ПРОБЛЕМЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

Том XXIII 1987 Вып. 1

УДК 621.391.1–503.5

БИНАРНЫЕ ПРОГРАММЫ И ИХ РЕАЛИЗАЦИЯ АСИНХРОННЫМИ АВТОМАТАМИ

Сагалович Ю. Л., Шалыто А. А.

Рассматривается кодирование состояний асинхронных автоматов, предназначенных для вычисления значений систем булевых функций посредством бинарных программ. Показано, что для достаточно широкого класса систем булевых функций противогоночное кодирование состояний оказывается безызбыточным.

§ 1. Введение

В последнее время в связи с достижениями в области микроэлектроники все большее внимание уделяется вопросам программной реализации систем булевых функций (СБФ).

Бинарные программы [1] как для одной булевой функции, так и для СБФ можно реализовать в виде дискретных автоматов. Это особенно важно, например, тогда, когда набор значений переменных известен не сразу, а значения переменных появляются последовательно одно за другим.

В случае реализации бинарной программы (БП) для СБФ асинхронным автоматом возникает проблема противогоночного кодирования его состояний. Настоящая статья посвящена первоначальному рассмотрению этого вопроса. В § 2 строится граф и таблица переходов автомата, который реализует БП для заданной СБФ. § 3 целиком посвящен противогоночному кодированию состояний автомата и опирается на терминологию и систему понятий, введенных в § 2.

§ 2. Граф и автомат для реализации бинарной программы

Пусть дана такая система m булевых функций

(1) $$y_j=f_j(x_0, x_1, \ldots, x_{l-1}), \quad j=1, 2, \ldots, m,$$

что каково бы ни было $i=1, \ldots, l$ найдется хотя бы одна функция системы, зависящая существенно от x_{i-1}.

Бинарной программой для системы (1), *следуя* [1], *будем называть программу последовательного вычисления коэффициентов* $f_i(\sigma_0, \sigma_1, \ldots$ $\ldots, \sigma_{i-1}, x_i, \ldots, x_{l-1})$ *при конъюнкциях* $x_0^{\sigma_0}x_1^{\sigma_1}\ldots x_{i-1}^{\sigma_{i-1}}$, $i=1, 2, \ldots, l$ *в разложении*

$$f_j=\bigvee_{\substack{\text{по всем}\\ \text{наборам } \sigma}} x_0^{\sigma_0}x_1^{\sigma_1}\ldots x_{i-1}^{\sigma_{i-1}} f_j(\sigma_0, \sigma_1, \ldots, \sigma_{i-1}, x_i, \ldots, x_{l-1}),$$

$$j=1, 2, \ldots, m.$$

системы (1) *на произвольном последовательно формирующемся наборе* $\sigma=(\sigma_0, \sigma_1, \ldots, \sigma_{l-1})$ *значений переменных* [1].

[1] Вообще бинарной программой называется программа, состоящая из условных переходов.

Бинарную программу для системы (1) можно представить в виде графа с одной начальной и не более, чем 2^m конечными вершинами. Этот граф строится следующим образом [1, 2].

Возьмем дихотомическое дерево, т. е. дерево, из каждой вершины которого, кроме конечных, выходит в точности по два ребра, и в каждую вершину которого, кроме начальной, входит одно ребро. Одно из выходящих из каждой вершины ребер помечено символом 0, а другое — символом 1. Вершины дерева отнесены к $l+1$ ярусам, имеющим номера $0, 1, \ldots, l$. Каждая вершина принадлежит в точности к одному ярусу. В каждую вершину i-го яруса ведет единственный путь из начальной вершины дерева, состоящий из $i-1$ ребер. Вершины i-го яруса можно, таким образом, перенумеровать двоичными числами $\sigma_0\sigma_1\ldots\sigma_{i-1}$, где $\sigma_j=0, 1$, $j=0, 1, \ldots, i-1$, каждое из которых в точности описывает путь из начальной вершины в данную. Каждая вершина i-го яруса является одновременно начальной вершиной поддерева с $l+1-i$ ярусами. Поддеревья с начальными вершинами, расположенными в одном ярусе, не пересекаются. В вершине с номером $\sigma_0\sigma_1\ldots\sigma_{i-1}$ поместим систему коэффициентов разложения системы (1) по переменным $x_0, x_1, \ldots, x_{i-1}$, при одной и той же конъюнкции $x_0^{\sigma_0}x_1^{\sigma_1}\ldots x_{i-1}^{\sigma_{i-1}}$. Дерево с размещенными системами коэффициентов назовем заполненным. В силу предположения о существенной зависимости системы (1) от переменных $x_0, x_1, \ldots, x_{l-1}$ две системы коэффициентов, помещенные в вершины первого яруса, не совпадают. Если уже рассмотрены вершины $(i-1)$-го яруса, перейдем к вершинам i-го яруса $(i=2, \ldots, l-1)$. Если совпадают системы коэффициентов в двух вершинах с номерами

