Глава 10
Таблицы решений и графы переходов

Одним из языков спецификации задач являются таблицы решений (ТР) [105, 111]. Достоинства таблиц решений состоят в компактности первичного описания задачи, а самое главное, в их декларативном характере — независимости результатов вычислений от порядка расположения строк в таблице.

При программировании в большинстве случаев [105] предлагается переходить от декларативной спецификации к процедурной реализации в виде граф-схем алгоритмов, структура которых обычно не содержит последовательно расположенных блоков с одним входом и одним выходом и не обладает тем свойством, что результаты вычислений по ГСА не зависят от порядка расположения этих блоков.

Цель настоящего раздела состоит в рассмотрении методов компилятивного построения программных реализаций, являющихся в максимальной степени непроцедурными, для таблиц решений, описывающих автоматы без памяти и автоматы с памятью.

При этом необходимо отметить, что специфика автоматных таблиц решений позволяет эффективно реализовывать их условными выражениями и булевыми формулами, что в литературе по таблицам решений, однако, не нашло достаточного отражения.

Настоящий раздел можно рассматривать как введение в технику работы с таблицами решений, описывающими автоматы.

10.1. Основные определения

Введем эти определения для одной булевой функции *г.*

Определение 1. ТР называется полностью определенной, если она задает, возможно в неявной форме, значения функции на всех входных наборах.

Определение 2. ТР называется непротиворечивой, если не существует таких входных наборов, на которых функция одновременно принимает противоположные значения.

Определение 3. ТР называется безызбыточной, если она не содержит строк, соответствующих одному и тому же значению функции и «покрывающих» друг друга.

Определение 4. ТР называется негенерирующей, если ни при каких значениях входных переменных она не генерирует [111].

Эффективные методы обнаружения неполноты, противоречивости и наличия генерации в таблицах решений могут быть разработаны на основе алгебры кортежей [242] и поэтому в настоящей работе не описываются. Не рассматриваются также вопросы, связанные с избыточностью, так как она влияет лишь на эффективность описания и не влияет на правильность.

Ниже излагаются вопросы обеспечения полноты и устранения противоречивости, если она обнаружена. Вопрос обеспечения полноты носит принципиальный характер, так как при проектировании таблицы решений Разработчик обычно все внимание уделяет заполнению ее строк и, закончив построение таблицы, не до конца понимает, какое поведение должно быть обеспечено на наборах, не описанных в ней. Кроме того, этот вопрос недостаточно отражен в литературе и поэтому в настоящей главе он рассматривается весьма подробно.

10.2. Реализация непротиворечивых неполных таблиц решений с одним столбцом значений

Пусть в качестве примера задана непротиворечивая неполная таблица решений (табл. 10.1). Она задает значения функции z на $4 + 4 + 2 + 4 = 14$ наборах из 16. Таблица неполна для двух входных наборов: $x_1 = x_2$ - 0, x_3 - 1, $x_4 = 0$ и $x_1 = 1$, $x_2 = 0$, $x_3 = 1$, $x_4 = 0$.

Таблица 10.1

x_1	x_2	x_3	x_4	z
1	-	0	-	1
0	—	0	-	0
—	1	1	0	1
		1	1	0

Рассмотрим, как будет выглядеть эта таблица при различных вариантах ее доопределения. При этом необходимо отметить, что если исходное задание может быть неполно, то программа, его реализующая, всегда определена полностью и на любой входной набор (в том числе и не рассмотренный в исходном задании) как-либо реагирует в зависимости от сознательного или подсознательного доопределения задания.

10.3. Доопределение нулями

Предположим, что в результате доопределения принято решение, что на незаданных наборах входных переменных $z = 0$ (табл. 10.2).

Таблица 10.2

x_1	x_2	x_3	x_4	z
1	—	0	—	1
0	—	0	—	0
—	1	1	0	1
—	—	1	1	0
Иначе				0

Эта таблица описывает автомат без памяти, который реализуется булевой формулой

$$z = x_1 \,\&\, \bar{x}_3 \vee x_2 \,\&\, x_3 \,\&\, \bar{x}_4. \qquad (10.1)$$

10.4. Доопределение единицами

Предположим, что в результате доопределения принято решение, что на незаданных наборах входных переменных $z = 1$ (табл. 10.3).

Как и в предыдущем случае, эта таблица описывает автомат без памяти, который реализуется булевой формулой

$$z = \overline{\bar{x}_1 \,\&\, \bar{x}_3 \vee x_3 \,\&\, x_4} = x_1 \,\&\, \bar{x}_3 \vee x_3 \,\&\, \bar{x}_4. \qquad (10.2)$$

10.5. Безразличное доопределение

Предположим, что в результате доопределения принято решение о том, что на незаданных наборах входных переменных значения функции

Таблица 10.3

x_1	x_2	x_3	x_4	z
1	—	0	0	1
0	—	0	—	0
—	1	1	0	1
—	—	1	1	0
Иначе				1

Таблица 10.4

x_1	x_2	x_3	x_4	z
1	—	0	—	1
0	—	0	—	0
-	1	1	0	1
-	-	1	1	0
0	0	1	0	-
1	0	1	0	-

z безразличны. Так как в этом случае появляется возможность, так выбрать значения z, чтобы получаемая булева формула была минимальна по числу букв по крайней мере в дизъюнктивной нормальной форме, то правило «Иначе» в этом случае применять нецелесообразно и незаданные входные наборы следует записать в таблице в явном виде (табл. 10.4).

