Введение

В данном пособии рассмотрены этапы проектирования управляющих дискретных устройств.

В первой главе вводится понятие конечного автомата как функциональной модели дискретного устройства, рассматриваются классификация автоматов и способы их задания, а также некоторые другие модели описания дискретных устройств.

Вторая глава посвящена методам минимизации конечных автоматов, т.е. сокращению числа их состояний.

В третьей главе изложены основы теории логических сетей, приведена классификация логических элементов и логических сетей, изложены принципиальные методы анализа и синтеза комбинационных и последовательностных схем. Изложены основные проблемы асинхронной схемотехники. Введено понятие опасных состязаний (гонок) в асинхронной схеме.

В четвертой главе рассмотрены методы противогоночного кодирования в асинхронных схемах.

В пятой главе изложены методы кодирований состояний в синхронных схемах, направленные на синтез наиболее простых синхронных схем и на минимизацию числа переключений триггеров.

В шестой главе ставится задача тестирования схемы, рассмотрены методы псевдослучайной генерации тестов и структурные методы построения тестов для комбинационных схем. Также изучается синтез легкотестируемых комбинационных схем и построение для них кратчайшего полного теста по функциональному описанию.

В седьмой главе вводится понятие самопроверяемого дискретного устройства, и излагаются методы синтеза самопроверяемых детекторов неупорядоченных кодов.

Таким образом, в пособии последовательно изложен процесс проектирования дискретного устройства, от функционального описания до структурной реализации в виде логической сети. Каждая глава сопровождается заданиями для самостоятельной работы.

1. Функциональные модели дискретных устройств

1.1. Автоматы

Определение. *Автомат* определяется как пятерка объектов $< X, Q, Y, \psi, \varphi >$, где

- X – входной алфавит;
- Q – внутренний алфавит;
- Y – выходной алфавит;
- $\psi : \Psi \to Q,\ \Psi \subseteq X \times Q$ – функция переходов;
- $\varphi : \Phi \to Y,\ \Phi \subseteq X \times Q$ – функция выходов.

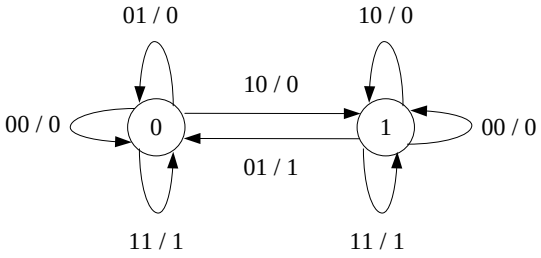
Символы алфавитов X, Q и Y называются соответственно *входными, внутренними* и *выходными* символами. Символы в Q также называются *состояниями*.

Всякий автомат можно рассматривать как математическую модель дискретного устройства, функционирующего в дискретные моменты (*такты*) следующим образом: если в момент времени $t = 1, 2, 3...$ устройство находится в состоянии $q(t)$ и на его вход поступает сигнал $x\ t$, то на выходе устройства вырабатывается сигнал $y\ t = \varphi\ x\ t\ , q\ t$, и устройство переходит в состояние $q\ t+1 = \psi\ x\ t\ , q\ t$.

Если роль входных и выходных переменных интуитивно понятна – это переменные, которыми можно управлять, или которые можно наблюдать, то природа внутренних переменных часто может остаться неизвестной, а их измерение – невозможным. Смысл внутренних переменных можно определить исходя из той роли, которую они играют в определении автомата: выходной символ в текущий момент и состояние в следующий момент однозначно определяются входным символом и состоянием в текущий момент. Таким образом, состояния осуществляют дополнительную связь между входами и выходами, т.е. играют роль *памяти*, поскольку состояние в текущий момент, вообще говоря, зависит от всех символов, поступивших на вход до текущего момента.

Пример. Рассмотрим автомат, описывающий работу турникета метро. Турникет имеет два входа – «монета» и «человек». Турникет пропускает человека, если он опустил монету. Если монета посту-

пила на вход, а человек – нет, то автомат запоминает это событие, и пропускает следующего человека без монеты. Работа такого устройства может быть представлена следующей диаграммой



Автомат имеет два состояния: 0 и 1, которые представлены на диаграмме в виде соответственно обозначенных вершин. Пометки на дугах демонстрируют значения входов, по которым происходит переход из состояния в состояние (до черты, первое значение показывает, поступила или нет на вход монета, второе – человек), и выход (после черты). В состоянии 0 автомат находится в начале работы, и остается в нем, если ни монета, ни человек не поступают на вход, при этом автомат не пропускает никого, т.е. выход равен нулю. На диаграмме эта ситуация представлена петлей, начинающейся и заканчивающейся в вершине 0, с пометкой 00/0. Если человек пытается пройти, не опустив монету, (т.е. значение входа «монета» равно 0, значение входа «человек» – 1), автомат также остается в состоянии 0, и не пропускает человека. Эта ситуация представлена петлей, начинающейся и заканчивающейся в вершине 0, с пометкой 01/0. Наконец, если человек проходит, опустив монету, автомат также остается в состоянии 0, но при этом он пропускает человека, т.е. значение выхода становится равно 1. Эта ситуация представлена петлей, начинающейся и заканчивающейся в вершине 0, с пометкой 11/1. В состояние 1 из состояния 0 автомат переходит, когда на вход поступила монета, а человек – нет, тогда автомат «запоминает» монету. Поведение автомата в состоянии 1 отличается от его поведения в состоянии 0 на входном наборе 01, т.е. когда на вход поступает человек, и не поступает монета – автомат пропускает человека, т.к. «запомнил» брошенную ранее монету, и возвращается в состояние 0. Отметим, что данный автомат запоминает лишь одну монету, чтобы запомнить $2,3,\ldots,n$ монет, требуется $2,3,\ldots,n$ состояний, кроме начального состояния 0.

Алфавиты автомата могут быть декартовыми произведениями других алфавитов. В частности, если $X = X_1 \times X_2 \times ... \times X_n$, то говорят, что автомат имеет n входов, при этом X_i называется *алфавитом i-го входа*.

1.2. Классификация автоматов

Определение. Автомат $A =< X, Q, Y, \psi, \varphi >$ называется *полным*, если области определения его функций выходов и переходов совпадают с $X \times Q$, иначе автомат A называется *частичным*.

Определение. Автомат $A =< X, Q, Y, \psi, \varphi >$ называется *автоматом Мура*, если его функция выходов не зависит существенно от входного символа, то есть если

$$\forall q \in Q, x_1 \in X, x_2 \in X : \varphi\ x_1, q\ = \varphi\ x_2, q\ .$$

Очевидно, что функцию выходов автомата Мура можно рассматривать как функцию только от состояния: $\varphi(x, q) = \varphi(q)$. Но так как состояние автомата зависит от входа на предыдущем такте, то выход также будет зависеть от входа на предыдущем такте:

$$y\ t\ = \varphi\ q\ t\ \ = \varphi\ \psi\ x\ t-1\ , q\ t-1\ \ = \tilde{\varphi}\ x\ t-1\ , q\ t-1\ \ .$$

Определение. Автомат $A =< X, Q, Y, \psi, \varphi >$ называется *автоматом Мили*, если его функция выходов зависит существенно от входного символа, то есть если

$$\exists q \in Q, x_1 \in X, x_2 \in X : \varphi\ x_1, q\ \neq \varphi\ x_2, q\ .$$

Модели автоматов Мура и Мили не являются принципиально различными, напротив, любая из них может быть сведена к другой. Привести модель Мура к модели Мили просто – достаточно добавить в аргументы функции выходов фиктивную переменную, описывающую вход. Приведение модели Мили к модели Мура осуществляется за счет увеличения числа состояний.