(2) $$\sigma_0\sigma_1\ldots\sigma_{i-2}\sigma_{i-1} \text{ и } \sigma_0\sigma_1\ldots\sigma_{i-2}\bar{\sigma}_{i-1}$$

(это означает, что система коэффициентов в вершине с номером $\sigma_0\sigma_1\ldots\sigma_{i-2}$ не зависит от x_{i-1}), то вершина $\sigma_0\sigma_1\ldots\sigma_{i-2}$ и две вершины (2) объединяются и помещаются в i-м ярусе. В новую вершину будет входить ребро, входившее в вершину $\sigma_0\sigma_1\ldots\sigma_{i-2}$. После объединения вершин вида (2) в i-м ярусе все еще могут оказаться вершины с одинаковыми системами коэффициентов. Объединим такие вершины, при этом входившие в них ребра будут входить в одну объединенную вершину. Вместе с объединением вершин объединяются и заполненные поддеревья, для которых объединяемые вершины являются начальными, так как заполненные поддеревья с одним и тем же числом ярусов и с одинаковыми системами коэффициентов в начальных вершинах полностью совпадают.

Так как каждый путь от начальной вершины к конечной изображает набор значений переменных, на котором система (1) принимает систему значений, сопоставленную данной конечной вершине, то только у конечных вершин и следует оставить системы коэффициентов; из всех других вершин их следует удалить.

Обозначим вершины i-го яруса одним и тем же символом x_i. Этим завершается построение графа. При заданной нумерации переменных x_i построенный таким способом граф единственный, и его конфигурация может варьироваться лишь в зависимости от нумерации переменных. По системам коэффициентов последнего яруса и графу система (1) воспроизводится однозначно [2].

---

[2] Для построения графа БП СБФ можно применить метод, предложенный В. И. Рубиновым и А. А. Шалыто. Он является развитием канонического метода Л. Ш. Блоха [3]. Метод базируется на представлении СБФ с помощью таблицы истинности, а его суть состоит в том, что строки значений функций в этой таблице обозначаются одинаковыми номерами, если строки совпадают, в то время как различные строки обозначаются различными номерами. Затем построение графа выполняется на основе канонического метода [3].

