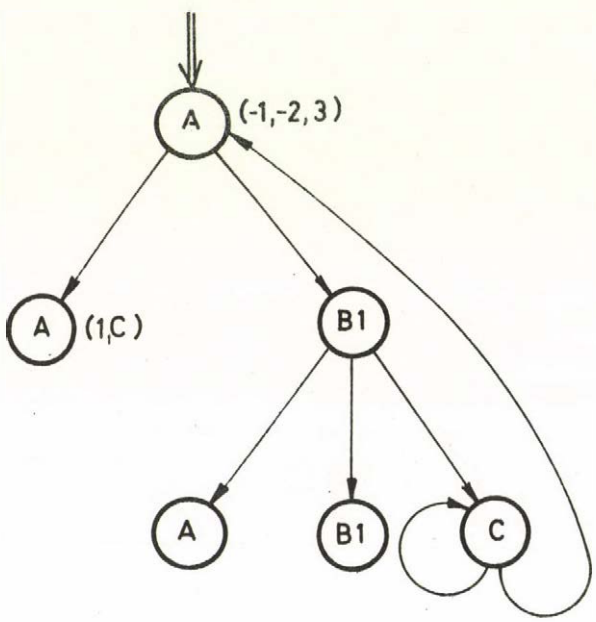
Если множество дуг, исходящих из какой-то вершины представляемого графа, является пустым, то голова соответствующего вершинного элемента фигурирует со знаком T. В противном случае каждой исходящей из вершины дуге соответствует одна и только одна дуговая связь соответствующего вершинного элемента, ссылающаяся на тот вершинный элемент списка, который соответствует вершине представляемого графа, в которую входит данная дуга. Дуговые

связи каждого неконечного вершинного элемента списка образуют такой линейный список содержащихся в поле связи одного машинного слова элементов, который расположен начиная со слова, следующего за головой данного вершинного элемента, и последний элемент которого фигурирует со знаком L . В простейшем случае линейный список дуговых связей одного неконечного вершинного элемента списка расположен последовательно. С целью упрощения процедуры изменения структуры информации этот линейный список может быть расположен так, что образуется непоследовательно расположенный линейный список последовательно расположенных подсписков дуговых связей данного неконечного вершинного элемента. Непоследний такой подсписок, занимающий $k>0$ последовательных машинных слов, в первых $k-1$ словах содержит дуговые связи, а в фигурирующем со знаком C последнее слово содержит связь продолжения, ссылающуюся на первое слово следующего подсписка дуговых связей данного неконечного вершинного элемента.

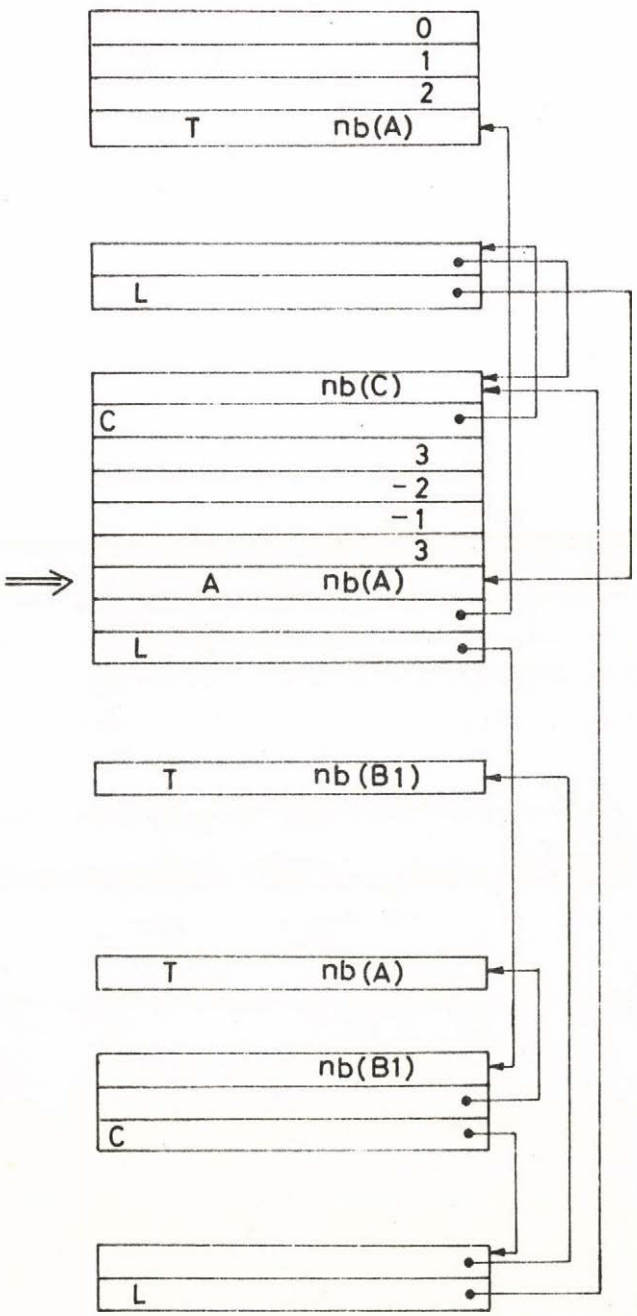
Пример представления в системе ЛИДИ-72 направленного графа, приведенного на рисунке I, указан на рисунке 2, где через nb(A), nb(B1) и nb(C) обозначены внутренние названия, соответсвующие названиям A, B1 и C соответственно.

Из описания внутримашинного представления вершин графа видно, что когда выполняются такие операции над графами, при которых расширяется множество атрибутов некоторой вершины или добавляется другая, исходящая из какой-то конечной вершины, тогда приходится перенести соответствующий изменяемой вершине вершинный элемент на более длинное свободное поле, если у этого вершинного элемента нет соседних свободных слов. В таких случаях для избежания полного прохождения списка на оригинальном месте перенесенного вершинного элемента оставляется слде в виде однословного переходного элемента, который является фигурирующей со знаком S связью, указывающей на новое место перенесенного вершинного элемента.

Из описанного представления направленных графов в системе ЛИДИ-72 видно, что в связном поле, отданном представлению информации, не занятые вершинными и переходами элементами списка машинные слова образуют свободные поля разной длины. Для экономического использования отданного списку связного поля в системе ЛИДИ-72 управление свободными словами производится с учитыванием длин свободных полей на основе каталога. Каталог свободных полей является особым вершинным элементом списка, фигурирующем со знаком T, но содержащим фигурирующие со знаком CL связи. Для $1 \le i \le 20$ i-тая связь указывает на первый элемент линейного списка свободных полей длины i, а пос-



Puc. 1.



Puc. 2.

ледняя 21-ая связь указывает на первый элемент линейного списка свободных полей длины выше 20-ти. Каждое свободное поле является элементом - естетственно, не последовательно расположенного - линейного списка свободных полей данной длины, исходящего из каталога свободных полей. Каждый элемент этих списков обладает однословной головой, в которой содержится со знаком CL связь, в случае непоследних элементов указывающая на следующий элемент данного списка, а в случае последних элементов, как и в соответствующих пустым спискам словах каталога, нулевая. Свободные поля длины больше 1 содержат и обратную связь со знаком CL , указывающую в случае непервых элементов на предшествующий элемент в данном списке свободных полей, а в случае первых элементов на соответствующее слово каталога. В свободных полях длины 3 в предшествующем голове слове содержится значение ω, поступающее в качестве значения атрибутов с малой вероятностью. В свободных полях длины $k>3$ в предшествующем голове слове указано значение $k-3$ и в предшествующих этому слову $k-3$ словах содержатся значения ω. Поля занимаются и освабождаются в основном по спискам, но знак CL по отношению остальных употребляемых знаков и значение ω определены таким образом, что при последовательному осмотру слов поля, отданного списку, обеспечивается распознавание свободных полей при осмотре сверху вниз определенно, а снизу вверх с большой вероятностью.

Пример структуры свободных полей приведен на рисунке 3.

Операции над информацией

В системе ЛИДИ-72 исходное состояние устанавливается с помощью подпрограмм, которые придают соответствующие значения параметрам, зависящим от используемой вычислительной машины (как например длина машинного слова, длина поля, отданного списку и т.п.), и определяют первоначальные значения некоторых переменных, передаваемых между подпрограммами системы через областей COMMON . С помощью одной из этих подпрограмм и список может быть приведен в начальное состояние.

Ввод информации в системе ЛИДИ-72 осуществляется с помощью подпрограммы ввода, которая считывает и интерпретирует предписания входного языка системы. В предписаниях могут задаваться описания вводимой информации в допущенных в входном языке пяти форматах. Каждое описание информации является описанием некоторого леса, внутреннее представление которого включается под вершиный элемент списка, соответствующий конечной вершине описанного в данном предписании пути представленного уже внутри машины направлен-

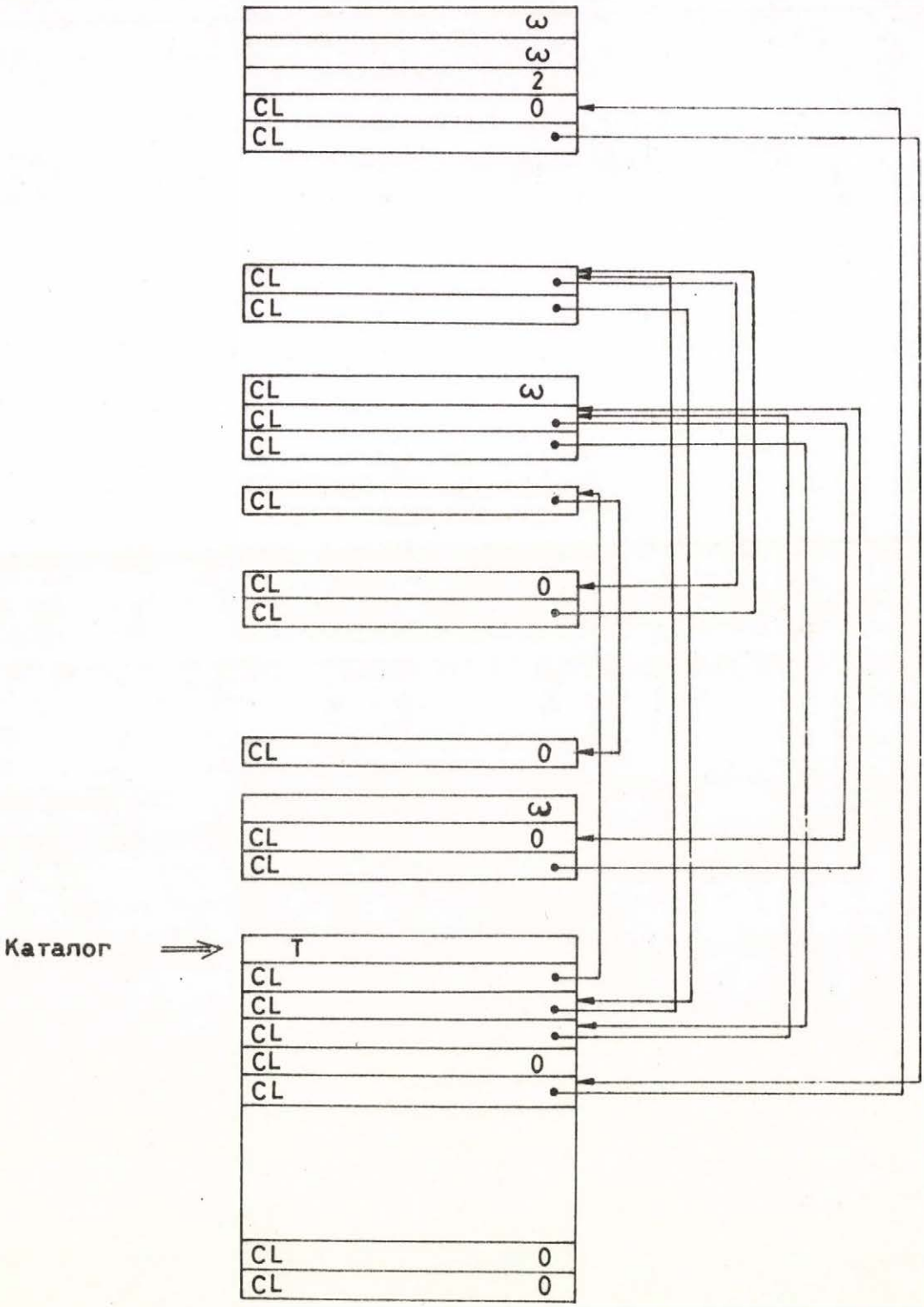


Рис. 3.

ного графа. Пример предписания входного языка системы ЛИДИ-72 приведен на бланках на следующих страницах. В результате интерпретации данного предписания получается внутренне представление направленного графа, показанного на рисунке 4, который является графовым представлением графической схемы, изображенной на рисунке 5.

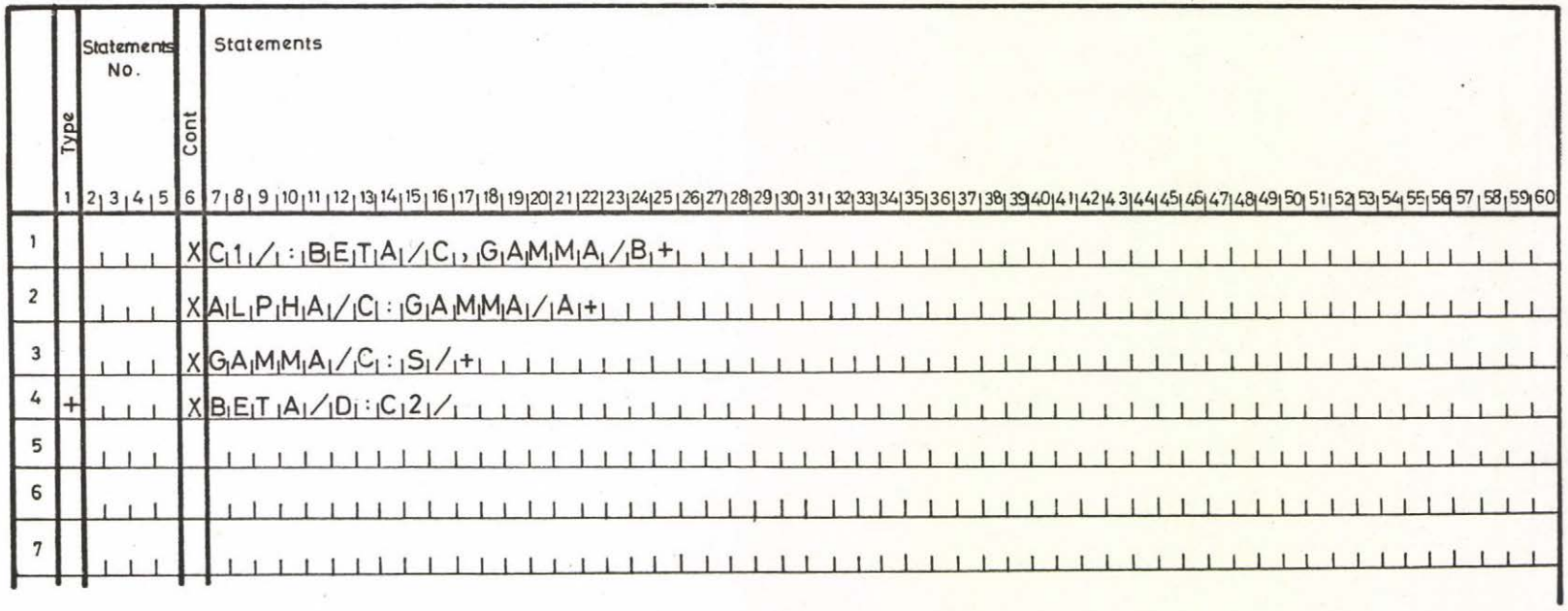
Во время работы на основе описания информации, задающего некоторый лес с помощью внешних названий вершин и различных разделителей согласно выбранному формату описания информации, описанный лес подпрограммой ввода представляется в виде скобочного выражения, в котором содержатся внутренние названия вершинных элементов. Внутреннее представление данного леса на основании его скобочного выражения включается в список с помощью соответствующей подпрограммы включения информации. Последние подпрограммы используют подпрограммы включения в список вершинных элементов и дуговых связей. Непосредственным вызовом подпрограмм включения дуговых связей может быть изменена и структура представления введенного леса таким образом, что получается представление более сложного графа, содержащего контуры. Имеются подпрограммы и расширения множеств атрибутов отдельных вершин.

В системе ЛИДИ-72 имеются соответствующие подпрограммы и для исключения информации из списка. С помощью этих подпрограмм может быть исключен из списка подсписок,представляющий некоторый подграф представленного списком графа. Эти подпрограммы используют и непосредственно вызываемые подпрограммы исключения отдельных вершинных элементов и дуговых связей списка. Имеются и подпрограммы исключения отдельных или всех атрибутов определенных вершинных элементов списка.

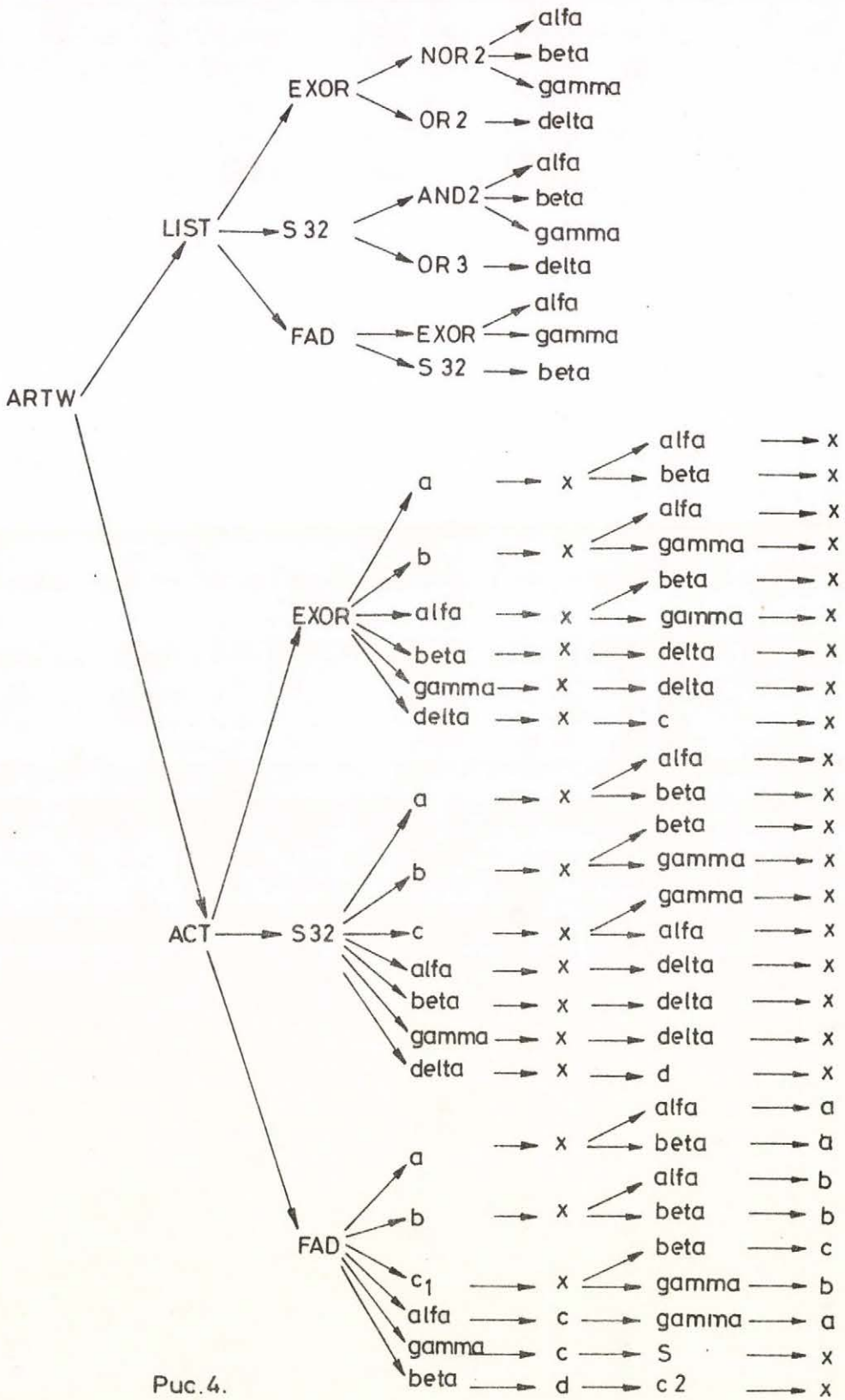
Важную группу подпрограмм системы ЛИДИ-72 образуют подпрограммы, реализующие прохождения по различным правилам вершин представленного графа. Подпрограммы этой группы при каждом пройденном вершинном элементе выполняют операцию, заданную пользователем, которая в случае одних подпрограмм может быть условной, а других - безусловной. Некоторые подпрограммы реализуют прохождение всех вершин, достигаемых исходя из определенной вершины представленного графа, а другие проходят только те вершинные элементы, которые соответствуют вершинам, лежащим на определенном уровне или до определенного уровня от заданной вершины.

Отдельную группу подпрограмм системы образуют подпрограммы, осуществляющие анализ структуры представленного списком направленного графа, которые были разработаны в рамках дипломной работы студента - математика Янош

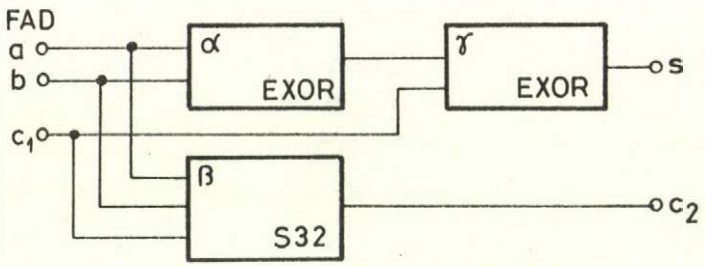
	Type	Statements No.	Cont	Statements
	1	2 3 4 5	6	7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
1	P	ATH		ARTW
2	L	IST		EXOR=
3			X	NOR2:ALPHA,BETA,GAMMA+
4			X	OR2:DELTA
5				S32=AND2:ALPHA,BETA,GAMMA+OR3:DELTA
6	A	CT		EXOR=
7			X	A/:ALPHA/,BETA/+
8			X	B/:ALPHA/,GAMMA/+
9			X	ALPHA/:BETA/,GAMMA/+
10			X	BETA/:DELTA/+
11			X	GAMMA/:DELTA/+
12			X	DELTA/:C/
13				S32=
14			X	A/:ALPHA/,BETA/+B/:BETA/,GAMMA/+C/:GAMMA/,ALPHA/+
15			X	ALPHA/:DELTA/+BETA/:DELTA/+GAMMA/:DELTA/+
16			X	DELTA/:D/
17	L	IST		FADD=EXOR:ALPHA,GAMMA+S32:BETA
18	A	CT		FADD=
19			X	A/:ALPHA/A,BETA/A+
20			X	B/:ALPHA/B,BETA/B+

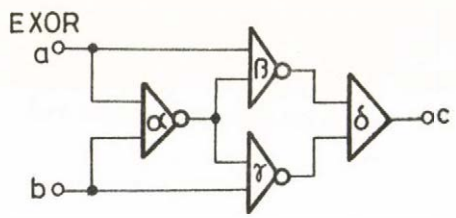


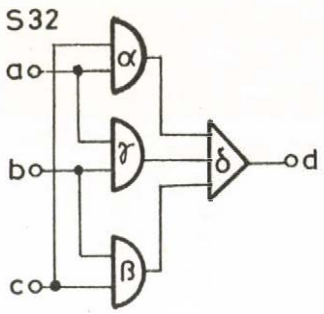
	Type	Statements No.	Cont	Statements
	1	2 3 4 5	6	7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
1			X	C 1 / : B E T A / C , G A M M A / B +
2			X	A L P H A / C : G A M M A / A +
3			X	G A M M A / C : S / +
4	+		X	B E T A / D : C 2 /
5				
6				
7				

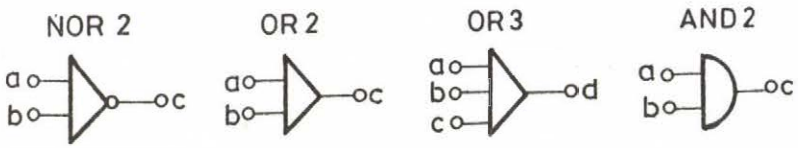

ARTW
LIST
EXOR
NOR 2
alfa
beta
gamma
OR 2
delta
S 32
AND2
alfa
beta
gamma
OR 3
delta
FAD
EXOR
alfa
gamma
S 32
beta
ACT
EXOR
a
b
alfa
beta
gamma
delta
x
alfa
beta
gamma
delta
c
S32
a
b
c
alfa
beta
gamma
delta
x
alfa
beta
gamma
alfa
delta
d
FAD
a
b
c_1
alfa
gamma
beta
x
alfa
beta
gamma
delta
c
S
c 2

Puc. 4.


FAD
a
b
c_1
α
EXOR
γ
EXOR
s
β
S32
c_2


EXOR
a
b
α
β
γ
δ
c


S32
a
b
c
α
β
γ
δ
d


NOR 2
a
b
c
OR 2
a
b
c
OR 3
a
b
c
d
AND 2
a
b
c

Puc. 5.

Ежефа. С помощью подпрограмм этой группы можно получить ответ на такие вопросы, как является-ли представленный списком направленный граф деревом, или определен-ли уровень каждой его вершины однозначно, или является-ли графом без контуров. В этой группе имеется и подпрограмма, определяющая минимальное множество таких вершин, после исключение которых представленный списком направленный граф превращается в граф без контуров.

Дальнейшее развитие системы

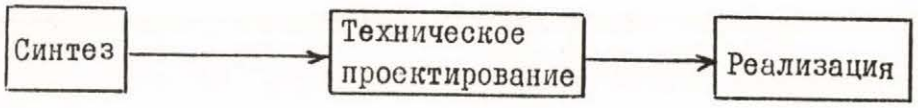
Первой задачей по развитию системы является унификация ввода и вывода информации. Дальнейшие усилия будут направлены на разработку управляющей программы системы и на расширение внутреннего представления информации на внешние накопители.

ПРОЕКТИРОВАНИЕ ПЛАТ ПЕЧАТНЫХ СХЕМ С ПРИМЕНЕНИЕМ ЭВМ

М. Ковач, А. Винце[x]

В Отделе Цифровой техники Исследовательского Института вычислительной техники и автоматизации ВАН разрабатывается система программ, служащая для проектирования цифровых устройств с помощью вычислительной машины. Языком программирования выбран ФОРТРАН IУ.

Процесс проектирования был разделен на отдельные этапы, как это показано на схеме:



Теперь система программ подразделяется согласно разделению задачи проектирования:

1. Программ, помогающие при логическом проектировании.
2. Техническое проектирование: а/ проектирование платы
   б/ кабляже рамки
3. Пост-процессорные программы, составляющие перфоленту, управляющую реализациями автоматами.

Результаты одних программ являются частью вводимых данных для других программ. Поэтому, а также ради возможности осуществления интерактивного проектирования нам пришлось выбрать соответствующую структуру данных. Разработанная нами структура данных является списковой структурой. Описание списковой структуры находится в стате Илона Фидрих/ Эта структура данных делает возможным сообщение данных, обмен данными между программами и единый способ модификаций в интерактивных точках, а также одновременную обработку данных, относящихся к различным задачам. В этой структуре данных возможно образование связей (графа) между данными и между отдельными частями информации, таким

x Исследовательский Институт вычислительной техники и автоматизации ВАН

образом, что представление данных избыточно в очень незначительной степени и в то же время обращение с данными динамическое.

В данном сообщении рассматривается только проектирование печатных схем.

Программа, проектирующая платы печатных схем

Задачей программы является техническое проектирование плат печатных схем, которые в матричном расположении на плоскости платы содержат интегральные схемы. Данные,используемые программой, могут быть причислены к двум основным типам: это данные, описывающие логическую систему (напр. схема соединений) и данные, описывающие конструкцию платы (размер платы, внешние контакты). Отправляясь от этих данных программа проектирует плату на основе описанного ниже нового принципа, применяемого вместо традиционно используемого в проектирующих программах такого типа алгоритма ЛЕЕ.Благодаря этому новому принципу на платах не остаются неспроектированные линии.

Построение программы делает возможным проектирование плат с матричным расположением различных размеров, размеры задачи ограничиваются только размером памяти имеющейся в распоряжении машины. Хотя программа, благодаря её принципу, не нуждается в активном сотрудничестве проектировщика в процессе проектирования, мы встроили в неё несколько интерактивных точек: замену интегральных схем, осуществляющих схему, схему функциональных элементов, реализуемых логическими (базовыми) элементами, находящихся в интегральных схемах, ввод дополнительных линий.

Вышеназванные интерактивные точки разделяют программу на следующие части: покрытие, размещение, трасировка.

I. Покрытие

Целью данного этапа является определение минимального числа интегральных схем, реализующих элементы логической схемы.

Для этого необходимо знать набор интегральных схем, имеющихся в расположении и список элементов логической схемы.

Процедура минимизации заключается в том, что свободные базовые элементы занятых интегральных схем используются для осуществления других логических функций, если такая подстановка возможна, и если это проводит к умень-

шению числа использованных схем. (Например, для осуществления схеми совпадения с двумя входами используем схему совпадения с тремя входами.) Такую процедуру подстановки в дальнейшем будем называть дегенерацией.

Результатом покрытия является некоторое множество интегральных схем, из которого возможно составление данной схемы соединений, но точного "привязывания" (т.е. точного соответствия между логической схемой и реализаций) в этой фазе ещё не происходит.

После этого следует интерактивный шаг, в котором проектировщик модифицирует набор интегральных схем, определённый в стадии покрытия.

II. Размещение

Для проектировщика обеспечивается возможность определить на листе печатной схемы геометрически отделённые части (категории). Это имеет особенное значение тогда, когда лист выполняет несколько электрически различных функций возможенных быть разделёнными также и геометрически. Задачей размещения является упределение элементов логических схем в интегральные схемы и определения месторасположение интегральных схем на плате таким образом, чтобы печатную проводку можно было бы осуществить на как можно меньшей территории.

Для размещения необходимо:

- электрическое описание проводников-носителей информации;
- определение картины листа печатной схемы (задание числа столбцов и рядов модулей).