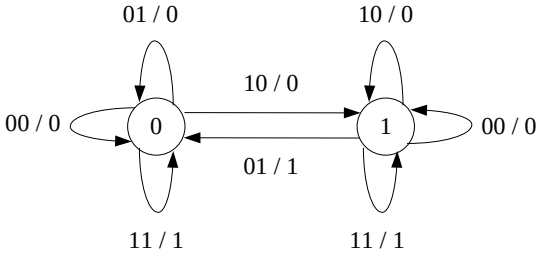
Рассмотрим этот процесс подробно. Пусть $A =< X, Q, Y, \psi, \varphi >$ – автомат Мили. Определим автомат Мура $A' =< X, Q', Y, \psi', \varphi' >$:

- $Q' = Q \times Y$, т.е. состояниями автомата A' являются пары q, y, где q – состояние, а y – выход автомата A;
- $\forall x \in X, q \in Q, y \in Y : \psi'\ x,\ q, y\ \ = q', y'$, если $\psi\ x, q\ = q'$,

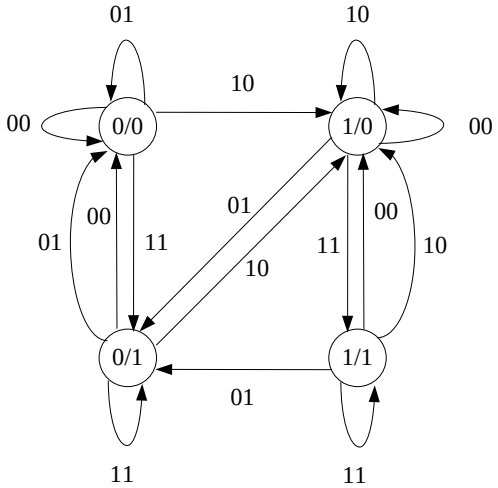
$\varphi(x, q) = y'$;

- $\forall q \in Q, y \in Y : \varphi(q, y) = y$.

Пример. Рассмотрим автомат из предыдущего примера. Легко видеть, что он является автоматом Мили.



Эквивалентный ему автомат Мура можно представить диаграммой. Автомат имеет 4 состояния, выход в каждом состоянии – это вторая цифра, первая цифра соответствует состоянию исходного автомата, т.е. вместо обозначения q, y используется обозначение q/y для упрощения записи.



Определение. Автомат, в котором $Q = Y$ и $\varphi(x, q) = \psi(x, q)$ называется *автоматом состояний*, или *автоматом без выхода*, и

обозначается $< X, Q, \psi >$.

Автомат состояний используется как модель устройства, функция выходов которого нас не интересует.

Дальнейшая классификация автоматов может быть проведена путем наложения ограничений на мощности их алфавитов.

Определение. Автомат с одноэлементным входным алфавитом $|X| = 1$ называется *автономным* и обозначается $A =< Q, Y, \psi, \varphi >$.

Автономный автомат является математической моделью дискретного устройства, в котором

$$\begin{cases} y\ t = \varphi\ q\ t\ ; \\ q\ t+1 = \psi\ q\ t\ . \end{cases}$$

Если в таком устройстве зафиксировать начальное состояние $q\ 0$, то в случае всюду определенных функций $\varphi \cdot$ и $\psi \cdot$ можно однозначно определить единственную выходную последовательность $y\ 1\ y\ 2\ ...$ и единственную внутреннюю последовательность $q\ 1\ q\ 2\ ...$. Если $|Q| = k$, то уже среди значений $q\ 1\ q\ 2\ ...q\ k+1$ найдутся повторяющиеся, то есть внутренняя и выходная последовательности окажутся периодическими с периодом, не превосходящим k, и, возможно, с некоторым предпериодом. Иначе говоря, полный автономный автомат является математической моделью генератора случайных чисел.

Определение. Автомат с одноэлементным внутренним алфавитом $|Q| = 1$ называется *комбинационным* или *автоматом без памяти* и обозначается $A =< X, Y, \varphi >$. Последнее название оправдано тем, что величину $\log_2 |Q|$ принято называть *объемом памяти* автомата. Для автомата без памяти $\log_2 |Q| = \log_2 1 = 0$.

Пример. Если $X = B^n$, $Y = B^m$, $B = \{0,1\}$, то автомат без памяти представляет систему m булевых функций от n переменных.

Определение. Автомат с конечными внутренним и входным алфавитами называется *конечным* автоматом.

Именно конечные автоматы являются математическими моделями реальных дискретных устройств. Далее мы будем рассматривать только конечные автоматы, поэтому договоримся употреблять термины «автомат» и «конечный автомат» как синонимы.

1.3. Способы задания конечных автоматов

1. *Диаграмма переходов-выходов.* Это ориентированный граф, вершины которого соответствуют состояниям автомата, а дуги – переходам их состояниям автомата. Если по символу x автомат переходит из состояния q в состояние q', то дуга направляется от вершины q к вершине q' и помечается символом x. Если автомат является автоматом Мили, то дуга помечается еще и символом $y = \varphi(x, q)$. На диаграмме автомата Мура символ выходного алфавита указывается прямо в вершине, соответствующей связанному с ним состоянию.

Примеры диаграмм переходов-выходов были приведены ранее.

В диаграмме переходов автономного автомата из каждой вершины выходит не более чем одна дуга. Диаграмма переходов автомата без памяти содержит ровно одну вершину, в которой начинаются и кончаются все дуги этой диаграммы. Дуги диаграммы переходов автомата без выхода помечены только входными символами.

2. *Таблица переходов и выходов.* Так называются таблицы, которыми задаются соответственно функция переходов и функция выходов автомата. Строки таблиц поставлены во взаимно однозначное соответствие входным символам автомата, а столбцы – состояниям. В клетке x, q записывается в первой таблице состояние $q' = \psi(x, q)$, если функция $\psi(x, q)$ определена для данных x, q; а во второй таблице выходной символ $y = \varphi(x, q)$, если функция $\varphi(x, q)$ определена для данных x, q. Для тех пар x, q, для которых функция не определены, клетка остается пустой или заполняется прочерком. Иногда, для достижения большей компактности, обе таблицы совмещают в одну, и в клетке x, q слева от разделительного знака (, или /) пишут переход, а справа – выход.

Пример. Рассмотрим задание автомата Мили из предыдущего примера функцией переходов и выходов.

$X \setminus Q$	0	1
00	0 / 0	1 / 0
01	0 / 0	0 / 1
10	1 / 0	1 / 0
11	0 / 1	1 / 1

В случае автомата Мура таблица выходов состоит из одной строки, в которой для каждого состояния q указано соответствующее значение функции выходов $\varphi(q)$, если оно определено. Для автономного автомата обе таблицы являются однострочными. Для автомата без памяти таблица переходов и выходов вырождаются в одностолбцовую таблицу для φ q .

3. *Канонические уравнения.* Если функции переходов и выходов допускают аналитическое выражение (например, в алгебре логики), то автомат может быть задан уравнениями вида

$$\begin{cases} y = \varphi(x, q); \\ q' = \psi(x, q). \end{cases}$$

Эти уравнения описывают связь между входным символом x и состоянием q с одной стороны и выходным символом y и состоянием q' с другой стороны. Второе уравнение является рекуррентным.