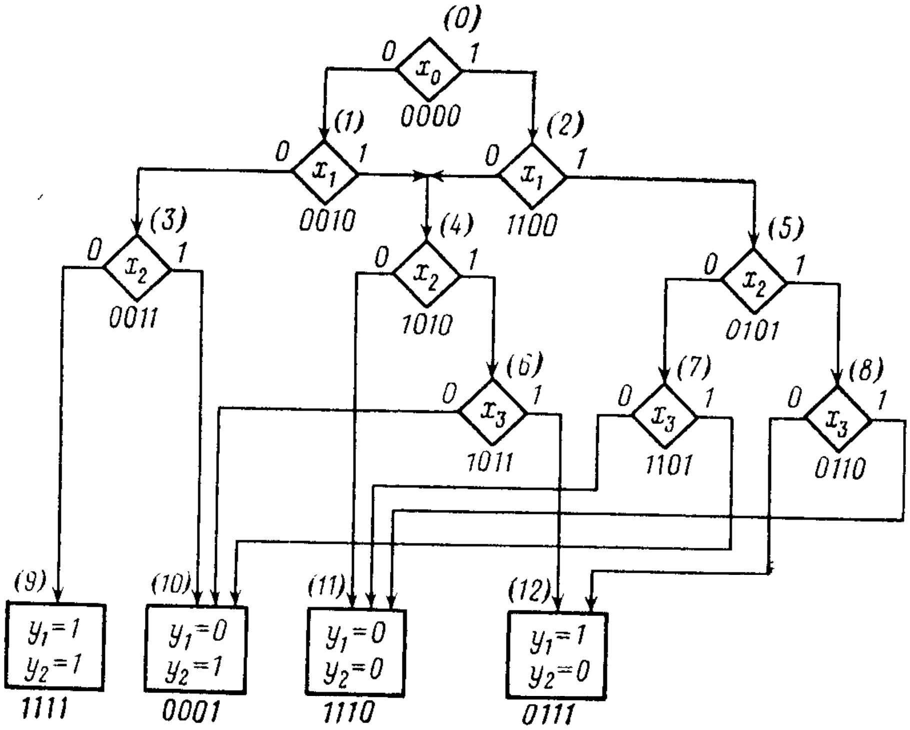

Рис. 1

В качестве примера на рис. 1 изображен граф БП для СБФ

$$y_1=\bar{x}_0(\bar{x}_1\bar{x}_2\vee x_1x_2x_3)\vee x_0x_2(x_1\oplus x_3),\qquad (3)$$
$$y_2=\bar{x}_0(\bar{x}_1\vee x_2\bar{x}_3)\vee x_0x_1\bar{x}_2x_3\vee\bar{x}_1x_2\bar{x}_3.$$

Если вершины графа БП занумеровать произвольным образом, а нахождение системы функций по этому графу вести в той же последовательности, в которой определяются коэффициенты разложения по переменным x_i, $i=0, 1, \ldots, l-1$, то реализацию БП можно осуществить в виде дискретного автомата. Состояния автомата суть номера вершин графа; *состоянием входа являются номера переменных $x_0, x_1, \ldots, x_{l-1}$ и их значения.* Переход автомата из состояния с номером a i-го яруса в состояние с номером b (обязательно яруса с бо́льшим номером) при заданном значении σ_i переменной x_i интерпретирует нахождение системы коэффициентов, которая отвечает вершине с номером b, если уже вычислена система коэффициентов, которая отвечает вершине с номером a. Выходом автомата может быть номер состояния, а в заключительном состоянии — набор значений функций системы. Если на выходе появится номер одной из конечных вершин, то значение функций системы (1) на данном наборе значений переменных вычислено. Одной из удобных автоматных реализаций БП является микропрограммный автомат Уилкса, допускающий параллельную выдачу результатов вычислений.

При реализации БП посредством синхронного автомата, время определения значений системы (1) при некоторых условиях может оказаться слишком большим. Поэтому для повышения быстродействия обращение к асинхронному режиму работы автомата будет единственно возможным.

§ 3. Противогоночное кодирование состояний автомата, реализующего бинарную программу

Асинхронный режим автомата, реализующего БП, требует противогоночного кодирования состояний.

При противогоночном кодировании: 1) все состояния автомата должны быть закодированы различными двоичными кодовыми векторами (будем обозначать их буквами с индексами); 2) любые два перехода $S_1\to S_2$ и $S_3\to S_4$, происходящие при одном и том же состоянии входов, не должны

Состояние	x_0		x_1		x_2		x_3	
	0	1	0	1	0	1	0	1
0	1	2	—	—	—	—	—	—
1	—	—	3	4	—	—	—	—
2	—	—	5	6	—	—	—	—
3	—	—	—	—	7	8	—	—
4	—	—	—	—	9	10	—	—
5	—	—	—	—	11	12	—	—
6	—	—	—	—	13	14	—	—
7	—	—	—	—	—	—	15	16
8	—	—	—	—	—	—	17	18
9	—	—	—	—	—	—	19	20
10	—	—	—	—	—	—	21	22
11	—	—	—	—	—	—	23	24
12	—	—	—	—	—	—	25	26
13	—	—	—	—	—	—	27	28
14	—	—	—	—	—	—	29	30
15	—	—	—	—	—	—	—	—
.								
.								
.								
30	—	—	—	—	—	—	—	—

иметь общих промежуточных состояний, которые могут возникнуть из-за состязаний (гонок) элементов памяти автомата.