На первоначальном шагу программа генерирует случайное размещение, которому ставит в соответствие некоторый коэффициент эффективности (вес). По ходу процедуры модули полагаются точками

$$W = \sum_{i=1}^{m} \sum_{j=1}^{n} l_{ij}$$

где

m – число логических элементов

n – число элементов, находящихся в потенциональной связи с i-тым логическим элементам $n = f(i)$

ij – число, характеризирующее взаимное расположение двух элементов:

Вертикальные каналы

Горизонтальные или направленные на модулю каналы

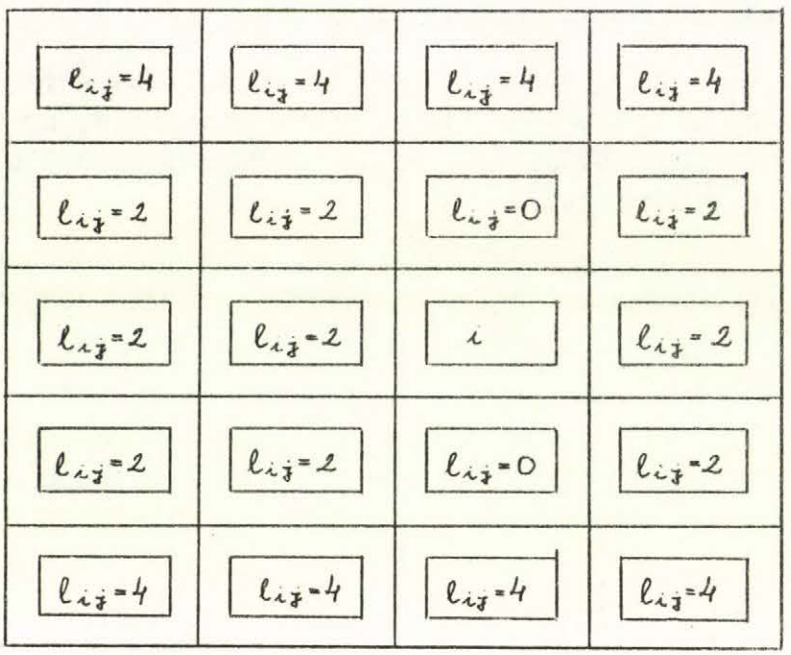


Рис. I.

Исходное распределение в дальнейшем оптимизируется путём случайных парных обменов.

В процедуре базначаются два логических элемента и потом исследуется, могут ли реализующие базисные элементы взаимно исполнять функции дугого. Если пара логических элементов удовлетворяет этим условиям и проведение обмена уменьшает суммирный все, эти два элемента меняются местами.

Если элементы пары не являются взаимозаменяемыми в программе испытываются взаимозаменяемость актуальных модулей и решение принимается на основе подобных принципов.

Неудачные выборы пар "наказываются" и если эта сумма перейдёт заданную границу, тогда процедура останавливается и выдаётся документация размещения.

Достоинствам метода решения является то, что в противоположность последовательно размещающим процедурам, отдельные модули и группы элементов и

оптимальное местоположение отделных модулей определяется на основе полной картины проводки.

Итак, программа выдаёт местоположение используемых модулей и реализованные в них элементы, но внутри модуля реализация происходит лишь на следующем шагу.

После изу ения документации проектировщик может пересмотреть результаты размещения и работа может продолжаться с модифицированным результатом.

На следующем шагу мы получаем новую документацию о том, что какого типа базисными элементами реализируются элементы схемы соединений. Если типы не совпадают, проектировщик решает, каким образом должна выполняться дегенерация. Напр. если для реализации схемы совпадения с тремя входами используется схема совпадения с тремя входами, то на основе решения проектировщика четвёртый вывод используется на напряжение питания.

III. Канализация

Проводка готовилась для технологии с двухсторонным печатанием, с проводниками, идущими горизонтально, со стороны элементов и с проводниками, идущими вертикально со стороны пайки.

Электрический контакт обеспечивается металлизированными отверстиями.

а/ Проектирование каналов

Задачей этого этапа является определение геометрической формы отдельных потенциалов в системе координат вертикальных и горизонтальных каналов таким образом чтобы нагрузка в сечениях каналов была равномерной.

В процессе решения предполагаем, что вывод модуля достижим с обеих сторон (рис.2.)

По ходу проектирования интегральные схемы рассматриваются точками и в дальнейшем и, проводники, исходя из вышеописанной схемы, могут подходить к любому выводу модуля как сверху, так и снизу.

В процедуре важную роль играет карта каблирования которая определяется матрицей каналов. Значение элемента матрицы указывает, сколько проводников проходит по сечению данного канала.

По окончанию работы программа выдаёт в качестве документации матрицу нагрузки. После определения картины монтажа выполняется точное соответствие внутри модуля. Элементы логической схемы соединений реализуются каталоговыми

элементами, таким образом, чтобы это сопровождалось пропорциональным уменьшением проводников в схеме соединений.

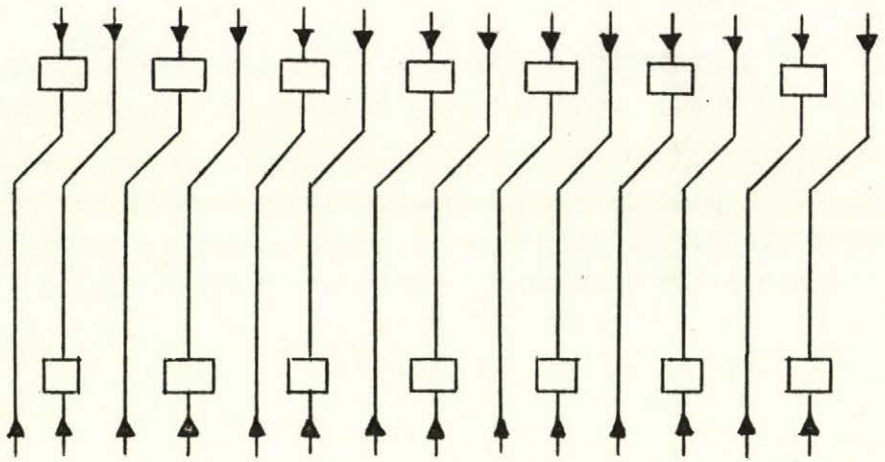


Рис. 2.

б/ Соединение эквивалентных групп выводов, распределение выводов

На предыдущих этапах была определена форма проводников, т.е. известно, какие отрезки проводников идут в горизонтальных каналах (в каналах идущих в направлении модулей), какие - в вертикальных каналах (в перекрёстном направлении).

Мы стремимся использовать как можно меньше "улиц" путём попарного соединения отдельных отрезков проводников.

На этом этапе составляется граф канала процедуры минимизации ширины канала, которую покажем на следующем примере.

Пусть форма канала следующая:

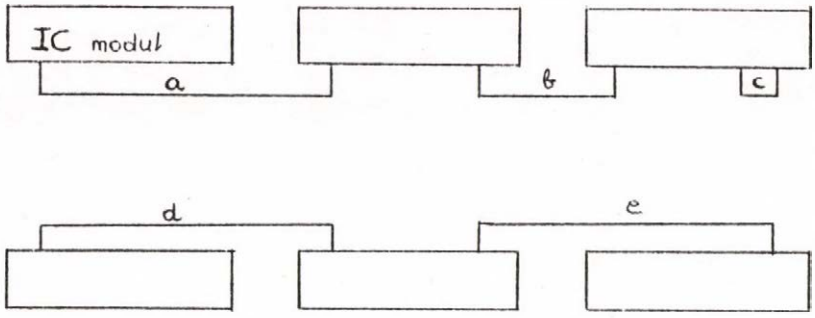


Рис. 3.

На соответствующем этому рисунку графе узлы соответствуют отдельным отрезкам проводников, а направленные стрелы соответствуют совмесности, возможности помещения в одной "улице".

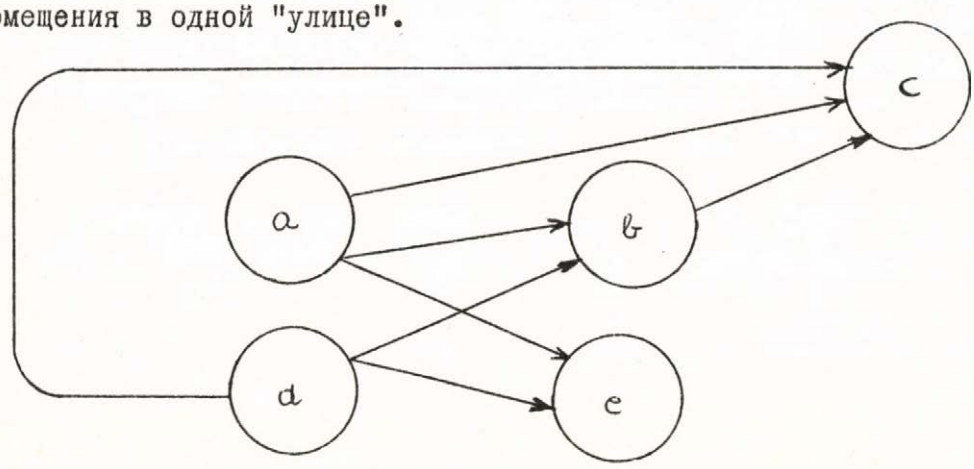


Рис. 4.

Целью алгоритма привязывания выводов заключается в том, чтобы к данному отрезку проводника выбрать среди эквивалентных выводов тот вывод, при выборе которого возникает соотношение наилучшей совместности.

В программе исследуются пары проводников внутри каналов и только в такой ситуации производится привязывание выводов, когда решение с точки зрения совместности является однозначным.

Процедура проводится дважды для каждого канала, потом непривязывание выводы распределяются в произвольном порядке.

Это процедура показала себя очень эффективной; примено 75% эквивалентных выводов определяется соотношениями совместности и образующиеся картины каналов очень благоприятны.

Для находящихся в каналах элементов, кроме совместности, определяем ещё одну зависимость которая описывает упорядочение элементов в перекрёстном направлении. Эта упорядочение обладает свойствами транзистивной иррефлексии и иссимметричности. Введение упорядочения в перекрёстном направлении необходимо для осуществления физической связи между элементами.

д/ разделение на подграфи

На следующем этапе проектирования нужно упорядочить находящие в каналах элементы линий таким образом, чтобы требуемая ширина канала была бы наименьшем. Назавём путём часть канала, занимаемую одним проводникам, тогда задача состоит в том, чтобы канал содержал минимальный путь. Задачу сформилируем в описанном выше графе таким образом, что граф канала нужно разделить на минимальное число таких подграфов, элементы которых образуют полный граф с точки зрения отношения совместности, и, если считать этот подграф точкой, будет существовать определённая на исходном упорядоченность в перекрёстном направлении. Это означает, что в некоторый подграф может попасть не более одного элемента из упорядоченного в перекрёстном направлении множества. Эту задачу мы разрешили с помощью такой эвристической процедуры, которая в случае отсутствия упорядоченности в перекрёстном направлении является алгоритмом т.е. даёт оптимальное решение, а в противном случае даёт квазиоптимальное решение. В решенных доселе не удалось вмешательством человека неправить выдаваемое программой решение.

IУ. Метризация

После этапа разделения на подграфы имеется в распоряжении принципальная, независимая от действительных размеров платы картина печатной платы. На настоящем этапе вместе с метризацией платы выполняется проведение линий к внешним разъёмам и определение порядка подграфов внутри канала в

перекёрстном направлении. Для выполнения задачи необходимо знать размер платы, число внешних разъёмов (максимально 4, на каждую сторону не более одного) и геометрические характеристики. Порядок подграфов определяем, принимая во внимание порядковое отношение между ними таким образом, чтобы значение длины проводников находящихся в каналах, соседних с внешними разъёмами, на так называемых упорядочивающих дорожках, ожидалось минимальным. Это выполняется для того, чтобы вероятностно увеличивалось бы число умещающихся в упорядочивающих дорожках проводников. Внешние сигналы фиксируем к контактам заданных внешних разъёмов (несколько сигналов подводятся к заданным контактам) и генерируем граф канала упорядочивающей дорожки таким образом, чтобы на графе создать как можно больше отношений совместимости.

После этого подобным образом определим разделение на подграфы и определим порядок подграфов для других каналов. В результате выполнения программы получаем геометрические размеры каналов, порядок подграфов и описание линий лежащих в упорядочивающих дорожках. При обладании этими данными мы приготовим линии, описывающие всю печатную плату. В некоторых случаях может быть, что полученные в результате проектирования физическое размеры платы больше требуемых. В этом случае проектировщик может выбирать один из следующих возможных выводов:

- модифицирует ход линий полученной картины печатания или снова запускает программу с новыми параметрами;
- может обратиться программу контракции, которая попробует сократить размеры картины печатания.

Поскольку программу имеет дело только с элементами интегральных схем или с другими элементами, объединёнными в интегрально-схемом модуле, после этапа метризации проектировщик может определить на свободное место платы различные схемные элементы вместе с их связями. Описываемая ниже программа контракции, может обрабатывать также и заданные таким образом данные.

Контракция

Программа работает с данными, описывающими проводники печатной платы, на каждом отдельном шаге может обрабатывать только линейные элементы принадлежащие одному каналу. Линейные элементы отжимаются к краю канала с малыми координатами. При проведении сжатия необходимо раздробление определённых линейных элементов удастся добиться указанного сжатия другие элементы канала и элементы с большими значениями координат параллельно отодвигаются

на полученное расстояние.

Если предыдущие функции проектирования были успешны, в нашем распоряжении теперь имеет геометрическое описание печатной платы, и это описание может оцениваться проектировщиком. Если печатная плата соответствует требуемой может вызываться программа пост-процессора, которая изготовит перфоленту, необходимую для управления машины с цифровым управлением. В нашем институте имеется программа пост-процессора, которая изготовляет перфоленту для установки АДМАП.

Кёгст Манфред, ГДР, г. Дрезден

О проблеме декомпозиции булевых функций

При автоматизации промышленности проблема синтеза управляющих устройств играет большую роль. До сих пор строили автоматы, прежде всего, на основе логических элементов ИЛИ, И, НЕ. Синтез автоматов из этих элементов создали при помощи теории дисъюнктивных нормальных форм. Эта теория хорошо известна из литературы. Однако, она обладает тем недостатком, что существует только взаимо-однозначное отношение между выражениями нормальных форм и параллельно -последовательными схемами глубины два.

Но в настоящее время всё больше изготовляются, так называемые, интегрированные или функциональные элементы, причём, такие элементы реализуют более сложные булевые функции, чем ИЛИ, И, НЕ. К сожалению, знакомые методы нельзя употребить для синтеза схем из интегрированных элементов. Следовательно, необходимо разработать новые методы и алгоритмы, пригодные для синтеза. Общий метод синтеза схем из интегрированных элементов представляет собой декомпозиция.

Настоящая работа ограничится синтезом комбинационных схем при помощи декомпозиции. В частности, интересует следующая проблема:
как можно найти для заданной булевой функции реализацию и даже оптимальную реализацию из интегрированных элементов данного набора?
При этом, конечно, нужно предполагать функциональную полноту набора. При решении данной проблемы возникают две подпроблемы. Первая подпроблема состоит в создании методов и алгоритмов для декомпозиции, т.е. для выделения булевой функции f_1 относительно другой булевой функции f_2 и для вычисления функции остатка f_3 .При этом f_2 - функция реализуемая интегрированным элементом. Вторая подпроблема состоит в более эффективном повторении выделения. Эта проблема особенно трудна, если необходимо найти реализацию, которая должна быть в каком-нибудь смысле оптимальна. Такой случай потребует большое усилие.

Первая подпроблема /т.е. задача выделения функции f_1 относительно функции f_2 и вычисления функции остатка f_3 / означает в языке алгебры нахождение следующего изображения:

$$f_1(x_1, \dots, x_{k1}) = f_3(z_1, \dots, z_{k3}; f_2(y_1, \dots, y_{k2}))$$

Причём, между множествами переменных величин существуют следующие отношения:

$$\{y_1, \dots, y_{k2}\} \subseteq \{x_1, \dots, x_{k1}\} \quad u \quad \{z_1, \dots, z_{k3}\} \subseteq \{x_1, \dots, x_{k1}\}.$$

В случае, если функции f_1 , f_2 и f_3 являются полными фукциями, и кроме того,

$$\{z_1, \dots, z_{k3}\} \cap \{y_1, \dots, y_{k2}\} = \phi$$

т.е. пересечение между этими множествами равно пустому множеству, то говорят что это - дисъюнктивное выделение. В противоположном случае это не является дисъюнктивным выделением. Из литературы известен целый ряд условий, когда

для двух данных функций f_1 и f_2 существует дисъюнктивное выделение, и также существуют методы, которыми можно найти такое выделение. С точки зрения нахождения оптимального решения дисъюнктивные выделения не играют особенную роль. Это подтверждается на примерах.

В общем случае как функция f_1 так и функция f_3 являются частичными, т. е. не всюду определённые функции. Частичная функция обозначается следующим образом: $f = (f^1, f^0)$. Если f зависят от n переменных, тогда для всех элементов x множества $\{0,1\}^n$ определяются следующим образом:

$$f^1(x) = \begin{cases} 1, & \text{если } f(x) = 1 \\ 0 & \text{иначе} \end{cases} \quad \text{и}$$

$$f^0(x) = \begin{cases} 1, & \text{если } f(x) = 0 \\ 0 & \text{иначе} \end{cases}$$

Пусть $f_{11} = (f_1^1(x_1, \dots, x_{k1}), f_1^0(x_1, \dots, x_{k1}))$ частичная функция, тогда можно получить решение выделения f_1 относительно f_2 в качестве частичной функции $f_3 = (f_3^1(z_1, \dots, z_{k3}), f_3^0(z_1, \dots, z_{k3}))$
При этом

$$f_3^1(z_1, \dots, z_{k3}) = z_{k3}(f_2 f_1^1) \vee \bar{z}_{k3}(\bar{f}_2 f_1^1) \text{ и } f_3^0(z_1, \dots, z_{k3}) = z_{k3}(\bar{f}_2 f_1^0) \vee \bar{z}_{k3}(f_2 f_1^0).$$

Таким образом первая подпроблема решена. Анализ реальной схемы из элементов заданного набора показывает, что принципиально возможно получать реализацию функции f из заданных элементов при помощи декомпозиции. Однако, относительно синтеза оптимальной схемы не известно, каким элементом надо начинать, чтобы получить оптимальное решение. Поэтому надо проверить все возможности, т.е. надо выделить f относительно всех элементов набора при всех интерпретациях их переменных. Но с точки зрения практического синтеза нельзя пользоваться этим подходом, так как в зависимости от числа переменных величин функции f и от числа элементов набора, затрата увеличивается значительно. Поэтому все практические методы декомпозиции предусматривают поиск реализаций, близких к оптимальным. Естественно, существуют многие возможности методов поиска реализации, близких к оптимальным. Предлагается метод, для которого потребуется два критерия.

Первый критерий K_1 - определяет порядок переменных величин данной функции в зависимости от их важности относительно реалицации этой функции. Второй критерий K_2 - определяет порядок функций данного набора в зависимости от их важности относительно реализации другой заданной функции.

Существует целый ряд таких критерий. Например, один критерий типа K_1 предлагают М.А. Гаврилов и В.М. Копиленко в их работе "Метод переходных таблиц синтеза многовыходных комбинационных структур произвольных элементов" - Москва 1970. К сожалению, математическим методом нельзя сравнить различные критерии. Сравнение можно провести только на примерах.

Применяя критерии K_1 и K_2 можно написать алгоритм для декомпозиции булевой функции следующим образом:

задан функциональный полный набор элементарных функций $f_{21}, \dots, f_{2p}$ и функция f .

0. шаг: $f_1 := f$.

I. шаг: Определяется порядок переменных функции $f_{21}, \dots, f_{2p}$ на основе критерия K_1.

2. шаг: Определяется порядок переменных функции f_1 на основе критерия K_1.

3. шаг: В зависимости от полученных порядков переменных величин функций f_1 и $f_{21}, \dots, f_{2p}$ переменные функции f_1 определяются как вход каждой функции $f_{21}, \dots, f_{2p}$. Полученные функции обозначаются как $f'_{21}, \dots, f'_{2p}$.

4. шаг: Выделяется функция f_1 относительно функции $f'_{21}, \dots, f'_{2p}$. Остатками этих выделений являются функции $f_{31}, \dots, f_{3p}$.

5. шаг: Из множества функций $f_{31}, \dots, f_{3p}$ выбирается на основе критерия K_2 одна функция f_3.

6. шаг: Спрашивается: равна ли реализованная f_3 заданной f ? Да, решение существует, т.е. можно реализовать f схемой из данных элементов . Нет, то $f_1 := f_3$ и перейти к 2. шагу.

Конечно, качество полученного решения зависит от качества критерий K_1 и K_2.

Вехлер Вольфганг, ГДР, г. Дрезден

О поведении конечных однородных структур

0. п-мерная конечная однородная структура (т. е. клеточный автомат) является интуитивно одной конечной п-мерной решёткой, причем каждая точка решётки представляет собой копию одного и того же конечного автомата (клетка). Состояние одной клетки в момент времени $t + 1$ определено состоянием этой клетки в момент времени t и состояниями постоянного числа соседных клеток в момент времени t. Для каждой клетки геометрическое расположение соседных клеток одно и тоже. Все клетки работают синехронны. Внешные вход и выход производятся через свободные входы и выходы клеток на переферии структуры.

Клеточные автоматы применяются в качестве математических моделей для адаптивных систем, для вычислительных структур (особенно для больших интегральных схем), для биологических систем и т. п.

Здесь рассматривается только автоматно-теоретический анализ поведения конечных клеточных автоматов. При этом установится только на поведении в качестве акцептора.

I. Конечный клеточный автомат $\underline{C}$ размерности п (при этом п-ететственное число) является следующим 4-набором.

$\underline{C} = (C, \Sigma, X, (b_1, \ldots, b_n))$.

Причём $C = (S,\delta)$ - отдельная клетка, определенная конечным непустым множетвом S внутренных состояний и локальной фунцией перехода $\delta: S^m \to S$. $\Sigma = (\sigma_1,\ldots,\sigma_m)$, где $\sigma_i \in \{-1,0,+1\}^n$ для $i = 1, \ldots, m$, называется схема соседства /образец соединения/. Входное множество X состоит из всех возможных интерпретаций периферии E от $\underline{C}$ с элементами из S :

$$E = \bigcup_{i=1}^{n} E_i \quad \text{с} \quad E_i = E_i^{(1)} \cup E_i^{(r)} \quad \text{и}$$

$$E_i^{(1)} = \{\alpha|\alpha = (\alpha_1,\ldots,\alpha_n) \text{ с } \alpha_i = 0 \text{ и } \alpha_j \in \{0,1,\ldots,b_j+1\} \text{ для } j \neq i\},$$

$$E_i^{(r)} = \{\alpha|\alpha = (\alpha_1,\ldots,\alpha_n) \text{ с } \alpha_i = b_i+1 \text{ и } \alpha_j \in \{0,1,\ldots,b_j+1\} \text{ для } j \neq i\}.$$

Так, X - множество свех отображений от E в S, т. е. $X = S^E$. n - набор $(b_1,\ldots,b_n)$ с b_i - ететственное число для $i = 1, \ldots, n$ содержит так называемые параметры ограничения от $\underline{C}$.

Чтобы приводить определение поведения вкачестве акцептора, придаётся $\underline{C}$ автомат $A(\underline{C}) = (Z, X, d)$. При этом Z - множество состояний, состоящее из всех возможных интерпретаций /конфигураций/ "Клеточного пространства" R от $\underline{C}$ с элементами из S :
$R = \{\alpha|\alpha = (\alpha_1,\ldots,\alpha_n) \text{ с } \alpha_i \in \{1,2,\ldots,b_i\}\}$.
Так, Z является множеством всех отображений от R в S, т. е. $Z = S^R$. Глобальная функция переходов $d: Z \times X \to Z$ придает каждой парой (z, x) состояние z', определение которого установленно следующим образом:

$$z': R \xrightarrow{\Sigma^o} (R \cup E)^m \xrightarrow{(z \cup x)^m} S^m \xrightarrow{\delta} S \in S^R$$

причем $\sum^{o}(\alpha) = (\alpha + \sigma_1, \alpha + \sigma_2, \ldots, \alpha + \sigma_m)$

и $(z \cup x)^m (\alpha + \sigma_1, \ldots, \alpha + \sigma_m) = (s_{\alpha+\sigma_1}, \ldots, s_{\alpha+\sigma_m})$.

Задача заключается в том, определить множество всех входных слов $w \in X^*$, которые переводят заданное начальное состояние $z_0 \in Z$ в состояние выбранного частичного множество вонечных состояний $Z_f \subseteq Z$, т. е. определить

$$L(z_0,Z_f) = \{w \in X^* | d(z_0,w) \in Z_f\}$$

для любых $z_0 \in Z$ и $Z_f \subseteq Z$. Эти множества слов $L(z_0,Z_f)$ можно вычислить с помощью так называемой матрицы переходов $\overline{d}: Z \times Z \longrightarrow A(X)$, где $A(X)$ - алгебра событий по X. При этом $\overline{d}$ определенно следующим образом: $\overline{d}(z',z'') = \{x \in X | d(z',x) = z''\}$.

При помощи произведения матриц вводим итерацию, так что действует $L(z_0,Z_f) = \bigcup_{z \in Z_f} \overline{d}^*(z_0,z)$.

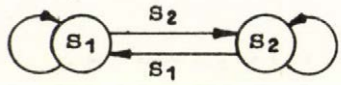
II. Идея, на которую основанно вычисление матрицы $\overline{d}^*$.

Из функциональной связи между $\overline{d}$ и соответствующей матрицы переходов $\overline{\delta}$ клетки выводится соотношение $\overline{d}^* = f(\overline{\delta}^*,K)$, причём K - такназываемая полугруппа связи, зависящая от C и $\sum$ [1,2].

δ^* преполагается известным, т. е. определить общее поведение в зависимос ' от поведения отдельных клеток.

Пример. $n = 1$; $\Sigma = (\sigma_1, \sigma_2)$ с $\sigma_1 = 0$, $\sigma_2 = -1$

/последовательная схема/; $C = (S, \delta)$ с $S = \{s_1, s_2\}$ и

δ: s_1  s_2 .

Матрица перехода клетки:

$$\overline{\delta} = \begin{bmatrix} s_1 & s_2 \\ s_1 & s_2 \end{bmatrix} .$$

Матрица перехода последовательной схемы

$$\overset{b}{\overline{d}} = \begin{bmatrix} s_1\overset{b}{\overline{K}}_1 & s_2\overset{b}{\overline{K}}_1 \\ s_1\overset{b}{\overline{K}}_2 & s_2\overset{b}{\overline{K}}_2 \end{bmatrix} = \begin{bmatrix} \overset{b}{\overline{K}}_1 & 0 \\ 0 & \overset{b}{\overline{K}}_2 \end{bmatrix} \overline{\delta} \quad ,$$

$\overset{b}{\overline{K}}_1$, $\overset{b}{\overline{K}}_2$ – элементы из $\overset{b}{\overline{K}}$ /полугруппа связи последовательной схемы/.

Причём b – число клеток в последовательной схеме.

Так справедливо:

$$\overset{b}{\overline{d}}^* = 1 \cup \overset{b}{\overline{d}} \cup \overset{b}{\overline{d}}^2 \cup \ldots \cup \overline{\delta}^* . \overset{b}{\overline{d}}^{b-1} .$$

Литература

[1] Wechler, W.: Eine Analysemethode für das Verhalten von zellularen BOOLEschen Feldern.
ZKI-Berichte (Akad. d. Wiss. d. DDR), Berlin 1973 .

[2] Wechler, W.: Kopplungshalbgruppen von Serienschaltungen.
Erscheint demnächst.

Бэр Гунтер, ГДР, г. Лейпциг

К решению проблем дискретного программирования посредством алгебры высказываний

При минизации булевых функций, при редукции автоматов и в других случаях, возникают частные проблемы, которые часто решаются методами целочисленного программирования.

Мы пытались пройти обратный путь, т.е. мы ставили перед собой цель решить проблемы целочисленного или - специально-дискретного программирования с помощью средств, вытекающих из теории автоматов /подобно псевдобулевому программированию/.

I. Принципиальный подход

Пусть задана следующая проблема программирования:

$$\left.\begin{array}{l} Z(x_1,\dots,x_n) \Longrightarrow \min \\ \text{при условиях ограничения} \\ \qquad g_i(x_1,\dots,x_n) \geq 0 \qquad \text{для} \quad i = 1,2,\dots,m \\ \qquad x_j \in \{0,1\} \qquad \text{для} \quad j = 1,2,\dots,n \end{array}\right\} \quad (1)$$

Искомы оптимальные решения от I.

Для специальных значений переменных $(x_1^0,\dots,x_n^0)$ условие ограничений $g_i(x_1,\dots,x_n)\geq 0$ либо выполнимо либо не выполнимо, т.е. либо истинно$^{(T)}$ либо ложно$(\mathcal{F})$. Иными словами, $g_i \geq 0$ являя однозначным отображением, которое придаёт n-набору от значений 0 или I логическое значение $\mathcal{F}$ или $\mathcal{T}$. Если здесь у переменных величин вместо 0 и I также принимаются логические значения $\mathcal{F}$ и $\mathcal{T}$, то $g_i \geq 0$ является однозначным отображением, которое n-набору логических значений опять придаёт логическое значение, т.е. $g_i \geq 0$ представляет собой логическую функцию классической двухзначной логики.

Пока неравенство $g_i \geq 0$ не является синтаксически правильно созданным /например для $4x_1x_2 + 3x_2 \geq 3$, где $x_j \in \{\mathcal{F},\mathcal{T}\}$, не дано определение/. Чтобы избавиться от этого положения, в работе /I/создан формализм арифметико-логического высказывания. В этом формализме можно заменить каждую численную переменную x_j логической переменной p_j следующим образом:

$$x_j = (1 \wedge p_j) \vee (0 \wedge \bar{p}_j) \qquad (2)$$

и для интерпритации $\mathcal{J}$ имеет место

$$\mathcal{J}(p_j) = \mathcal{F} \longrightarrow \text{значение}\,(x_j,\mathcal{J}) = 0$$

и

$$\mathcal{J}(p_j) = \mathcal{T} \longrightarrow \text{значение}(x_j,\mathcal{J}) = 1$$

Посредством подхода (2) из проблемы программирования (1) вытекает следующая проблема программирования:

$$\left.\begin{array}{l} Z(p_1,\dots,p_n) \Longrightarrow \min \\ \text{с условиями ограничения} \\ \qquad g_i(p_1,\dots,p_n) \geq 0 \quad \text{для} \quad i = 1,2,\dots,m \\ \text{и} \qquad p_j \in \{\mathcal{F},\mathcal{T}\} \quad \text{для} \quad j = 1,2,\dots,n \end{array}\right\} \quad (3)$$

Постоянные числа, которые пока ещё имеются в условии ограничения $g_i \geq 0$ можно исключить пригодным понятием выведения, причём, получается выражение H_i классического формализма двухзначной логики высказывания.

Конъюнкция $K = \bigwedge_{i=1}^{m} H_i$ со следующим свойством соответствует в конце концов совокупности всех ограничений.

Для замещения f переменных $p_1, \ldots, p_n$ справедливо значение $(K, f) = T$ тогда и только тогда, когда $(x_1^o, \ldots, x_n^o)$ является допустимой точкой проблемы программирования (1).

Наконец, с использованием целевой функции можно подобрать оптимальное решение.

Пример: $Z = 14 x_1 + 4 x_2 + 8 x_1 x_3 \Rightarrow \min$

$$2 x_1 + 3 x_2 + 4 x_3 \geq 6 \quad (\equiv H_1)$$
$$5 x_1 x_2 + 2 x_2 x_3 - 6 x_3 \geq -4 \quad (\equiv H_2)$$

Действительно $H_1 = p_1 p_3 \vee p_2 p_3$, $H_2 = p_2 \vee \bar{p}_3$

и следовательно $K \equiv H_1 H_2 = (p_1 p_3 \vee p_2 p_3) \wedge (p_2 \vee \bar{p}_3) = p_2 p_3$

$$Z(x_1, x_2, x_3) = Z(x_1, 1, 1) = 14 x_1 + 4 + 8 x_1 = 22 x_1 + 4 \Rightarrow \min.$$

так $x_1 = 0, x_2 = 1, x_3 = 1$, причём $Z = 22$.