Полностью использовать эту информацию можно лишь при применении для минимизации карт Карно, требующих, к сожалению, заполнения всех 2" клеток в ней, что сводит на нет преимущества использования таблиц решений в качестве языка спецификации для автоматов без памяти по сравнению с таблицами истинности. В данном примере из карты Карно следует, что минимальная реализация достигается при замене прочерков единицами. При этом табл. 10.4 может быть заменена табл. 10.3, по которой строится соотношение (10.2).

10.6. Доопределение с сохранением значений выходной переменной

Предположим, что в результате доопределения принято решение, что на незаданных наборах входных переменных предыдущие значения функции z должны сохраняться (табл. 10.5).

Эта таблица описывает работу автомата с памятью. При этом верхняя часть табл. 10.5 (до правила «Иначе») является однотактной (значения выхода зависят только от значений входных воздействий), а часть таблицы, соответствующая правилу «Иначе», — многотактная, так как каждое значение выхода в этом случае зависит не только от значений входных воздействий, но и от предыдущего значения z. (В общем случае в верхней части таблицы решений, в столбце z, кроме нулей и единиц также могут быть указаны и символы г и г).

Рассмотрим различные методы реализации этой таблицы.

При использовании п е р в о г о м е т о д а, записывая формулу по нулям столбца z, получим условие перехода z из 1 в 0:

$$z_0 = \bar{x}_1 \,\&\, \bar{x}_3 \vee x_3 \,\&\, x_4. \qquad (10.3)$$

Таблица 10.5

x_1	x_2	x_3	x_4	z
1	—	0	—	1
0	—	0	—	0
—	1	1	0	1
—	—	1	1	0
Иначе				z

Записывая теперь формулу по единицам столбца г, получим условие перехода z из 0 в 1:

$$z_1 = x_1 \,\&\, \bar{x}_3 \vee x_2 \,\&\, x_3 \,\&\, \bar{x}_4. \tag{10.4}$$

Запишем формулу, определяющую условие, при выполнении которого состояние автомата сохраняется:

$$z_c = \overline{z_0 \vee z_1} = \bar{x}_2 \,\&\, x_3 \,\&\, \bar{x}_4. \tag{10.5}$$

На основе полученных соотношений может быть записана система из трех условных выражений, реализующая табл. 10.5:

$$\begin{aligned} &\textbf{if } (z_0) \quad z = 0; \\ &\textbf{if } (z_1) \quad z = 1; \\ &\textbf{if } (z_c) \quad z = z. \end{aligned} \tag{10.6}$$

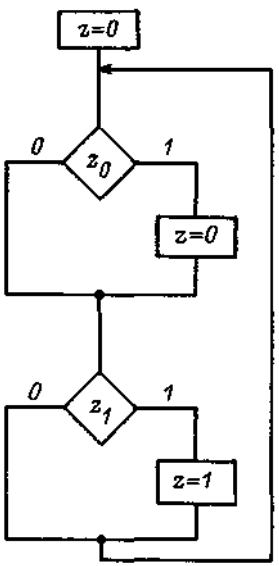

Рис. 10.1

Эта реализация является непроцедурной, так как условные выражения в приведенной системе могут быть записаны в произвольном порядке. Более того, так как в программировании считается, что переменная z обладает памятью, то последнее выражение в системе (10.6) может быть исключено:

$$\begin{aligned} &\textbf{if } (z_0) \quad z = 0; \\ &\textbf{if } (z_1) \quad z = 1. \end{aligned} \tag{10.7}$$

Эти выражения также могут быть записаны в произвольном порядке.

На основе систем (10.7) может быть построено тело структурированной ГСА

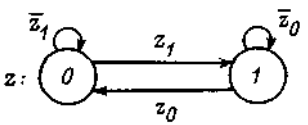

Рис. 10.2

(рис. 10.1), в котором блоки можно менять местами. Эта ГСА реализует табл. 10.5.

Для того чтобы от системы условных выражений перейти к системе булевых формул, используем z_0 в качестве функции сброса двухвходового триггера, а z_1 — в качестве функции его установки. Так как в этом случае ни на одном входном наборе не выполняется соотношение

$$z_0 \,\&\, z_1 = 1,$$

то в качестве двухвходового триггера может быть выбран любой из триггеров следующих типов — *R, S, E* или *JK*.

Выбирая для определенности *S*-триггер, получим реализацию табл. 10.5 системой из трех булевых формул:

$$z_0 = \bar{x}_1 \,\&\, \bar{x}_3 \vee x_3 \,\&\, x_4;$$

$$z_1 = x_1 \,\&\, \bar{x}_3 \vee x_2 \,\&\, x_3 \,\&\, \bar{x}_4; \qquad (10.8)$$

$$z = z_1 \vee \bar{z}_0 \,\&\, z.$$

Выполняя подстановку первых двух формул в третью, определим описание табл. 10.5 с помощью одной булевой формулы:

$$z = x_1 \,\&\, \bar{x}_3 \vee x_2 \,\&\, x_3 \,\&\, \bar{x}_4 \vee \overline{(\bar{x}_1 \,\&\, \bar{x}_3 \vee x_3 \,\&\, x_4)} \,\&\, z =$$

$$= x_1 \,\&\, \bar{x}_3 \vee (x_2 \vee z) \,\&\, x_3 \,\&\, \bar{x}_4.$$

Второй метод построения булевой формулы, реализующей табл. 10.5, состоит в ее записи по графу переходов (рис. 10.2), построенному на основе соотношений (10.3), (10.4).