Когда в переходах $S_1 \to S_2$ и $S_3 \to S_4$ нет общих промежуточных состояний, то говорят, что переходы разделены. Переходы $S_1 \to S_2$ и $S_3 \to S_4$ разделены тогда и только тогда, когда в векторах состояний S_1, S_2, S_3, S_4 найдется хотя бы одна такая компонента, в которой S_1 и S_2 имеют 1, а S_3 и S_4 — 0 или наоборот.

Графы, реализующие БП, имеют весьма специфические свойства, которые в значительной степени определяют выбор метода противогоночного кодирования.

Рассмотрим такую систему функций (1), в которой все системы коэффициентов ее разложения по переменным различны. Это означает, что условий для объединения вершин и ребер графа нет, и поэтому графом БП для такой системы функций будет полное дерево, имеющее в каждом i-м ярусе 2^i вершин (здесь $m \geqslant l$).

Если нумерация вершин графа проведена в ярусе слева направо, начиная с 0-го яруса, то вершины i-го яруса будут иметь номера от 2^i-1 до $2^{i+1}-2$.

Пусть состояния автомата, реализующего БП с таким графом, расположены в таблице переходов в том же порядке, в каком пронумерованы отвечающие им вершины графа. (Для удобства вместо S_i в таблице используется индекс i).

Предположим, что $2l$ столбцов таблицы расположены в порядке возрастания номеров переменных x_i и для определенности в каждой паре столбцов, соответствующих одной переменной, столбец $x_i=0$ предшествует столбцу $x_i=1$.

Тогда таблица переходов имеет стандартный вид. Она имеет $l+1$ горизонтальных полос, а i-я полоса содержит 2^i строк ($i=0, 1, \ldots, l$). На пересечении столбцов $x_i=0$ и $x_i=1$ с i-й полосой помещены номера конечных состояний переходов. В остальных клетках таблицы переходы не определены. Конечные состояния переходов i-й полосы являются начальными состояниями переходов $(i+1)$-й полосы. По диагонали таблицы слева направо и сверху вниз расположены «ящики», не имеющие общих элементов. Ширина «ящика» — два столбца, высота — 2^i строк. В последней l-й полосе переходы не определены. Поэтому эту полосу можно опустить.

Таблицу в этой полосе можно доопределить, предполагая, например, что состояния переходят в самих себя.

Табл. 1 является таблицей переходов для автомата с 31 состоянием, реализующего БП системы функций, зависящих от четырех переменных.

Назовем путем последовательность состояний, которые переходят друг в друга под действием смены состояний входов. Например, путем будет последовательность 0, 1, 3, 8, 17 или 0, 2, 6, 14, 30.

Для удобства вместо термина «разделение переходов» будем пользоваться выражением «разделение строк», имея в виду, что в двух разных строках расположены конечные состояния переходов.

Справедливо следующее утверждение. *Все* $2^{l+1}-1$ *состояний автомата, реализующего БП системы функций, все системы коэффициентов которой в ее разложении по переменным* $x_0, x_1, \ldots, x_{l-1}$ *различны, можно закодировать безызбыточным противогоночным кодом, т. е. кодом длины* $k=l+1$.

Доказательство этого утверждения конструктивно, т. е. ведется фактическим противогоночным кодированием состояний. Разделения строк нулевой полосы таблицы не требуется.

Для разделения строк первой полосы требуется один разряд. Поэтому в первом разряде векторов всех состояний всех путей, начинающихся состоянием 1, поместим символ $\bar{\sigma}_1$, а всех состояний всех путей, начинающихся состоянием 2, символ σ_1.