2. Способ решения

а/ В абзаце I излагался подход к решению проблемы только принципиально. Однако, эффективность решения проблемы зависит значительно от эффективности преобразования отдельного условия ограничения в логическое выражение.

Линейные условия ограничения можно преобразовать в логическое выражение довольно хорошо:

Существуют необходимые и достаточные критерии, чтобы установить, есть ли заданное условие ограничения H_i всегда выполнимо или никогда не выполнимо.

Если $H_i \equiv a_0 + a_1 x_1 + \ldots + a_k x_k - b_0 - b_1 x_{k+1} - \ldots - b_l x_{k+l} \geq 0$

и $a_0, \ldots, a_k, b_0, \ldots, b_l$ – это не отрицательные вещественные числа, то действительно: $H = T$ тогда и только тогда, когда $a_0 - \sum_{j=0}^{l} b_j \geq 0$

Если ни один из обоих случаев не имеет место, то в соответствии с

$$H_i = \bar{p}\, H_i^{\,p/F} \vee p\, H_i^{\,p/T}$$

разложится H_i в два сокращённых условия ограничения $H_i^{\,p/F}$ и $H_i^{\,p/T}$, а затем исследуются опять эти условия ограничения, и т.д. Даже возможно заданное линейное условие ограничения непосредственно свести к минимальной альтернативной нормальной форме.

б/ К решению линейных проблем оптимального программирования, при которых переменные принимают два значения: 0 и I.

Каждое условие ограничения преобразуется в минимальную конъюнктивную нормальную форму K_i . Конъюнкции $K = \bigwedge_{i=1}^{m} K_i$ соответствуют совокупности всех условий ограничения, которые представляют собой опять конъюнктивную нормальную форму и по способу Нельзона её можно преобразовать в альтерна-

тивную нормальную форму. Каждой простой конъюнкции в соответствует множество допустимых точек задачи оптимального программирования. Наконец, с помощью целевой функции отбирается оптимальное решение.

Этот способ решения также применим для нелинейных задач оптимального программирования, хотя прт этом получается пониженная эффективность /потому, что в большинстве случаев нельзя переводить нелинейные условия ограничения в минимальные нормальные формы/.

Эти результаты сравнимы с результатами, полученными Камму и Рудеану /2/.

3. Возможности преобразования условий ограничения проблемы оптимального программирования.

Интересные результаты даёт следующая постановка вопросов:
Задано любое выражение H классического двухзначного формализма высказывания. Спрашивается о существовании эквивалентного к H неравенства с переменными, принимающими два значения: 0 и I. На этот вопрос всегда можно давать положительный ответ и его можно аналитически решить.
Если H -любая альтернативная нормальная форма, то непосредственно можно указать эквивалентную к H нелинейное неравенство.

Пример: пусть $H \equiv p_1 \bar{p}_2 \vee \bar{p}_1 p_4 p_5 \vee p_3$

отсюда вытекает $H = x_1(1-x_2) + (1-x_1)x_4 x_5 + x_3 \geq 1$

Так для выражения R вида $R \equiv p_1 \vee p_2 \vee \ldots \vee p_i \vee \bar{p}_{i+1} \vee \ldots \vee \bar{p}_{i+l}$

следует $R = (x_1 + x_2 + \ldots + x_i - x_{i+1} - \ldots - x_{i+l} \geq 1-l)$

Выводы для проблем оптимального программирования, при которых переменные принимают два значения:0 и I.

а/ Всегда можно любое линейное условие ограничения заменить некоторыми /в общем случае/ специальными линейными условиями ограничения.

Пример:Пусть задано линейное условие ограничения

$H: \quad 7x_1 + 3x_2 + 6x_3 + 5x_4 + 10x_5 \geq 6$

Тогда $H = (p_1 \vee p_2 \vee p_3 \vee p_5)(p_1 \vee p_3 \vee p_4 \vee p_5)$

и потому $H = \begin{cases} x_1 + x_2 + x_3 + x_5 \geq 1 \\ x_1 + x_4 + x_3 + x_5 \geq 1 \end{cases}$

Таким образом, любые линейные проблемы оптимального программирования можно привести к специальным линейным.

б/ Всегда можно любое нелинейное условие ограничения заменять в /общем случае/ некоторыми линейными условиями ограничения.

Пример: Пусть задано нелинейное условие ограничения H:

$4 + 3x_1x_3 + 2x_2 - 6x_2x_3 - 2x_1x_4 - 3x_1 \geq 0$

Тогда $H = (\bar{p}_1 \vee p_2 \vee p_3 \vee \bar{p}_4)(\bar{p}_1 \vee \bar{p}_2 \vee \bar{p}_3 \vee \bar{p}_4)$

и потому $H = \begin{cases} -x_1 + x_2 + x_3 - x_4 \geq -1 \\ -x_1 - x_2 - x_3 - x_4 \geq -3 \end{cases}$

Таким образом, любые нелинейные проблемы оптимального программирования

можно свести к одному единственному нелинейному условию ограничения.

Пример: Заданы 3 условия ограничения

$$H_1 \equiv 14x_1 + 7x_2 + 3x_3 + x_4 \geq 8$$

$$H_2 \equiv x_1 + 8x_2 + 7x_3 + 10x_4 \geq 15$$

$$H_3 \equiv 8x_1 + 3x_2 + 4x_3 - 2x_4 \geq 7$$

Тогда $H_1 H_2 H_3 = p_1 p_2 p_3 \vee p_1 p_2 p_4 \vee p_1 p_3 p_4 \vee p_2 p_3 \bar{p}_4$

и далее $H = x_1 x_2 x_3 + x_1 x_2 x_4 + x_1 x_3 x_4 + x_2 x_3 - x_2 x_3 x_4 \geq 1$

Такое преобразование преимущественно для случая, когда вначале число условий ограничения является большим.

г/ Некоторые нелинейные условия ограничения можно заменить даже одним условием ограничения.

Пример: Пусть задано $H \equiv x_1 + x_1 x_2 + x_1 x_3 - x_1 x_4 + x_2 x_3 - x_2 x_3 x_4 \geq 1$

то и $H \equiv 2x_1 + x_2 + x_3 - x_4 \geq 2$

При изучении способов преобразования условий ограничения можно применять итоги пороговой логики.

Литература

/1/ Bär,G.,Rohleder,H.: Über einen arithmetisch-aussagenlogischen Kalkül und seine Anwendung auf ganzzahlige Optimierungsprobleme. EIK 3 (1967), S.171-195

/2/ Kammu,P.L.,Rudeanu,S.: Boolean methods in operations research and related arrays. Springer-Verlag Berlin, Heidelberg 1970

Языки переработки графической информации

Зенон КУЛЬПА [x]

Р е з ю м е

Статья посвящена построении специализированных языков программирования, предназначенных для оперирования с графической информацией при помощи ЭЦВМ (автоматическое чтение текста, анализ снимков, карт и планов, технических чертежей и др.).

Описывается коротко язык "PICTURE ALGOL 1204 " сделанный для машины ОДРА 1204, который уже находится в успешной эксплуатации в течении полтора лет. Представляется также концепция нового, более сложного и имеющего на много большие возможности языка "PAL ", находящегося сейчас в стадии детальной разработки в Институте Прикладной Кибернетики ПАН.

I. Введение

Бурное развитие теории и практики опознавания образов, в особенности графических /зрительных/ делает необходимым разработку подходящих средств для формулировки, описания и испытания алгоритмов переработки, анализа и опознавания зрительных образов.

Один способ осуществления этой цели базируется на создании специальных устройств предназначенных для конкретных задач опознавания. Такой подход однако имеет смысл только для хорошо разработанных задач, например чтения стилизованного печатного текста. Для более сложных задач такой подход связан с большими затратами времени и требует больших усилий на разра-

[x] Институт Прикладной Кибернетики ПАН, Варшава, Польша

ботку громоздких электронных устройств.

Более эффективным средством исследований в этой области является очевидно использование ЭЦВМ. Это делается либо путем разработки специализированных машин /крупнейшим таким проектом является машина ILLIAC III [1,3]/ либо путем снабжения обычной универсальной ЭЦВМ устройствами ввода и вывода графической информации и соответственной системой программирования /язык программирования, библиотека подпрограмм/. В мире сделано уже несколько таких систем, наиболее распространенная из них группа систем имеет общее название "PAL" [2] , базируется на языке ФОРТРАН и представляет собой в принципе что-то вроде симулятора некоторых технических узлов машины ILLIAC III.

В нашем институте ведутся тоже работы в этом направлении. Разработано специальное устройство ввода - вывода образов в ЭЦВМ ОДРА 1204 /на основе телевизионной камеры/ и соответственное опрограммирование, основой которого является язык PICTURE ALGOL 1204 [5,6] . Краткое описание этого языка дано в следующем разделе настоящего доклада. Эта система работает уже в течении некоторого времени и накопленный до сих пор опыт /смотри например [7] / показывает её высокую целесообразность в испытании разного рода алгоритмов переработки изображений.

Но организованные возможности выше упомянутой системы уже нас не удовлетворяют. Параллельно с разработкой лучшего устройства ввода - вывода изоображений ведутся у нас работы над усовершенствованным, более удобным языком программирования для целей переработки графической информации. Основные принципы конструкции этого языка, получившего название PAL /Picture Analysing Lang. [8]/, представлены в третьем разделе доклада.

2. Язык PICTURE ALGOL 1204

Этот язык базируется в большой степени на языке Нарасимхана [3], имеет тоже его название. Язык тот среализован в виде так называемого "погружения" в языке АЛГОЛ 1204 /которой является реализацией полного АЛГОЛа 60 на ЭЦВМ ОДРА 1204/.

Операции языка сделаны в виде набора взаимно связанных процедур АЛГОЛа. Тела этих процедур в большинстве написаны в некотором подмножестве машинного языка. Тем самым используются возможности предлагаемые транслятором языка АЛГОЛ 1204, который позволяет писать фрагменты АЛГОЛ - программы с использованием машинных команд. Эти возможности значительно повышают

эффективность работы программы и использования памяти /смотри также разд.2.I/

Операции языка реализуют, в принципе, так называемые однородные, локальные операции над образами [3, 4] . Реализацию специальных структур данных языка /оброзов и списков направлении/ описывается ниже.

Полное описание языка находится в [5], почти полное, в более компактном виде можно найти в [6]. В работе [7] описано использование этого языка для испытания разного рода алгоритмов утоньшения линии.

2.I. Представление образа

Язык в принципе оперирует образами в двух уровнях яркости /черно--белыми/. Много-уровенные образы можно представлять в виде набора нескольких двухровенных, обрабатываемых отдельно, соответственно с принятым кодированием уровней зачерненности.

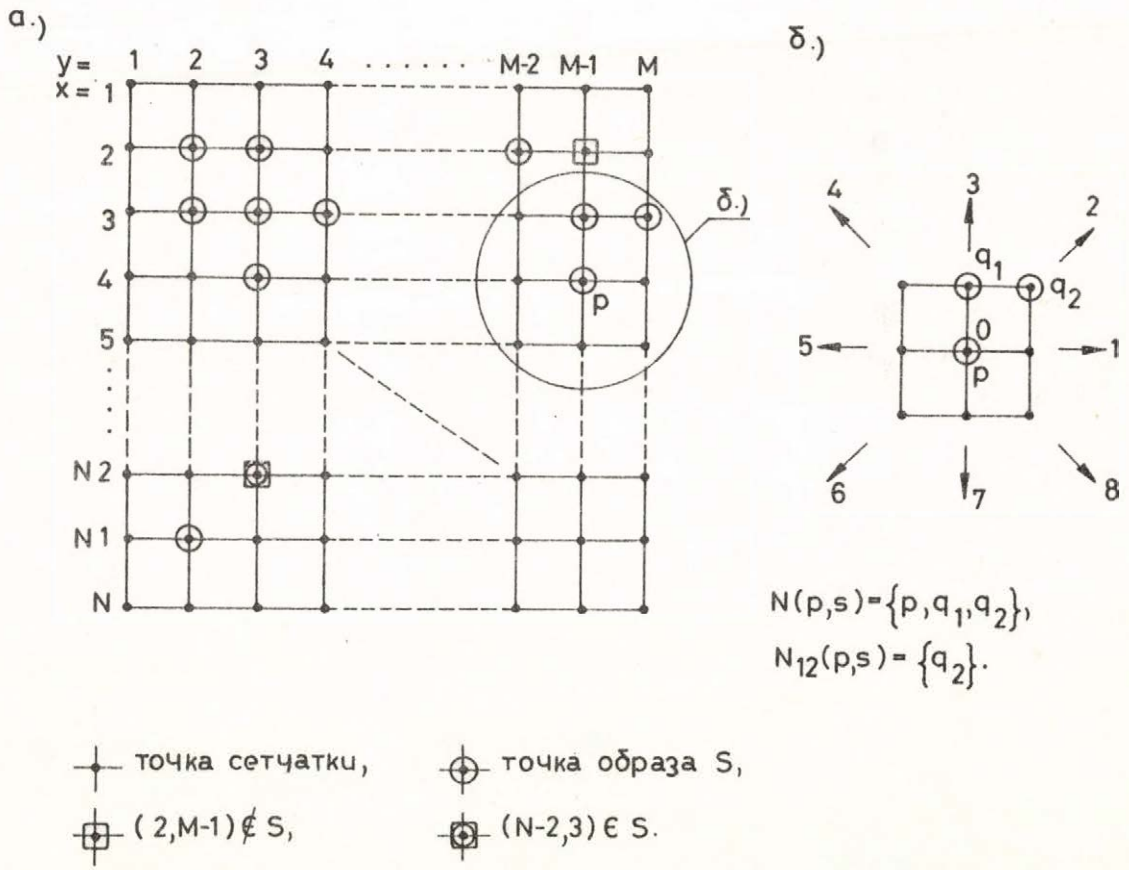


Рис.I. Образ и окрестность

Образ в языке представляется /рис.I./ на прямоугольном растре /сетчатке/, размером M на N точек /клеток/. В виду бинарности образа, можно принять, что образ есть просто некоторое подмножество точек целого растра, т.е. если растр /"универсальный образ" [3]/ обозначим через:

$$U = \{(x,y) \mid 1 \le x \le N \,\&\, 1 \le y \le M\}$$

то произвольный образ S будет некоторым подмножеством образа U: $S \subseteq U$.

В машине образ записывается в виде одномерного массива типа integer АЛГОЛа I204, размером в N элементов, по M клеток растра в слове. Одна точка представляется одним битом /разрядом/ слова, который равен единице когда точка принадлежить образу и равен нулю в противном случае. Такое представление образа в машине обеспечивает большую компактность упаковки образов в памяти а также повышает скорость обработки /квази-параллельность, см. ниже/.

С каждой точной p растра связано подмножество ее непосредственно соседних точек - окрестность N (p) - рис.I. Множество тех точек окрестности, которые принадлежат одноврменно некоторому образу S означается N (p,s). Все соседние точки имеют свои номера, начиная с нуля /означает самую точку p/ до восьми. Эти номера носят название "направлений" относительно точки p /рис. I.б./.

Одними из параметров операции соседства /см. ниже/ являются списки направлений, представляемые в программе в виде "строк" /string/ АЛГОЛа, т.е. последовательности номеров направлений замкнутых в строчные скобки /апострофы/. Значение такой строки декодируется внутри процедур языка при использовании машинных команд /так как в АЛГОЛе нет операции над строками/.

2.2. Операции языка

Операции над образами, находящиеся в языке, принадлежат классу так называемых однородных, локальных операции над образами [4] т.е. операции, которые зависят только от значений точек в некоторой /обычно невеликой/ окрестности рассматриваемой точки, в независимости от ее положения на растре и результатов операции для других точек образа. Такого рода операции могут быть выполнены параллельно и на таком принципе построена машина ILLIAC III [I,3] . В языке PICTURE ALGOL как и в системах PAX [2] некоторый уровень параллельности достигается путем хранения нескольких точек образа в одном машинном слове, используя факт, что все разряды слова обрабатываются одновременно.

Операции языка составляют несколько групп, с которых кратко будут описаны две важнейшие.

Первая группа операции /операции теории множеств/ включает операции на образах как на множествах точек растра /см. разд. 2.I./. К этим операциям относятся операции переписывания образов, присваивания нулевого значения /пустого множества точек/, пересечения, объединения, дополнения /до универсального образа U/, суммы по модулю два, разности, проверки пустоты образа и равенства образов.

Вторая группа операции языка это операции соседства. Это наиболее сложные операции языка, результат которых для данной точки p образа зависит от значений точек окрестности N(p,s) этой точки. За исключением одной, все эти операции зависят от двух образов - "аргумента" и "контекста". Результатом этих операции являются те точки образа - аргумента, которых окресность в образе - контексте выполняет определеные условия, заданные другими параметрами операции.

В той группе можно выделить три подгруппы - генерирующие операции, которые выделяют точки растра находящиеся в заданном направлении от точек аргумента в образе - контексте; операция трансформации, выделяющая точки, которых окрестность выполняет условия заданные булевой функцией от номеров направлений, и пороговые операции, выделяющие точки, для которых число точек в разного типа окрестностях исполняет булево выражение АЛГОЛа, являющееся одним из параметров операции.

Пример использования пороговых операции для устранения шума и выделения горизонтальных и вертикальных линий образа показывает рис.2.

3. Язык PAL

Конценция языка PAL [8] предполагает, что язык должен удовлетворят следующим требованиям:

I. Удобное оперирование образами и их структурными описаниям,
2. Возможность легкого расширения состава его операции,
3. Эффективность выполнения программ,
4. Сравнительная легкость имплементации.

Для выполнения этих требований в языке предусматривается соответствующие составные структуры данных /см. разд. 3.I./, механизм модулярной библиотеки подпрограмм /см. разд. 3.2./ и возможности оперирования, на нисшем уровне языка, машинными единицами данных /слово, адрес/ и машинными командами.

INPUT PICTURE X :

```
D D D D D D D D D D D D D D D D D D D D D D D D
D                                             D
D                       X X                   D
D           X X X X X X X X X     X X     X X X X X X     D
D             X X X X X X X       X       X X X X X       D
D                X X X X X     X X         X X X           D
D X X X          X X X X                   X X X           D
D                X X X X           X       X X X X         D
D              X X X X X         X         X X X   X       D
D              X X X X X         X         X X X           D
D           X    X X X X X X X X X X X X X X X X X         D
D        X       X X X X X X X X X X X X X X X X X         D
D     X  X       X X X X X X X X X X X X X X X X X       X D
D X X            X X X X                   X X X           D
D     X          X X X X     X             X X X           D
D     X X        X X X X                   X X X           D
D                X X X X               X X X X           D
D              X X X X X X             X X X X X X       D
D           X X X X X X X X           X X X X X X X     D
D                       X             X X X             D
D                          X                            D
D D D D D D D D D D D D D D D D D D D D D D D D D D D D
```

THRESHOLDCMARK X,X,H,'12345678',THR 4 :

```
W W W W W W W W W W W W W W W W W W W W W W W W W W
W                                             W
W                                             W
W             H H H H H H H           H H H   W
W           H H H H H H H H H       H H H H H   W
W             H H H H H H             H H H     W
W             H H H H H               H H H     W
W             H H H H H               H H H     W
W             H H H H H               H H H     W
W           H H H H H                 H H H     W
W             H H H H H H H H H H H H H H H H   W
W             H H H H H H H H H H H H H H H H   W
W             H H H H H H H H H H H H H H H H   W
W             H H H H H               H H H     W
W             H H H H H               H H H     W
W             H H H H H               H H H     W
W             H H H H H               H H H     W
W           H H H H H H H             H H H H H W
W             H H H H H               H H H H   W
W                                             W
W                                             W
W W W W W W W W W W W W W W W W W W W W W W W W W W
```

Рис.2. Операции порога

```
      THRESHOLDCHAIN  X,X,P,'15',THR 5   :

W W W W W W W W W W W W W W W W W W W W W W W W W
W                                               W
W                                               W
W       P P P P P P P P         P P P P P P P   W
W         P P P P P P P                         W
W                                               W
W                                               W
W                                               W
W                                               W
W                                               W
W           P P P P P P P P P P P P P P P P     W
W           P P P P P P P P P P P P P P P P     W
W           P P P P P P P P P P P P P P P P     W
W                                               W
W                                               W
W                                               W
W                                               W
W                                               W
W       P P P P P P P P       P P P P P P P P   W
W                                               W
W                                               W
W W W W W W W W W W W W W W W W W W W W W W W W W

      THRESHOLDCHAIN  X,X,L,'1357',THR 15   :

W W W W W W W W W W W W W W W W W W W W W W W W W
W                                               W
W                   L                           W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L L L L L L L L L L L L L     W
W           L L L L L L L L L L L L L L L L     W
W           L L L L L L L L L L L L L L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W           L L L L                   L L L     W
W                                     L L       W
W                                               W
W W W W W W W W W W W W W W W W W W W W W W W W W
```

Рис.2. /ок./ Операции порога

В языке предполагается динамическое резервирование памяти, подчиненное полному контролю программиста и не связанное с какойлибо блочной структурой программы.

Нет в языке булевых выражений и переменных - еих роль выполняют ограниченные целье числа, представляющие собой значения переменно- значной логики [9]. Условные инструкции заменены инструкциями выбора одной среди многих альтернатив по значении целочисленного выражения.

3.I. Структуры данных

Кроме элементарных типов данных /целые и действительные числа, знаки, адреса, слова, метки/ в языке существует ряд составных структур данных.

Первым из них является тип "set" /множество/, представляющий собой одномерный "плавающий" вектор объектов некоторого элементарного типа /тип всех элементов данного множества одинаков/. Элементы множества доступны либо по индексу, либо с начала или конца вектора. Число элементов множества может произвольно менятся в ходе программы. Оперирование множествами адресов позволяет моделировать в языке произвольные, сложные структуры данных, кроме стандартных.

Второй составной структурой является образ "picture" представляемый в виде двухмерного массива целых чисел из некотрого предела значений. Данный образ имеет определенные размеры /пределы координат точек/ и предел значений "зачерненности" точек. С образом тесно связан следующий тип данных, а именно список направлений, представляющий собой множество пар целых чисел, определяющее относительные координаты точек окрестности некоторой точки образа.

Самой сложной структурой языка являются так называемые "полиграфы" представляющие собой некоторые реляционные структуры. От обычных графов они отличаются тем, что их "ребра" могут иметь более чем два конца. Такие "полиребра" представляют собой, очевидно, многоаргуметные реляции в множестве вершин полиграфа. Полиграф является совокупностю множества вершин и множества полиребер разного порядка, такой, что в множестве его вершин входят все концы его полиребер. Как вершины так и полиребра это так называемые "атомы". С каждым атомом может быть динамически связан динамически изменяемый набор "свойств", являющихся произвольными объектами элементарных типов. Один и тот

же атом может являться элементом нескольких полиграфов, как вершина и как полиребро. Наличие полиграфов в языке дает возможность удобно оперировать и трансформировать структурные описания образов.

3.2. Модулярная библиотека подпрограмм

В языке PAL определена конструкция модулярной библиотеки подпрограмм, которая является средством расширения языка. В состав библиотеки входят модули, каждый из которых имеет свое наименование, список параметров и содержит определения переменных, процедур и операторов.

Определенные в данном модуле переменные и подпрограммы доступны в каждой программе имеющей в начале, в так называемым списке присоединений, наименование этого модуля и определение его параметров.

Стандартными операциями языка являются лишь самые элементарные действия на единицах данных. Все более сложные операции предполагается определять в соответственных библиотечных модулях. Для этой цели язык содержит богатый и эффективный в использованию аппарат определения новых операторов и процедур [8].

3.3. Эффективность машинной реализации

Эффективное использование программой всех возможностей ЭЦВМ является важным фактом качества программы, особенно при переработке больших массивов сложной по структуре информации при помощи больших и сложных программ. Использование в таких случаях языков программирования высшего уровня накладывает большое требования во первых на качество компиляторов и во вторых на конструкцию самого языка.

Язык программирования высшего уровня обеспечивающий высокую эффективность использования машины программой должен, прежде всего, иметь свойство "прозрачности" до уровня слов и команд машины и должен допускать возможность не использования программистом конструкции высшего уровня языка если ухудшают они эффективность программы.

Язык PAL в большой степени удовлетворяет этим требованиям. В языке существует ряд типов данных нисшего уровня, а именно "слово" /последовательность битов/, "адрес" и "метка", являющиеся адресом данных или инструкции соответственно. Для этих типов данных определен ряд операций нисшего уровня /выделение битов слова, модификация и извлечение адресов и др./, обеспечивающих эффективность и гибкость их использования и позволяющих на моделирование программистом довольно сложных структур данных наи-

более эффективным способом. Этой же цели служит механизм динамической резервировки памяти находящихся под контролем программиста.

В языке допускается тоже использование команд машины, в форме обращения к процедурам. Состав команд, число и значение их параметров определяется конкретной имплементацией языка.

Еще одним механизмом этого рода является возможность выбора одного из двух способов трансляции подпрограмм - допускающий рекурсивные подпрограммы или нет. Существует еще несколько конструкции подобного типа, но из-за недостатка места не будем их здесь рассматривать.

4. Заключительные замечания

Описанные в докладе языки программирования представляют собой очередные шаги в направлении создания удобных средств записи и испытания алгоритмов переработки графической информации при помощи ЭЦВМ. Язык PICTURE ALGOL имеет характер первого, экспериментального шага в новую область при помощи как возможно простых средств /"погружение" в уже существующий язык высшего уровня/ и служит больше всего для накопления некотрого опыта. Язык PAL в свою очередь, как результат накопления этого опыта, является уже богатым и соответственно сложным языком, предназначенным с одной стороны как основа ряда потребительных систем переработки изображений в разных практических областях использования, а с другой стороны, наличие в нем таких элементов, как многозначения логика и оперирование реляционными структурами расширяет круг его возможных применений на другие области переработки сложных по структуре массивов информации.

5. Литература

[1] Mc Cormick B.H.: The Illinois Pattern Recognition Computer - ILLIAC III. IEEE Trans. on El. Comp., EC-12, 1963. pp. 791-818 .

[2] Johnston E.G.: The PAX II Picture Processing System, in: Picture Processing and Psychopictorics, Lipkin B.S., Rosenfeld A., eds., Academic Press, 1970.

[3] Narasimhan R.: Labeling Schemata and Syntactic Description of Pictures, Inf. and Control, vol. 7., 1964. pp. 151-179. /русский перевод в: Анализ сложных изображений, Мир. Москва, 1969/.

[4] Rosenfeld A.: Picture Processing by Computer, Academic Press, 1969.

[5] Kulpa Z. Szydlo H.: PICTURE ALGOL 1204 Ab - jezyk przetwarzania informacji graficznej. Prace ICS PAN, 1972.

[6] Kulpa Z.: A Picture Processing System PICTURE ALGOL 1204, Proc. VIII-th Yugoslav International Symp. on Inf. Proc. /FCIP 72/, Bled, October 1972 /paper nr a3/.

[7] Kulpa Z.: Algorytmy pocieniania linii. Prace ICS PAN, 1972.

[8] Kulpa Z.: Jezyk przetwarzania obrazów PAL, opis koncepcji. Opracowanie wewn. ICS PAN, 1972.

[9] Michalski R.S.: A Variable - Valued Logic System as Applied to Picture Description and Recognition, Proc. of the IFIP Conf. on Graphic Lang., Vancouver, Canada, May, 1972.

Декомпозиция комбинационных автоматов

М.И. Гаврилов, чл.корр.АН СССР [x]

В статье рассматриваются методы декомпозиции произвольных комбинационных автоматов, условия работы которых заданы в виде булевой функции в нормальной или скобочной форме, на произвольные подавтоматы, заданные в такой -же форме.

Последние годы характеризуются существенным изменением требований к синтезу дискретных автоматов в связи с ростом размерностей практических задач и усложнением структурных свойств элементов.

При большой размерности решаемых задач классические формы задания условий работы дискретных автоматов в виде таблиц переходов, таблиц состояний и т.п., также как и алгоритмы, связанные с необходимостью оперирования отдельными состояниями, становятся практически непригодными в связи с их чрезмерной громоздкостью.

При формулировке условий практических задач обычно применяются существенно более компактные формы записи в виде списков функций возбуждений и выходов или в виде функций переходов и выходов, выраженных в нормальной или скобочной форме булевых функций. Это однако требует достаточно коренного пересмотра большинства из существующих методов, которые не приспособлены к такой форме записи.

Существенные изменения в методы синтеза вносит также усложнение структурных свойств элементов.

Помимо тех же затруднений, что и выше, связанных большой размерностью элементов по числу входов и необходимостью компактных форм описания их структурных свойств, возникают определенные трудности, связанные с быстрым изменением самых структурных свойств элементов. В этих случаях разработка методов синтеза, ориентированных на какую-либо определенную элементную базу,

Институт проблем управления /ИАТ/, Москва

становится нерациональной, так как требует переработки соответствующих процедур синтеза каждый раз при появлении какого либо нового базиса. Поэтому все большее значение приобретают универсальные методы, расчитанные на любые виды элементов.

Поскольку элемент с большим числом входов представляет собой по существу сам по себе некоторый автомат, то в общем случае задачу синтеза структуры на таких элементах можно трактовать как задачу декомпозиции заданного дискретного автомата на заданные подавтоматы.

В настоящей работе эта задача рассматривается применительно к т.н. "комбинационным" автоматам, т.е. таким, которые не имеют элементов памяти.

Существующие методы декомпозиции комбинационных автоматов на подавтоматы, можно подразделить на две группы:

а/ методы, основанные на принципах т.н. "подбора" и заключающиеся в сравнении заданной функции, характеризующей условия работы автомата, с функцией, характеризующей структурные свойства подавтомата (или частями ее), и "припасовывания" этих функций друг к другу путем выбора комбинаций входных переменных и констант, подаваемых на входы подавтомата;

б/ методы, основанные на принципах т.н. "направленного поиска" оптимальной декомпозиции и заключающиеся в выборе входных переменных, подаваемых на входы подавтоматов с помощью некоторых оценочных критериев.

Обе группы методов пригодны как для одновыходных, так и для многовыходных комбинационных автоматов. Ниже, однако, для простоты они будут рассматриваться применительно к одновыходным автоматам.

Проанализируем проблемы, возникающие при декомпозиции комбинационных автоматов на произвольные подавтоматы, на примере рассмотрения методов первой группы, наиболее приближающихся к интуитивным методам, применяемым в инженерной практике и обладающих присущими им недостатками, которые должны быть устранены в оптимальных методах.