При этом

$$z = z_1 \,\&\, \bar{z} \vee \bar{z}_0 \,\&\, z =$$

$$= x_1 \,\&\, \bar{x}_3 \,\&\, \bar{z} \vee x_2 \,\&\, x_3 \,\&\, \bar{x}_4 \,\&\, \bar{z} \vee x_1 \,\&\, x_3 \,\&\, z \vee x_3 \,\&\, \bar{x}_4 \,\&\, z =$$

$$= x_1 \,\&\, \bar{x}_3 \vee (x_2 \vee z) \,\&\, x_3 \,\&\, \bar{x}_4.$$

Третий метод построения булевой формулы базируется на использовании соотношения:

$$z = z_1 \vee z_c \,\&\, z = x_1 \,\&\, \bar{x}_3 \vee x_2 \,\&\, x_3 \,\&\, \bar{x}_4 \vee \bar{x}_2 \,\&\, x_3 \,\&\, \bar{x}_4 \,\&\, z =$$

$$= x_1 \,\&\, \bar{x}_3 \vee (x_2 \vee z) \,\&\, x_3 \,\&\, \bar{x}_4.$$

Четвертый метод реализации табл. 10.5 состоит в следующем. Построим по табл. 10.5 процедурную ГСА без обратных связей (рис. 10.3).

По этой граф-схеме может быть записана булева формула, реализующая табл. 10.5:

$$z = x_1 \,\&\, \bar{x}_3 \vee x_2 \,\&\, x_3 \,\&\, \bar{x}_4 \vee \bar{x}_2 \,\&\, x_3 \,\&\, \bar{x}_4 \,\&\, z =$$

$$= x_1 \,\&\, \bar{x}_3 \vee (x_2 \vee z) \,\&\, x_3 \,\&\, \bar{x}_4.$$

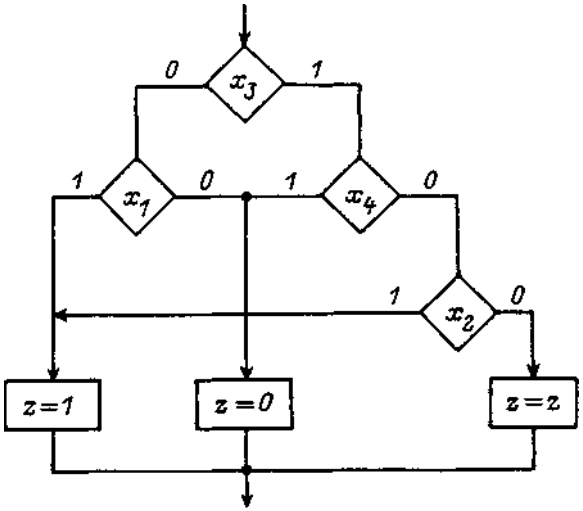

Рис. 10.3

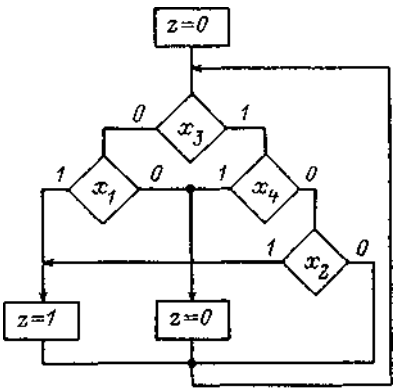

Рис. 10.4

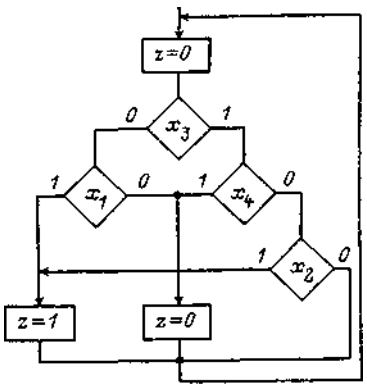

Рис. 10.5

Если в полученной граф-схеме исключить операторную вершину $z = z$ и применить преобразованную ГСА в качестве тела новой граф-схемы (рис. 10.4), то последняя будет реализовывать заданную таблицу.

В этой ГСА в неявном виде учитывается тот факт, что в операторных вершинах осуществляется запоминание значений выходной переменной z, так как в противном случае не удается сохранить ее значения при прохождении пути $(x_3 \bar{x}_4 \bar{x}_2)$, не содержащего вершин этого типа.

Наличие путей в ГСА, не содержащих хотя бы для одной выходной переменной операторных вершин, в которых не устанавливается ни нулевое, ни единичное ее значение (значение рассматриваемой переменной на этих путях сохраняется), является признаком того, что граф-схема реализует автомат с памятью.

Интересным является тот факт, что если расположить стрелку не под операторной вершиной $z = 0$ (рис. 10.4), а над ней (рис. 10.5), то новая ГСА будет реализовывать не автомат с памятью, а комбинационную схему, так как в теле граф-схемы не существует ни одного пути, при прохождении которого выходная переменная может принимать различные значения, а условные вершины с пометкой z отсутствуют. ГСА (рис. 10.5), реализующая табл. 10.2, может быть упрощена, например, как показано на рис. 10.6.