Для разделения четырех строк второй полосы требуется еще только один разряд кодового вектора, так как две пары начальных состояний этой полосы уже были разделены как конечные состояния переходов с различными начальными состояниями предыдущей полосы. Во втором разряде векторов состояний 3 и 5 поместим символ $\bar{\sigma}_2$, а состояний 4 и 6 — символ σ_2. Эти же символы поместим во втором разряде векторов тех состояний, которые принадлежат путям с соответствующим начальным состоянием. Так будем поступать с каждой полосой. Каждая следующая полоса (в которой есть «ящик») для разделения еще неразделенных строк требует одной новой переменной. Имеется $l-1$ таких полос. Поэтому для разделения всех переходов, происходящих при одинаковых состояниях входов, требуется $l-1$ разрядов. Последовательное разделение переходов означает и частичное различение состояний: все состояния, принадлежащие пути, который начинается в некотором состоянии i-й полосы, различены от всех состояний, принадлежащих путям, которые начинаются в другом начальном состоянии i-й полосы. После разделения переходов получим, что все начальные состояния i-й полосы таблицы закодированы различными кодовыми векторами длины i. Однако по построению вместе с каждым начальным состоянием i-й полосы одинаковый набор символов в первых i разрядах имеют еще $2^{l+1-i}-2$ состояний. Это значит, что каждый набор i символов встречается в точности $2^{l+1-i}-1$ раз, и поэтому оставшихся $l+1-i$ разрядов достаточно для различения состояний с этими одинаковыми наборами.

Для завершения кодирования последнее указание является исчерпывающим. Недостающие до $l+1$ символы дописывают с учетом единственного требования, состоящего в том, что все кодовые комбинации должны быть различными. Это полностью выполнимо ввиду приведенных выше количественных соотношений. Дописывание символов следует проводить в порядке убывания номера полосы таблицы переходов, так как в процессе разделения переходов состояния полосы с большим номером оказались закодированными более длинными кодовыми комбинациями.

После того, как различными кодовыми векторами закодированы все состояния с номерами $1, 2, \ldots, 2^{l+1}-2$, остается еще два неиспользованных набора длины $l+1$. Одним из них следует закодировать состояние с номером 0. Этим кодирование заканчивается. В результате все переходы, происходящие при одном состоянии входов, разделены и все состояния различны. Утверждение доказано.

Для примера выполним кодирование таблицы переходов (табл. 1).

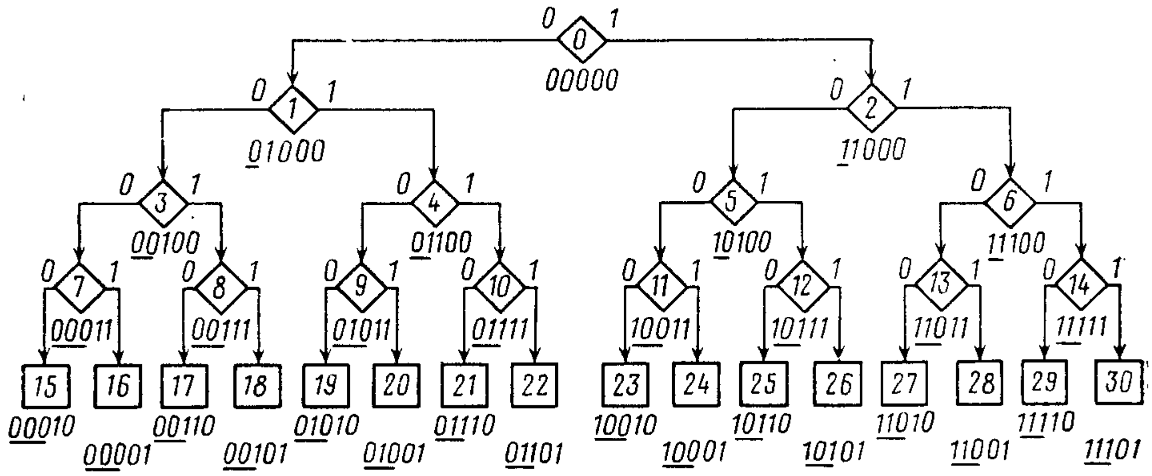

Рис. 2

После разделения переходов состояния закодированы следующим образом (эти части кодовых векторов подчеркнуты).