В методах подбора припасовывание функций, характеризующих структурные свойства подавтоматов, к функции, характеризующей заданный комбинационный автомат, производится или на основе графического представления этих функций [I] или в аналитическом виде. В последнем случае функции, реализуемые заданными подавтоматами, разлагаются на более простые бесповторные функции с помощью раскрытия скобок или замены входных переменных константами. Затем

полученные бесповторные функции сравниваются с функцией (или функциями) характеризующей заданный комбинационный автомат, и путем подбора, в частности ранжирования длин конъюнкций и дизъюнкций заданной функции, определяются наиболее подходящие члены бесповторных функций, полученных при разложении. При этом, если ранг конъюнкций или дизъюнкций заданной функции оказывается меньшим ранга конъюнкции или дизъюнкции бесповторных функций элемента, то на избыточные входы элемента подаются константы, а если большим, то применяется многоуровневая реализация. В случае задания реализуемой функции в скобочной форме применяется многоуровневая реализация: в первую очередь указанным выше путем реализуются наиболее глубокие скобки. Затем они заменяются дополнительной переменной, с учётом этой переменной реализуются скобки следующей глубины и т.д. до полной реализации заданной функции.

Наиболее существенным принципиальным недостатком этого метода является то, что он предусматривает реализацию функции в форме, заданной проектировщиком, без какого либо преобразования ее.

Укажем, что в ряде случаев, как показывает практика, такое задание может быть сильно избыточным и не отвечать структурным свойствам подавтоматов, на которых реализуется заданный автомат.

Поэтому возникают следующие задачи:

а/ удавление избыточности в заданной функции и

б/ преобразование ее в форму, наиболее оптимальную к декомпозиции на заданных подавтоматах.

Уже сама задача устранения избыточности (определения т.н. "тупиковых форм) при большом числе переменных является исключительно громоздкой в связи с быстрым ростом числа членов тупиковых форм (т.н. "первичных имплицантов), резким увеличением размерности т.н. "таблиц покрытий", применяемых для решения этих задач в классических методах минимизации булевых функций, и усложнением задачи определения минимального покрытия [2] .

Ещё более сложным является задача преобразования заданной булевой функции к виду, наиболее оптимальному для реализации на заданных подавтоматах, поскольку, как было показано в [3] , для образования, например, оптимальной скобочной формы булевой функции, что в принципе становится необходимым уже при двухровневой реализации заданного автомата, необходимо перебрать все возможные расширения полученной тупиковой формы.

Следует указать ещё на одно обстоятельство, связанное с заданием комбинационного автомата булевой функцией в нормальной или скобочной фор-

мах. В этом случае по существу возможные реализации ограничиваются лишь определенной областью решений, в которую наиболее оптимальные решения могут и не входить. Последние могут быть найдены только с помощью перехода к таблице состояний, построение которой для большого числа переменных практически невозможно в связи с большой размерностью ее.

Поэтому возникает задача получения всей информации для оптимальной реализации заданного комбинаторного автомата на заданных подавтоматах и содержащейся в таблицах состояния, не строя последнюю, а используя лишь некоторые модификации заданных булевых функций, характеризующих комбинационный автомат и подавтоматы, на которых он должен быть реализован, и определяя необходимые параметры таблиц состояний путем подсчета.

Отметим, что реализация комбинационных автоматов, на основе функции заданной проектировщиком, в ряде случаев может привести к структуре, существенно отличающейся от оптимальной. Рассмотрим три модификации одной из функций, характеризующих рабочие состояния комбинационного автомата.

$$F_1 = x_5 + x_6 + x_1x_2\bar{x}_3 + x_1x_2\bar{x}_4 + x_1x_2\bar{x}_7 + x_1x_2\bar{x}_8 + x_1x_2\bar{x}_9 + x_1x_2x_{10}x_{11} \quad \ldots \text{(Iа)}$$

$$F_2 = x_5 + x_6 + x_1x_2(\bar{x}_3 + \bar{x}_4 + \bar{x}_7 + \bar{x}_8 + \bar{x}_9 + x_{10}x_{11}) \quad \ldots \text{(Iб)}$$

$$F_1 = x_5 + \bar{x}_1\bar{x}_5x_6 + x_1(\bar{x}_5x_6 + x_2\bar{x}_5\bar{x}_6x_{10}x_{11}) + x_1x_2(\bar{x}_3 + \bar{x}_4 + \bar{x}_7 + \bar{x}_8 + \bar{x}_9) \quad \ldots \text{(Iв)}$$

Пусть эти функции нужно реализовать на подавтомате, характеризующемся функцией:

$$Y = \bar{y}_1 + \bar{y}_2 + \bar{y}_3 + y_4y_5y_6$$

На рис. I, 2 и 3 приведены реализации указанных выше модификаций методом подбора .

В таблице № I приведены данные о необходимом количестве подавтоматов и эффективности их использования.

Таблица № I

Модификация функции	Количество подавтоматов	Количество входов подавтоматов, заполненных нулями и единицами
(а)	II	35
(б)	5	I7
(в)	8	22
(а), (б), (в)	3	3

В последней строке таблицы № I представлены результаты реализации всех трех модификаций с помощью алгебраического метода направленного поиска оптимальной декомпозиции который будет описан ниже и который дает одинаковую реализацию вне зависимости от того в какой модификации дана функция, описывающая заданный комбинационный автомат. Реализация автомата в этом виде представлена на рис 4.

Отметим, что такая реализация при методе подбора требует представления заданной функции в виде:

$$F_1 = x_5 + x_6 * \overline{(\bar{x}_1 + \bar{x}_2) x_3 x_4 x_7} + x_1 x_2 (\bar{x}_8 + \bar{x}_9 + x_{10} x_{11}) \qquad \ldots(Iг)$$

что существенно отличается от представленных выше первоначальных форм.

Упомянем еще о нескольких недостатках, присущих методу подбора, как и всем интуитивным методам, основанным на реализации заданной формы булевой функции, характеризующей комбинационный автомат.

а/ заданная функция реализуется как полностью определенная, что при наличии недоопределенных состояний может привести к избыточной реализации.

б/ Необходиомсть разложения функций, характеризующих подавтоматы, на бесповторные обедняет их структурные свойства. Например, свойства элемента, реализующего функцию суммы по модулю два, выраждаются в этом случае в функцию повторителя или инвертора.

в/ Не решается задача выбора оптимального подавтомата на выходе структуры из числа заданного базиса подавтоматов.

г/ Не предусматривается никаких процедур для выравнивания глубины получаемых структур, что в ряде случаев является практически весьма важным.

Перейдем теперь к рассмотрению методов направленного поиска.

Предложенные до настоящего времени методы, относящиеся к этой группе, базируются в основном на некоторой гипотезе, заключающейся в том, что если при реализации структуры выбирать на каждом этапе совокупность переменных и констант, подаваемых на входы выходного подавтомата, таким образом, чтобы функция, реализуемая на его выходе, получалась возможно более близкой к заданной функции, то реализация всей заданной функции в целом на заданном наборе элементов будет получаться наиболее простой.

Если реализация заданной функции на выбранном наборе переменных будет противоречивной, то переменные на входах выходного подавтомата частично или полностью заменяются т.н. "доопределяющими" функциями; которые уст-

раняют эту противоречивность. I/ Для каждой доопределяющей функции снова подбирается наиболее оптимальный выходной подавтомат и т.д. Следует указать, что методы направленного поиска, так-же как и методы подбора дают лишь приближенное решение задачи оптимальной декомпозиции комбинационных автоматов. В методах подбора это требует полного перебора всех возможных решений, а в методах направленного поиска зависит от точности критериев, применяемых на каждом этапе для выбора оптимального подавтомата. Как показывает практика, такие методы "локальной" оптимизации дают достаточно удовлетворительные результаты, однако требуют отыскания достаточно хороших критериев оптимизации для всех этапов синтеза.

В разработке методов направленного поиска в настоящее время имеются два направления.

В первом из них [5, 6] сформулированная выше гипотеза используется в прямом смысле, т.е. на входы заданных подавтоматов подаются все возможные комбинации переменных реализуемой функции и констант 0 и I, и определяются реализуемые при этом на выходе функции. Затем с помощью специального критерия определяется близость каждой из этих функций к реализуемой функции и из всех возможных функций выбирается по полученным значениям критерия наиболее близкая к реализуемой функции.

Основным недостатком этого метода является необходимость определения на каждом этапе всех возможных функций, реализуемых каждым из подавтоматов.

Например, в общем случае для подавтоматов с несимметричными входами необходимо на каждом этапе определять и сравнивать с заданной функцией число функций, равное:

$$R = q! C_{n+2}^{q}$$

где: n -число входных переменных и q -число входов подавтомата.

Для подавтомата с 6-ю входами и двумя группами входов , в пределах которых каждые три входа являются симметричными, приведенного в рассмотренном

I/ Метод "доопределения" членов, противоречиво реализующих заданную функцию, был впервые предложен автором в [4] , но только в последние годы получил практическое применение.

выше примере, число различных функций, рассматриваемых на каждом этапе, будет равно:

$$R = \frac{q!}{q_1!\, q_2!} C^{q}_{n+2} = \frac{6!}{3!\,3!} C^{6}_{13} = 34.320$$

Это весьма сильно ограничивает размерность решаемых задач, даже при использовании УВМ. В [5,6] этот метод рассматривается применительно к заданию функции комбинационного автомата и подавтоматов, на которых он должен быть реализован, в виде таблиц состояний, что также сужает размерность решаемых задач.

В отличие от только что рассмотренного метода, который основан на принципе "выбора" оптимального подавтомата из всех возможных, в другом направлении, использующем направленный поиск, применяется метод "конструирования" оптимального автомата с помощью определения совокупности переменных и констант, подаваемых на входы выходного подавтомата. Перебор в этом случае ограничивается числом переменных функций, описывающий заданный комбинационный автомат, и растет линейно с ростом их числа, что позволяет существенно увеличить размерность решаемых задач.

Основные идеи этого метода применительно к заданию функций, описывающих заданный комбинационный автомат и подавтоматы с помощью таблиц состояний, изложены в [7, 8]. Особенности этого метода заключаются в следующем:

I. Комбинационный автомат и подавтоматы на которых он должен быть реализован задаются в виде некоторого множества определенных состояний (рис. 5а):

$M(f_i) = M_1(f_i) \cup M_0(f_i)$ и $H(\mathcal{P}_j) = H_1(\mathcal{P}_j) \cup H_0(\mathcal{P}_j)$

где: $M_1(f_i)$ – множество состояний входных переменных, для которых на i-М – выходе комбинационного автомата должно быть получено воздействие, равное единице (т.н. "рабочие" состояния), а

$M_0(f_i)$ – множество состояний входных переменных, для которых на i-М выходе должно быть получено воздействие, равное нулю (т.н. "нерабочие" состояния),

$H_1(\mathcal{P}_j)$ – множество состояний входов j-го подавтомата, для которых воздействие на его выходе равно единице, и

$H_0(\mathcal{P}_j)$ – множество состояний j-го подавтомата, для которых воздействие на выходе равно нулю.

Очевидно, что задача реализации на некотором выходном подавтомате $\mathcal{P}_j$ функции f_i, заданной этими множествами, будет решена, если на его входы будет подан такой набор входных переменных и констант, или функции от них, или и тех и других одновременно, что для каждого из состояний, принадлежащих множеству $M_1(f_i)$, на выходе этого подавтомата будет появляться воздействие, равное единице, а для каждого из состояний, принадлежащих $M_0(f_i)$ -воздействие, равное нулю.

Назоваем "переходной" таблицей состояний таблицу, содержащую $N_1(f_i)$ строк, соответствующих всем состояниям из $M_1(f_i)$, $N_0(f_i)$ строк, соответствующих всем состояниям из $M_0(f_i)$ и $q(\mathcal{P}_j)$ столбцов, где $q(\mathcal{P}_j)$ - число входов подавтомата $\mathcal{P}_j$ (рис.5б). Подадим на входы подавтомата некоторую совокупность входных переменных и проставим в клетках переходной таблицы значения этих переменных, соответствующие значениям их в строчках, принадлежащих $M_1(f_i)$ и $M_0(f_i)$ (в прямом или обратном виде, в зависимости от того в каком виде - прямом или инверсном подана переменная на вход подавтомата). Если при этом комбинации этих значений для строк, входящих в $M_1(f_i)$ будут образовывть в переходной таблице состояния, входящие в $H_1(\mathcal{P}_j)$, а для строк, входящих в $M_0(f_i)$ -состояния, входящие в $H_0(\mathcal{P}_j)$, то сформулированные выше условия реализации функции будут полностью выполнены, а структура, реализующая эту функцию, будет состоять из одного подавтомата $\mathcal{P}_j$ с поданной на его входы указанной совокупностью входных переменных.

Если совокупности входных переменных, удовлетворяющих этим условиям, не существует, или с помощью соответствующих критериев их не удастся подобрать, то для части или всех состояний из $M_1(f_i)$ и из $M_0(f_i)$ будет получаться противоречивая реализация, т.е. состояния переходной таблицы в строках, принадлежащих $M_1(f_i)$, будет входить в подмножество $H_0(\mathcal{P}_j)$, а в строках принадлежащих $M_0(f_i)$, -входить в подмножество $H_1(\mathcal{P}_j)$. В этом случае некоторые или все переменные должны быть заменены т.н. "доопределяющими" функциями f_{yi}, устраняющими противоречивость реализации. Анализ состава нулей и единиц в каждой строке переходной таблицы и сравнение его с рабочими состояниями подавтомата, позволяют определить таблицу состояний доопределяющих функций для каждого вида выходного подавтомата. Реализация таблиц состояния доопределяющих функций производится также, как и таблицы состояний первоначальной функции.

Переходную таблицу, в которой входные переменные заменены необходимыми доопределяющими функциями и в которой полностью соблюдены условия непротиворечивой реализации заданной функции f_i -будем называть "приведенной" таблицей функции f_i по выходному элементу $\mathcal{P}_j$.

Реализация доопределяющих функций производится послойно, т.е. при замене переменных, поданных на входы выходного автомата доопределяющими функциями, в каждой из этой функции реализуется лишь один выходной подавтомат. Если при подаче функций, реализуемых этими подавтоматами, на все входы основного выходного подавтомата непротиворечивая реализация состояний $M_1(fi)$ и $M_0(fi)$ не будет еще получена, переходят к заполнению второго слоя подавтоматов и т.д.

Указанная последовательность реализации доопределяющих функций обеспечивает получение равномерной глубины структуры и, кроме того, сходимость процесса реализации даже в тех случаях, когда множество $M_1(fy_i)$ и $M_0(fy_i)$ доопределяющих функций получаются равными с множествами $M_1(fi)$ и $M_0(fi)$ реализуемых функций.

Определение таблиц состояний доопределяющих функций на основе переходной таблицы и с учетом реализации состояний $M_1(fi)$ и $M_0(fi)$, полученной на предыдущих этапах обеспечивает получение безизбыточной реализации заданной функции.

2. Выбор переменных, подаваемые на входы выходного подавтомата, выбор очередности реализации выходов заданного комбинационного автомата, очередность зап · ения входов выходного подавтомата в случае его несимметричности и т.д. осуществляется с помощью ряда следующих критериев.

а/ Выбор очередности реализации выходов заданного комбинационного автомата производится с помощью критерия предложенного Л. Шеломовым [9] и дающего общую оценку сложности реализации булевой функции.

$$I_i = N(fi)\log_2 N(fi) - N_1(fi)\log_2 N_1(fi) - N_0(fi)\log_2 N_0(fi) \qquad \ldots(2)$$

где: i -номер выхода,

$N(fi)$ -число состояний в $M(fi)$,

$N_1(fi)$ -число состояний в $M_1(fi)$ и

$N_0(fi)$ -число состояний в $M_0(fi)$.

Для реализации в первую очередь выбирается выход с наименьшим значением.

Предполагается, что указанная последовательность оптимальна для связанной реализации выходных функций, при которой выходы подавтоматов, реализующих какую либо одну из выходных функций автомата, могут быть использованы в реализации какой-либо одной или нескольких других выходных функций.

б/ Выбор оптимальной совокупности переменных, подаваемых на входы выходного подавтомата, определяется в соответствии с критерием:

$$b_к = d_{1к} - d_{0,к} \qquad d_{1,к} = \frac{n'_{1,к}}{N_1(fi)} \qquad d_{0,к} = \frac{n'_{0\,1,к}}{N_0(fi)} \qquad \ldots (3)$$

где: $N_1(fi)$ и $N_0(fi)$ число рабочих и нерабочих состояний реализуемой функции, $n'_{1,к}$ и $n'_{0,к}$ число рабочих и нерабочих состояний реализуемой функции, в которых соответственно переменная $x_к$ имеет значение единицы.

В первую очередь выбираются переменные $x_к$ с максимальным значением по абсолютной величине критерия $b_к$. При этом, если $b_к > 0$, то переменная $x_к$ подается на вход с неинверсным значением, а если $b_к < 0$, то с инверсным значением. Если при подаче на вход очередной переменной некоторые состояния из $M_1(fi)$ и $M_0(fi)$ оказываются реализованными, то критерии для последующих переменных подсчитываются без учета этих состояний [I/].

в/ Выбор оптимального выходного подавтомата производится по критерию

$$Q_i = \frac{n^*_{1,i}}{N_1(fi)} + \frac{n^*_{0,i}}{N_0(fi)} \qquad \ldots (4)$$

где $n^*_{1,i}$ и $n^*_{0,i}$ –число противоречиво реализованных состояний соответственно в $M_1(fi)$ и $M_0(fi)$ переходной таблицы i –го выхода. Выбирается подавтомат, функция выхода которого наиболее близка к реализуемой функции, т.е. для которого критерий Q_i имеет наименьшее значение.

г/ Выбор оптимального порядка доопределения переменных, поданных на входы выходного подавтомата.

Очередность замены переменных, поданных на входы выходного элемента, доопределяющими функциями определяется по критерию

$$n_{эyi} = \frac{\Delta n_{1,y} + \Delta n_{0,yi}}{N_{1,yi} + N_{0,yi}} \qquad \ldots (5)$$

где: $N_{1,yi}$ и $N_{0,yi}$ – число состояний соответственно в рабочей и нерабочей частях таблицы состояний доопределяющей функции на входе yi а $\Delta n_{1,yi}$ и $\Delta n_{0,yi}$ – число состояний соответственно в рабочей и нерабочей частях этой таблицы, требующих устранения противоречивости.

В первую очередь производится доопределение входа, для которого $n_{эyi}$–имеет максимальное значение. Подсчет $n_{э,yi}$ производится заново после

I/ Для автоматов с несимметричными входами разработаны критерии, которые одновременно с определением оптимальных переменных определяют также и оптимальные входы подавтомата, на которые они должны быть поданы. Эти критерии базируются на тех-же параметрах таблиц состояний, что и критерии (3), но определяемых не только для таблицы состояний выходов заданного комбинационного автомата, но также и таблиц состояний, характеризующих подавтоматы.

реализации доопределяющей функции каждого из предыдущих входов, поскольку это изменяет значение $n_{э,yi}$ для всех остальных, еще нереализованных доопределяющих функций.

д/ Выбор оптимальной связности структуры между отдельными её выходами и выходами отдельных её частей.

Выбор оптимальной связности производится с помощью критерия (3). Для этого выход каждого из элементов в уже реализованной части структуры, включая и реализованный выход структуры в целом, принимается за новую переменную, которая присоединяется к входным переменными и рассматривается наравне с ними при подсчете критерия (3). Выгодность использования этой переменной, а вместе с тем и выгодность введения связности определяется значением критерия $b_к$ для неё.

Как уже указывалось в начале настоящей статьи, одним из наиболее эффективных методов преодоления трудностей, связанных с большой размерностью практических задач, является использование наиболее компактных форм записи условий работы заданного комбинационного автомата и структурных свойств в подавтоматов в виде нормальных или скобочных форм булевых функций. При этом возникает ряд задач, связанных с переводом всех операций рассмотренного выше алгоритма в алгебраическую форму.

Рассмотрим некоторые основные из этих задач.

Как уже упоминалось, одной из основных операций при переходе к алгебраическому методу синтеза является определение по заданным функциям, записанным в аналитическом виде, значения критериев, вычисляемых на основе соответствующих этим функциям таблиц состояний подавтоматов, переходных таблиц и т.п.

Обозначим через $F_{1,i}$ и $F_{0,i}$ функции, характеризующие соответственно множества рабочих и нерабочих состояний $M_1(fi)$ и $M_0(fi)$ i-го выхода заданного комбинационного автомата. Эти функции характеризуют, в частности, и параметры этих множеств, но для удобного подсчета необходимо приведение заданных функций к канонической форме, при которой множество состояний, входящее в данный критерий, могло-бы определяться как сумма отдельных, независимых друг от друга, частей канонической формы [1/].

1/ Канонические формы, операции и методика преобразования нормальных и скобочных форм были разработаны под руководством автора канд.техн.наук В.М. Копыленко.

Для функций, записанных в нормальной дизъюнктивной форме, канонической формой будет являться такой вид их, при котором все члены нормальной дизъюнктивной формы будут взаимно ортогональны т.е. произведение любых двух конъюнкций U_i и U_j :

$$U_i \cdot U_j = 0$$

Для определения канонической скобочной формы введем некоторые обозначения. Введем нумерацию скобок, обозначая индексом I внешние скобки, заключающие всю функцию в целом, и повышая величину индекса, по мере увеличения глубины скобок. Выражение, заключенное в скобки с i-м индексом будем называть i-м термом дизъюнктивным или конъюнктивным в зависимости от знаков, которые соединяют входящие в него $i+1$-е термы. Для единообразия знаки инверсии над выражениями будем заменять квадратными скобками и нумеровать их подобным-же образом.

Например с учетом указанных обозначений скобочная форма, рассмотренной выше функции F_1 (вариант (Iв) запишется а в виде

$$F_1 = \Big(x_5 + \overline{x_1}\,\overline{x_5} x_6 + \Big(\big(x_1 \big)_3 \big(\overline{x_5} x_6 + x_2 \overline{x_5}\,\overline{x_6} x_{10} x_{11} \big)_3 \Big)_2 +$$
$$+ \Big(\big(x_1 x_2 \big)_3 \big(\overline{x_3} + \overline{x_4} + \overline{x_7} + \overline{x_8} + \overline{x_9} \big)_3 \Big)_2 \Big)_1$$

В целом функция F_1 представляет собой дизъюнктивный терм (скобки с индексом I). В скобках с индексом 2 содержатся конъюнктивный термы, а в скобках с индексом 3 - дизъюнктивные термы.

Канонической скобочной формой, пригодной для определения числа состояний, является такая, в которой все члены, входящие в дизъюнктивные термы, являются взаимно ортогональными, а все члены, входящие в конъюнктивные термы, не должны содержать одинаковых переменных (т.е., если какая либо переменная x_k входит в какую либо из скобок i-го конъюктивного терма, то они не должны входить ни в какую другую скобку этого терма). Преобразование к канонической скобочной форме производится с помощью разложения заданной скобочной формы по переменным с помощью формулы Де-Моргана.

Рассмотрим пример. Пусть задано скобочное выражение функций F_1 (вариант в) и кроме того известно, что состояния, характеризуемые функцией:

$$F_\sim = x_3 x_4 + \overline{x_8}\,\overline{x_9}$$

являются неиспользуемыми. Необходимо определить функции F_1' и F'_0, характеризующие рабочие и нерабочие состояния соответствующей недоопределенной таблицы состояний.

Очевидно, что:

$$F_1' = \overline{F_1} - \overline{F_\sim} \qquad \text{и} \qquad F_0' = \overline{F_1} - \overline{F_\sim}$$

Используя для обозначения инверсий квадратные скобки, будем иметь:

$$F_1' = \Big(\big(x_5 + \bar{x}_1 \bar{x}_5 x_6 + \big(\big(x_1 \big) \big(\bar{x}_5 x_6 + x_2 \bar{x}_5 \bar{x}_6 x_{10} x_{11} \big) \big) + \big(\big(x_1 x_2 \big) \big(\bar{x}_3 + \bar{x}_4 + \bar{x}_7 + \bar{x}_8 + \bar{x}_9 \big) \big) \big) \big[x_3 x_4 + x_8 x_9 \big] \Big)$$

$$F_0' = \Big(\big[x_5 + \bar{x}_1 \bar{x}_5 x_6 + \big((x_1)(\bar{x}_5 x_6 + x_2 \bar{x}_5 \bar{x}_6 x_{10} x_{11}) \big) + \big((x_1 x_2)(\bar{x}_3 + \bar{x}_4 + \bar{x}_7 + \bar{x}_8 + \bar{x}_9) \big) \big] \big[x_3 x_4 + \bar{x}_8 \bar{x}_9 \big] \Big)$$

Разлагая эти функции по переменным X_1, X_2, X_3 и X_4 и производя соответствующие преобразования, получим

$$F_1' = \Big(\big(\big(x_1 x_2 x_3 \bar{x}_4 \big) \big[\bar{x}_8 \bar{x}_9 \big] \big) + \big((x_1 x_2 \bar{x}_3) \big[\bar{x}_8 \bar{x}_9 \big] \big) + \big((x_1 \bar{x}_2)(x_5 + \bar{x}_5 x_6)(\bar{x}_3 + x_3 \bar{x}_4) \big[\bar{x}_8 \bar{x}_9 \big] \big) + \big((\bar{x}_1) (x_5 + \bar{x}_5 x_6)(\bar{x}_3 + x_3 \bar{x}_4) \big[\bar{x}_8 \bar{x}_9 \big] \big) \Big) \qquad \ldots (6.1)$$

$$F_0' = \Big(\big((x_1 \bar{x}_2) \big| x_5 + \bar{x}_5 x_6 \big| (\bar{x}_3 + x_3 \bar{x}_4) \big| \bar{x}_8 \bar{x}_9 \big| \big) + \big((\bar{x}_1) \big| x_5 + \bar{x}_5 x_6 \big| (\bar{x}_3 + x_3 \bar{x}_4) \big| \bar{x}_8 \bar{x}_9 \big| \big) \Big) \qquad \ldots (7.1)$$

Рассмотрим теперь методику подсчета числа состояний.

Очевидно, что в дизъюнктивном терме, приведенном к каноничсекой форме, число состояний, соответствующее ему, равно сумме числа состояний, соответствующих конъюнктивным членам, входящих в его состав.

Конъюнктивному члену будет соответствовать число состояний, равное

$$d_i = 2^{n-k} \quad \beta_1, \beta_2 \cdots \beta_m$$

где: n – общее число переменных, к–число переменных, входящих в конъюнктивный терм, и $\beta_1, \beta_2 .. \beta_m$–число состояний, реализуемых дизъюнктивными термами, входящими термами, входящими в состав данного конъюнктивного члена. Если в состав какого либо из термов входит член в квадратных скобках (инверсия), то соответствующий член определяется по формуле:

$$\beta_j = 2^2 - \beta_j^*$$

где: τ –общее число переменных, входящих в этот член, и β_j^* –число состояний, соответствующее выражению, заключенному в квадратные скобки.

Так например, для выражения: $[x_5 + \bar{x}_5 x_6]$ будем иметь:

$\tau = 2$; $\beta^* = 3$. Поэтому $\beta_j = 4-3 = 1$.

Для выражения $[\bar{x}_8 \bar{x}_9]$ получим:

$\tau = 2$; $\beta_j^* = 1$; $\beta_j = 4-1=3$

Перепишем написанные выше выражения для F_1' и F_0', удалив лишние скобки и проставив в качестве верхнего индекса у оставшихся скобок величины 2^{n-k} и β.

$$F_1' = \left(\left(x_1x_2x_3x_4\right)^{2^8}_{12}\left[x_8x_9\right]^{2^4}_{3}\right)^{2^2\cdot3}_{32} + \left(\left(x_1x_2x_3\right)^{2^8}_{2}\left[x_8x_9\right]^{2^5}_{3}\right)^{2^3\cdot3}_{32} + \left(\left(x_1x_2\right)^{2^8}_{2}\left(x_5+x_5x_6\right)^{2^6}_{33}\left(x_3+x_3x_4\right)^{2^4\cdot3}_{33}\left[x_8x_9\right]^{2^2\cdot9}_{32}\right) + \left(\left(x_1\right)^{2^8\cdot2^7}_{1}\left(x_5+x_5x_6\right)^{2^5\cdot3}_{33}\left(x_3+x_3x_4\right)^{2^3\cdot9}_{33}\left[x_8x_9\right]^{2^1\cdot27}_{321}\right) \quad \ldots (6.2)$$

$$F_0 = \left\{\left(x_1x_2\right)^{2^8}_{12}\left[x_5+x_5x_6\right]^{2^6}_{33}\left(x_3+x_3x_4\right)^{2^4\cdot1}_{33}\left[x_8x_9\right]^{2^2\cdot3}_{32} + \left(x_1\right)^{2^8\cdot2^7}_{1}\left(x_5+x_5x_6\right)^{2^5\cdot1}_{33}\left(x_3+x_3x_4\right)^{2^3\cdot3}_{33}\left[x_8x_9\right]^{2^0\cdot9}_{321}\right\} \quad \ldots (7.2)$$

Таким образом числа состояний в рабочей и нерабочей частях таблицы состояний заданной функции будет равно:

$$N_1' = 2^2\cdot3 + 2^3\cdot3 + 2^0\cdot27 + 2^1\cdot27 = 117$$

$$N_0' = 2^0\cdot9 + 2^1\cdot9 = 27$$

Очевидно, что такие-же результаты будут получены при задании функции F_1 , в виде (а), или (в) или в любом другом виде, что указывает, что результаты декомпозиции в рассматриваемом методе не зависят от вида, в котором задается комбинационный автомат.

В тех случаях, когда функции, характеризующие комбинационный автомат, заданы в нормальной форме и желательно проводить все преобразования не переходя к скобочным формам, удобно пользоваться т.н. " Z -операцией, представляющей собой аналог логической операции "правый запрет" I/. Будем обозначать Z-операцию символом * . Тогда соотношение

$$f_1 = f_2 * f_3$$

будет означать функцию, характеризующую таблицу состояний, содержащую состояния, соответствующие функции f_2 , без состояний, характеризующихся функцией f_3 .

Очевидны следующие состояния:

$$1 * f = \bar{f} \quad \ldots (8.1)$$

$$0 * f = 0 \quad \ldots (8.2)$$

$$f * 1 = 0 \quad \ldots (8.3)$$

$$f * 0 = 0 \quad \ldots (8.4)$$

$$f * f = 0 \quad \ldots (8.5)$$

$$f * \bar{f} = f \quad \ldots (8.6)$$

Если обозначть через U_i , U_j и $U_к$ некоторые конъюнкции, входящие в нормальную дизъюнктивную форму, то легко получить также следующие состояния

$$U_i * (U_j + U_к) = U_i * U_j * U_к \quad \ldots (8.7)$$

$$(U_i + U_j) * U_к = U_i * U_к + U_j * U_к \quad \ldots (8.8)$$

I/ Z-операция была разработана В.М. Копыленко совместно с автором.

Если $U_j \geqslant U_i$, т.е. множество букв, содержащееся в U_j , полностью совпадает, а по числу равно или меньше множества букв, содержащихся в U_i то:

$$U_i * U_j = 0$$

Если, наконец, U_i и U_j имеют общую часть U_i' и буквы $x_{p1}, x_{p2} \ldots x_{pt}$ дополняют U_i' до U_j , то справедливое соотношение:

$$U_i * U_j = U_i \left(x_{p1} \overset{t}{\vee}_{s=2} \bar{x}_{ps} \overset{s-1}{\wedge}_{j=1} x_{pj}\right) \qquad \ldots (3.10)$$

Это соотношение справедливо и в том случае, если $U_i' = \emptyset$.