Пример. Рассмотрим автомат из предыдущего примера. Обозначим входы через a и b. Выпишем таблицу истинности для q' и y.

a	b	q	q'	y
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	0	1
1	1	1	1	1

Выпишем кратчайшие ДНФ этих функций.

$$\begin{cases} y = a b \vee b q; \\ q' = a \bar{b} \vee a q \vee \bar{b} q. \end{cases}$$

Интерпретировать эти уравнения очень просто: турникет пропускает человека ($y = 1$), когда он опустил монету и сразу прошел ($a = b = 1$), либо монета была опущена ранее, и автомат это запомнил ($b = q = 1$). Турникет должен будет пропустить человека без монеты (автомат перейдет в состояние $q = 1$) если монета опущена, а человек не прошел ($a = 1, b = 0$). Автомат остается в состоянии 1, если он уже находится в этом состоянии ($q = 1$), и при этом либо

брошена еще одна монета ($a = 1$), либо человек еще не прошел ($b = 0$).

Канонические уравнения автомата также записывают в виде

$$\begin{cases} y\ t\ = \varphi\ x\ t\ , q\ t\ ; \\ q\ t+1\ = \psi\ x\ t\ , q\ t\ . \end{cases}$$

Каждое из канонических уравнений автомата может быть векторным, если алфавиты Y или Q являются декартовыми произведениями других алфавитов. В этом случае их можно расписать как совокупность скалярных уравнений.

Построение конечного автомата для дискретного устройства с заданным поведением представляет собой нетривиальную задачу, в первую очередь потому, что сложно определить необходимое число состояний и функцию переходов. Далее мы рассмотрим некоторые функциональные модели дискретных устройств, которые могут служить промежуточными этапами при построении конечного автомата.

1.4. Система формул переходов

Пусть задано *множество логических условий* $X = x_1, \ldots, x_n$, принимающих значения из булева множества $B = 0,1$, и *множество микроопераций* $Y = y_1, \ldots, y_m$. Под микрооперацией будем понимать единый неделимый акт обработки информации. *Микрокоманда* (или *оператор*) $Y_j \subset Y$ есть набор микроопераций.

Пусть заданы множество операторов $Y^* = Y_1, \ldots, Y_M, Y_0, Y_k$, где $Y_0 = \varnothing$ – начальный и $Y_k = \varnothing$ – конечный операторы. Рассмотрим произвольный оператор $Y_j \neq Y_k$.

Определение. *Формула перехода* из оператора Y_j – это выражение вида

$$Y_j \rightarrow U_1^j Y_1 \vee \ldots \vee U_M^j Y_M \vee U_k^j Y_k .$$

Здесь $U_1^j, \ldots, U_M^j, U_k^j$ – булевы функции, зависящие от переменных множества логических условий X, U_p^j – *функция перехода* от оператора Y_j к оператору Y_p. Некоторые из этих функций могут быть равны нулю, и никакие две из них не могут принимать значе-

ние единица на одном наборе значений переменных. Как правило, функции перехода являются ортогональными элементарными конъюнкциями.

Слева от знака $\rightarrow$ в данной формуле находятся оператор, который выполняется в текущий момент времени (такт), справа – операторы, один из которых, в зависимости от значений логических условий, выполнится на следующем такте. Формула означает, что после оператора Y_j на следующем такте выполняется оператор Y_p (то есть происходит переход к оператору Y_p из оператора Y_j), если $U_p^j = 1$. Для конечного оператора формулы перехода не существует.

Определение. Множество формул перехода для операторов $Y_1,\ldots,Y_M,Y_0$ образует *систему формул перехода* (или *СФП*).

Пример. Пусть количество логических условий $n = 4$, и количество микроопераций $m = 5$. Рассмотрим СФП для шести операторов (включая начальный оператор Y_0).

$$Y_0 \rightarrow x_1\bar{x}_2Y_1 \vee x_1x_2Y_2,$$

$$Y_1 \rightarrow x_2x_3x_4Y_2 \vee x_2x_3\bar{x}_4Y_3 \vee x_2\bar{x}_3Y_1 \vee \bar{x}_2Y_4, \qquad Y_1 = y_1, y_3, y_5\ ;$$

$$Y_2 \rightarrow \bar{x}_3Y_1 \vee x_3x_4Y_2 \vee x_3\bar{x}_4Y_3, \qquad Y_2 = y_1, y_4\ ;$$

$$Y_3 \rightarrow Y_k, \qquad Y_3 = y_2, y_4\ ;$$

$$Y_4 \rightarrow x_4Y_2 \vee x_3\bar{x}_4Y_k \vee \bar{x}_3\bar{x}_4Y_5, \qquad Y_4 = y_2, y_3, y_5\ ;$$

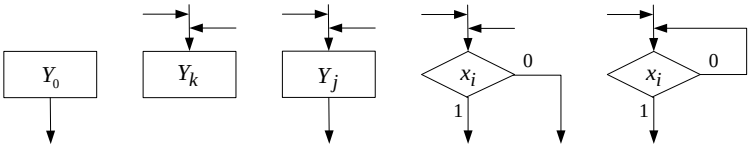
$$Y_4 \rightarrow x_4Y_2 \vee x_3\bar{x}_4Y_k \vee \bar{x}_3\bar{x}_4Y_5, \qquad Y_5 = y_2, y_5\ .$$

Из формул следует, что, например, после выполнения оператора Y_0 при $x_1 = 1,\ x_2 = 0$ и произвольных значениях остальных переменных логических условий на следующем такте выполняется оператор Y_1 (то есть микрооперации y_1, y_3, y_5), а при $x_1 = 1,\ x_2 = 1$ выполняется оператор Y_2 (микрооперации y_1, y_4). При $x_1 = 0$ не происходит перехода ни к какой микрокоманде, это означает, что реализующее алгоритм дискретное устройство не выполняет никаких микроопераций, пока на некотором такте на входе x_1 не появится единица.

После выполнения оператора Y_3 на следующем такте происходит *безусловный переход* к оператору Y_k, то есть оператор Y_k выполняется при любых значениях переменных x_i. •

1.5. Граф-схема алгоритма

Определение. *Граф-схема алгоритма* (*ГСА*) – это ориентированный слабо связный граф, содержащий одну *начальную* вершину Y_0, одну *конечную* вершину Y_k, и множество $P = P_1,\ldots,P_r$ *условных* вершин и множество $A = A_1,\ldots,A_t$ *операторных* вершин.



Вершины ГСА удовлетворяют следующим условиям:

- из начальной вершины исходит одна дуга, в начальную вершину не заходит ни одной дуги;
- в конечную вершину заходят не менее одной дуги, из конечной вершины не исходит ни одной дуги;
- в операторную вершину заходят не менее одной дуги, из операторной вершины исходит одна дуга;
- в условную вершину заходят не менее одной дуги, из условной вершины исходит две дуги, помеченные 0 и 1.

В каждой условной вершине записывается один из элементов множества логических условий. В каждой операторной вершине записывается микрокоманда.

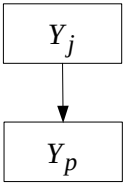
ГСА удовлетворяет следующим условиям:

- каждая вершина графа лежит по крайней мере на одном пути, соединяющем начальную вершину с конечной;
- одна их исходящих дуг условной вершины может заходить в эту же вершину (такие вершины называются *ждущими* или *возвратными*);
- исходящие из операторных вершин дуги не могут заходить в эти же вершины.

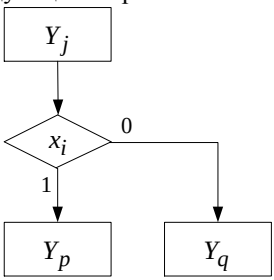
1.6. Переход от СФП к ГСА

Пусть алгоритм функционирования устройства задан системой формул переходов. Рассмотрим построение граф-схемы алгоритма.