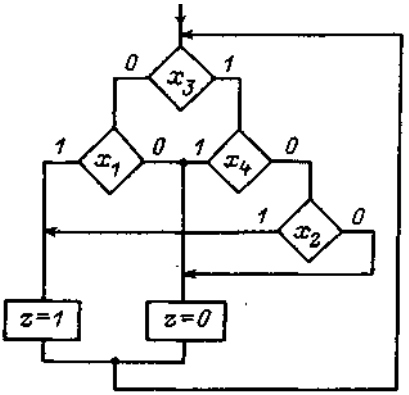

Рис. 10.6

10.7. Доопределение с инвертированием значений выходной переменной

Предположим, что в результате доопределения принято решение, что на незаданных входных наборах предыдущее значение z должно инвертироваться (табл. 10.6).

Таблица 10.6

x_1	x_2	x_3	x_4	z
1	—	0	—	1
0	—	0	—	0
—	1	1	0	1
—	—	1	1	0
Иначе				$\bar{z}$

В этом случае формулы для z_0 и z_1 не отличаются от найденных в разд. 10.6, а условие инвертирования значений имеет вид:

$$z_и = \overline{z_0 \vee z_1} = \bar{x}_1 \,\&\, x_3 \,\&\, \bar{x}_4.$$

Отсюда следует, что табл. 10.6 реализуется системой из трех условных выражений:

```
if (z0)   z = 0;
if (z1)   z = 1;
if (zи)   z = z̄.
```

Результат вычисления по этим выражениям, как и в предыдущем случае, не зависит от порядка их расположения, но в отличие от случая сохранения значений z последнее выражение не может быть исключено.

Табл. 10.6 реализуется булевой формулой следующим образом:

$$z = z_1 \vee z_н \,\&\, \bar{z} = x_1 \,\&\, \bar{x}_3 \vee x_2 \,\&\, x_3 \,\&\, \bar{\bar{x}}_4 \vee x_2 \,\&\, x_3 \,\&\, \bar{\bar{x}}_4 \,\&\, \bar{z} =$$
$$= x_1 \,\&\, \bar{x}_3 \vee (x_2 \vee \bar{z}) \,\&\, x_3 \,\&\, \bar{\bar{x}}_4.$$

Доопределение незаданных наборов инверсией функции редко встречается на практике. Значительно чаще встречается следующий случай.

10.8. Сложное доопределение

Учитывая физические особенности реализуемого процесса, необходимо решить, какую часть неопределенных наборов следует отнести к $z = 0$, какую часть — к $z = 1$, а на каких наборах следует сохранить предыдущее значение z.

Если вновь введенные строки таблицы решений «склеиваются» с имеющимися, то таблица упрощается, так как в левой ее части некоторые нули и единицы могут быть заменены прочерками.

Пусть задана таблица решений (табл. 10.7). Эта таблица непротиворечива, но неполна, так как задана лишь на $16+16+4+4+2+2=44$ входных наборах из 64.

Таблица 10.7

x_1	x_2	x_3	x_4	x_5	x_6	z
0	0	—	—	—	—	0
0	1	—	—	—	—	1
1	1	0	—	—	0	0
1	1	1	—	—	0	1
1	1	—	0	1	1	0
1	1	—	1	0	1	1

Для доопределения незаданных наборов будем строить по табл. 10.7 граф переходов. Для этого введем в рассмотрение две вершины, в первой из которых $z = 0$, а во второй — $z = 1$.

Рассматривая нулевую вершину в качестве исходной, построим по табл. 10.7 фрагмент графа переходов, в котором указаны пути, ведущие из нулевой вершины в нулевую (строки таблицы, помеченные значением $Z = 0$) и из нулевой вершины в первую (строки таблицы, помеченные значением $z = 1$). В этот фрагмент ГП должны быть включены также пути, соответствующие неопределенным строкам таблицы решений.

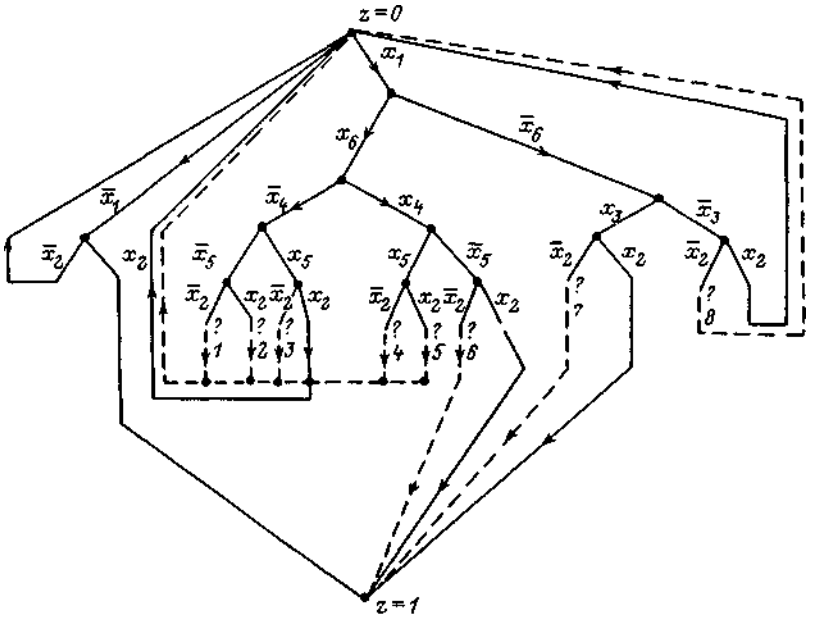

Рис. 10.7

В этот момент частично построенный фрагмент превращается в анкетный язык [179], так как, исходя из знаний о процессе, Разработчик должен указать в явном виде, в какую из вершин, помеченных нулем или единицей, «отправить» дугу, отмеченную символом «?» (рис. 10.7).