0. 0 0000;	8. 00111	16. 000 01	24. 100 01
1. 0 1000	9. 01011	17. 001 10	25. 101 10
2. 1 1000	10. 01111	18. 001 01	26. 101 01
3. 00 100	11. 10011	19. 010 10	27. 110 10
4. 01 100	12. 10111	20. 010 01	28. 110 01
5. 10 100	13. 11011	21. 011 10	29. 111 10
6. 11 100	14. 11111	22. 011 01	30. 111 01
7. 00 011	15. 00010	23. 100 10	

На рис. 2 приведен граф, соответствующий табл. 1, с изображением предпринятого кодирования состояний.

Из рассмотрения подчеркнутых частей векторов следует, что, например, состояния 7, 15, 16 имеют одну и ту же комбинацию 000 первых трех символов. Двух символов достаточно для различения этих трех состояний, так же, как и состояний 8, 17, 18. Состояния 3, 7, 8, 15—18 имеют одну и ту же комбинацию 00 первых двух символов. Трех символов достаточно для различения этих семи состояний. Однако, состояния 7, 16, 15, равно как 8, 17, 18, уже были различены с помощью трех символов. Для седьмого состояния 3 берем любую, еще не использованную комбинацию трех символов, например 100.

Точно так же поступаем со всеми путями, имеющими начальное состояние 4. Далее, из табл. 1 следует, что все состояния 1, 3, 4, 7—10, 15—22 имеют один и тот же первый символ. Но все состояния 3, 7, 8, 15—18 были отделены от состояний 4, 9, 10, 19—22 вторым символом еще при разделении переходов, а также внутри каждой из этих групп состояний по предыдущему построению.

Таким образом, 14 состояний уже различены с помощью 14 различных комбинаций из четырех символов. Остаются еще две комбинации, одну из которых — 1000 приписываем к первому символу 0 состояния 1. Точно так же поступаем со всеми путями, имеющими начальное состояние 2. После этого все состояния, принадлежащие путям, которые начинаются состоянием 1, равно как и состоянием 2, различены между собой и между состояниями, принадлежащими путям с различными начальными состояниями. Таким образом, тридцать состояний различены с помощью тридцати различных комбинаций из пяти символов, т. е. минимально возможным числом символов.

Таблица 2

Состояние	x_0		x_1		x_2		x_3	
	0	1	0	1	0	1	0	1
0	1	2	—	—	—	—	—	—
1	—	—	3	4	—	—	—	—
2	—	—	4	5	—	—	—	—
3	—	—	—	—	9	10	—	—
4	—	—	—	—	11	6	—	—
5	—	—	—	—	7	8	—	—
6	—	—	—	—	—	—	10	12
7	—	—	—	—	—	—	11	12
8	—	—	—	—	—	—	12	11
9	—	—	—	—	—	—	—	—
10	—	—	—	—	—	—	—	—
11	—	—	—	—	—	—	—	—
12	—	—	—	—	—	—	—	—

Отметим, что при переходах обязательно будут появляться промежуточные состояния, в точности равные другим состояниям автомата. Например, при переходе из состояния 3 в состояние 7 могут появиться все остальные шесть состояний с двумя нулями в первых двух разрядах — состояния 0, 8, 15, 16, 17, 18. Однако по построению ни одно из этих состояний не появится ни при одном из переходов с другим начальным и другим конечным состояниями при том же состоянии входов.

Заметим, что и табл. 1 и данный код годятся для любой системы булевых функций, зависящих от четырех переменных, все системы коэффициентов которых в разложении по этим переменным различны. Разница между системами функций будет выявляться лишь в тех наборах констант, которые соответствуют состояниям с номерами от 15 до 30.

Заметим к тому же, что если граф БП не имеет такой регулярной структуры, которая была рассмотрена выше, и некоторые ее вершины и ребра объединены, то таблица переходов также имеет «ящичное» строение с недоопределенностью вне ящиков. Поэтому разделения переходов, происходящих при одинаковом состоянии входов, также достаточно для исключения критических состязаний.