Пусть задана формула переходов вида $Y_j \to Y_p$, на ГСА она изображается следующим образом.



Пусть задана формула переходов вида $Y_j \to x_i Y_p \vee \bar{x}_i Y_q$, на ГСА она изображается следующим образом.



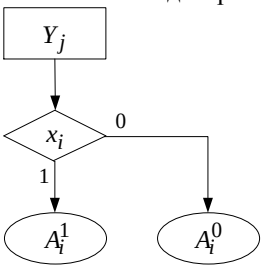
Пусть теперь задана формула перехода общего вида:

$$Y_j \to U_1^j Y_1 \vee \ldots \vee U_M^j Y_M \vee U_k^j Y_k .$$

Разложим ее правую часть по Шеннону по переменной, которая встречается во всех функциях перехода U_p^j, получим формулу вида $Y_j \to \bar{x}_i A_i^0 \vee x_i A_i^1$, где A_i^0, A_i^1 – подформулы вида:

$$\tilde{U}_1^j Y_1 \vee \ldots \vee \tilde{U}_M^j Y_M \vee \tilde{U}_k^j Y_k .$$

Соответствующая разложению часть диаграммы выглядит так:



Если, например, $A_i^0 = Y_j$, то вершина x_i является ждущей: исходящая дуга, помеченная нулем, заходит в эту же вершину. Если $A_i^0 = Y_p$, то часть диаграммы, сооьветствующая A_i^0, представляет

собой операторную вершину. Иначе разложим подформулу A_i^0 по очередной переменной, нарисуем в диаграмме на месте A_i^0 условную вершину, помеченную этой переменной, и продолжим разложение, если это необходимо. Так действуем до тех пор, пока коэффициенты разложения не будут представлять собой микрокоманды. Таким образом мы построим подграф, соответствующий формуле перехода. Объединив подграфы для всех формул перехода в один граф, получим ГСА. Следует отметить, что если одна и та же подформула встречается в нескольких формулах перехода, соответствующий ей подграф нарисовать лишь один раз и показать переходы к ней с помощью дуг.

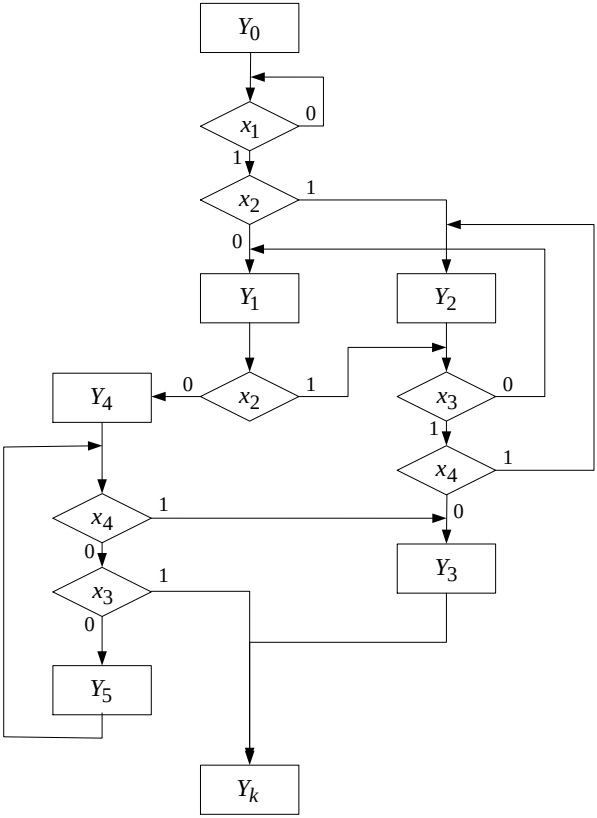
Пример. Рассмотрим СФП из предыдущего примера.

$$Y_0 \to x_1\overline{x}_2Y_1 \vee x_1x_2Y_2;$$
$$Y_1 \to x_2x_3x_4Y_2 \vee x_2x_3\overline{x}_4Y_3 \vee x_2\overline{x}_3Y_1 \vee \overline{x}_2Y_4;$$
$$Y_2 \to \overline{x}_3Y_1 \vee x_3x_4Y_2 \vee x_3\overline{x}_4Y_3;$$
$$Y_3 \to Y_k;$$
$$Y_4 \to x_4Y_2 \vee x_3\overline{x}_4Y_k \vee \overline{x}_3\overline{x}_4Y_5;$$
$$Y_5 \to x_4Y_2 \vee x_3\overline{x}_4Y_k \vee \overline{x}_3\overline{x}_4Y_5.$$

В первой формуле переменная x_1 встречается только без инверсии, значит, вершина x_1 является ждущей. Разложим правую часть формулы по переменной x_1. Разложим по Шеннону также остальные формулы, каждый раз выбирая переменную, которая встречается во всех функциях перехода.

$$Y_0 \to x_1\ \ \overline{x}_2Y_1 \vee x_2Y_2\ \ ;$$
$$Y_1 \to x_2\ \ \underline{x_3\ \ x_4Y_2 \vee \overline{x}_4Y_3\ \ \vee \overline{x}_3Y_1}\ \ \vee \overline{x}_2Y_4;$$
$$Y_2 \to \underline{\overline{x}_3Y_1 \vee x_3\ \ x_4Y_2 \vee \overline{x}_4Y_3}\ \ ;$$
$$Y_3 \to Y_k;$$
$$Y_4 \to \underline{\underline{x_4Y_3 \vee \overline{x}_4\ \ x_3Y_k \vee \overline{x}_3Y_5}}\ \ ;$$
$$Y_5 \to \underline{\underline{x_4Y_3 \vee \overline{x}_4\ \ x_3Y_k \vee \overline{x}_3Y_5}}\ \ .$$

Здесь подчеркнуты одинаковые подформулы. Соответствующая ГСА имеет следующий вид.



1.7. Переход от ГСА к автомату

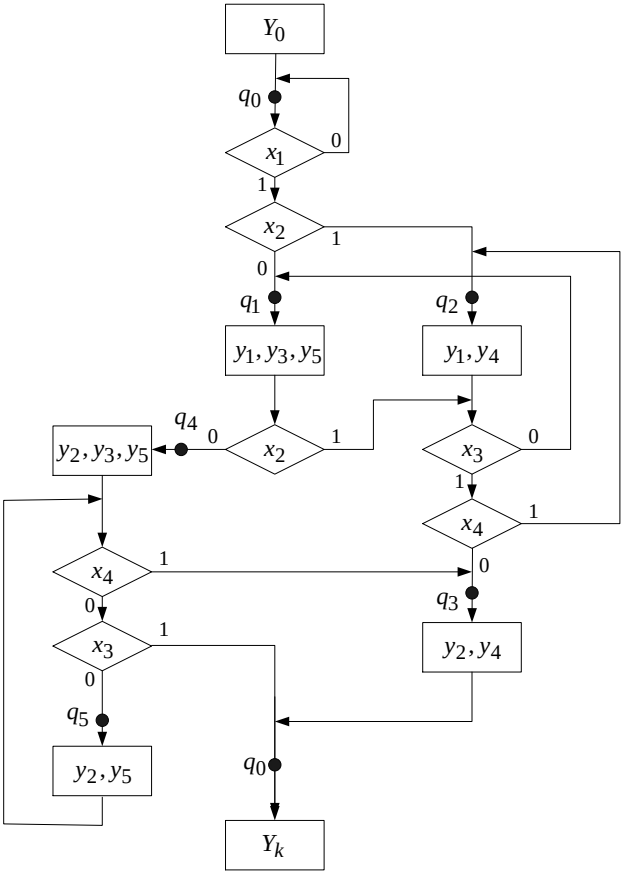
Пусть задана ГСА. Построим автомат $A = \langle X, Y, Q, \varphi, \psi \rangle$, реализующий данный алгоритм. Здесь входной алфавит X – булево пространство B^n, выходной алфавит Y – булево пространство B^m. Мы будем описывать входные и выходные сигналы на языке элементарных конъюнкций. Внутренний алфавит Q и функции переходов $\psi(\cdot)$ и выходов $\varphi(\cdot)$ строятся по-разному для автоматов Мура и Мили.