Полагая, что при $x_4 = x_5 = 0$ необходимо сохранять значения z, направим дуги *1* и 2 в вершину «0» (z = z). Полагая, что при $x_4 = x_5 = 1$ необходимо, чтобы z был равен нулю, отправим дуги *4* и *5* в вершину «ноль» ($z = 0$). Исходя из знаний о процессе, отправим дуги *3* и *8* в вершину «0» ($z = 0$), а дуги 6 и 7 — в вершину «1» ($z = 1$).

Из изложенного следует, что в результате доопределения в таблицу решений вводятся дополнительные строки, помеченные в столбце значений нулями, единицами и символами z. Так как при этом некоторые строки расширенной таблицы могут отличаться значениями только одной переменной и поэтому склеиваются, то заданная таблица, возможно, упрощается. В рассматриваемом примере для $z = 1$ при $x_1 — 1$ значения переменной x_2 могут быть заменены прочерками.

Запишем по этому фрагменту ГП формулу, определяющую переходы из вершины, помеченной нулем, в вершину, отмеченную единицей:

$$z_1 = \bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, x_6 \,\&\, x_4 \,\&\, \bar{x}_5 \vee x_1 \,\&\, \bar{x}_6 \,\&\, x_3 =$$

$$= \bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_4 \,\&\, \bar{x}_5 \,\&\, x_6). \qquad (10.10)$$

Формула, соответствующая путям из нулевой вершины в нулевую, равна $\bar{z}_1$.

Используя табл. 10.7 и принятые предположения, может быть построен второй фрагмент ГП, содержащий пути из первой вершины в первую и из первой вершины в нулевую. При этом

$$z_0 = \bar{x}_1 \,\&\, \bar{x}_2 \vee x_1 \,\&\, x_6 \,\&\, x_5 \vee x_1 \,\&\, \bar{x}_6 \,\&\, \bar{x}_3 =$$

$$= \bar{x}_1 \,\&\, \bar{x}_2 \vee x_1 \,\&\, (\bar{x}_3 \,\&\, \bar{x}_6 \vee x_5 \,\&\, x_6). \qquad (10.11)$$

Формула, соответствующая путям из первой вершины в первую, равна $\bar{z}_0$. По полученным формулам может быть построен ГП (рис. 10.2).

Из изложенного следует, что граф переходов с большим числом неустойчивых вершин и простыми пометками дуг (его фрагмент приведен на рис. 10.7) эквивалентен ГП с двумя вершинами и сложными пометками дуг.

При этом

$$\begin{aligned} &\textbf{if } (\bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_4 \,\&\, \bar{x}_5 \,\&\, x_6)) \quad z = 1; \\ &\textbf{if } (\bar{x}_1 \,\&\, \bar{x}_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_5 \,\&\, x_6)) \quad z = 0, \end{aligned} \qquad (10.12)$$

а так как $z_c = \overline{\bar{z}_0 \vee \bar{z}_1} = x_1 \,\&\, \bar{x}_4 \,\&\, \bar{x}_5 \,\&\, x_6$, то

$$z = z_1 \vee z_c \,\&\, z =$$

$$= \bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_4 \,\&\, \bar{x}_5 \,\&\, x_6) \vee x_1 \,\&\, \bar{x}_4 \,\&\, \bar{x}_5 \,\&\, x_6 \,\&\, z =$$

$$= \bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee (x_4 \vee z) \,\&\, \bar{x}_5 \,\&\, x_6). \qquad (10.13)$$

Из соотношений 10.10 и 10.11 следует, что исходная таблица решений (табл. 10.7) после доопределения приобретает вид (табл. 10.8):

x_1	x_2	x_3	x_4	x_5	x_6	z
0	0	—	—	—	—	0
0	1	—	—	—	—	1
1	—	0	—	—	0	0
1	—	1	—	—	0	1
1	—	—	—	1	1	0
1	—	—	1	0	1	1
Иначе						z

В этой таблице ее верхняя часть уже определяет не 44 набора входных переменных, как в табл. 10.7, а 16+16+ 8 + 8 + 8 + 4 = 60 наборов из 64.

10.9. Учет дополнительных условий и ограничений

Пример 10.1. Пусть задана таблица решений (табл. 10.9), в которой используются следующие обозначения: $x_1 = 0$ — выбор местного поста управления; $x_1 = 1$ — выбор центрального поста управления — функциональной клавиатуры, входящей в состав управляющей ЦВМ, размещенной на центральном посту; $x_2 = 0$ — объект управления (насос) отключен; $x_2 = 1$ — ОУ включен; $x_3 = 0$ — команда с центрального поста на отключение ОУ; $x_3 = 1$ — команда с центрального поста на включение ОУ; $z = 0$ — обобщенная команда на отключение ОУ; $z = 1$ — обобщенная команда на включение ОУ.

Таблица 10.9

x_1	x_2	x_3	z
0	0	—	0
0	1	—	1
1	—	0	0
1	—	1	1

Кроме того, необходимо учесть условие, состоящее в том, что переменная x_3 находится в битовой ячейке памяти, на которую, в частности, воздействует счетный триггер, управляемый от кнопки, расположенной на функциональной клавиатуре.