Процесс противогоночного кодирования уже не будет столь прозрачен и алгоритмичен, как в рассмотренном случае полного графа.

Для автомата, реализующего граф БП системы (3), который изображен на рис. 1, с таблицей переходов (табл. 2) противогоночный код получается отбрасыванием лишних столбцов матрицы Адамара порядка 12 (см. [4, с. 145]). Отбрасывание столбцов можно производить с помощью алгоритма спуска, предложенного А. Е. Янковской [5].

Получающийся противогоночный код является безызбыточным и имеет вид:

0. 0000	3. 0011	6. 1011	9. 1111	12. 0111.
1. 0010	4. 1010	7. 1101	10. 0001	
2. 1100	5. 0101	8. 0110	11. 1110	

Если же логическое строение автомата заранее неизвестно, то следует обращаться к универсальному коду. Правда, универсальный код может оказаться избыточным. В качестве универсального кода, который обладает противогоночными свойствами безотносительно к виду конкретного автомата с 12 состояниями, можно взять код, построенный из матрицы Адамара порядка 12 отбрасыванием одного разряда без применения метода спуска (который учитывает логическое строение автомата). В [4] показано, что для этого кода минимальное число столбцов вида 0011 и 1100 в произвольной упорядоченной четверке векторов S_1, S_2, S_3, S_4 не менее 1, так как рассматриваемый код эквидистантный с расстоянием $d=6$. Отметим, что длина универсального кода является грубой верхней оценкой длины кода.

Для сочетания противогоночного кодирования с помехоустойчивым имеется несколько подходов. Первый, наиболее простой подход состоит в том, что каждая кодовая комбинация повторяется $2t+1$ раз. Этим достигается устойчивость к любым t и менее ошибкам в условиях состязаний элементов памяти. Второй подход состоит в применении универсального противогоночного и помехоустойчивого кода [4] с последующим удалением лишних столбцов. Еще два подхода связаны с применением методов, связанных с перебором. Эти методы описаны в [4, 6]. По-видимому, специфические особенности графа автомата, реализующего бинарную программу, проявляются особенно ярко именно при противогоночном кодировании.

§ 4. Заключение

Предложено минимальное противогоночное кодирование автомата, реализующего бинарную программу для системы булевых функций, которая обладает специфическим свойством: ее граф — полное дерево.

Длина противогоночного кода в случае полного дерева является тривиальной верхней оценкой длины противогоночного кода состояний произвольного ациклического графа, реализующего бинарную программу для системы булевых функций, число переменных которой совпадает с числом ярусов.

Авторы высказывают предположение, что каков бы ни был граф автомата (какова бы ни была система реализуемых этим автоматом функций), длина его противогоночного кода равна минимальной длине кода, предназначенного только для различения состояний.

ЛИТЕРАТУРА

1. *Кузнецов О. П.* О программной реализации логических функций и автоматов I // Автоматика и телемеханика, 1977, № 7, с. 163—174.
2. *Кузьмин В. А.* Реализация функций алгебры логики автоматами, нормальными алгорифмами и машинами Тьюринга // Проблемы кибернетики. М.: Физматгиз, 1965. Вып. 13. С. 75—96.
3. *Блох А. Ш.* Граф-схемы и их применение. Минск: Вышэйш. шк. 1975.
4. *Сагалович Ю. Л.* Кодирование состояний и надежность автоматов. М.: Связь, 1975.
5. *Янковская А. Е.* Алгоритмы спуска при решении некоторых задач синтеза дискретных устройств и их приложения // Теория дискретных управляющих устройств. М.: Наука, 1982. С. 206—214.
6. *Закревский А. Д., Янковская А. Е.* Помехоустойчивое кодирование внутренних состояний асинхронных автоматов // Информационные материалы. М.: АН СССР, Научный совет по комплексной проблеме «Кибернетика», 1971. Вып. 3(50). С. 53—58.

Поступила в редакцию
27.II.1985
После переработки
12.VI.1985

АКАДЕМИЯ НАУК СССР

ПРОБЛЕМЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

(ОТДЕЛЬНЫЙ ОТТИСК)

I

МОСКВА · 1987