1.7.1. Переход от ГСА к автомату Мура

Расставим на ГСА отметки состояний:

- отметка q_0 ставится на дуге, исходящей из начальной вершины, и на дуге, заходящей в конечную вершину;
- отметки $q_1,\ldots,q_t$ ставятся около операторных вершин;
- отметки $q_{t+1},\ldots,q_k$ ставятся на дугах, заходящих в ждущие вершины.

Пример. Расставим отметки состояний на ГСА из предыдущего примера.

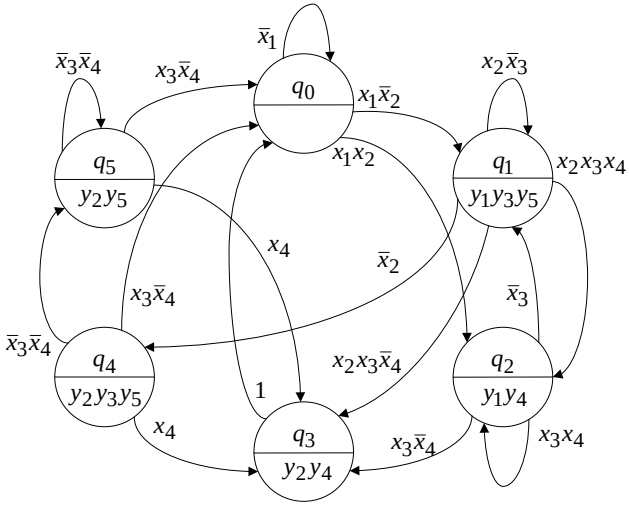


Здесь отметки $q_1,\ldots,q_5$ поставлены около операторных вершин $Y_1,\ldots,Y_5$, а микрокоманды заменены соответствующими множествами микроопераций. Отметим, что исходящая из начальной вершины дуга заходит в ждущую вершину, значит, в данном случае достаточно одной отметки q_0.

Далее построим диаграмму переходов-выходов автомата. С состоянием q_j, расположенным на графе около операторной вершины, связываем выход – множество микроопераций, записанных в этой вершине. Для состояния q_0 и состояний, связанных со ждущими вершинами, функция выхода есть пустое множество микроопераций. Автомат переходит из состояния q_j в состояние q_p под воздействием сигнала U_p^j, который формируется следующим образом.

- Находится очередной простой путь, соединяющий q_j и q_p и не содержащий отметок других состояний.
- По выбранному пути составляется элементарная конъюнкция: если путь проходит через условную вершину x_i, то x_i входит в конъюнкцию. При этом если исходящая из x_i дуга помечена символом 1, то x_i входит в конъюнкцию без инверсии, иначе x_i входит в конъюнкцию с инверсией. Если путь не проходит ни через одну условную вершину, конъюнкция равна 1.
- Составляется дизъюнкция всех таких элементарных конъюнкций – это функция перехода U_p^j, то есть $\psi\left(U_p^j, q_j\right) = q_p$.

Пример. Автомат, построенный по последней ГСА, имеет шесть состояний. Функции выхода $\varphi(q_0) = \varnothing$, $\varphi(q_1) = y_1, y_3, y_5$, и т.д. Найдем функцию переходов U_1^0. Состояния q_0 и q_1 соединяет один простой путь, проходящий через вершины x_1 (дуга помечена единицей) и x_2 (дуга помечена нулем), значит, $U_1^0 = x_1\overline{x}_2$. Аналогично находятся остальные функции переходов. Диаграмма переходов-выходов автомата имеет вид:

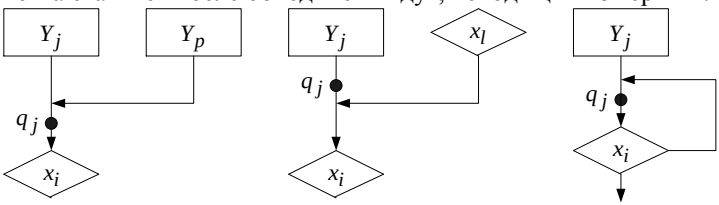


1.7.2. Переход от ГСА к автомату Мили

В случае автомата Мили отметки состояний ставятся:

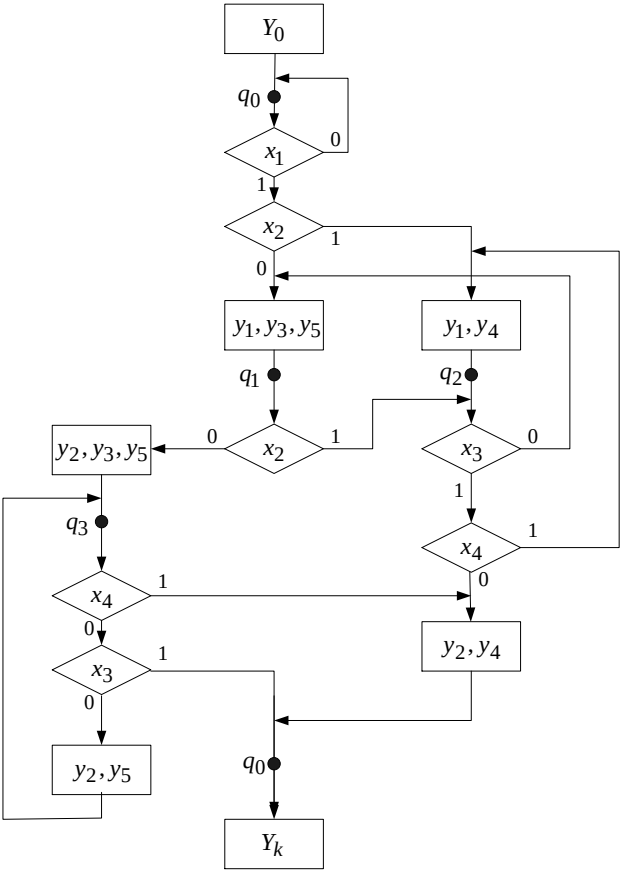
- отметка q_0 ставится на дуге, исходящей из начальной вершины, и на дуге, заходящей в конечную вершину;
- отметки $q_1,\ldots,q_t$ ставятся на дугах, исходящих из операторных вершин;
- отметки $q_{t+1},\ldots,q_k$ ставятся на дугах, заходящих в ждущие вершины.

Заметим, что если дуги, исходящие из двух операторных вершин, заходят в одну вершину, отметка ставится после объединения этих дуг. Если же в одну вершину (не конечную!) заходят дуги, исходящие из операторной и условной вершины, отметка ставится до объединения этих дуг. Если за операторной вершиной следует ждущая, отметка ставится после объединения дуг, исходящих из вершин.