Задано также ограничение: только при переключении значения переменной x_1 (без нажатия кнопки на функциональной клавиатуре) значение выходной переменной г, установленное с местного поста, не должно изменяться.

Табл. 10.9 полна и непротиворечива и реализуется булевой формулой:

$$z = \bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, x_3. \qquad (10.14)$$

Так как эта булева формула реализует автомат без памяти, то ей соответствует полный граф переходов автомата Мура, содержащий переходы между его любыми двумя вершинами, причем дуги, входящие в каждую вершину, имеют одинаковую пометку, совпадающую с пометкой петли в рассматриваемой вершине.

В табл. 10.9 — четыре строки, и поэтому построим ГП, содержащий четыре вершины, каждая из которых определяет одно значение, заданное в столбце z таблицы. Из каждой вершины исходят четыре дуги, одна из которых является петлей.

Все дуги, входящие в вершину, соответствующую выбранному значению z, помечаются условием, записанным в строке таблицы, определяющей это значение выходной переменной (рис. 10.8). Из рассмотрения этого графа переходов следует, что он не удовлетворяет указанному в

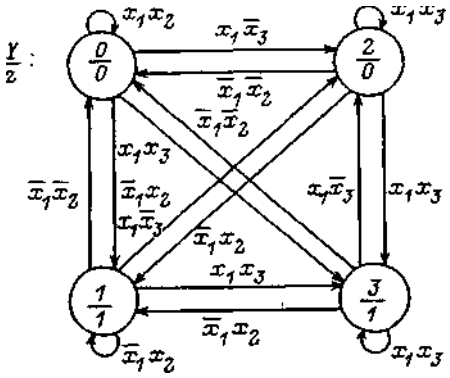

Рис. 10.8

постановке задачи ограничению, так как, например, если в вершине «1» с помощью кнопки функциональной клавиатуры переменной x_3 присвоить значение «ноль», то только при изменении значения x_1 выходная переменная z изменит значение с 1 на 0.

Для выполнения этого ограничения в графе переходов должны быть устранены диагональные переходы. Переходы «3—0» и «2—1» при исправном объекте управления ($x_2 = 1$ при включенном объекте и $x_2 = 0$ при отключенном объекте) невозможны. Для исключения переходов «1—2» и «0—3» введем дополнительное соотношение

$$x_3 = z, \tag{10.15}$$

располагаемое в программе после соотношения (10.14).

Соотношение (10.15) в вершине «0», в которой $z = 0$, исключит переход «0—3», а в вершине «1», в которой $z = 1$, исключит переход «1—2».

Следовательно, доказано, что таблица решений (табл. 10.9) с учетом приведенных условия и ограничения реализуется следующей системой булевых формул:

$$\begin{gathered} z = \bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, x_3; \\ x_3 = z \end{gathered} \tag{10.16}$$

или системой условных выражений вида:

$$\begin{gathered} \text{if } (\bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, x_3) \quad \{z = 1;\ x_3 = 1;\} \\ \text{if } (\bar{x}_1 \,\&\, \bar{x}_2 \vee x_1 \,\&\, \bar{x}_3) \quad \{z = 0;\ x_3 = 0;\} \end{gathered} \tag{10.17}$$

только при выполнении еще одного условия, не указанного в формулировке задачи и состоящего в том, что объект управления должен быть исправным.

Из (10.17) следует, что если один из условных операторов выполняется, то для реализации второго из них при x_1 - 1 значение переменной x_3 должно быть проинвертировано извне (кнопкой).

Из изложенного следует, что для исправного объекта управления задача может быть решена с помощью графа переходов (рис. 10.9), полученного из ГП (рис. 10.8) исключением диагональных дуг. В новом графе пометки петель по умолчанию обеспечивают полноту в каждой вершине. Из сравнения таблицы решений (табл. 10.9) и графа переходов (рис. 10.9) следует, что последний обладает существенно более высокими

изобразительными возможностями [156], так как включает не только условия, задаваемые таблицей, но и условие и ограничение, задаваемые текстом.

Анализ графа переходов (рис. 10.9) показывает, что он обладает двумя недостатками. Первый из них состоит в том, что переходы «2—0» и «3—1» зависят от переменной x_2 и происходят при исправном объекте управления. Для устранения этого недостатка откорректируем предыдущий и получим новый ГП (рис. 10.10). Отметим также, что выполненная корректировка не имеет изоморфного отражения в исходной таблице решений, которая формально была полна и непротиворечива. По новому ГП может быть построена новая таблица решений, которая совершенно не будет похожа на исходную таблицу.