Для получения по заданной функции F_1 , функции F_0 достаточно очевидно применить соотношение (3.1) в виде:

$$F_0 = \overline{1} * F_1 = (x_i + \bar{x}_i) * F$$

при этом в качестве x_i полезно взять переменную, наиболее часто входящую в члены F_1 . Отметим, что использование соотношения (3.10) позволяет получить F_0 непосредственно в каноническ ой форме, т.е. со взаимно ортогональными членами.

Для преобразования в каноническую форму функции F_1 необходимо соотношение (3.1) использовать дважды, т.е. сперва получить каноническую форму для F_0 , а затем произведя Z -операцию в виде:

$$F_1 = 1 * F_0$$

получить каноническую форму для F_1 .

Рассмотрим в качестве примера ту-же функцию F_1 , но в виде (а) и ту-же функцию $F_\sim$, что и в предыдущем примере.

Очевидно, что

$$F_0' = (1 * F_1) * F_\sim = F_0 * F_\sim$$

Приняв $I = X_1 + \overline{X_1}$, определим сперва F_0 .

$$F_0 = (x_1 * x_5 * x_6 * x_1 x_2 \bar{x}_3 * x_1 x_2 \bar{x}_4 * x_1 x_2 \bar{x}_7 * x_1 x_2 \bar{x}_8 * x_1 x_2 \bar{x}_9 * x_1 x_2 x_{10} x_{11}) + (\bar{x}_1 * x_5 * x_6) = x_1 \bar{x}_5 \bar{x}_6 \bar{x}_2 +$$
$$+ x_1 \bar{x}_5 \bar{x}_6 x_2 x_3 x_4 x_7 x_8 x_9 \bar{x}_{10} + x_1 \bar{x}_5 \bar{x}_6 x_2 x_3 x_4 x_7 x_8 x_9 x_{10} \bar{x}_{11} + \bar{x}_1 \bar{x}_5 \bar{x}_6$$

Отсюда:

$$F_0' = F_0 * F_\sim = (x_1 \bar{x}_5 \bar{x}_6 \bar{x}_2 + x_1 \bar{x}_5 \bar{x}_6 x_2 x_3 x_4 x_7 x_8 x_9 \bar{x}_{10} + x_1 \bar{x}_5 \bar{x}_6 x_2 x_3 x_4 x_7 x_8 x_9 x_{10} \bar{x}_{11} + x_1 \bar{x}_5 x_6) * x_3 x_4 *$$
$$* \bar{x}_8 \bar{x}_9 = x_1 \bar{x}_5 \bar{x}_6 \bar{x}_2 x_3 \bar{x}_8 + x_1 \bar{x}_5 \bar{x}_6 x_2 x_3 \bar{x}_8 x_9 + x_1 \bar{x}_5 \bar{x}_6 x_2 x_3 x_4 \bar{x}_8 + x_1 \bar{x}_5 \bar{x}_6 x_2 x_3 x_4 \bar{x}_8 \bar{x}_9 +$$
$$+ \bar{x}_1 \bar{x}_5 \bar{x}_6 \bar{x}_3 \bar{x}_8 + \bar{x}_1 \bar{x}_5 \bar{x}_6 \bar{x}_3 x_8 x_9 + \bar{x}_1 \bar{x}_5 \bar{x}_6 \bar{x}_3 x_4 \bar{x}_8 + \bar{x}_1 \bar{x}_5 \bar{x}_6 \bar{x}_3 \bar{x}_4 \bar{x}_8 x_9$$

Число состояний

$$N_0' = 2^3 + 3.2^2 + 3.2^1 + 2^0 = 27$$

Аналогично определяется и каноничная форма для F_1' по формуле:

$$F_1' = (1 * F_0) * F_{\sim}$$

Каноническая форма позволяет достаточно просто вычислить все необходимые параметры таблиц состояний для определения критериев направленного поиска. Проиллюстрируем это на примере вычисления критерия $b_к$. Для определения этого критерия необходимо знать, как это следует из формулы (2), величины: N_1, N_0, n_1^1, n_0^1.

Для определения величин n_1^1 и n_0^1 достаточно положить в формулах F_1' и F_0' соответствующую переменную x_i, равной единице.

Определим в качестве примера значения n_1^1 и n_0^1 для выражений (6.I) и (7.I) по отношению к переменной x_i.

$$F_{1,x}' = ((x_2 x_3 \bar{x}_4 [\bar{x}_8 \bar{x}_9]) + (x_2 x_3 [\bar{x}_8 \bar{x}_9]) + (\bar{x}_2 (x_5 + x_5 x_6) (\bar{x}_3 + x_3 \bar{x}_4) [\bar{x}_8 \bar{x}_9]))$$

Отсюда: $n_1^1 = 2^2.3 + 2^3.3 + 2^0.27 = 63$

$$F_{0,x}' = (x_2 [x_5 + x_5 x_6] (x_3 + x_3 x_4) [x_8 x_9])$$

Отсюда: $n_1^1 = 9$

Таким образом

$$b_к = \frac{63}{117} - \frac{9}{27} = 0{,}15$$

Аналогично определяются и другие параметры, входящие в перечисленные выше критерии.

Одной из основных операций в рассматриваемом алгоритме является определение доопределяющих функций, которые должны быть поданы на входы выходного подавтомата вместо выбранных на первом этапе алгоритма переменных. Метод доопределения позволяет сделать алгоритм интеративным путем применения к подавтоматам, разрешаемым в I-м, 2-м и т.д. уровнях структуры, тех-же операций, что и к выходному подавтомату.

Суть противоречивой реализации заключается в том, что при заполнении переходной таблицы значениями выбранных переменных в строках этой таблицы в рабочей части могут оказаться состояния из $H_0(\mathcal{Y}_j)$ выходного подавтомата, а в нерабочей части состояния из $H_1(\mathcal{Y}_j)$. При определении рабочих и нерабочих состояний доопределяющей функции для какого либо столбца переходной таблицы анализируют состав нулей и единиц в ней и определяют какое значение должна обеспечить доопределяющая функция для того, чтобы ликвидировать

имеющуюся противоречивость. Очевидно, что, если правильная реализация в какой либо строке переходной таблицы обеспечивается в других столбцах, то данное состояние должно быть отнесено в доопределяющей функции данного столбца к неиспользуемым состояниям. Для произвольных подавтоматов в рабочих и нерабочих частях переходной таблицы переходов при устранении противоречивости в рассматриваемом столбце в различных строках может понадобиться обеспечение реализации в доопределяющей функции как нулей, так и единиц. Поэтому при построении доопределяющей функции определяют первоначально некорую порождающую функцию, содержащую как рабочие, так и нерабочие состояния доопределяющей функции, а затем из нее определяют функции, соответствующие рабочей и нерабочей части таблицы состояний доопределяющей функции.

Порождающая функция определяется с помощью выражения:

$$\Psi y_i = (\Gamma_1 * \mathcal{S}_1[y_i] * \mathcal{S}_1(\overline{y_i})) + \Gamma_0 * \mathcal{S}_0[y_i] * \mathcal{S}_0(\overline{y_i}) \qquad \ldots (9.1)$$

или

$$\Psi y_i = \Gamma_1 \cdot \overline{\mathcal{S}_1[y_i]} \cdot \mathcal{S}_1(y_i) + \Gamma_0 \cdot \overline{\mathcal{S}_0[y_i]} \cdot \mathcal{S}_0(y_i) \qquad \ldots (9.2)$$

где: $\mathcal{S}_1|y_i|$ и $\mathcal{S}_0|y_i|$ -члены функций $\mathcal{S}_1$ и $\mathcal{S}_0$, не содержащие символов y_i после замены переменной, поданной на вход y_i, символом y_i и $\mathcal{S}_1[y_i]$ и $\mathcal{S}_0(y_i)$ -члены этих функций содержащие y_i. При этом во всех членах, содержащих незаполненные символы входов, эти символы независимо от их вхождения в инверсном или неинверсном виде заменяются единицами. Выражения характеризующие рабочие и нерабочие состояния доопределяющей функции определяются как:

$$\Gamma_{1,y_i} = \Psi_{y_i}(y_i = 1) \qquad \Gamma_{0,y_i} = \Psi_{y_i}(y_i = 0) \qquad \text{I/}$$

Доопределяющие функции определяются отдельно для инверсного и неинверсного значения входа y_i для выбора в дальнейшем наиболее оптимальной из них. При этом при определении порождающей функции Ψ_{y_i} для y_i в $\mathcal{S}_1(y_i)$ нужно брать члены с y_i, а в $\mathcal{S}_0(y_i)$ с $\overline{y}_i$, а при определении порождающей функции для y_i нужно брать наоборот, для $\mathcal{S}_1(y_i)$ -члены с $\overline{y_i}$ и для $\mathcal{S}_0(y_i)$ члены с y_i.

Рассмотрим пример. Пусть необходимо реализовать функцию:

$$\Gamma_1 = x_1x_2 + \overline{x}_1x_2x_3 + \overline{x}_2\overline{x}_3\overline{x}_4$$

$$\Gamma_0 = \overline{x}_1x_2\overline{x}_3 + \overline{x}_2x_3 + \overline{x}_2\overline{x}_3x_4$$

I/ Если при этом функции Γ_{1,y_i} и Γ_{0,y_i} получаются противоречивыми, т.е.: $\Gamma_{1,y_i}\ \Gamma_{0,y_i} \neq 0$, то состояния, на которых они противоречивы, удаляются соотственно из Γ_{1,y_i} и Γ_{0,y_i} применением, например, Z -операции.

на подавтомате: $\mathcal{P}_1 = y_1 y_2 + \bar{y}_1 \bar{y}_2 \bar{y}_3 + y_2 y_3 \bar{y}_4$

$\mathcal{P}_0 = \bar{y}_1 y_2 \bar{y}_3 + \bar{y}_1 y_2 y_3 y_4 + \bar{y}_2 y_3 + y_1 \bar{y}_2 \bar{y}_3$

Пусть для этого подавтомата уже определена переменная X_2 подаваемая на входы y_2 и y_3 . Вход y_4 еще не заполнен. Определим доопределяющие функции для неинверсного и инверсного значения входа y_1 .

При указанном заполнении входов y_2 и y_3 функции $\mathcal{P}_1$ и $\mathcal{P}_0$ примут вид:

$\mathcal{P}_1 = y_1 x_2 + \bar{y}_1 \bar{x}_2 + x_2 \bar{y}_4$

$\mathcal{P}_0 = \bar{y}_1 x_2 y_4 + y_1 \bar{x}_\varepsilon$

Для неинверсного значения y_1 получим:

$\mathcal{P}_1[y_1] = x_2 \bar{y}_4 = x_2 \qquad \mathcal{P}_1[y_1] = y_1 x_2 \qquad \overline{\mathcal{P}_1[y_1]} = x_2 \bar{y}_1 + \bar{x}_\varepsilon$

В этом случае в первой части порождающей функции получим

$\Gamma_1 * x_2 * \bar{x}_2 * x_2 \bar{y}_1 = 0$

Это говорит о том, что доопределяющая функция для неинверсного значения входа y_i не реализует ни одного из рабочих состояний функции Γ_1 . Поэтому использовать доопределяющую функцию для неинверсного значения в этом случае нецелесообразно.

Для инверсного значения y_1 получим

$\mathcal{P}_1[\bar{y}_1] = x_2 \qquad \mathcal{P}_1[\bar{y}_1] = \bar{y}_1 \bar{x}_2 \qquad \overline{\mathcal{P}_1[y_1]} = x_2 + \bar{x}_2 y_1$

$\mathcal{P}_0[y_1] = 0 \qquad \mathcal{P}_0[y_1] = y_1 \bar{x}_2 \qquad \overline{\mathcal{P}_0[y_1]} = x_2 + \bar{x}_2 \bar{y}_1$

$\Psi = ((x_1 x_2 + \bar{x}_1 x_2 x_3 + \bar{x}_2 \bar{x}_3 \bar{x}_4) * x_2 * x_2 * \bar{x}_2 y_1) + (\bar{x}_1 x_2 \bar{x}_3 + \bar{x}_2 x_3 + \bar{x}_2 \bar{x}_3 x_4) * x_2 * \bar{x}_2 \bar{y}_1) = \bar{x}_2 \bar{x}_3 \bar{x}_4 \bar{y}_1 + \bar{x}_2 x_3 y_1 + \bar{x}_2 \bar{x}_3 x_4 y_1$

Отсюда: $\Gamma_{1,\bar{y}} = \bar{x}_2 x_3 + \bar{x}_2 \bar{x}_3 x_4 \qquad \Gamma_{0,\bar{y}} = \bar{x}_2 \bar{x}_3 \bar{x}_4$

Следующей основной операцией в рассматриваемом алгоритме является операция ввода новых переменных.

Как указывалось, вопрос о реализации заданной функции решается в описываемом алгоритме на основе рассмотрения переходной или приведенной таблицы. При этом значение какой-либо переменной в соответствующем столбце переходной таблицы заменяются значениями соответсвующей ей доопределяющей функции. Заданная функция будет реализована полностью и непротиворечиво, если каждому состоянию из $M_1(fi)$ таблицы состояний комбинационного автомата в приведенной таблице будет соответствовать одно из состояний $H_1(\mathcal{P}_j)$ подавтомата, а каждому из состояний из $M_0(fi)$ - будет соответствовать одно из состояний из $H_0(\mathcal{P}_j)$.

Если обозначить через $\varphi_1(x,f)$ и $\varphi_0(x,f)$ функции, реализуемые на выходе выходного подавтомата при подаче на его входы выбранных переменных или заменяющих их доопределяющих функций, то условия полной и непротиворечивой реализации заданной функции выразятся в виде:

$$T_1 * \varphi_1(x,f) = 0 \quad ; \quad T_0 * \varphi_0(x,f) = 0$$

или

$$T_1 \cdot \overline{\varphi_1(x,f)} = 0 \quad ; \quad T_0 \cdot \overline{\varphi_0(x,f)} = 0$$

Замена какой либо переменной соответствующей ей доопределяющей функцией осуществляется с помощью введения дополнительной переменной $x_{n+i} = f(x_1, x_2, \ldots, x_n)$ подаваемой на соответствующий вход выходного подавтомата. Для определения факта реализации функции $T_1(fi)$ и $T_0(fi)$ эту переменную нужно ввести в последние. Воспользуемая для этого тем фактом, что x_{n+i} является функцией от входных переменных, поэтому выражение:

$$\Delta = \overline{x}_{n+i} f(x_1, x_2 \ldots x_n) + x_{n+i} \overline{f(x_1, x_2 \ldots x_n)} = 0$$

В соответствие с этим состояния которым соответствует Δ , будут неиспользуемыми. Удаление этих состояний из $T_1(fi)$ и $T_0(fi)$ обеспечивает введение в эти функции переменной x_{n+i} .

Проиллюстрируем это на примере.

Пусть при реализации функции прерыдущего примера на том-же подавтомате выбрано следующее заполнение входов последнего:

$$\varphi = (x_3^1, x_2^2, x_2^3, \overline{x}_1^4)$$

Проверка показывает, что при таком заполнении заданная функция реализуется противоречиво.

Определено, что в первую очередь должен быть доопределен вход y_3 , для которого получена доопределяющая функция в виде:

$$T_{1,y3} = x_4 + x_2 \overline{x}_4 \qquad T_{0,y3} = \overline{x}_2 \overline{x}_4$$

Обозначим переменную. соответствующую этой функции, через x_5 и введем ее в $T_1(fi)$ и $T_0(fi)$. Определим Δ :

$$\Delta = \overline{x}_5 (x_4 + x_2 \overline{x}_4) + x_5 \overline{x}_2 \overline{x}_4$$

Введем Δ в $T_1(fi)$ и $T_0(fi)$ с помощью Z -операции.

$$T_{1,x5} = T_1 * \Delta = (x_1 x_2 + \overline{x}_1 x_2 x_3 + \overline{x}_2 \overline{x}_3 \overline{x}_4) * \overline{x}_2 \overline{x}_4 x_5 * x_4 \overline{x}_5 * x_2 \overline{x}_4 \overline{x}_5 = x_1 x_2 x_4 x_5 + \overline{x}_1 x_2 x_3 x_4 x_5 + x_1 x_2 \overline{x}_4 x_5 +$$
$$+ \overline{x}_2 \overline{x}_3 \overline{x}_4 \overline{x}_5 + \overline{x}_1 x_2 x_3 \overline{x}_4 x_5$$

$$T_{0,x5} = T_0 * \Delta = (\overline{x}_1 x_2 \overline{x}_3 + \overline{x}_2 x_3 + \overline{x}_2 \overline{x}_3 x_4) * \overline{x}_2 \overline{x}_4 x_5 * x_4 \overline{x}_5 * x_2 \overline{x}_4 \overline{x}_5 = \overline{x}_1 x_2 \overline{x}_3 x_4 x_5 + \overline{x}_1 x_2 \overline{x}_3 \overline{x}_4 x_5 +$$
$$+ \overline{x}_2 x_3 \overline{x}_4 \overline{x}_5 + \overline{x}_2 x_3 \overline{x}_4 \overline{x}_5 + \overline{x}_2 \overline{x}_3 x_4 x_5$$

Заменим переменную x_2 на 3-ем входе выходного подавтомата на x_5 .

$$\varphi_1 = x_3x_2 + \bar{x}_3\bar{x}_2\bar{x}_5 + x_2x_5x_1 \qquad \varphi_0 = \bar{x}_3x_2\bar{x}_5 + \bar{x}_3x_2x_5\bar{x}_1 + \bar{x}_2x_5 + x_3\bar{x}_2\bar{x}_5$$

Проверим реализацию функций F_{1,x_5} и F_{0,x_5} .

$$F_{1x_5}{}^{*}\varphi_1 = (x_1x_2x_4x_5 + \bar{x}_1x_2x_3x_4x_5 + x_1x_2x_4x_5 + x_2x_3x_4x_5 + x_1x_2x_3x_4x_5)^{*}x_3x_2 {}^{*} x_3x_2x_5 {}^{*} x_2x_5x_1$$

$$F_{0,x_5}{}^{*}\varphi_0 = (\bar{x}_1x_2\bar{x}_3x_4x_5 + \bar{x}_1x_2\bar{x}_3\bar{x}_4x_5 + \bar{x}_2x_3x_4x_5 + \bar{x}_2x_3\bar{x}_4\bar{x}_5 + \bar{x}_2\bar{x}_3x_4x_5)^{*}\bar{x}_3x_2\bar{x}_5 + \bar{x}_3x_2x_5\bar{x}_1 + \bar{x}_2x_5 + x_3\bar{x}_2\bar{x}_5$$

Легко заметить, что в этих выражениях для каждого из членов в F_{1,x_5} и F_{0,x_5} можно найти соответственно в выражениях для φ_1 и φ_0 член за знаком z-операции, содержаний совпадающие буквы, но в меньшем количестве. В соответствии с (8.9) это обращает этот член при осуществлении Z-операции в 0. Поскольку такие условия имеют место для всех исключения членов F_{1,x_5} и F_{0,x_5} то получим:

$$F_{1x_5}{}^{*}\varphi_1 = 0 \qquad F_{0,x_5}{}^{*}\varphi_0 = 0$$

т.е. заданная функция будет полностью реализована.

Для функций, характеризующих недоопределенные таблицы состояний, существенное сокращение вычислений дает исключение несущественных переменных Несущественной называют переменную, в столбце которой в таблице состояний не содержится обязательных букв, т.е. такую переменную для которой в рабочих и нерабочих состояниях нельзя найти пару таких состояний, (одно в рабочей части и другой в нерабочей), которые отличались друг от друга только значением данной переменной. Очевидно такую переменную можно исключить из F_1 и F_0 без нарушения непротиворечивости этих функций.

Признаком несущественной переменной является навенство нулю функции:

$$\Delta F_{x_к} = F_1(x_к=1) \cdot F_0(x_к=0) + F_1(x_к=0) \cdot F_0(x_к=1)$$

Исключение несущественных переменных следует производить в порядке возрастания критерия:

$$R = n_1^1 n_0^0 + n_1^0 n_0^1$$

где: n_1^1 и n_0^1 имеют то же значение, что и в (3), а n_1^0 и n_0^0 есть число рабочих и нерабочих состояний реализуемой функции, в которых соответствующая переменная $x_к$ имеет значение нуля.

После исключения каждой очередной несущественной переменной критерии подсчитываются заново и для всех оставшихся переменных снова проверяется их несущественность.

Как уже указывалось исключение несущественных переменных в ряде случаев является весьма эффективным. Так например для функций (6.I) и (7.I) несущественными являются переменные: x_3, x_4, x_8 и x_9. . Исключая их получим:

$$F_1' = x_1x_2 + (x_1\bar{x}_2 + \bar{x}_1)(x_5 + \bar{x}_5x_6) \qquad F_0' = (x_1\bar{x}_2 + \bar{x}_1)\bar{x}_5\bar{x}_6$$

Эти функции непротиворечиво реализуются всего на одном элементе типа:

$$\varphi = \bar{y}_1 + \bar{y}_2 + \bar{y}_3 + y_k y_5 y_6$$

со следующим заполнением входов:

$$\varphi = (\bar{x}_5^1, \bar{x}_6^2, 1^3, x^4, x_2^5, 1^6)$$

Описанные операции преобразования нормальных и скобочных форм булевых функций дают представление лишь об основных идеях, на которых базируется метод декомпозиции комбинационных автоматов при формулировке их условий работы и структурных свойств подавтоматов, в нормальной или скобочной форме. Изложенный метод является достаточно эффективным, что в частности показывает рассмотренный в начале статьи пример. Следует, однако, отметить, что операции его для ручной работы достаточно громоздки и требуют применения УВМ. Тем не менее следует, вероятно, признать, что переход при декомпозиции комбинационных автоматов к алгебраическим методам является единственным, который позволит преодолеть трудности, возникающие при реализации комбинационных автоматов большой размерности.

Цитированная литература

I В.А. Горбатов: Схемы управления ЦВМ и графы. Изд. "Энергия", 1971.

2 Р. Миллер: Теория переключательных схем. т.I, Изд. "Наука", 1970.

3 W. Buckhart: Theorem minimisation. An.Comp.Machinery, May, 1955.

4 М.А. Гаврилов: Минимизация булевых функций, характеризующих релейные цепи . "Автоматика и телемеханика", т.2. №9. 1959. .

5 П.П. Пархоменко: Синтез структур релейных устройств методом замены входных переменных. "Автоматика и телемеханика", №I. 1967.

6. В.Р. Горовой: Синтез релейных структур методом замены выходных функций. "Автоматика и телемеханика", №I. 1967.

7. М.А. Гаврилов, В.М. Копыленко: Метод переходных таблиц синтеза многовыходных комбинационных структур на произвольных элементах. Изд. Института проблем управления /Автоматика и телемеханика/, 1970 г.

8. М.А. Гаврилов: Теоретические проблемы практического приложения теории конечных автоматов. Труды международного семинара по прикладным аспектам автоматов. Изд. Болгарской Академии Наук. Институт технической кибернетики, 1971 г.

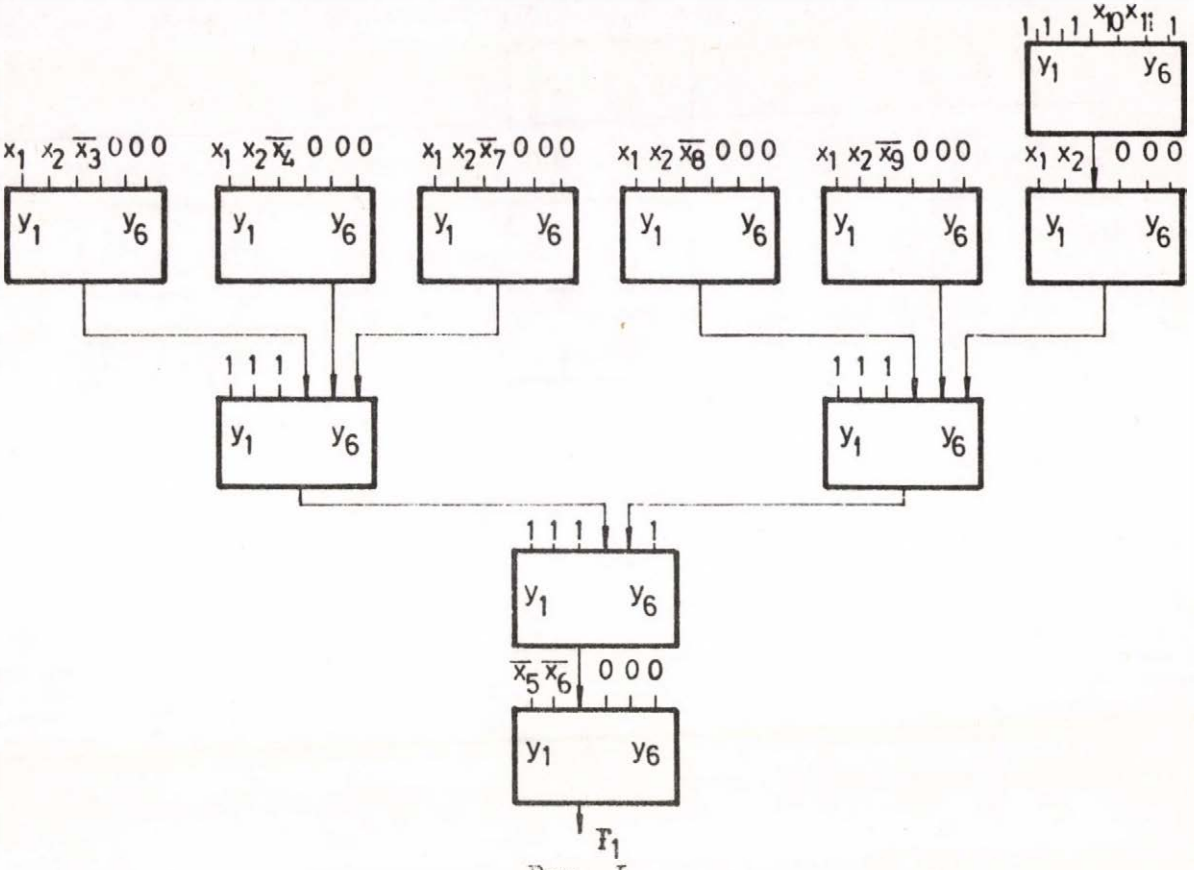


Рис. I.

Реализация функции F_1 (вариант а) методом подбора

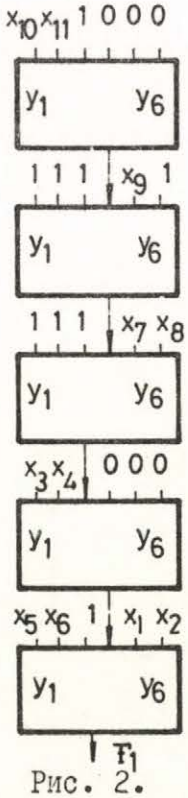


Рис. 2.

Реализация функции F_1 (вариант б) методом подбора

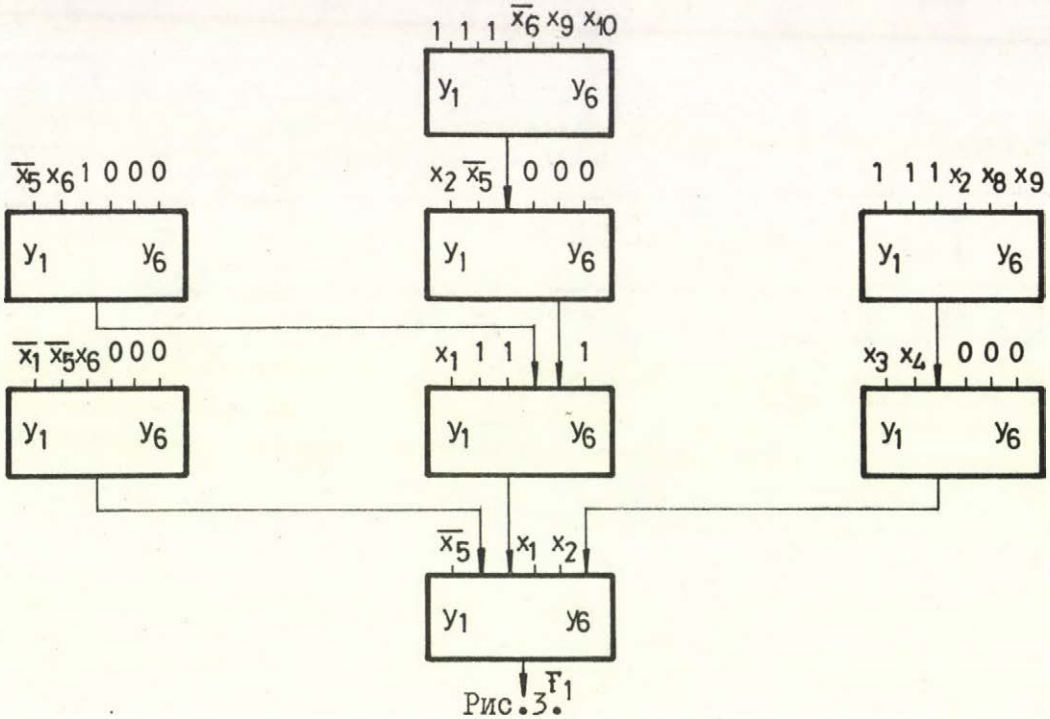


Рис.3.

Реализация функции F_1 (вариант в) методом подбора

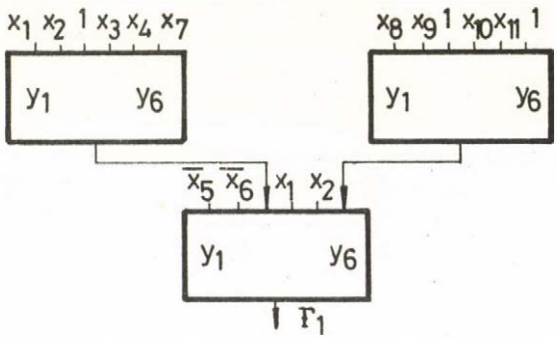


Рис. 4.

Реализация функции F_1 (вариант а, б и в) методом переходных таблиц

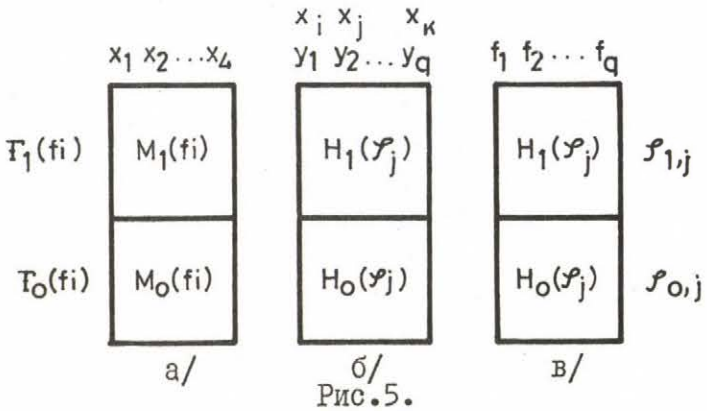


Рис.5.

а–таблица состояний, б–переходная таблица, в–приведенная таблица

Принципы организации автоматизированной системы проектирования логики дискретных управляющих устройств

В. В. Девятков [x]

Настоящий доклад посвящён изложению основных принципов, положенных в основу создания диалогово-дисплейной автоматизированной системы проектирования логики дискретных управляющих устройств, называемой ДАСП и разрабатываемой в Институте проблем управления. Общие требования на разработку автоматизированных систем синтеза логики дискретных управляющих устройств /ДУУ/ изложены в [I] . Основные принципы, положенные в основу ДАСП, учитывают эти требования.