Так как одна и та же вершина может следовать за несколькими операторными вершинами (иллюстрация справа), в общем случае число состояний в автомате Мили меньше, чем число состояний в эквивалентном ему автомате Мура.

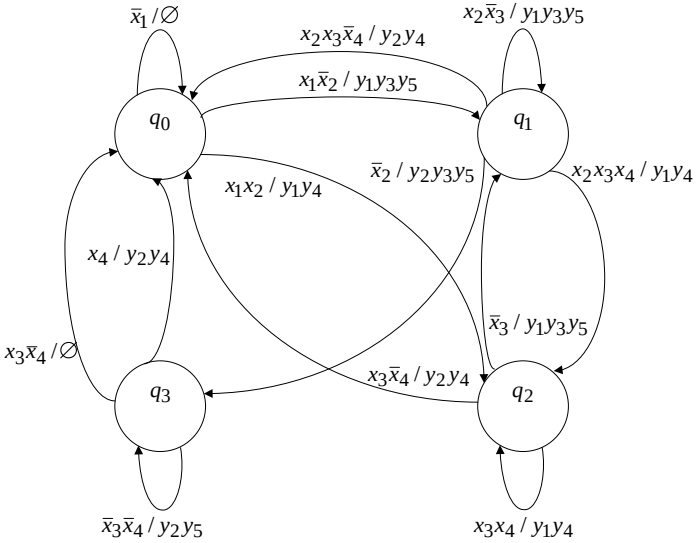
Пример. Для предыдущей ГСА отметки состояний расставляются следующим образом. Число состояний здесь на два меньше, чем в эквивалентном автомате Мили.



Функция переходов автомата Мили находится так же, как и для автомата Мура. Функция выходов находится следующим образом: если простой путь, соединяющий состояния q_j и q_p, заканчивается

операторной вершиной, то функция выхода, соответствующая этому пути, есть множество записанных в операторной вершине микроопераций, иначе функция выхода есть $\varnothing$.

Пример. Диаграмма переходов-выходов автомата Мили, построенного по последней ГСА, выглядит следующим образом.



1.8. Задачи

Задача 1. Опишите систему, которая может быть представлена конечным автоматом. Задайте входной, выходной и внутренний алфавиты и функции переходов и выходов. Обоснуйте выбор множества состояний.

Задача 2. Построить конечные автоматы.

а) Автомат выдает единицу, если суммарная длина серий, содержащих не менее двух единиц, нечетная, и ноль в противном случае. Пример:

вход	1	0	1	1	1	0	1	0	0	1	1	0	1	1	1	0
выход	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0

b) Автомат выдает единицу, если число серий, содержащих не менее двух единиц, нечетно, и ноль в противном случае. Пример:

вход	1	0	1	1	1	0	1	0	0	1	1	0	1	1	1	0
выход	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1

c) Автомат выдает единицу, если число серий, содержащих нечетное число единиц, нечетно, и ноль в противном случае. Пример:

вход	1	0	1	1	1	0	1	0	0	1	1	0	1	1	1	0
выход	0	1	1	1	1	0	0	1	1	1	1	1	1	1	1	0

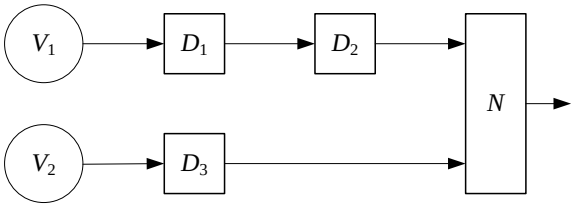
d) Автомат выдает единицу, если число серий, содержащих ровно две единицы, нечетно, и ноль в противном случае. Пример:

вход	0	1	0	1	1	0	1	0	0	1	1	0	1	1	1	0
выход	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0

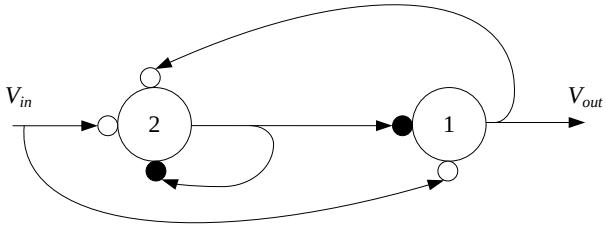
Задача 3. Грузовой лифт, обслуживающий трехэтажный магазин, имеет кнопку вызова на каждом этаже. Если нажата одна кнопка, то лифт движется на этаж, где расположена данная кнопка; если нажаты две или три кнопки, то лифт движется на самый нижний из всех этажей, на которых нажаты кнопки. Ни одна кнопка не может быть нажата во время движения лифта. Описать автомат, управляющий работой лифта (вход – нажатые кнопки, выход – направление, в котором движется лифт, и число этажей, которые он пройдет без остановки).

Задача 4. На рисунке изображена схема N, на которую поступают сигналы от двух импульсных генераторов напряжения V_1 и V_2. Каждый генератор генерирует положительный или отрицательный импульс с периодом 1 микросекунда. Элемент D задерживает импульс на 1 микросекунду. Схема N выдает положительный импульс, когда оба поступающих на ее входы импульса положительны, и отрицательный во всех остальных случаях. Описать закон

функционирования схемы конечным автоматом.



Задача 5. На рисунке представлена модель нервной сети, где нервное волокно V_{in} соединено с источником, который генерирует стимулы 0 или 1 в моменты времени t_i. Большой кружок представляет нейрон. Выход нейрона в момент t_i возбужден (т.е. принимает значение 1) только в том случае, когда разность между числом его возбужденных «возбуждающих входов» (маленькие белые кружки) и числом его возбужденных «тормозящих входов» (маленькие черные кружки) в момент t_{i-1} равна или превышает его «порог» (число, записанное внутри большого кружка). Описать с помощью конечного автомата закон функционирования сети.



Задача 6. Работа вычислительного устройства, имеющего вход x и выход y, определяется булевыми уравнениями

$$y_i = x_i \oplus y_{i-1} \oplus z_{i-1};$$
$$z_i = x_i \oplus z_{i-1}.$$

Записать эквивалентную диаграмму переходов-выходов автомата.

Задача 7. На вход устройства поступают нули и единицы. Построить конечный автомат, осуществляющий суммирование единиц по модулю n.

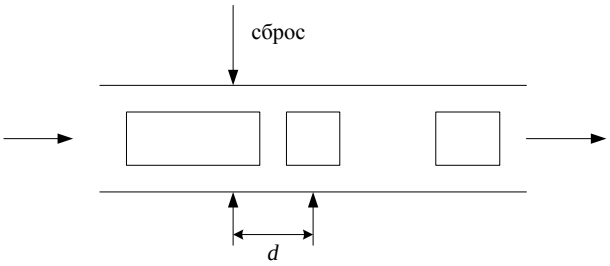
Задача 8. Синтезировать конечный автомат, осуществляющий поразрядное сложение двух чисел в двоичной системе счисления.

Задача 9. Показать, что с помощью конечного автомата нельзя реализовать умножение двоичных чисел.

Задача 10. Построить конечный автомат, регулирующий пешеходный переход по запросу пешеходов.

Задача 11. На дороге с односторонним движением установлены два датчика, расстояние между которыми равно d. Длина легковой машины меньше d, длина грузовой машины больше d. С датчиков поступает сигнал на сигнальное устройство, проезжает ли мимо датчика машина. Сигнальное устройство должно сообщать, какая именно машина проезжает по дороге. Синтезировать автомат, описывающий работу сигнального устройства.