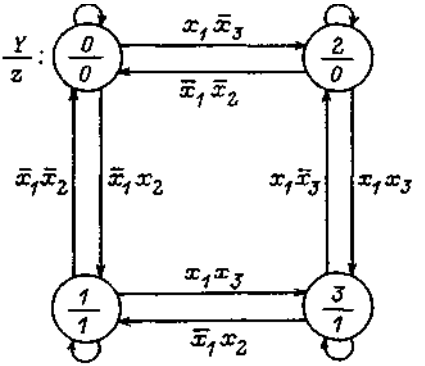

Рис. 10.9

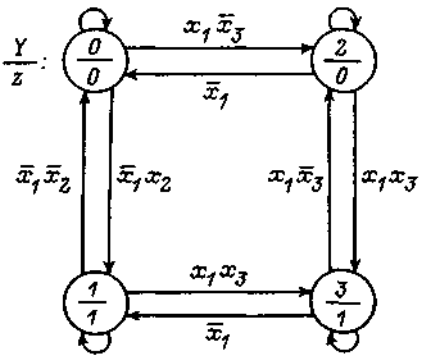

Рис. 10.10

В этом графе сохраняется второй недостаток ГП (рис. 10.9), состоящий в том, что в ряде случаев для переключения объекта управления кнопку на функциональной клавиатуре после появления x_1, = = 1 приходится нажимать дважды. Действительно, пусть, например, в то время как автомат находился в вершине «0», с помощью кнопки переменной x_3 была присвоена единица. Тогда при x_1 = 1 для обеспечения переходов «0—2—3» кнопка должна быть нажата дважды.

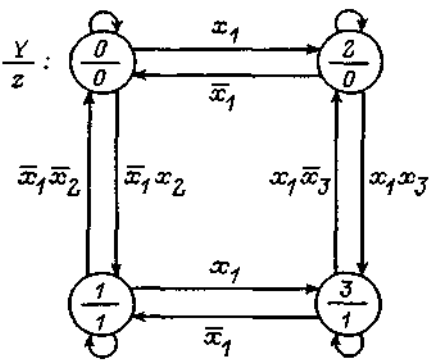

Рис. 10.11

В рамках одного ГП автомата Мура этот недостаток не устранить. Он может быть исключен, если, с одной стороны, в графе переходов (рис. 10.10) устранить первые нажатия кнопки (рис. 10.11), а с другой — дополнительно ввести соотношение (10.15). Оно исключает, например, возможность перехода «1—3—2» при x_1 - 1 без нажатия кнопки. Таким образом, соотношение (10.15) при x_1 - 1 заменяет первое нажатие кнопки.

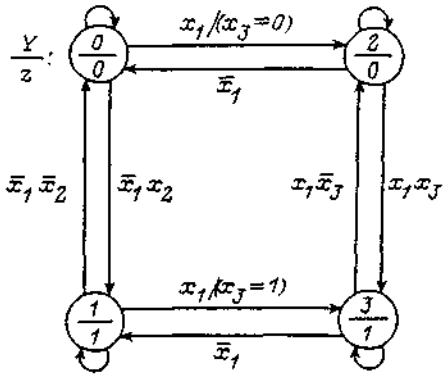

Рис. 10.12

Из изложенного следует, что изобразительных возможностей только ГП автомата Мура в данном случае не хватило для детального отображения всех особенностей решаемой задачи, не говоря уже о возможностях использования для этих целей таблицы решений.

Проведем расширение модели, перейдя к модели смешанного автомата — автомату Мура—Мили [16]. Однако этот переход задачу не решает. Поэтому выполним дальнейшее расширение модели, для чего применим С-автомат с флагом (рис. 10.12), который решает поставленную задачу в полном объеме. Автоматы с флагами характеризуются тем, что в одном графе переходов одна и та же переменная может одновременно использоваться как в качестве входной, так и выходной переменной. На рис. 10.12 на переходах «2—3» и «3—2» переменная x_3 — входная, а на переходах «0—2» и «1—3» она является выходной. Реализуем этот ГП программой на языке СИ, имеющей в значительной степени декларативный характер и обладающей структурой, изоморфной графу (рис. 10.12):

```
        switch (Y) {
case 0: z = 0;
        if (x̄1 & x2)     Y = 1;
        if (x1)        {x3 = 0; Y = 2; }
        break;
case 1: z = 1;
        if (x̄1 & x̄2)     Y = 0;
        if (x1)        {x3 = 1; Y = 3; }
        break;
case 2: z = 0;
        if (x̄1)          Y = 0;
        if (x1 & x3)     Y = 3;
        break;
case 3: z = 1;
        if (x̄1)          Y = 1;
        if (x1 & x̄3)     Y = 2;
        break; }
```

Из приведенной программы следует, что граф переходов на рис. 10.12 является не «картинкой», а алгоритмической моделью для построения изоморфной программы.

Пример 10.2. Пусть вместо табл. 10.9 задана более сложная таблица решений (табл. 10.8), а также условие и ограничение, введенные в примере 70.7.

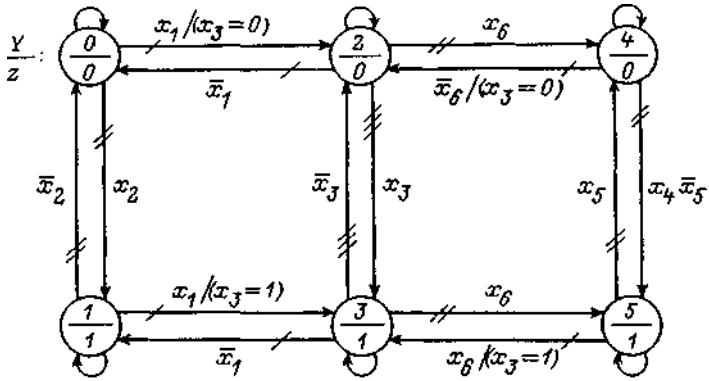

Рис. 10.13

Эта таблица соответствует управлению объектом с двух постов — местного ($x_1 = 0$), при выборе которого обеспечивается функционирование автомата по сигналу x_2, и центрального ($x_1 — 1$). При $x_1 — 1$ возможны два вида управления — с функциональной клавиатуры (x_6 - 0) от кнопки, воздействующей через счетный триггер на переменную x_3, и автоматический ($x_6 = 1$) — от сигнализаторов x_4 («пуск») и x_5 («стоп»).