Языки описания условий работы ДУУ в ДАСП могут быть разделены на три группы: языки высшего уровня, базовые языки и стандартные языки. Языки высшего уровня - это проблемно-ориентированные языки описания условий работы ДУУ /алгоритма функционирования ДУУ/, обладающие широким набором изобразительных средств, удобных для описания условий работы ДУУ определённого класса, зависящего от области применения этого ДУУ. Словарный состав и синтаксис этих языков близок к словарному составу и синтаксису тех профессиональных языков, которые используются в организациях, проектирующих ДУУ этого класса. Языками высшего уровня в ДАСП в настоящее время являются языки УСЛОВИЕ [I] и ФОРУМ [2] .

Язык условие предназначен для описания условий работы ДУУ широкого класса. Описание ДУУ в языке УСЛОВИЕ состоит из описания условий работы его блоков. Разбивка на блоки определяется самим проектировщиком, каждый блок проектируется в ДАСП независимо от других, если проектировщик не потребует в дальнейшем его композиции с другими блоками.

Описание условий работы блока в языке УСЛОВИЕ состоит из описания в терминах управляемого объекта входных и выходных переменных блока /в том числе векторных/, возможных состояний этих переменных в терминах действий

управляемого объекта, соответствующих этим состояниям /в этих состояниях не обязательно значения всех переменных должны быть определены/ и алгоритма функционирования блока, состоящего из различных типов предложений: безусловных, простых условных, макропредложений /последовательности из простых условных предложений/, взаимосвязанных друг с другом.

При описании алгоритма функционирования блоков в языке УСЛОВИЕ возможно задание параллелизма в работе отдельных частей блока, называемых процессами /внутри процессов параллелизм не допускается/, в свою очередь состоящих предложений различного типа. Между процессами возможно компактное задание задание приоритетов, синхронизаций и прерываний. Процессы из-за описания входных и выходных переменных и их состояний в терминах управляемого объекта могут рассматриваться как описание параллельной работы отдельных частей управляемого объекта. В языке УСЛОВИЕ предусмотрено также задание различных стандартных блоков /функций/, явное задание времени, входо-выходных ограничений и т.п.

Язык ФОРУМ предназначен для использования его при проектировании ДУУ различными управляемыми механизмами /УМ/, в которых можно выделить отдельные компоненты - агрегаты, каждый из которых имеет возможность с помощью исполнительных устройств по сигналам от ДУУ перемещаться в пространстве. Траектория этого перемещения фиксируется конечным числом координат. Язык ФОРУМ предназначен не для описания ДУУ (условий его работы), а для описания условий работы УМ, из которого извлекается задание на проектирование блоков ДУУ, поэтому изобразительные средства языка ФОРУМ таковы, чтобы в нём было как можно удобней описывать условия работы /функционирование/ УМ, заключающееся в перемещении агрегатов в пространстве по координатам, где в общем случае каждая последущая координата является функцией предыдущей и сигнала от ДУУ. В языке ФОРУМ предусмотрены средства для описания параллельной работы агрегатов, их вхаимодействия, блокировок и т.п.

Математической моделью ДУУ в ДАСП является конечный автомат или их совокупность. Основным языком представления конечного автомата в ДАСП является система его функций возбуждения и выходов /ФВВ/, заданных в нормальной или скобочной форме. Система ФВВ называется стандартным языком, являющимся наиболее компактной формой задания конечного автомата в ДАСП. Стандартный язык удобен для дальнейших преобразований ДУУ и его блоков вплоть до реализации ДУУ, но использование его непосредственно для семантического истолкования языков высшего уровня вызывает больше трудности и, как следствие этого, непосредственная трансляция с языков высшего уровня в стандартный

язык /особенно при построении транслятора синтаксического типа/ также весьма неудобна и затруднена.

Для устранения этих трудностей в ДАСП имеется язык промежуточный между языками высшего уровня и стандартным языком, называемый базовым. Этот язык близок к изложенному в 3.

Для взаимодействия ДАСП с пользователем предусмотрено три режима: режим обычной пакетной обработки, режим разделения времени и комбинированный режим. В первом случае ввод описания условий работы производится с перфокарт, а результаты работы выдаются на широкую печать, во втором случае ввод и вывод происходит через дисплей, а в третьем случае ввод происходит с перфокарт по команде, отдаваемой с дисплея, общение пользователя с машиной в процессе проектирования происходит через дисплей, вывод может быть осуществлён как на широкую печать, так и на дисплей.

При работе пользователя за дисплеем ДАСП направляет его работу с помощью запросов, требуя от него ответа, формат которого указывается пользователю в случае необходимости. Пользователь также может воздействовать на ход проектирования с помощью приказов, которые он может отдавать ДАСП в ответ на любой запрос с её стороны или сообщение о том, что ДАСП в настоящее время делает. Приказы подразделены на общесистемные: объявления начала и конца работы с ДАСП, указаний режима работы, вызова отдельных подсистем системы /см. ниже/, обеспечивающих проектирование ДУУ на различных этапах и т.п.; подсистемные: вызова информации о результатах проектирования данной подсистемой, редактирования, принятия к исполнению решения о применении той или иной программы данной подсистемой и т.п.

Функциональный состав ДАСП с точностью до подсистем показан на рис.I.

Главный монитор управления подсистемами управляет потоками информации между внешними устройствами и подсистемами, осуществляет вызов мониторов подсистем и передачу им управления.

Подсистема общеязыковой логической обработки /ПОЛО/ воспринимает описание условий работы ДУУ на языках высшего уровня, выполняет поиск ошибок в описаниях, трансляцию в базовых язык, выяснение полноты описания, установление непротиворечивости описания, выдачу и получение запросов и ответов относительно выполняемых подсистемой функций, переход к поблочному описанию ДУУ в виде систем ФВВ.

Подсистема внутриблочных и межблочных преобразований /ПВМП/ воспринимает поблочное описание условий работы ДУУ в виде системы ФВВ, осу-

ществляет внутриблочные преобразования с целью оптимизации памяти блока /минимизации числа внутренних состояний/ и межблочные преобразования с целью композиции и декомпозиции блоков, которые в обоих случаях заданы системой ФВВ в нормальной форме, выдачу и получение запросов и ответов относительно выполняемых подсистемой функций и переход к новому поблочному описанию в виде системы ФВВ или графа переходов.

Подсистема реализации памяти /ПРП/ воспринимает поблочное описание ДУУ в виде системы ФВВ, осуществляет выбор программы кодирования внутренних состояний блока для реализации его с учётом тех или иных требований к ДУУ /синхронное с учётом простоты реализации в последующем комбинационной части ДУУ, асинхронное с устранением состояний, для типовой реализации памяти, например, на двоичных сдвиговых регистрах, с учётом заданной надёжности и т.п./, выдачу и получение запросов и ответов относительно выполняемых подсистемой функций и переход к новому поблочному описанию ДУУ в виде системы ФВВ.

Подсистема реализации комбинационной части /ПРКЧ/ воспринимает поблочное описание ДУУ в виде системы ФВВ, осуществляет выбор алгоритма и реализацию на конкретных наборах функциональных элементов комбинационной части блоков ДУУ, выдачу и получение запросов и ответов относительно выполняемых подсистемой функций и переход к описанию блоков ДУУ в виде функциональных схем.

Подсистема моделлирования может воспринимать описание ДУУ в виде системы ФВВ или в виде функциональной схемы /т.е. от любой из подсистем/ и осуществляет проверку правильности функционирования отдельных блоков ДУУ и ДУУ в целом, получая входо-выходные последовательности от пользователя или генерируя их автоматически и осуществляя слежение за запрещёнными выходными состояниями, выдачу временных диаграм, выдачу и получение запросов и ответов относительно выполняемых подсистемой функций.

Алгоритмы, положенные в основу ДАСП, созданы по методам, многие из которых разработаны специально для ДАСП и ориентированы на работу с системами булевых функций в интервальной и скобочной форме. Использование систем булевых функций в качестве основного языка /стандартного/ для представления и преобразования блоков ДУУ на всех этапах его проектирования является характерной чертой ДАСП, выделяющей её из известных и разрабатываемых систем автоматизированного проектирования ДУУ и позволяющей обойти многие трудности связанные с оптимизацией ДУУ большой размерности.

Так в основу подсистемы ПВМП положены алгоритмы, представление о которых могут дать работы [I] , [4] . Эти алгоритмы позволяют производить оптимизацию блоков, их композицию и параллельную декомпозицию в стандартном языке - по системам ФВВ в интервальной форме без построения таблиц переходов. В основу подсистемы ПРП положены алгоритмы, основанные на методах, часть из которых изложена в работах [5] - [I0]. Большинство из этих алгоритмов предназначено для кодирования внутренних состояний блоков с учётом различных требований /отсутствия состояний, простоты структуры, типовой реализации памяти и т.п./, причем решение о выборе того или иного метода /алгоритма/, применяемого для кодирования внутренних состояний блока, осуществляется с помощью оценки параметров этого блока, влияющих на эффективность применения того или иного алгоритма. Применение алгоритма, разработанного по методу, изложенному в работе [9] , позволяет применять для кодирования состояний блоков, в которых недопустемы состояния элементов памяти, не только алгоритмы кодирования с устранением состояний, но и алгоритмы кодирования не устраняющие состязания /но с другими достоинствами/, с последующими анализом на наличие состязаний и коррекцию для их устранения.

В основу подсистемы ПРКЧ положены алгоритмы, основанные на методах, представление о которых дают работы [II] , [I2]. Эти алгоритмы в общем случае относятся к приближённым и не дают самого лучшего решения, но работают с системами булевых функций большой размерности и позволяют получать безизбыточные структуры из функциональных элементов близкие к минимальным.

Машинная реализация ДАСП осуществляется на системе ICL 4-70 с использованием языка ФОРТРАН в качестве основного языка программирования. На настоящем этапа развития ДАСП язык ФОРТРАН выбран исходя из того, что он прост и широко распространён как в СССР, так и за его границами, трансляторы с него имеются на машинах второго поколения и отдельные программы ДАСП могут использоваться на них, замедление в скорости работы программ и перерасход по памяти, которые являются следствием применения ФОРТРАНА для программирования логических задач, окупаются скоростью отладки программ на нём и интервальным или скобочным представлением булевых функций в ДАСП. Применение ФОРТРАНА не исключает использования для отдельных программ других языков программирования, в частности, автокода. Некоторые сведения о составе программного обеспечения подсистемы ПОЛО содержаться в [I3] и касаются синтаксического транслятора с языка УСЛОВИЕ, состава языка приказов, реализации режима взаимодействия пользователя с машиной через дисплей в процессе проектирования. В [I4] приведено описание программы минимизации полностью определённого автомата, относящейся к подсистеме ПВМП, исходным заданием

для которой является система ФВВ блока ДУУ, представленная в нормальной форме. Здесь же содержится описание комплекса подпрограмм работы с булевыми функциями таких как склеивание, инверсия, дизъюнкция, кнъюнкция и т.д., носящих универсальный характер, а также описание применяемого в рамках ФОРТРАНА представления в машине булевых функций в нормальной форме.

Класс задач, на решение которых рассчитана ДАСП, в общем случае не фиксируется, однако в настоящем её виде она больше подойдет проектировщику дискретных управляющих устройств для объектов и процессов промышленности, чем проектировщику электронных вычислительных машин, рассматриваемых как единое целое, состоящее из устройства управления, операционной части и памяти.

Литература

I. Гаврилов М.А., Девятков В.В., Потехин А.И., Чичковский А.Б., Пупырев Е.И. Технические требования на автоматизированную систему логического синтеза дискретных управляющих устройств. Применение процедур диалога с машиной при их синтезе. Информационные материалы, 7 /54/, М., 1971.

2. Амбарцумян А.А., Девятков В.В. ФОРУМ - язык формального описания работы управляемых механизмов. Сб. Вопросы кибернетики, М., 1973.

3. Заславский И.Д. Граф-схемы с памятью. Труды математического института, АН СССР, 72, 1964.

4. Девятков В.В., Потехин А.И. Интервальный метод анализа поведения автомата, заданного системой функций возбуждения и выходов. Автоматика и телемеханика, 5, 1972.

5. Лебедев А.Г., Потехин А.И. Алгоритм кодирования внутренних состояний релейного устройства. Сб. Абстрактная и структурная теория релейных устройств и конечных автоматов, вып. 2, 1972.

6. Девятков В.В. Алгоритм реализации конечного автомата на сдвиговых регистрах. Там же.

7. Девятков В.В. Реализация конечных автоматов на сдвиговых регистрах. Автоматика и телемеханика, 11, 1968.

8. Девятков В.В. Алгоритмы синтеза устройств на сдвиговых регистрах. М., ИАТ, 1969.

9. Потехин А.И. Метод анализа устойчивости асинхронного автомата, заданного функциями возбуждения и выходов. Автоматика и вычислительная техника, 5, 1970.

10. Булат М.С. Кодирование внутренних состояний автомата с учётом связности многовыходных структур. Сб. трудов Кишинёвского политехнического института, Кишенёв, 1969.

11. Гаврилов М.А., Копыленко В.М. Алгоритм синтеза многовыходных комбинационных структур большой размерности на произвольных элементах. Сб. Абстрактная и структурная теория релейных устройств и конечных автоматов, вып. 2, 1972.

12. Пупырев Е.И. Минимизация булевых функций в базисе ИЛИ-НЕ, И-НЕ. Автоматика и телемеханика, 3, 1972.

13. Чичковский А.Б., Девятков В.В. Состав программного обеспечения для начального этапа логического проектирования дискретного управляющего устройства с использованием дисплея. Сб. Вопросы кибернетики, М., 1972.

13. Девятков В.В., Малевич А.Н., Пупырев А.И. Программная реализация метода минимизации памяти дискретного управляющего устройства, заданного системой функций возбуждения и выходов. Сб. Вопросы кибернетики, М., 1973.

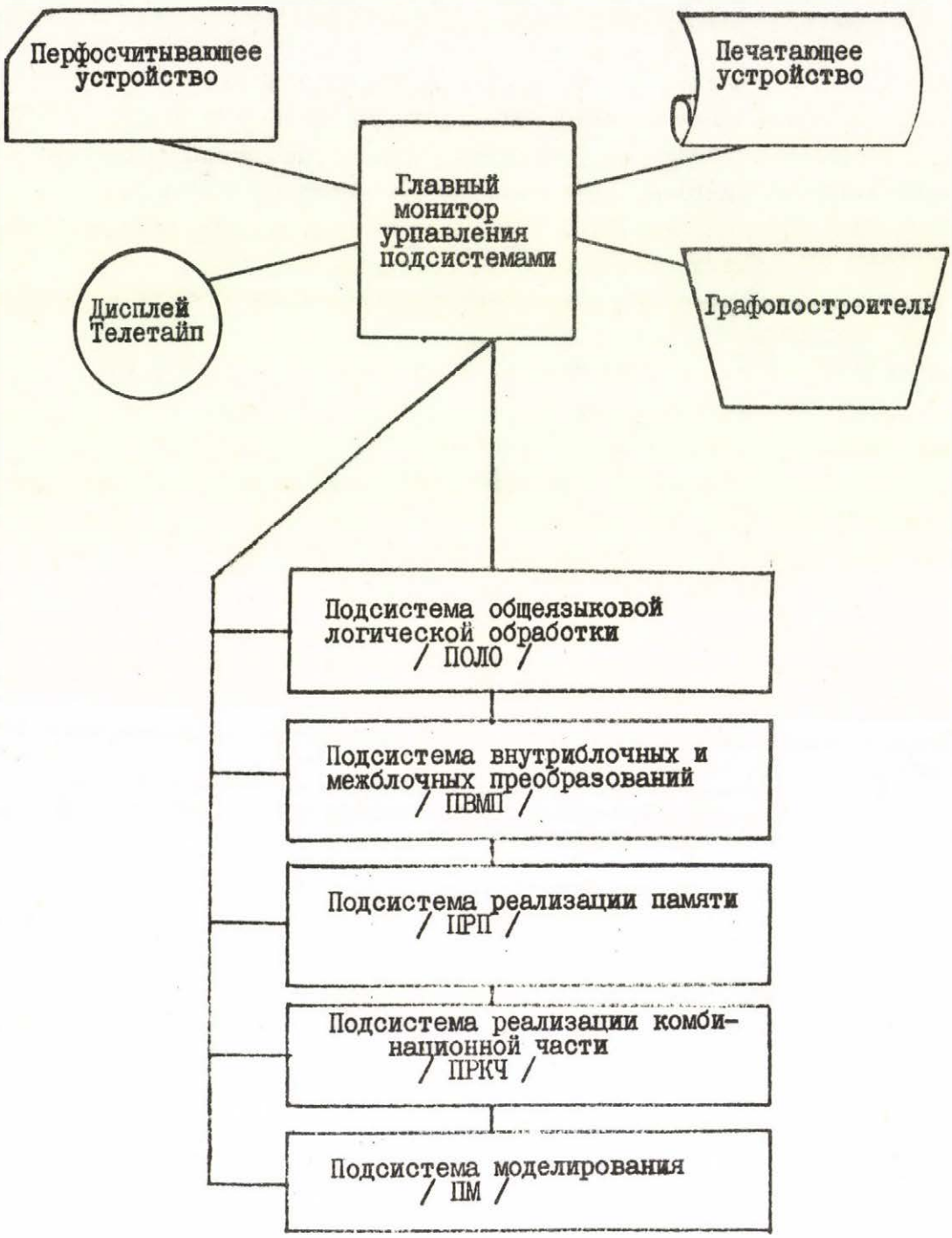

Перфосчитывающее устройство
Печатающее устройство
Главный монитор урпавления подсистемами
Дисплей Телетайп
Графопостроитель
Подсистема общеязыковой логической обработки / ПОЛО /
Подсистема внутриблочных и межблочных преобразований / ПВМП /
Подсистема реализации памяти / ПРП /
Подсистема реализации комбинационной части / ПРКЧ /
Подсистема моделирования / ПМ /

Рис. I

ОБ ОДНОМ ПОДХОДЕ К АВТОМАТИЗАЦИИ СИНТЕЗА АСИНХРОННЫХ КОНЕЧНЫХ АВТОМАТОВ

Фрицнович Г.Ф.

(СССР)

I.Введение

Переход к интегральной технологии и стремление максимально использовать быстродействие базисных элементов приводят к широкому использованию на практике устройств логической переработки дискретной информации,работающих в асинхронном режиме.Поэтому значительный практический интерес приобретает разработка формальных методов построения логической структуры проектируемых устройств,где в качестве математической модели используется асинхронный конечный автомат и которые позволяют полностью автоматизировать этап логического проектирования этих устройств.Решение этой задачи заключается в разработке взаимосвязанных алгоритмов решения всех задач,возникающих при синтезе асинхронных конечных автоматов,и в создании системы соответствующих машинных программ,позволяющей осуществить синтез при помощи ЭЦВМ.

Ниже рассмотрим единый подход к решению основных задач синтеза,на базе которого может быть создана компактная и гибкая система автоматизации сквозного синтеза асинхронных конечных автоматов и который основан на сведении всех основных задач синтеза к раскраске вершин конечных неориентированных графов [I].

Настоящий доклад в основном является обзором,систематизацией и некоторым обобщением ранее полученных результатов.Принимая во внимание это обстоятельство и учитывая ограничения на объём доклада,в третьем разделе при формулировке основных результатов опущены доказательства,а в четвертом разделе приведено лишь описание упрощенной блок-схемы системы алгоритмов синтеза асинхронных конечных автоматов,разработанных на основе предлагаемого подхода.

2.Постановка задачи

Решение целого ряда задач,возникающих при синтезе асинхронных конечных автоматов(минимизация числа внутренних состояний,противогоночное кодирование внутренних состояний при отсутствии ограничений на быстродействие,минимизация булевых функций в классе д.н.ф. и др.) заключается в нахождении не-

которого разбиения конечного множества M с заданным бинарным отношением π, которое обычно обладает свойствами симметричности и рефлексивности, но не является транзитивным на всем множестве M, на минимальное число классов эквивалентности. Оказывается, что решение этой задачи в свою очередь может быть сведено к нахождению минимальной по числу цветов в заданном смысле правильной раскраски вершин конечного неориентированного графа.

Точные решения задач минимизации числа внутренних состояний, кодирования внутренних состояний и минимизации булевых функций может быть сведены к решению сформулированной ниже задачи*).

Пусть дано некоторое конечное множество M и бинарное отношение π на этом множестве. Требуется найти минимальное по числу элементов множество $\mathcal{M}_g = \{M_{g1}, M_{g2}, \ldots, M_{gw_{min}}\}$, которое обладает следующими свойствами:

1) $M_{gi} \subseteq M \quad (i = 1,2,\ldots, w_{min})$,

2) любые два элемента каждого $M_{gi} \in \mathcal{M}_g$ находятся в отношении π,

3) $\bigcup_{i=1}^{i=w_{min}} M_{gi} = M$,

4) разбиение $\mathcal{M}_g$ замкнуто.

При решении задачи минимизации числа внутренних состояний множеством M является множество всех внутренних состояний синтезируемого автомата, отношение π определяется как отношение совместимости любых двух состояний, а наличие свойства замкнутости интерпретируется как возможность объединения всех элементов каждого $M_{gi} \in \mathcal{M}_g$ в одно состояние без нарушения условий детерминированности работы синтезируемого автомата.

При кодировании внутренних состояний асинхронного конечного автомата кодом, обладающим способностью устранения опасных состязаний между промежуточными переменными при отсутствии ограничений на порядок изменения значений промежуточных переменных в переходах между внутренними состояниями(П-код), в качестве множества M рассматривается множество всех строк матрицы пар переходов. Отношение π в этом случае определяется как объединимость любых двух строк этой матрицы(отношение объединимости), а замкнутость - как возможность объединения всех элементов любого $M_{gi} \in \mathcal{M}_g$ в одно объединение (в один разряд кода).

Под матрицей пар переходов при этом понимается частично определенная булева матрица M_{nn} размерностью $N \times s$, где N - число пар переходов, а

---

*) Под точным решением задачи минимизации числа и кодирования внутренних состояний будем понимать нахождение минимального числа внутренних состояний и кода минимальной длины, а при минимизации булевых функций - - нахождение кратчайшей д.н.ф.

s – число внутренних состояний синтезируемого автомата. Строкам этой матрицы взаимнооднозначно сопоставлены пары переходов, а столбцам – внутренние состояния. Значения элементов матрицы M_{ns} определяются следующим образом:

$$m_{ij} = \begin{cases} 1, & \text{если } j\text{-тое внутреннее состояние содержится в первом переходе } i\text{-той пары переходов,} \\ 0, & \text{если } j\text{-тое внутреннее состояние содержится во втором переходе } i\text{-той пары переходов,} \\ -, & \text{если } j\text{-тое внутреннее состояние не содержится в } i\text{-той паре переходов,} \end{cases}$$

$$i = 1,2,\ldots, N \quad , \qquad j = 1,2,\ldots,s \quad *) \; .$$

Любые две строки c_q и c_p матрицы пар переходов называются объединимыми, если и только если c_q совпадает с точностью до неопределенных элементов со строкой c_p или $\bar{c}_p$, где $\bar{c}_p$ – дополнение строки c_p. Дополнением строки c_p называется строка $\bar{c}_p$, которая образуется из c_p путем замены всех определенных значений элементов этой строки их отрицаниями. Объединением любых двух объединимых строк c_q и c_p матрицы M_{ns} называется строка, значения элементов которой определяются значениями элементов строк c_q или $\bar{c}_q$ и c или $\bar{c}_p$.

При минимизации булевых функций предполагается, что булева функция от n переменных задана на n-мерном булевом пространстве $\mathcal{M}_s$ разбиением его на три подмножества $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_\sim$, где $\mathcal{M}_1$ – множество элементов булевого пространства, на которых функция принимает единичное значение (множество рабочих состояний), $\mathcal{M}_0$ – множество элементов булевого пространства, на которых функция принимает нулевое значение (множество нерабочих состояний), $\mathcal{M}_\sim$ – множество элементов булевого пространства, на которых значение функции неопределено (множество безразличных состояний). Множество M при этом отождествляется с множеством $\mathcal{M}_1$ рабочих состояний минимизируемой функции, отношение π определяется как наличие интервала, определенного на множестве $\mathcal{M}_1 \cup \mathcal{M}_\sim$ и покрывающего любые два элемента множества M, а замкнутость интерпретируется как наличие для каждого $M_{gi} \in \mathcal{M}_g$ интервала, определенного на множестве $\mathcal{M}_1 \cup \mathcal{M}_\sim$ и покрывающего все элементы рассматриваемого $\overset{**}{M_{gi}}$.

---

*) Нумерация переходов в паре переходов произвольная.

**) Здесь и ниже при рассмотрении задачи минимизации булевых функций, в основном, используется терминология, введенная в работе [2].

Как следует из работ [3,4,5,6], решение сформулированной задачи сводится к решению следующих двух частных задач:

I) образование некоторого множества $\mathcal{M}$ подмножеств элементов множества M, все элементы которых попарно находятся в отношении π,

2) нахождение такого минимального по числу элементов покрытия множества M множеством $\mathcal{M}$, которое обладает свойством замкнутости.

При минимизации числа внутренних состояний в качестве множества $\mathcal{M}$ рассматривается множество всех (всех простых) множеств совместимости [3, 7]. При решении задачи кодирования множеством $\mathcal{M}$ является множество всех максимальных объединений строк матрицы пар переходов [4], а при минимизации булевых функций - множество всех максимальных интервалов (простых импликант) [2,5,6].

Как показывает опыт использования этой схемы решения, основные трудности её практического применения заключаются в том, что мощность множества $\mathcal{M}$ при решении задач реальной сложности является очень большой. Некоторое представление о возможном числе элементов множества $\mathcal{M}$ в зависимости от числа элементов множества M дает известные в теории графов оценки числа клик конечного неориентированного графа [8,9].

Пусть множество M является множеством вершин графа $\bar{G}$ и пусть m_M - мощность этого множества. Любые две вершины графа $\bar{G}$ соединяются ребром тогда и только тогда, когда соответствующие им элементы множества M находятся в отношении π. Нетрудно убедиться, что при таком построении графа $\bar{G}$ мощность D множества $\mathcal{M}$ удовлетворяет неравенству $D \geq Q$, где Q - число клик графа $\bar{G}$. Максимальное возможное число Q_{max} клик этого графа определяется следующим образом:

$$Q_{max} = \begin{cases} 3^r, & \text{если } m_M = 3r, \\ 2.3^r, & \text{если } m_M = 3r - 1, \\ 4.3^r, & \text{если } m_M = 3r + 1, \end{cases}$$

где $r = 1,2,3,\ldots$.

Несмотря на то, что в практических задачах мощность множества $\mathcal{M}$ является, как правило, значительно меньше максимального возможного числа клик графа $\bar{G}$, часто встречаются задачи, где D достигает $10^3 - 10^4$. Это приводит к необходимости образования, запоминания и обработки больших массивов информации и решения задач поиска кратчайшего покрытия настолько большой размерности, что даже представление этих задач в ЭЦВМ является серьёзной проблемой. Следует также отметить, что существующие методы нахождения множества $\mathcal{M}$, как правило, мало отличаются от полного перебора, а область практического применения существующих методов поиска кратчайшего покрытия ограничивается решением этой задачи для множеств, мощность которых не превышает 10^2.

Ниже предлагается другой подход к решению задач минимизации числа и противогоночного кодирования внутренних состояний асинхронных конечных автоматов и минимизации булевых функций в классе д.н.ф., который в значительной

ст ени позволяет преодолеть указанные трудности.

Пусть множество M является множеством вершин графа G, любые две вершины $m_i, m_j \in M$ которого соединены ребром тогда и только тогда, когда $\overline{m_i \pi m_j}$ *). Множество $\mathcal{M}$ при этом интерпретируется как некоторое множество внутренне устойчивых множеств вершин графа G. Решение рассматриваемой задачи при этом может быть сведено к поиску в заданном смысле минимального по мощности покрытия множества вершин графа G некоторым множеством внутренне устойчивых множеств вершин. Эта задача, в свою очередь, сводится к нахождению минимальной по числу цветов в заданном смысле правильной раскраски вершин графа G.

Наличие в теории графов таких методов, которые дают возможность найти минимальную по числу цветов правильную раскраску вершин графа без предварительного нахождения множества всех внутренне устойчивых множеств вершин даёт основание ставить и успешно решить задачу разработки на базе предлагаемого подхода эффективных методов минимизации и кодирования внутренних состояний асинхронных конечных автоматов и минимизации булевых функций, которые не требуют предварительного нахождения множества $\mathcal{M}$ и последующего решения задачи поиска кратчайшего покрытия.

3. Основные результаты

В настоящем разделе исследованы возможности сведения трех задач к раскраске вершин конечного неориентированного графа, приведены основные результаты этих исследований и на базе этих результатов сформулированы общие схемы решения указанных задач.

При этом синтезируемый автомат рассматривается как совокупность пяти объектов $\mathcal{A}(R, K, L, \varphi, \psi)$, где $R = \{\rho_1, \rho_2, \ldots, \rho_{\ell_R}\}$ – множество состояний входа ($\ell_R \leq 2^n$, где n – число входных переменных), $K = \{x_1, x_2, \ldots, x_{\ell_K}\}$ – множество внутренних состояний ($\ell_K \leq 2^k$, где k – число промежуточных переменных), $L = \{\lambda_1, \lambda_2, \ldots, \lambda_{\ell_L}\}$ – множество состояний выхода ($\ell_L \leq 2^m$, где m – число выходных переменных), φ – функция переходов, ψ – функция выходов. Все переменные двоичные. Предполагается, что условия работы синтезируемого автомата заданы в виде таблицы переходов, свойства которой в ряде случаев специально оговорены. При минимизации булевых функции предполагается, что минимизируемая функция представлена в виде таблицы состояний, где перечислены все рабочие и нерабочие состояния рассматриваемой функции [10].

3.1. Минимизация числа внутренних состояний.

О п р е д е л е н и е I. Два внутренних состояния $x_i, x_j \in K$ конечного автомата $\mathcal{A}$ назовем несовместимыми ($x_i \pi' x_j$) тогда и только тогда, когда множество допустимых последовательностей состояний входа автомата $\mathcal{A}$

---

*) Граф G является дополнительным графом графа $\bar{G}$.

содержит такую последовательность длины $d \geq 1$, которая переводит состояния x_i и x_j в состояния x_q и x_p, соответственно, и имеет последнюю булеву $\rho_v \in R$ такую, что

$$\psi(\rho_v, x_q) = \lambda_s,$$
$$\psi(\rho_v, x_p) = \lambda_t,$$

где $\lambda_s, \lambda_t \in L$; $x_q, x_p \in K$ и $\lambda_s \neq \lambda_t$.

В противоположном случае состояния x_i и x_j назовем совместимыми.

О п р е д е л е н и е 2. Графом несовместимости $G_{нс}$ автомата $\mathcal{A}$ назовем конечный неориентированный граф, вершинам которого взаимнооднозначно сопоставлены внутренние состояния автомата $\mathcal{A}$ и любые две вершины i и j которого соединяются ребром тогда и только тогда, когда сопоставленные им внутренние состояния x_i и x_j такие, что $x_i \pi' x_j$.