Задача 12. Рядом с конвейером установлены два датчика-фотоэлемента, передающие сигнал на устройство сброса. Расстояние между датчиками равно d. Детали на конвейере могут быть различной длины (как больше, так и меньше d). Деталь сбрасывается, если расстояние между ней и предыдущей деталью меньше d. Синтезировать автомат, управляющий устройством сброса.



Задача 13. По заданной СФП построить ГСА, а затем графы автоматов Мили и Мура.

$Y_0 \to x_1 Y_1;$

$Y_1 \to x_2 Y_2 \vee \overline{x}_2 x_3 Y_3 \vee \overline{x}_2 \overline{x}_3 Y_4;$

$Y_2 \to Y_5;$

$Y_3 \to x_2 x_4 Y_2 \vee \overline{x}_2 x_3 x_4 Y_3 \vee \overline{x}_2 \overline{x}_3 x_4 Y_4 \vee \overline{x}_4 Y_k;$

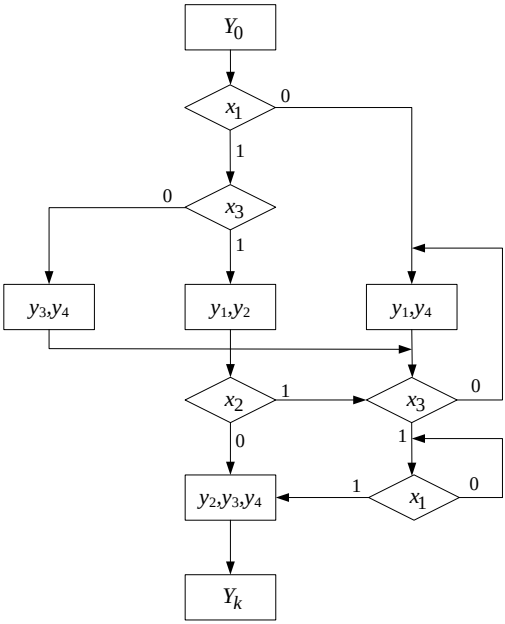
$Y_4 \to Y_1;$

$Y_5 \to x_2 \overline{x}_5 Y_1 \vee \overline{x}_2 \overline{x}_5 Y_4 \vee x_5 Y_6;$

$Y_6 \to x_2 Y_3 \vee \overline{x}_2 Y_4.$

$Y_1) y_1, y_2;\ Y_2) y_1, y_2, y_3;\ Y_3) y_3, y_5;\ Y_4) y_2, y_4;\ Y_5) y_3, y_5;\ Y_6) y_1, y_3$

Задача 15. По заданной ГСА построить диаграммы переходов-выходов автоматов Мили и Мура. Записать СФП, эквивалентную заданной ГСА.



2. Минимизация автоматов

Если в автомате выделить некоторое начальное состояние и подать на вход некоторую последовательность входных символов, мы получим последовательность выходных символов. Этот переход однозначен, а обратный переход многозначен, то есть существует множество автоматов, решающих одну и ту же задачу, но отличающихся друг от друга, в частности, числом внутренних состояний. В конечных автоматах не нужно интересоваться физической природой состояний. Единственная их функция состоит в том, чтобы участвовать в определении зависимостей между входами и выходами автомата. Следовательно, любое множество состояний, выполняющих эту функцию, является приемлемым, и желательно минимизировать число состояний автомата. Известные методы синтеза не гарантируют получения автомата с минимальным числом состояний, и тогда возникает задача минимизации числа состояний уже построенного автомата, называемая кратко *задачей минимизации автомата*.

2.1. Минимизация полных автоматов

2.1.1. Неотличимость состояний

Для решения задачи минимизации введем необходимые определения. Рассмотрим автомат $A = < X, Q, Y, \psi, \varphi >$

Определение. *Словом* в алфавите X называется любая последовательность символов алфавита X. Длиной слова α называют число символов в нем (обозначается $|\alpha|$). Отдельно выделяют *пустое слово* (обозначается ε).

Определение. *Языком* над алфавитом X (обозначается X^*) называется множество всех слов в алфавите X, иначе говоря

$$X^* = \varepsilon \cup X \cup X \times X \cup \ldots \cup X \times \ldots \times X \cup \ldots$$

Пример. Пусть $X = a, b, c$. Тогда a, bc, $aacb$ – слова в алфавите X, а язык

$$X^* = \varepsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, \ldots .$$

Распространим функции $\psi(x, q)$ и $\varphi(x, q)$ на слова из языка X^*. Дадим их рекуррентное определение

$$\psi\ \varepsilon, q\ = q;$$

$$\psi\ \alpha x, q\ = \psi\ x, \psi\ \alpha, q\ \ ;$$

$$\varphi\ \alpha x, q\ = \varphi\ x, \psi\ \alpha, q\ \ .$$

Другими словами, под воздействием последовательности β автомат пробегает цепочку состояний и выдает цепочку выходных символов, последние элементы этих цепочек суть соответственно переход и выход $\psi\ \beta, q$ и $\varphi\ \beta, q$.

Пример. Рассмотрим автомат, заданный таблицами переходов и выходов

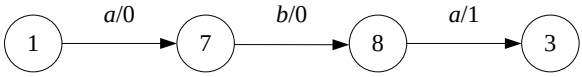
ψ

*X**Q*	1	2	3	4	5	6	7	8
a	7	7	8	8	7	5	1	3
b	6	2	6	2	4	1	8	8
c	1	1	3	3	3	1	3	1

φ

*X**Q*	1	2	3	4	5	6	7	8
a	0	0	0	0	0	0	1	1
b	1	1	1	1	1	1	0	0
c	0	0	0	0	0	0	0	0

Пусть автомат находится в состоянии 1, построим диаграмму его работы для слова *aba*



Отсюда $\psi\ aba, 1\ = 3, \varphi\ aba, 1\ = 1.$

Рассмотрим автоматы с одинаковыми входными и выходными алфавитами $A_1 =< X, Q_1, Y, \psi_1, \varphi_1 >$, и $A_2 =< X, Q_2, Y, \psi_2, \varphi_2 >$.

Определение. Состояния $q_1 \in Q_1$ и $q_2 \in Q_2$ *неотличимы* (обозначается $q_1 = q_2$), если

$$\forall \alpha \in X^* : \varphi_1(\alpha, q_1) = \varphi_2(\alpha, q_2)\ .$$

Иначе эти состояния называются *отличимыми* (обозначается $q_1 \neq q_2$), а α *– отличающим словом*.

Другими словами, состояния q_1 и q_2 неотличимы, если при подаче на входы автоматов A_1 и A_2, находящихся в состояниях q_1 и q_2 соответственно, любой последовательности α на выходах автоматов будет сгенерирована одна и та же последовательность β. В частном случае, при $A_1 = A_2 = A$, говорят о неотличимых состояниях автомата A. Далее мы покажем, что задача минимизации полного автомата сводится к поиску неотличимых состояний.

Пример. Рассмотрим автомат из предыдущего примера

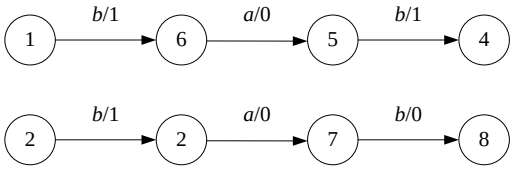
ψ

$X\backslash Q$	1	2	3	4	5	6	7	8
a	7	7	8	8	7	5	1	3
b	6	2	6	2	4	1	8	8
c	1	1	3	3	3	1	3	1

φ

$X\backslash Q$	1	2	3	4	5	6	7	8
a	0	0	0	0	0	0	1	1
b	1	1	1	1	1	1	0	0
c	0	0	0	0	0	0	0	0

Подадим последовательность *bab* на автомат, находящийся в состояниях 1 и 2



Из диаграммы видно, что $\varphi(bab,1) = 1$, $\varphi(bab,2) = 0$, значит, $1 \neq 2$, то есть эти состояния отличимы, и *bab* – отличающее слово. Заметим, что столбцы таблицы выходов у этих состояний одинаковы, т. е. для неотличимости этого недостаточно.