На основе результатов, полученных в примере *10.1* и разд. 10.8, можно утверждать, что поставленная задача при исправном объекте управления реализуется системой булевых формул:

$$z = \bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee (x_4 \vee z) \,\&\, \bar{x}_5 \,\&\, x_6);$$

$$x_3 = z,$$

или системой условных выражений:

if $(\bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_4 \,\&\, \bar{x}_5 \,\&\, x_6))$ $\{x_3 = 1;\ z = 1;\}$

if $(\bar{x}_1 \,\&\, \bar{x}_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_5 \,\&\, x_6))$ $\{x_3 = 0;\ z = 0;\}$.

При этом отметим, что в отличие от примера *10.1* (система 10.16) первая формула системы (10.18) описывает не комбинационную схему, а автомат с двумя устойчивыми состояниями.

Проводя анализ, аналогичный с выполненным в примере *10.1,* можно показать, что эта задача более корректно и понятно решается с помощью графа переходов С-автомата с флагом (рис. 10.13). Этот ГП содержит 6 вершин (по числу строк в табл. 10.8, расположенных до строки «Иначе»), в каждой из которых формируется одно из значений столбца z этой таблицы. Для упрощения изображения графа пометки петель, обеспечивающие полноту вершин, умалчиваются, а непротиворечивость дуг, исходящих из каждой вершины, достигается не ортогонализацией пометок на дугах, как это выполнялось ранее, а указанием приоритетов дуг, изображаемых штрихами, число которых тем меньше, чем выше приоритет перехода.

10.10. Реализация противоречивых таблиц решений

Пусть условия работы заданы таблицей решений, отличающейся от табл. 10.8 тем, что в строке, непосредственно предшествующей строке «Иначе», вместо нуля имеется прочерк.

Существуют два подхода устранения противоречия: алгоритмический и программный. При использовании первого из них таблица должна быть откорректирована, например построением табл. 10.8. При втором подходе теряется непроцедурный характер программы, так как булевы формулы в условных выражениях не ортогональны и выражение со значением, которому отдается приоритет, должно быть расположено в тексте программы ниже. Для рассматриваемого примера

$$\begin{aligned} &\textbf{if } (\bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_4 \,\&\, x_6)) \quad z = 1; \\ &\textbf{if } (\bar{x}_1 \,\&\, \bar{x}_2 \vee x_1 \,\&\, (\bar{x}_3 \,\&\, \bar{x}_6 \vee x_5 \,\&\, x_6)) \quad z = 0. \end{aligned} \tag{10.20}$$

Для получения булевой формулы, описывающей противоречивую таблицу решений, применим формулу двухвходового триггера, тип которого в этом случае существен. Для рассматриваемого примера должен использоваться *R*-триггер. При этом первоначально строится система из трех булевых формул, порядок расположения первых двух из которых несущественен:

$$\begin{aligned} z_0 &= \bar{x}_1 \,\&\, \bar{x}_2 \vee x_1 \,\&\, (\bar{x}_3 \,\&\, \bar{x}_6 \vee x_5 \,\&\, x_6); \\ z_1 &= (\bar{x}_1 \,\&\, x_2 \vee x_1 \,\&\, (x_3 \,\&\, \bar{x}_6 \vee x_4 \,\&\, x_6); \\ z &= \bar{z}_0 \,\&\, (z_1 \vee z). \end{aligned} \tag{10.21}$$

Выполняя подстановку первых двух формул в третью, получим соотношение (10.13).

Второй из рассмотренных подходов применять нецелесообразно, так как при этом либо спецификация остается противоречивой, либо, если противоречие устранено, спецификация не изоморфна реализации (10.20).

При этом отметим, что непроцедурность (10.20) позволяет упростить реализацию по сравнению с соотношением (10.12), в котором выражения можно менять местами.

Таким образом, в настоящем разделе показано, что автоматные таблицы решений могут быть весьма эффективно программно реализованы. Однако при их использовании в качестве языка спецификации для автоматов с памятью они обладают существенно меньшими изобразительными возможностями по сравнению с графами переходов и системами из них (последние заменяют весьма сложные «сцепленные» таблицы решений [111]), которые, по мнению автора, и должны применяться для описания этого класса задач вместо таблиц решений, так как значительно более наглядно (особенно при применении в случае необходимости совместно с ними соответствующего графа достижимых маркировок) отражают динамику переходов автоматов и заданные условия и ограничения. Кроме того, построение двоичных таблиц решений связано с описа-

нием отдельных компонент автомата, в то время как ГП позволяют мыслить об автомате в целом, используя понятие «состояние» и многозначное кодирование. Из изложенного следует, что таблицы решений целесообразно применять в качестве спецификации в традиционной для них области — для реализации автоматов без памяти, а также, например, при формализации сложных условий, помечающих дуги ГП. Как показано в разд. 10.9, таблицы решений, кроме того, могут использоваться на начальной стадии построения ГП для определения числа вершин, некоторых дуг и их пометок.

В заключение раздела отметим, что в [243, 244] предложены ассоциативные программируемые логические устройства, интерпретирующие таблицы решений.