Пусть s_{min} - минимальное число внутренних состояний, которое может быть получено путем объединения совместимых внутренних состояний автомата $\mathcal{A}$, и $\gamma(G_{нс})$ - хроматическое число графа $G_{нс}$.

Справедливы следующие две леммы.

Л е м м а 1. Для любого полностью определенного конечного автомата, имеющего π - хроматический граф несовместимости $G_{нс}$, $s_{min} = \gamma(G_{нс})$.

Л е м м а 2. Для любого частично определенного конечного автомата, имеющего γ-хроматический граф несовместимости $G_{нс}$,

$$s_{min} \geq \gamma(G_{нс}).$$

Справедливость леммы 1 следует из того, что отношение совместимости в случае полностью определенных автоматов является отношением эквивалентности. В справедливости леммы 2 легко убедиться, предположив, что $\gamma(G_{нс}) > s_{min}$.

В ряде частных случаев могут быть получены более сильные результаты.

В качестве первого такого частного случая рассмотрим класс автоматов, описанный в работе [II]. К этому классу принадлежат такие и только такие автоматы, таблицы переходов которых обладают следующими свойствами:

1) для любого внутреннего состояния $x_i \in K$ существует в точности одно состояние входа $\rho_r \in R$ такое, что $\varphi(\rho_r, x_i) = x_i$ и $\psi(\rho_r, x_i)$ определено;

2) если для $x_i \in K$ существует $\rho_v \in R$ такое, что $\varphi(\rho_v, x_i)$ неопределено, тогда не определены также $\varphi(\rho_v, x_j)$ для всех $x_j \in K$ таких, что $\varphi(\rho_r, x_j) = x_j$ $(r \neq v)$.

Все автоматы, принадлежащие к этому классу, аналогично [II] назовем автоматами типа A.

Т е о р е м а 1. Для любого автомата типа A, имеющего γ-хроматический граф несовместимости $G_{нс}$,

$$\gamma(G_{нс}) = s_{min}$$

и существует правильная раскраска вершин графа $G_{нс}$ в $\gamma(G_{нс})$ цветов, которая определяет полную и замкнутую совокупность множеств совместимости.

Справедливость теоремы I следует из определения I свойств правильной раскраски вершин конечных неориентированных графов и результатов работы [II].

Для нахождения минимальной по числу цветов правильной раскраски вершин графа G_{HC} предполагается использовать алгоритм, предложенный в работе [I2]. В результате применения этого алгоритма каждой вершине заданного конечного неориентированного графа G сопоставляется некоторый цвет из упорядоченного множества цветов $\mathfrak{S} = \{ S_1, S_2, \ldots, S_p \}$, где $p \geq \gamma(G)$. Введем для этого алгоритма понятие первого варианта раскраски.

О п р е д е л е н и е 3. Первым вариантом раскраски вершин конечного неориентированного графа G назовем такую правильную раскраску вершин этого графа p цветами, где каждая вершина окрашена в наименьший по номеру цвет.

Используя определения I и 3, свойства указанного алгоритма раскраски и автоматов типа A, может быть доказано, что при минимизации числа внутренних состояний автоматов типа A в случае использования для раскраски вершин графа G алгоритма, предложенного в работе [I2], первый вариант раскраски всегда определяет полную и замкнутую совокупность множеств совместимости [I3].

На основе этих результатов может быть предложена следующая схема решения задачи минимизации числа внутренних состояний автоматов типа A:

I) определение попарной несовместимости всех внутренних состояний минимизируемого автомата и построение графа несовместимости G_{HC};

2) нахождение первого варианта минимальной по числу цветов правильной раскраски вершин графа G_{HC} при помощи алгоритма, предложенного в работе [12];

3) объединение всех внутренних состояний, соответствующие которым вершины графа G_{HC} окрашены в один и тот же цвет и построение таблицы переходов минимизированного автомата.

В качестве второго частного случая рассмотрим класс конечных автоматов, таблицы переходов которых обладают следующими свойствами:

I) для любого внутреннего состояния $x_i \in K$ существует в точности одно состояние входа $\rho_r \in R$ такое, что $\varphi(\rho_r, x_i) = x_i$ и $\psi(\rho_r, x_i)$ определено;

2) для любого $x_i \in K$ существует не более одного $x_j \in K$ ($i \neq j$; $\varphi(\rho_h, x_j) = x_j$; $\rho_h \in R$; $h \neq r$) такого, что $\varphi(\rho_r, x_j) = x_i$ и не более одного состояния входа $\rho_v \in R$ ($v \neq r$) такого, что $\varphi(\rho_v, x_i)$ определено.

Автоматы, принадлежащие к этому классу назовем автоматами типа B.

Отметим одно важное свойство автоматов типа B.

Л е м м а 3. Для любого автомата типа B существует хотя бы одна полная и замкнутая совокупность $\mathfrak{S}_{min}$ непересекающихся множеств совместимости.

Справедливость леммы 3 может быть доказана, используя по существу то обстоятельство, что в автоматах типа B, кроме перехода "сам на себя" в каждое состояние задан переход только из одного состояния и из каждого состояния

задан переход только в одно состояние.

О п р е д е л е н и е 4 . Раскраску вершин графа G_{HC} автомата $\mathcal{A}$ назовем c -правильной,если эта раскраска удовлетворяет следующим двум условиям:

I)любые две одноцветные вершины графа G_{HC} не являются смежными;

2)сопоставленные любым двум одноцветным вершинам графа G_{HC} внутренние состояния автомата $\mathcal{A}$ при подаче любого $p_r \in R$ переходят в такие внутренние состояния ,соответствующие которым вершины графа G_{HC} также окрашены в один и тот же цвет.

О п р е д е л е н и е 5 . Наименьшее натуральное число p ,для которого существует c -правильная раскраска вершин графа G_{HC} в p цветов,назовем c -хроматическим числом графа G_{HC} и обозначим $\gamma_c(G_{HC})$.

Очевидно,что $\gamma(G_{HC}) \leq \gamma_c(G_{HC})$.

Справедливо следующее утверждение.

Т е о р е м а 2 . Для любого автомата типа B ,имеющего граф несовместимости G_{HC} ,

$$s_{min} = \gamma_c(G_{HC}),$$

и любая c -правильная раскраска вершин графа G_{HC} числом цветов $\gamma_c(G_{HC})$ определяет полную и замкнутую совокупность множеств совместимости.

Справедливость теоремы 2 следует из справедливости леммы 3 и определений 4 и 5.

Общая схема решения задачи минимизации числа внутренних состояний автоматов типа B отличается от ранее приведенной схемы минимизации автоматов типа A лишь тем,что после графа G_{HC} осуществляется поиск минимальной по числу цветов c -правильной раскраски его вершин.Для решения этой задачи может быть использована некоторая модификация алгоритма,предложенного в работе [I2] ,где,кроме попарной несмежности одноцветных вершин,проверяется также выполнение второго условия c -правильности раскраски.

Так как каждому варианту доопределения частично определенного конечного автомата $\mathcal{A}$ соответствует некоторое полное и замкнутое множество непересекающихся множеств совместимости,а каждому такому множеству совместимости, в свою очередь,соответствует некоторая c -правильная раскраска вершин графа G_{HC} автомата $\mathcal{A}$ и наоборот,то справедливо следующее утверждение.

Т е о р е м а 3 . Для любого частично определенного конечного автомата $\mathcal{A}$,имеющего граф несовместимости G_{HC} ,

$$s'_{min} = \gamma_c(G_{HC}),$$

где s'_{min} - минимальное число внутренних состояний,которое может быть получено путем перебора всех вариантов доопределения автомата $\mathcal{A}$ с последующим объединением эквивалентных внутренних состояний.

Следовательно,предложенная схема минимизации числа состояний автоматов типа B может быть также использована для приближенного решения задачи минимизации числа состояний в общем случае,когда минимизируемый автомат не

является автоматом типа A или типа B .

3.2.Кодирование внутренних состояний П-кодом

Пусть синтезируемый асинхронный конечный автомат $\mathcal{A}$ задан при помощи нормальной таблицы переходов и пусть построена матрица пар переходов M_{nn} этого автомата.Предположим,что $\mathcal{M}_{nn}$ - множество строк этой матрицы.Введем понятие отношения необъединимости на множестве $\mathcal{M}_{nn}$.

О п р е д е л е н и е 6 . Две строки $c_i = \{m_{i1}, m_{i2}, \ldots, m_{is}\}$ и $c_j = \{m_{j1}, m_{j2}, \ldots, m_{js}\}$ матрицы M_{nn} автомата $\mathcal{A}$ назовем необъединимыми $(c_i \bar{\pi} c_j)$ тогда и только тогда,когда существуют такие $q, p \in \{1, 2, \ldots, s\}$,что $m_{ip} = I(0)$, $m_{jp} = 0(I)$ и $m_{iq} = I(0)$, $m_{jq} = I(0)$.

О п р е д е л е н и е 7 . Конечный неориентированный граф $G_{нб}$, вершинам которого взаимнооднозначно сопоставлены строки матрицы M_{nn} и любые две вершины которого соединены ребром тогда и только тогда,когда соответствующие им строки являются необъединимыми,назовем графом необъединимости.

Пусть k_{min} - минимальная длина П-кода автомата $\mathcal{A}$,а $\gamma(G_{нб})$ - хроматическое число графа $G_{нб}$.Предположив,что $k_{min} < \gamma(G_{нб})$,легко убедиться в справедливости следующего утверждения.

Л е м м а 4 . Для любого асинхронного конечного автомата $\mathcal{A}$,имеющего граф необъединимости $G_{нб}$,

$$k_{min} \geq \gamma(G_{нб}).$$

О п р е д е л е н и е 8 . Раскраску вершин графа необъединимости $G_{нб}$ автомата $\mathcal{A}$ назовём d -правильной,если и только если:

I)любые две одноцветные вершины графа $G_{нб}$ не являются смежными,

2)все строки матрицы M_{nn} ,которые сопоставлены одноцветным вершинам графа $G_{нб}$,могут быть объединены в одно объединение.

О п р е д е л е н и е 9 . Наименьшее натуральное число p ,для которого существует d -правильная раскраска вершин графа необъединимости $G_{нб}$ автомата $\mathcal{A}$ в p цветов,назовем d -хроматическим числом графа $G_{нб}$ и обозначим $\gamma_d(G_{нб})$.

Отметим два свойства d -правильной раскраски:

I)для любого графа необъединимости $G_{нб}$

$$\gamma(G_{нб}) \leq \gamma_d(G_{нб});$$

2)любая d -правильная раскраска вершин графа $G_{нб}$ в p цветов определяет полную и замкнутую совокупность p непересекающихся множеств попарно объединимых строк матрицы M_{nn} (множеств объединимости)и наоборот.

Т е о р е м а 4 . Для любого асинхронного конечного автомата $\mathcal{A}$, имеющего граф необъединимости $G_{нб}$,

$$k_{min} = \gamma_d(G_{нб}),$$

и любой d -правильной раскраске вершин графа $G_{нб}$ в $\gamma_d(G_{нб})$ цветов соответствует полная и замкнутая совокупность k_{min} непересекающихся мно-

жеств объединимости (П-код минимальной длины).

Справедливость этого утверждения непосредственно следует из определений 7 и 8 и свойств d-правильной раскраски вершин графа $G_{нб}$.

Общая схема построения П-кода минимальной длины при этом может быть сформулирована следующим образом:

I)построение матрицы $M_{пп}$ с учетом требования попарного различения всех внутренних состояний;

2)определение попарной необъединимости всех строк матрицы $M_{пп}$ и построение графа необъединимости $G_{нб}$;

3)нахождение минимальной по числу цветов d-правильной раскраски вершин графа $G_{нб}$;

4)построение П-кода путем объединения всех строк матрицы $M_{пп}$,соответствующие которым вершины графа $G_{нб}$ окрашены в один и тот же цвет.

Следует отметить,что задача построения П-кода может быть сведена также к нахождению в обычном смысле правильной раскраски вершин некоторого конечного неориентированного графа.В этом случае не гарантируется нахождение П-кода минимальной длины,но упрощается процесс построения графа и раскраски его вершин.

Действительно,задача минимизации длины первоначального П-кода может быть сведена к сокращению числа различных строк частично определенной булевой матрицы(матрицы пар переходов $M_{пп}$)путем доопределения неопределенных значений элементов.Последняя задача,в свою очередь,может быть сведена к раскраске вершин некоторого конечного неориентированного графа - графа прямой необъединимости [I4] .

Пусть $M_б$ - произвольная частично определенная булева матрица размерами $w \times s$ и $\mathcal{M}_{M_б}$ - множество всех строк этой матрицы.Определим на множестве $\{0,1,-\}$ бинарное отношение π' ,полагая,что $0\,\pi'\,1$.Продолжим это отношение на множество $\mathcal{M}_{M_б}$ следующим образом: если $c_p = \{m'_{p1}, m'_{p2}, \ldots, m'_{ps}\}$ и $c_q = \{m'_{q1}, m'_{q2}, \ldots, m'_{qs}\}$ две строки матрицы $M_б$,то $c_p \pi' c_q$ тогда и только тогда,когда существует такое $j \in \{1,2,\ldots,s\}$,что $m'_{pj}\,\pi'\,m'_{qj}$.

Отношение π' назовем отношением прямой необъединимости,строки c_p и c_q матрицы $M_б$ - прямо необъединимыми.

О п р е д е л е н и е I 0 . Графом прямой необъединимости $G_{нн}$ назовем конечный неориентированный граф,вершинам которого взаимнооднозначно сопоставлены строки матрицы $M_б$ и любые две вершины которого соединены ребром тогда и только тогда,когда соответствующие им строки матрицы $M_б$ являются прямо необъединимыми.

Пусть γ_{min} - наименьшее число различных строк матрицы $M_б$,которое может быть найдено путем перебора всех возможных вариантов доопределения неопределенных значений элементов,а $\chi(G_{пн})$- хроматическое число графа $G_{пн}$.В работе [I4] доказана справедливость следующего утверждения.

Т е о р е м а 5 . Для произвольной частично определенной булевой

матрицы M_δ ,имеющей граф прямой необъединимости $G_{пн}$,

$$r_{min} = \gamma(G_{пн}),$$

и любая правильная раскраска вершин графа $G_{пн}$ в $\gamma(G_{пн})$ цветов определяет вариант сокращения числа строк матрицы M_δ до r_{min} .

Общая схема решения задачи построения П-кода в этом случае отличается от приведенной выше тем,что вместо графа необъединимости строится граф прямой необъединимости и вместо нахождения d -правильной раскраски вершин графа необъединимости осуществляется поиск минимальной по числу цветов в обычном смысле правильной раскраски вершин графа $G_{пн}$.

В работе [I4] показано,что длина k П-кода,найденного при помощи этого метода удовлетворяет неравенству

$$k_{min} \leq k \leq 2k_{min}$$

Предложенные методы могут быть использованы также в том случае,если вместо пар переходов рассматриваются такие более крупные образования как K -множества [4].Так как число K -множеств,как правило,значительно меньше числа пар переходов,то использование K -множеств позволяет расширить область практического применения предлагаемых методов.Однако,это происходит за счет снижения качества получаемого результата.

3.3. Минимизация булевых функций

Аналогичный подход может быть использован также при минимизации булевых функций в классе д.н.ф.

Пусть m'_i и m''_j - два элемента множества рабочих состояний $\mathcal{M}_1$ минимизируемой функции $F(x_1, x_2, \ldots, x_n)$.Введем бинарное отношение π' - отношение несклеиваемости на этом множестве.Будем считать,что $m''_i \, \pi' \, m''_j$,если и только если не существует такого интервала и на множестве $\mathcal{M}_1 \cup \mathcal{M}_\sim$,который покрывает как элемент m''_i ,так и элемент m''_j .

О п р е д е л е н и е I I . Конечный неориентированный граф $G_{нск}$, вершинам которого взаимнооднозначно сопоставлены элементы множества рабочих состояний булевой функции $F(x_1, x_2, \ldots, x_n)$ и любые две вершины которого соединены ребром тогда и только тогда,когда соответствующие им элементы множества $\mathcal{M}_1$ являются несклеиваемыми,назовем графом несклеиваемости булевой функции $F(x_1, x_2, \ldots, x_n)$.

Очевидно,что здесь также имеет место результат ,аналогичный результатам сформулированным в виде леммы 2 при рассмотрении минимизации числа внутренних состояний и леммы 4 при исследовании задачи кодирования внутренних состояний.Пусть d_{min} - минимальное число максимальных интервалов,определенных на множестве $\mathcal{M}_1 \cup \mathcal{M}_\sim$,которое в совокупности покрывают все элементы множества $\mathcal{M}_1$ (число дизъюнктивных членов кратчайшей тупиковой д.н.ф. минимизируемой функции),а $\gamma(G_{нск})$ - хроматическое число графа $G_{нск}$.

Л е м м а 5 . Для любой булевой функции $F(x_1, x_2, \ldots, x_n)$ имеющей граф несклеиваемости $G_{нск}$, $d_{min} \geq \gamma(G_{нск})$.

О п р е д е л е н и е I 2 . Раскраску вершин графа $G_{НСК}$ назовем f -правильной, если и только если:

I)любые две одноцветные вершины не являются смежными,

2)для любого подмножества элементов множества M_1 ,которым соответствует одноцветные вершины,существует интервал и на множестве $M_1 \cup M_0$,покрывающий все элементы этого подмножества.

О п р е д е л е н и е I 3 . Наименьшее натуральное число p ,при котором существует f -правильная раскраска вершин графа $G_{НСК}$ в p цветов,назовем f -хроматическим числом графа $G_{НСК}$ и обозначим $\chi_f(G_{НСК})$.

Очевидно,что $\chi(G_{НСК}) \leq \chi_f(G_{НСК})$.Так как,согласно определению I2, при f -правильной раскраске каждому множеству одноцветных вершин графа $G_{НСК}$ соответствует интервал u ,определенный на множестве $M_1 \cup M_0$,который покрывает все рабочие состояния,соответствующие вершинам этого множества, а каждый интервал,в свою очередь,может быть однозначно представлен при помощи конъюнкции своих внешних переменных [2] ,то каждая f -правильная раскраска вершин графа $G_{НСК}$ определяет некоторую д.н.ф.функции $F(x_1, x_2, \ldots, x_n)$, имеющей этот граф,и наоборот.Справедливо следующее утверждение,которое является аналогом теорем 3 и 4.

Т е о р е м а 6 . Для любой булевой функции $F(x_1, x_2, \ldots, x_n)$, имеющей граф несклеиваемости $G_{НСК}$,

$$d_{min} = \chi_f(G_{НСК}),$$

и любая f -правильная раскраска вершин этого графа в $\chi_f(G_{НСК})$ цветов определяет дизъюнктивную нормальную форму функции $F(x_1, x_2, \ldots, x_n)$,имеющую минимальное число дизъюнктивных членов.

Таким образом,в результате f -правильной раскраски вершин графа $G_{НСК}$ мы получим некоторый вариант покрытия множества M_1 интервалами,определяемыми на множестве $M_1 \cup M_0$ (при раскраске $\chi_f(G_{НСК})$ цветами - кратчайшее покрытие).Каждый из этих интервалов или является максимальным,или покрывается некоторым максимальным интервалом.Следовательно,общая схема построения кратчайшей тупиковой д.н.ф.булевой функции может быть сформулирована следующим образом:

I)определение попарной несклеиваемости всех рабочих состояний минимизируемой функции и построение графа несклеиваемости $G_{НСК}$;

2)нахождение минимальной по числу цветов f -правильной раскраски вершин графа $G_{НСК}$;

3)поиск для каждого подмножества элементов множества M_1 ,соответствующие которым вершины графа $G_{НСК}$ окрашены в один и тот же цвет,максимального интервала,покрывающего все элементы этого подмножества.

Следует отметить,что предложенный метод минимизации булевых функций может быть также успешно использован для минимизации булевых функций,заданных в интервальной форме [I0] .Однако,в этом случае результат зависит от первона-

чального задания и метод позволяет найти некоторую безызбыточную д.н.ф., которая может не оказаться кратчайшей тупиковой д.н.ф.

Возможность минимизации булевых функций, заданных в интервальной форме, позволяет использовать предлагаемый метод для синтеза комбинационных схем, свободных от логических состязаний.

Как известно, например, из работ [9, 15] , минимизация системы булевых функций путем введения дополнительных переменных может быть сведена к минимизации одной булевой функции. Этот результат позволяет приведенную выше схему успешно использовать также для минимизации системы булевых функций. В этом случае, однако, не гарантируется нахождение минимального решения.

4. Система алгоритмов

На основе приведенных выше результатов создана система алгоритмов сквозного синтеза асинхронных конечных автоматов, позволяющая строить быстродействующие и устойчивые относительно состязаний логические схемы этих автоматов (рис. I).

При этом предполагается, что условия работы синтезируемого автомата заданы в виде таблицы переходов или графа переходов.

Блок построения графа несовместимости содержит алгоритм определения попарной несовместимости любых двух внутренних состояний синтезируемого автомата и алгоритм построения графа несовместимости в виде некоторого упорядочения множества подмножеств вершин, где j-тое подмножество содержит все вершины, смежные с j-той вершиной графа.

Блок алгоритмов раскраски вершин графа является основной частью системы и содержит комплекс алгоритмов, позволяющих осуществить в обычном смысле правильную, c-правильную , d-правильную и f- правильную раскраску вершины графов. Этот блок содержит как алгоритмы нахождения требуемой раскраски вершин графа с минимальным числом цветов, так и алгоритмы приближенного решения задачи раскраски.

При помощи следующего блока алгоритмов по первоначальному заданию синтезируемого автомата и результатам раскраски вершин графа несовместимости осуществляется построение таблицы переходов (графа переходов) минимизированного автомата.

Блок построения графа необъединимости состоит из алгоритмов нахождения безызбыточного множества пар переходов, определения их попарной необъединимости и построения графа необъединимости.

После раскраски вершин графа необъединимости при помощи алгоритмов, содержащихся в блоке построения структурных таблиц (уравнений), осуществляется построение соответствующего этой раскраске П-кода путем объединения всех пар переходов, сопоставленные которым вершины графа необъединимости окрашены в один и тот же цвет. После этого, используя таблицу переходов (граф переходов) минимизированного автомата и найденный П-код, реализуется построение структурной таблицы (системы структурных уравнений), определяющих условия работы

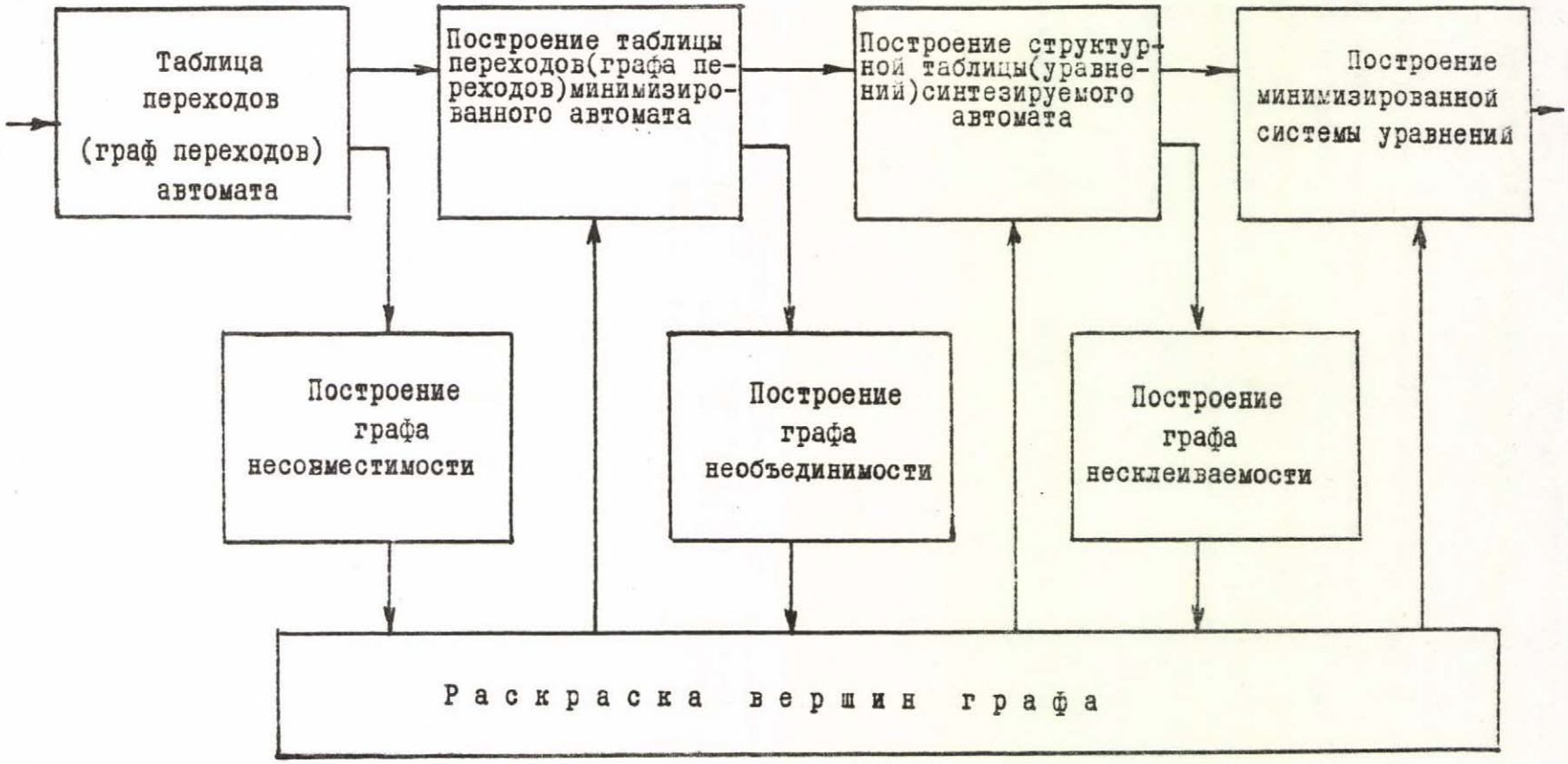


Рис.I. Блок-схема системы алгоритмов.

логической структуры синтезируемого автомата.

При помощи алгоритмов блока построения графа несклеиваемости осуществляется построение булевой функции, представляющей найденную структурную таблицу (систему уравнений), в виде перечисления её множеств рабочих и нерабочих состояний, определяется попарная несклеиваемость всех рабочих состояний и строится граф несклеиваемости.

Далее, используя результаты раскраски вершин графа несклеиваемости и найденную ранее структурную таблицу, осуществляется построение минимизированной системы уравнений логической структуры синтезируемого автомата в классе д.н.ф.

5. Заключение

В результате изучения существующих общих схем решения основных задач синтеза асинхронных конечных автоматов предложен единый подход к решению всех этих задач, который основан на сведении их к раскраске вершин конечных неориентированных графов и позволяет разработать весьма компактную и эффективную систему алгоритмов сквозного синтеза асинхронных конечных автоматов.

Методы минимизации числа внутренних состояний, кодирования внутренних состояний П-кодом и минимизации булевых функций, разработанные на основе предлагаемого подхода, не требуют предварительного нахождения множества $\mathfrak{M}$, позволяют сократить объём запоминаемой в процессе решения промежуточной информации и снизить общую трудоёмкость решения рассматриваемых задач.

Так как, с одной стороны, трудоемкость раскраски вершин графа зависит от соотношения числа вершин и ребер раскрашиваемого графа и снижается при уменьшении числа ребер [16], а, с другой стороны, увеличение мощности множества $\mathfrak{M}$, как правило, приводит к уменьшению числа ребер раскрашиваемых графов, то снижение трудоемкости решения рассматриваемых задач при помощи предложенных методов является наиболее значительным именно для тех случаев, когда мощность множества $\mathfrak{M}$ велика.

Кроме того, следует отметить, что предлагаемый подход имеет также определенное самостоятельное значение и может быть использован не только для решения рассмотренных задач синтеза, но и для решения ряда других задач комбинаторного характера, которые сводятся к поиску кратчайшего покрытия.

Литература

I. Берж К.: Теория графов и ее применение М., ИЛ. I962.

2. Новоселов В.Г.: Минимизация булевых функций (обзор) - В сб. "Итоги исследований по кибернетике", изд-во Томского ГУ, Томск, I968.

3. Paull M.C., Unger S.H.: Minimizing the Number of States in Incompletely Specified Sequential WSwitching Functions. IRE Irans. Electron. Comput., EC-8, 1959. 3.

4. Iracey J.H.: Internal States Assignment for Asynchronous Sequential Machines. IEEE, Irans. Electron. Comput. EC-15, 1966. 4.

5. McCluskey E.J.: Minimization of Boolean Functions, BSJJ, v. 35. 1956.6.

6. Рогинский В.Н.: Учет неиспользованных состояний при синтезе релейно-контактных схем. Автоматика и телемеханика, т. I5. I954. 3.

7. Graselli A., Luccio F.: A Method for Minimizing the Number of Internal States in Incompletely Specified Sequential Networks. IEEE Irans Electrion. Comput., EC-14, 1965. 3.

8. Moon J.W., Moser L.: On Cliques in Graphs. Israel J. Math. 3. 1965. 1.

9. Миллер Р.: Теория переключательных схем. Т.П., "Наука", I970.

I0. Гаврилов М.А,: Теоретические проблемы практического приложения теории конечных автоматов. Труды Международного симпозиума по прикладным аспектам теории автоматов. Варна, I97I.

II. Мак-Класки Э.Дж.: Многотактные схемы с минимальным числом состояний для ограниченного класса неполностью определеённых таблиц переходов. В кн.: Теория конечных и вероятностных автоматов. М., "Наука" I965.

I2. Гринберг Э.Я., Илзиня И.Г.: О раскраске вершин неориентированных графов. В кн.: Автоматика и вычислительная техника. 7. Рига, изд-во АН Латв.ССР. I964.

I3. Ланге Э.Э., Фричнович Г.Ф.: К минимизации числа состояний одного класса асинхронных конечных автоматов. Автоматика и вычислительная техника. I97I. 4.

I4. Илзиня И.Г., Фрицнович Г.Ф.: Кодирование внутренных состояний асинхронного конечного автомата П-кода. Автоматика и вычислительная техника I970. 6.

I5. Торопов Н.Р.: Сведение минимизации системы булевых функций к минимизации одной функции. В сб.: "Цифровые модели и интегрирующие структуры", Таганрог. I970.

I6. Илзиня И.Г., Фрицнович Г.Ф.: Об одном подходе к решению проблемы кратчайшего покрытия. Автоматика и вычислительная техника I972. I.

Алгоритмический метод факторизации минимальной формы булевой функции

Ян Коленичка,
кафедра вычислительных машин, ВУТ Брно, ЧССР

Идея факторизации или поиска общих множителей минимальной формы булевой функции так называемых факторов основана на поиске множества общих переменных в членах минимальной формы функции $F(x_1, x_2, \ldots, x_n)$

Пример:

Пусть существует функция

$$F_1 = ab\bar{c}\bar{d}ef + ab\bar{c}\bar{d}\bar{g}h + Y \qquad (1)$$

Будем предполагать, что для реализации этой функции можно пользоваться элементами НЕ-И с максимальным числом входов $d_v = 4$. Из уравнения (1) видно, что в первых двух членах число переменных больше d_v, то значит, эти члены мы должны подразделить так, чтобы каждая часть содержала только до d_v переменных. При случайном подразделении мы можем реализовать F_1, например как на рис.1, причем в реализации находится семь элементов НЕ-И.

При более подробном осмотре можно видеть, что первый и второй члены содержат общую часть переменных $-a, b, \bar{c}, d$, которую мы можем реализовать согласно рис. 2.

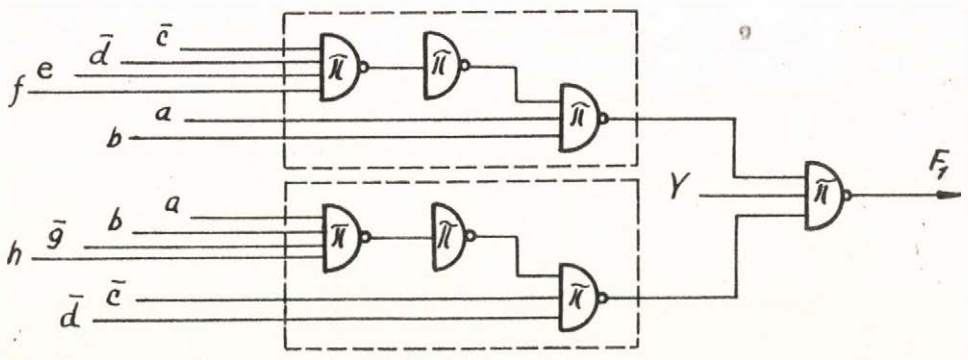


Рис. 1.

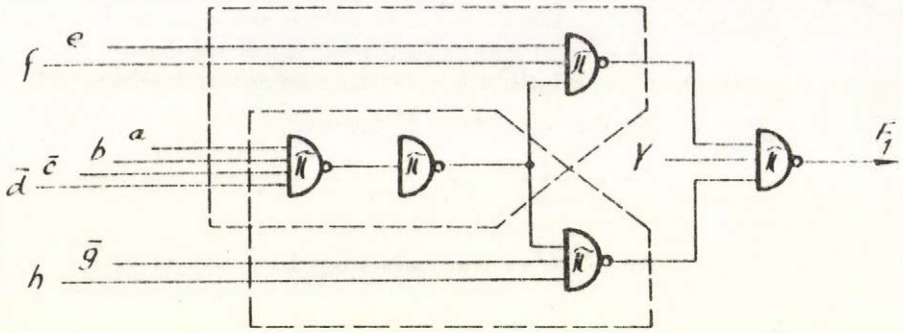


Рис. 2.

Реализация на рис.2 содержит только пять элементов НЕ - И, что значит, что задача поиска общей части членов минимальной формы имеет значение.

Приведенный пример изображает основную идею факторизации. Очевидно, что в процесс факторизации войдут только те члены минимальной формы, у которых число переменных больше d_v.

Алгоритм факторизации, разработанный Шнейдером и Диетмейером, является очень несообразительным, потому что он работает с некоторыми "потенциальными" факторами. Число этих факторов, оптимальный из которых выбирается после проведения огромного количества операций, является очень большим.

Предлагаемый метод поиска факторов (общих множителей) весьма быстрый и однозначный.

Определение 1.

Конъюнкцию по крайней мере двух общих букв по крайней мере двух членов минимальной формы функции $F(x_1, x_2, \ldots, x_n)$ будем называть <u>фактором</u> f (общим множителем f).

Процесс поиска факторов будем называть <u>факторизацией</u>.

Члены минимальной формы, входящие в процесс факторизации, будем называть <u>оригиналами</u>.

Определение 2.

Произведение $P \cdot D$, где P - число оригиналов K_j, для которых $f \subseteq K_j$ и D - длина фактора (число переменных в факторе), называется <u>мерой предпочтения</u> фактора f.

(Уже было показано, что фактор с максимальной мерой предпочтения является оптимальным для реализации в логической схеме.)

Исходным пунктом метода является матрица оригиналов. Она представляет собой изображение членов минимальчой формы, которые входят в факторизацию, т.е. оригиналов, причем каждому оригиналу соответствует пара строчек; каждый столбец в нее соответствует одной переменной x_i ; единица (соответственно нуль) в столбце x_i на первой строчке этой пары выражают, содержит ли член минимальной формы K_j перменную $\bar{x}_i$ (соответственно не содержит ее).
Подобно тому во второй строчке для переменной x_i единица (соответственно нуль) выражают, содержит ли K_j x_i (соответственно не содержит его).

1-й шаг

В первом шагу алгоритма создается матрица оригиналов.

Пример:

Пусть задана минимальная форма F :

$$F = K_1 + K_2 + K_3 + K_4 + K_5 + K_6 + K_7 =$$
$$= \bar{x}_1 x_2 \bar{x}_3 x_4 x_8 \bar{x}_9 + x_3 x_{10} + x_1 x_5 \bar{x}_6 x_7 \bar{x}_{13} +$$
$$+ x_2 \bar{x}_3 x_4 x_6 \bar{x}_7 x_{13} + \bar{x}_1 x_2 \bar{x}_3 x_4 x_{10} \bar{x}_{11} +$$
$$+ \bar{x}_1 x_5 \bar{x}_6 x_7 \bar{x}_8 + \bar{x}_1 x_2 \bar{x}_3 x_4 x_5 \bar{x}_6 x_7 x_{12} \bar{x}_{13}$$

Предполагаем, что $d_v = 4$, то значит, член K_2 не входит в факторизацию. Соответствующая матрица оригиналов находится на рис.3.

			x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}
1	K_7	$\bar{x}$	1	0	1	0	0	1	0	0	0	0	0	0	1
		x	0	1	0	1	1	0	1	0	0	0	0	1	0
2	K_1	$\bar{x}$	1	0	1	0	0	0	0	0	1	0	0	0	0
		x	0	1	0	1	0	0	0	1	0	0	0	0	0
3	K_5	$\bar{x}$	1	0	1	0	0	0	0	0	0	0	1	0	0
		x	0	1	0	1	0	0	0	0	0	1	0	0	0
4	K_4	$\bar{x}$	0	0	1	0	0	0	1	0	0	0	0	0	0
		x	0	1	0	1	0	1	0	0	0	0	0	0	1
5	K_3	$\bar{x}$	0	0	0	0	0	1	0	0	0	0	0	0	1
		x	1	0	0	0	1	0	1	0	0	0	0	0	0
6	K_6	$\bar{x}$	1	0	0	0	0	1	0	1	0	0	0	0	0
		x	0	0	0	0	1	0	1	0	0	0	0	0	0

Рис. 6.

Приведенное изображение оригиналов позволяет непосредственно создавать факторы.

Рассмотрим следующий случай: пусть два оригинала K_r, K_s заданы в виде двухстрочных субматриц $(K_r), (K_s)$ некоторой матрицы оригиналов следующим образом:

$$(K_r) = \begin{pmatrix} a_{01}, a_{02}, \ldots, a_{0m} \\ a_{11}, a_{12}, \ldots, a_{1m} \end{pmatrix} \qquad a_{ij} \in (0,1)$$

$$(K_s) = \begin{pmatrix} b_{01}, b_{02}, \ldots, b_{0m} \\ b_{11}, b_{12}, \ldots, b_{1m} \end{pmatrix} \qquad b_{ij} \in (0,1)$$

где m – число столбцов матрицы оригиналов. Очевидно, что общее множество переменных изображено как строчное произведение субматриц

$$(K_r)\cdot(K_s) = \begin{pmatrix} a_{01}\cdot b_{01}, & a_{02}\cdot b_{02}, & \ldots, & a_{0m}\cdot b_{0m} \\ a_{11}\cdot b_{11}, & a_{12}\cdot b_{12}, & \ldots, & a_{1m}\cdot b_{1m} \end{pmatrix}, \qquad (2)$$

причем если эта новая субматрица (2) содержит по крайней мере две единицы, она является изображением фактора.

Теорема 1

Фактор содержившийся по крайней мере в двух оригиналах

$$(K_{i_1}), (K_{i_2}), \ldots, (K_{i_r})$$

изображен в виде субматрицы, образованной строчным произведением

$$(K_{i_1})\cdot(K_{i_2})\cdot \ldots \cdot (K_{i_r})$$

которая содержит по крайней мере два ненулевых элемента. Иначе оригиналы не содержат общего фактора.

(Без доказательства).

На основе теоремы 1 можно приступить к созданию алгоритма, который основан на постепенном проведении строчных произведений всех субматриц матрицы оригиналов.

2-й шаг:

Во 2-ом шагу осуществляются строчные произведения таким путем:

2a) $i = 1$; $j = 1$

2b) Взять субматрицу с порядковым номером i и выполнить строчное произведение с $(i+j)$ -ной субматрицей. Если это произведение содержит по крайней мере два ненулевых элемента, сохранить ее и перейти на 2c. В обратном случае, если полученная субматрица не содержит по крайней мере два ненулевых элемента, она не является изображением фактора, сделать $j = j + 1$. Если $(i + j)$ не является последним порядковым номером последней строчки, вернуться на 2b. Если $(i + j)$ является последним порядковым номером, перейти к шагу 3.

2c) Записать во вспомогательную таблицу P1 длину созданного фактора и порядковые номера субматриц образующих этот фактор,

в поднимающейся последовательности. Вычислить меру предпочтения как произведение длины фактора и числа порядковых номеров в таб. P1. Сделать $j = j + 1$, и если $(i + j)$ не является последним порядковым номером, перейти на 2*b*, в обратном случае перейти к шагу 3.

3-й шаг:

Проверить, обнаружен ли любой фактор, т.е. содержит ли таб.P1 по крайней мере одну строчку. В положительном случае перейти к шагу 4; в отрицательном случае сделать $i = i + 1$, $j = 1$; если i не является последним порядковым номером, перейти на 2*b*, в обратном случае окончить алгоритм.

4-й шаг:

4a) Расставить строчки таб.P1 в зависимости от падающей длины факторов, и согласно тому расставить тоже им соответствующие субматрицы факторов. Сделать $p = 1$, $q = 1$, $W_1 = 0$, $W_2 = 0$.

4*b*) Взять субматрицу фактора, соответствующую p-ой строчке, и провести строчное произведение с субматрицей фактора соответствующей $(p + q)$-ой строчке, если удовлетворено следующее условие: если вход в шаг 4 был проведен из шага 3, провести произведение всегда; если вход в шаг 4 был проведен из шага 6 - провести произведение только тогда, если множество порядковых номеров субматриц (конъюнкций) матрицы оригиналов в $(p + q)$-ой строчке таб.P1 не содержит все множество порядковых номеров конъюнкций p-ой строчки; в случае, что множество содержит все множество порядковых номеров конъюнкций p-ой строчки, записать в другую вспомогательную таблицу P2 p-ую строчку таблицы P1. Если полученная субматрица содержит по крайней мере два ненулевых элемента, перейти на 4c, в обратном случае на 4*d*.

4c) Сохранить полученную субматрицу и записать в другую вспомогательную таблицу P2 длину по-новому созданного фактора и порядковые номера субматриц матрицы оригиналов, из которых мы получили фактор, и вычислить меру предпочтения.

В случае, что некоторая субматрица из пары субматриц факторов (строчное произведение которых было проведено) изображает фактор с большей мерой предпочтения чем имеет из них полученный новый фактор, записать строчку из таб. P1, соответ-

ствующую этой субматрице, в таблицу P2. Сделать $W_1 = 1$, $W_2 = 1$ и перейти к 4d.

4d) Сделать $q = q + 1$ и если $(p + q)$ является последним порядковым номером, перейти к 4e, в обратном случае к 4b.

4e) Проверить, существует ли в таб. P2 некоторый новый фактор (т.е. имеет ли силу $W_1 = 1$), созданный для как раз существующей величины p. В положительном случае перейти к 4f, в отрицательном случае записать в качестве новой строчки в таб. P2 p-ую строчку таб. P1 и перейти к 4f.

4f) Вычеркнуть в таб. P1 p-ую строчку. Сделать $p = p + 1$, $q = 1$, $W_1 = 0$ и если p совпадает с последним порядковым номером строчки таб.P1, записать эту строчку из таб. P1 в таб. P2, вычеркнуть ее из таб. P1 и перейти к шагу 5. Если p не совпадает с последним порядковым номером, перейти к 4b.

5-й шаг:

Проверить, если $W_2 = 0$. В положительном случае перейти на шаг 7, в обратном случае устранить из таб.P2 строчку, которая соответствует фактору длины d_1 полученному произведением субматриц $(K_{i_1})(K_{i_2}) \cdots\cdots\cdots (K_{i_r})$, если существует одновременно строчка, соответствующая фактору длины $d_2 \geq d_1$ созданному на основе произведения по меньшей мере следующих субматриц $(K_{i_1})\cdot(K_{i_2})\cdot \cdots\cdots \cdot(K_{i_r})$. Эту операцию провести для всех строчек таб. P2. Перейти к шагу 6.

6-й шаг:

Полученную таб. P2 считать новой таблицей P1. Перейти к шагу 4.

7-й шаг:

Выбрать фактор с максимальной мерой предпочтения, сохранить его и перейти к шагу 9. Если имеется несколько факторов, перейти к шагу 8.

8-й шаг:

Определить для каждого из выбранных факторов максимальной меры предпочтения число субматриц матрицы оригиналов, которые на основе подбора этого фактора можно вычеркнуть из дальнейшей факторизации. Провести это следующим образом:

вычесть от длины каждой конъюнкции в матрице оригиналов, которая содержит фактор, его длину и определить, меньше ли эта упрощенная длина конъюнкции чем d_v. В положительном случае рассматриваемая конъюнкция не войдет в процесс дальнейшей факторизации.

Если число субматриц матрицы оригиналов, которые можно вычеркнуть из дальнейшей факторизации, для рассматриваемых факторов одинаково, выбрать самый длинный фактор, длина которого максимально равна произведению $n \cdot d_v$, где n - минимальное натуральное число, для которого это условие выполнено. Перейти к шагу 9.

9-й шаг:

Выбрать субматрицу избранного фактора и создать ее отрицание, т.е. произвести отрицание каждого ее элемента. Потом провести строчное произведение отрицанию подверженной субматрицы фактора на каждую субматрицу матрицы оригиналов, из которой был фактор создан.

Таким же путем провести строчное произведение отрицанию подверженной субматрицы фактора на все субматрицы факторов, соответствующие таб.P2, и вычислить новые длины факторов и меры предпочтения. Из таб. P2 вычеркнуть строчки, которым соответствует "фактор" длины меньшей двух. Определить, пуста ли таб. P2. В отрицательном случае вернуться к шагу 7, в обратном случае перейти к шагу 10.

10-й шаг:

Выбросить из матрицы оригиналов все субматрицы от 1 до i включительно. Вычислить новые длины конъюнкций в матрице оригиналов и выбросить из матрицы оригиналов конъюнкции (и их

субматрицы), длина которых меньше d_v.

Оставшиеся субматрицы расставить в зависимости от падающей длины конъюнкций. Определить, содержит ли новая матрица оригиналов по крайней мере две субматрицы; в положительном случае перейти к шагу 2, в отрицательном случае алгоритм окончить.

После проведения шага 10 определены уже все факторы - в том случае, если они существуют.

Использование комплексных пар для кодирования внутренних состояний.

Р. Новански /ИИВТУ - Прага/

Предполагается, что задан /не должен быть польностью определенным/ конечный автомат

$$A = \langle I, Q, O, \delta, \lambda \rangle \qquad \begin{matrix} I_m \\ 2_n \end{matrix}$$

где I, Q, O соответствуют алфавитам входных, внутренних и выходных символов и δ и λ функции переходов и выходов.

В дальнейшем мы будем заниматься декомпозицией его функции переходов $\delta : Q \times I \longrightarrow Q$ в виде рис. I в систему запоминающих элементов в виде рисунков 2, 3, 4.
Функции возбуждения запоминающих элементов мы ищем таким образом, чтобы их зависимости от внутренних переменных были возможно максимально редуцированы. Для того мы пользуемся так называемыми парами комплексов.
Для объяснения понятия пары комплексов мы должны уточнить следующие понятия.

Опр. I: Симплексом в дальнейшем подразумевается любая пара элементов, не смотря на их очередь. В настоящем случае мы будем симплексом к.а. A подразумевать неупорядоченную пару его внутренних состояний.

Опр. 2: Любое множества $X = \bigcup_{e=1}^{n} \{(2\alpha e, 2\beta e)\}$ различных симплексов к.а. A будем в дальнейшем называть комплексом автомата A.

Комплекс X определяет на множестве симметричное, нерефлексивное отношение несовместимости, которое является вообще не транзитивным. Внутрение состояния 2α и 2β являются по отношению к комплексу X различимыми, если $(2\alpha, 2\beta) \in X$. В противоположном случае /$(2\alpha, 2\beta) \notin X$ / они по отношению к комплексу X являются совместимыми.

Из определения вытекает, что с комплексами возможно работать как с множествами симплексов.

Дизъюнкция комплексов обозначается отношением $X_I + X_2$
Конъюнкция комплексов обозначается отношением $X_I + X_2$
Отрицание комплекса X_I обозначается отношением $X_2 = \overline{X_I}$

Опр. 3: Мы говорим, что множество внутренних переменных $\{y_j\}\ j\in X$ различает комплекс $X = \bigcup_{e=1}^{n}\{(2\alpha_e, 2\beta_e)\}$ тогда, когда для каждого симплекса $(2\alpha_e, 2\beta_e)\in X$ существует переменная y_j, $j\in K$ так, что $y_j(2\alpha_e) \neq y_j(2\beta_e)$.

Введем теперь следующее обозначение

$$\{I_к^{-I}\ 2\gamma\} = \{2e/\delta(2e, I_к) = 2\gamma\}$$

Опр. 4: Пусть $\Phi = \{(2\alpha, 2\beta)\}$ комплекс содержащий только один симплекс $(2\alpha, 2\beta)$. Обозначим $\eta(\Phi) \equiv \eta(2\alpha, 2\beta)$ комплекс автомата, содержащий все возможные симплексы, которые отображаются под влиянием некоторого входного символа $I_к$, к = I, 2, ... м, на симплекс $(2\alpha, 2\beta)$.

Понятно, что

$$\eta(\Phi) \equiv \eta(2o, 2\beta) = \bigcup_{k=1}^{m}(\{I^{-1}2\alpha\} x \{I_k^{-1}\ 2\beta\})$$

В общем случае комплекса

$$\varphi = \bigcup_{e=1}^{n}\{(2\alpha_e, 2\beta_e)\}$$

мы можем определить

$$\eta(\varphi) = \bigcup_{e=1}^{n}\eta(2\alpha_e, 2\beta_e) \qquad /I/$$

Опр. 5: Мы говорим, что внутренняя переменная $y_i = y_i(t+I)$ зависит только от подмножества внутренних переменных $\{y_j / j\in K_i\}$ тогда, когда возможно её выразить в виду

$$y_i = F_i(\{y_j\}_{j\in K_i}, X),$$

где $X = \{x_I, x_2, \ldots, x_p\}$ множество входных переменных

Дальше мы скажем, что множество внутренних переменных $\{y_i / i\in L\}$ зависит в целом от подмножества внутренних переменных $\{y_j / j\in K_L\}$ тогда, когда возможно писать

$$y_i = F_i(\{y_j\}_{j\in K_L}, X) \qquad i\in L$$

Для кодирования внутренних состояний возможно доказать следующую теорему

Теорема I

Пусть на кодирование внутренних состояний наложено такое ограничение, чтобы некоторая внутренняя переменная y_i различала комплекс Ψ и чтобы она зависела только от подмножества внутренних переменных $\{y_j / j\in K_i\}$. В таком случае необходимым и достаточным условием для существования такого внутреннего кода является то, чтобы подмножество

внутренних переменных $\{y_j / j \in K_i\}$ различало комплекс $\gamma(\varphi)$. Различает ли некоторое подмножество внутренных переменных $\{y_j / j \in K_i\}$ комплекс $\Psi \geq \gamma(\varphi)$, потом оно различает тоже комплекс $\gamma(\varphi)$. На основе того можно ввести следующее определение.

Опр. 6: Упорядоченная пара комплексов автомата называется парой комплексов /комплексной парой/ тогда, когда для любого симплекса $(2\alpha, 2\beta) \in \varphi \Rightarrow \gamma(2\alpha, 2\beta) \leq \Psi$.

Для комплексов автомата нетрудно доказать следующее свойство. Из равенства $\varphi = \varphi_I + \varphi_2 \Rightarrow \gamma(\varphi) = \gamma(\varphi_I) + \gamma(\varphi_2)$

Для кодирования внутренних состояний удобно ввести так называемые полные бихроматические комплексы /сокращенно полные бикомплексы/.

Опр. 7: Полным бикомплексом называется такой комплекс φ, соответствующий график которого $G = (Q = \varphi)$ /см. например рис. 5/ соедржит точно две максимально независимые множества вершин.

Каждому полному бикомплексу φ_i можно присоединить одну внутреннюю переменную y_i, которая точно различает состояния всех симплексов комплекса φ_i. Значит всех состояниям /вершинам/ каждого из двух максимально независимых множеств графика $G_i = (Q, \varphi_i)$ можно пресоединить одинаковое значение внутренней переменной y_i.

Пусть значения внутренних переменных y_I, y_2, ..., y_s определяют внутрение состояния автомата A', который является расширением заданного автомата A. Тогда любая внутренняя переменная y_i $i = I, 2, .., s$ различает на множестве Q состояний автомата A точно один полный бикомплекс φ_i. В таком случае теорема I высказывает, что необходимым и достаточным условием для того, чтобы переменная y_i зависела от подмножества внутренних переменных $\{y_j / j \in K_i\}$/ и от входов/ является условие

$$\sum_{j \in K_i} \varphi_j \geq \gamma(\varphi_i), \qquad /2/$$

где K некоторое подмножество множества индексов всех внутренних переменных $K_0 = \{I, 2, ..., s\}$.

Соотношение /2/ мы называем неравенством информационного тока. В этом неравенстве $\gamma(\varphi_i)$ представляет наименьшее количество информации о состоянии автомата в такте t, из которого возможно вычислить комплекс φ_i в такте $t + I$. Неравенство /2/ высказывает, что для вычисления y_i в следующем состоянии необходимо знать о настоящем состоянии автомата A по крайней мере столько информации, сколько

её содержится в комплексе $\eta(\varphi_i)$. Надо отметить, что выражение

$$\sum_{j \in K_i} \varphi_j$$

в отношении /2/ определяет такой комплекс

$$\psi_i = \sum_{j \in K_i} \varphi_j$$

автомата A, что (φ_i, ψ_i) является парой комплексов.

Пример: Пусть функция переходов к.а. A задана таблицей переходов /табл. I/. Полный бикомплекс φ_I пусть задан графиком $G_I = (Q, \varphi_I)$ /рис. 5а/. Бикомплексу φ_I мы можем присоединить внутреннюю переменную y_I. В следствии теоремы I переменная y_I всегда будет зависет от переменных, которые различают комплекс $\psi_I = \eta(\varphi_I)$. Согласно определению $\eta(\varphi_I)$ с помощью равенства /I/ мы можем начертить график $H_I = (Q, \psi_I)$, соответствующий комплексу ψ_I /рис. 5б/. Из этого вытекает, что симплексы комплекса η /φ_I/ можно различать только при помощи подходящим образом присоединенной одной переменной y_2, потому что комплекс $\eta(\varphi_I)$ является полным бихроматическим комплексом. Если мы это осуществим, то будет $\varphi_2 = \eta(\varphi_I)$.

Аналогично мы находим, что комплекс $\eta(\varphi_2)$ можно исчислить так, что после присоединения к нему симплекса /4, 5/ мы получим полный бикомплекс $\psi_2 = \eta(\varphi_2) + \{(4,5)\}$ /см. рис. 6/. После присоединения следующей переменной y_3, соответствующей полному бикомплексу ψ_2 мы получим $\varphi_3 = \psi_2$ /см. рис. 7/ и т.д.

Из сверхсказанного вытекает, что в результате мы получили кодирование внутренних состояний /см. табл 2/. Из него обыкновенным способом мы получим следующие уравнения для внутренних переменных.

$y_I = F_I/x, y_2/ \quad /= x + y_2/$

$y_2 = F_2/x, y_3/ \quad /= x \oplus y_3/$

$y_3 = F_3/x, y_I, y_2, y_3/ \quad /= x y_2 \bar{y}_3 + \bar{x} y_2 y_3 + x \bar{y}_2 y_3 + y_I \bar{y}_3/$

$z = \bar{x}\bar{y}_3 + x y_2 / \bar{y}_I + y_3/$

Табл. 2 является таблицей расширенного конечного автомата A', который предоставляет возможность реализации с более удобной структурой чем первоначальный автомат A, и который с точки зрения внешнего поведения является с ним эквивалентным.

Из предыдущего примера мы видим, что пары комплексов предоставляют возможность анализа автоматов с точки зрения редукции зависимостей не одной, а нескольких внутренних переменных. Имея в виду совместные редукции зависимостей, необходимо ввести понятие системы пар комплексов:

Опр. 8: Закрытой сисмемой пар комплексов называется такая система пар комплексов (φ_I, ψ_I), (φ_T, ψ_2), ..., (φ_S, ψ_S), которая удовлетворяет следующим условиям

$$\sum_{i=1}^{s} \varphi_i = I$$

$$\psi_i = \sum_{j \in K_i} \varphi_j$$

$$\sum_{i=1}^{s} d_i + \sum_{\substack{i,j=1 \\ i \neq j}}^{s} d_{ij} + \sum_{\substack{i,j,k=1 \\ i \neq j \neq k \\ i \neq k}}^{s} d_{ijk} + \ldots \leq 2^s - n$$

$$L = \sum_{i=1}^{s} \ell(k_i) = \min$$

где $s = [\log_2 n]$ является минимальным числом внутренних переменных, необходимых для кодирования n внутренних состояний автомата A.

K_i – удобное подмножество множества K_0 индексов всех переменных $(K_0 = \{I, 2, \ldots, s\})$

d_i – число изолированных вершин в графиках комплексов φ_i/см. например рис. 5, 6, 7/

d_{ij} – число изолированных вершин совместно в графиках комплексов φ_i и φ_j

d_{ijk} – число изолированных вершин совместно в графиках комплексов φ_i и φ_j и φ_k и т.д.

$\ell(K_i)$ – число элементов множества индексов $K_i - \{i\}$

L – общее число зависимостей функций возбуждения всех элементов памяти от внутренних переменных для выбранной системы пар комплексов.

Замечания:

I/ $\ell(K_i) = \mathrm{card}(K_i - \{i\})$ – результат одной теоремы для закрытых систем пар комплексов.

2/ Понятия пар комплексов и закрытых систем пар комплексов можно использовать тоже тогда, когда используются в запоминающей части последовательной схемы не только элементы задержки,

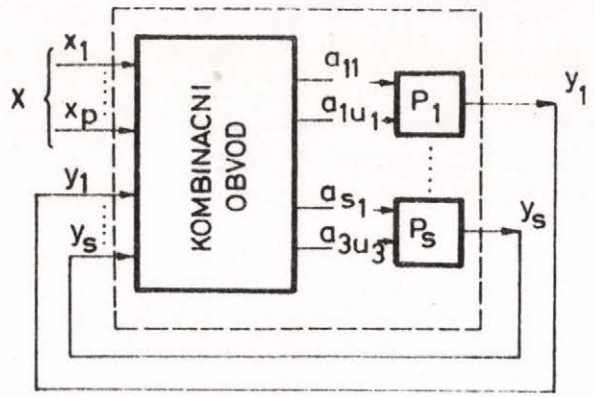
а также общие запоминающие элементы /тригеры/. Для таких реализаций были в /2/ выведены некоторые необходимые и достаточные условия, касающиеся кодировения внутренних состояний, для которых используется минимальное число внутренних переменных. В таком случае пары комплексов предоставляют возможность редуцирования зависимостей функций возбуждения элементов памяти на внутренних переменных. В работе /2/ приводится тоже соответствующий алгоритм выбора пар комплексов, который возможно было бы программировать на вычислительной машине.

Литература:

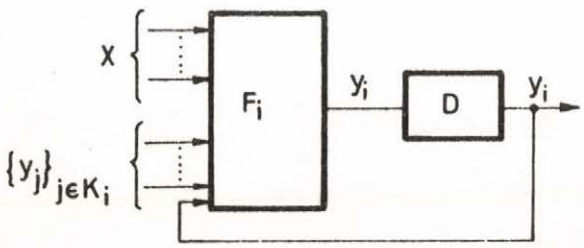
1/ **Novansky R. "The Complex Pairs and Their Application to the Syntheses of Finite Automata" Information Processing Machines, No 15, pp 97 - 121, Academia Prague 1971**

2/ **Novansky R. "Using of the Complex Pairs in the Coding of Internal States of Finite Automata" to appear in Information Processing Machines No 19**

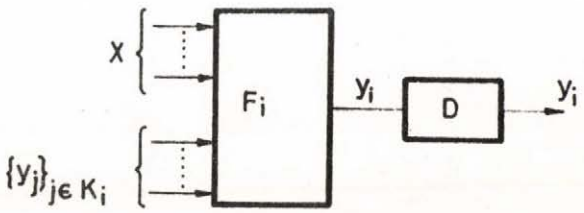
3/ **Novansky R. "Kodováni vnitrnich stavu konecnych automatu s syuzitim komplexovych páru",** Научный отчет № **UVVTR 15 008** ИИТУ Прага 1972



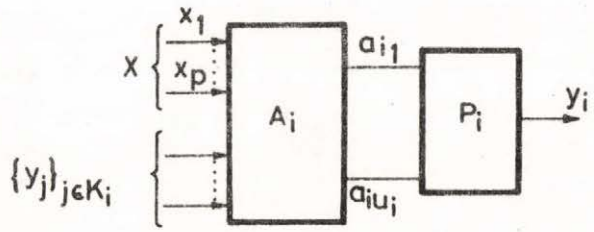
Puc.1.



Puc.2.



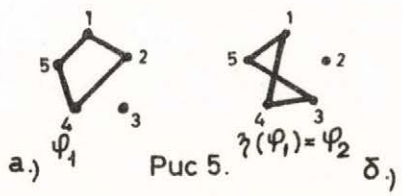
Puc. 3.



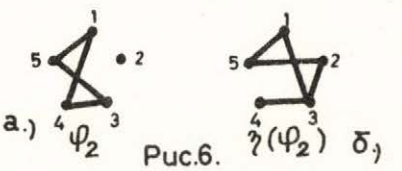
Puc. 4.

δ	0	I
1	1/I	2/0
2	3/I	5/0
3	4/0	3/0
4	2/I	5/I
5	5/0	2/I

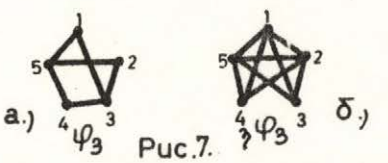
Taδ 1.



Puc 5.



Puc.6.



Puc.7.

$y_3y_2y_1$	δ'	0	I
000	– 1	1	2'
001	– 2	3	5
011	– 2'	3'	5
100	– 3	4	3'
101	– 3'	4	3'
010	– 4	2	5
111	– 5	5	2

Taδ 2.

Содержание

1756