Определение неотличимости состояний не является конструктивным в том смысле, что оно не дает алгоритма проверки состояний на неотличимость, поскольку множество слов бесконечно. Но из определения неотличимости состояний нетрудно вывести правила проверки состояний на неотличимость:

1. Неотличимым состояниям должны соответствовать равные столбцы матрицы выходов.
2. Соответствующие этим состояниям столбцы матрицы переходов должны быть равны или отличаться символами неотличимых состояний.

Проверка неотличимости заданной пары может повлечь за собой проверку неотличимости некоторых других пар, и этот процесс может иметь цепной характер. Однако, вследствие конечности числа состояний этот процесс конечен. Проверку можно произвести с помощью *графа условий неотличимости*.

Алгоритм поиска неотличимых состояний автомата

Начало. Задан автомат $A=<X,Q,Y,\psi,\varphi>$.

Шаг 1. Выбираем очередную пару p,q состояний с одинаковыми столбцами таблицы выходов. Если она еще не добавлена в граф условий неотличимости, добавляем ее как очередную вершину.

Шаг 2. Пусть по какому-то входному символу пара p,q переходит в пару различных состояний $p',q' \neq p,q$. Если пара p',q' не добавлена в граф условий неотличимости, добавляем ее в качестве очередной вершины. Проводим дугу из p,q в p',q'.

Шаг 3. Если пара p',q' имеет разные столбцы таблицы выходов, вычеркиваем ее.

Шаг 4. Просматриваем все вершины графа. Если пара s',t' вычеркнута, и есть дуга из s,t в s',t', вычеркиваем s,t. Повторяем этот шаг, пока возможны вычеркивания.

Шаг 5. Если не все пары с одинаковыми столбцами таблицы выходов просмотрены, идем на шаг 1.

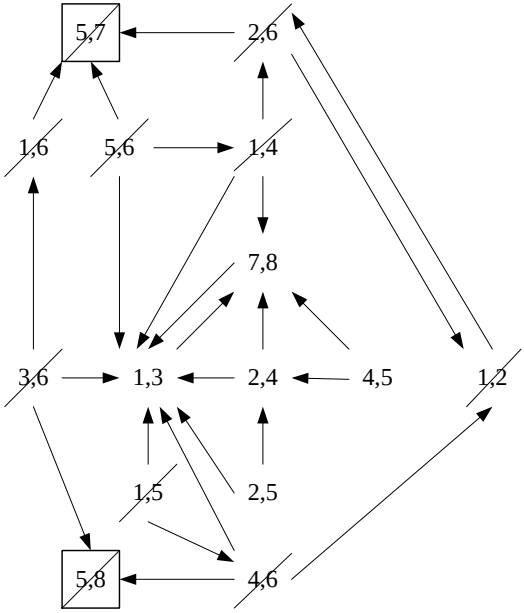
Конец. Не вычеркнутые пары суть пары неотличимых состояний.

Пример. Найдем все пары неотличимых состояний автомата $A=<X,Q,Y,\psi,\varphi>$ из предыдущего примера.

				ψ				
$X\backslash Q$	1	2	3	4	5	6	7	8
a	7	7	8	8	7	5	1	3
b	6	2	6	2	4	1	8	8
c	1	1	3	3	3	1	3	1

	φ							
*X**Q*	1	2	3	4	5	6	7	8
a	0	0	0	0	0	0	1	1
b	1	1	1	1	1	1	0	0
c	0	0	0	0	0	0	0	0

Граф условий неотличимости имеет вид



Пара (5,7) и (5,8), обведенные в рамку, имеют разные столбцы таблицы выходов, вычеркиваем эти пары. Вычеркиваем также пары, из которых исходят дуги, заходящие в вычеркнутые пары: (2,6), (1,6), (5,6), (3,6), (4,6), а затем и (1,2), (1,4), (1,5). Остальные пары состояний неотличимы: (1,3), (7,8), (2,4), (2,5), (4,5).

Замечание. Отношение неотличимости обладает свойством транзитивности, то есть

$$p = q,\ p = s \Rightarrow q = s;$$

$$p = q,\ p \neq s \Rightarrow q \neq s.$$

Использование этого свойства может сократить процесс проверки пар состояний на неотличимость.

2.1.2. Приведенная форма автомата

Вернемся к задаче минимизации полного автомата. Из определения неотличимых состояний следует, что поведение автомата в этих состояниях для любой входной последовательности одинаково. Поэтому для минимизации полного автомата достаточно объединить в одно состояние все неотличимые состояния.

Поставим формально задачу минимизации полного автомата. Рассмотрим два автомата с одинаковыми входным и выходным алфавитами $A_1 =< X, Q_1, Y, \psi_1, \varphi_1 >$ и $A_2 =< X, Q_2, Y, \psi_2, \varphi_2 >$.

Определение. Автоматы A_1 и A_2 *эквивалентны*, если для любого состояния автомата A_1 найдется неотличимое состояние автомата A_2, и наоборот, для любого состояния автомата A_2 найдется неотличимое состояние автомата A_1.

Задачу минимизации полного автомата можно сформулировать так: построить автомат, эквивалентный данному, с минимальным числом состояний.

Определение. Автомат $A =< X, Q, Y, \psi, \varphi >$ называется *приведенным*, если любые два состояния в нем отличимы.

Определение. Приведенный автомат, эквивалентный заданному автомату A, называется *приведенной формой* автомата A и обозначается $\overline{A}$.

Теорема. Приведенная форма автомата A есть автомат с минимальным числом состояний, эквивалентный автомату A.

Доказательство. Рассмотрим автомат $A =< X, Q, Y, \psi, \varphi >$, его приведенную форму $\overline{A} =< X, \overline{Q}, Y, \overline{\psi}, \overline{\varphi} >$, и произвольный автомат $\tilde{A} =< X, \tilde{Q}, Y, \tilde{\psi}, \tilde{\varphi} >$, эквивалентный автомату A. Поскольку отношение неотличимости является транзитивным, нетрудно показать, что автоматы $\overline{A}$ и $\tilde{A}$ также эквивалентны, т.е. для каждого состояния автомата $\overline{A}$ найдется неотличимое состояние автомата $\tilde{A}$. Рассмотрим два различных состояния $q, p \in \overline{Q}$, для них в автомате $\tilde{A}$ найдутся состояния $\tilde{q}, \tilde{p} \in \tilde{Q} : q = \tilde{q}, p = \tilde{p}$. Поскольку все состояния автомата $\overline{A}$ отличимы, то $\tilde{q} \neq \tilde{p}$. Следовательно, число состояний в автомате $\tilde{A}$ не меньше числа состояний в автомате $\overline{A}$. ■

Итак, минимизировать полный автомат – это значит найти его приведенную форму. Рассмотрим процесс ее построения.

Отношение неотличимости на множестве Q, как нетрудно убедиться, является отношением эквивалентности, и, следовательно, разбивает Q на классы эквивалентности, называемые *классами неотличимости*. Пусть $P=\{S_1,...,S_r\}$ – разбиение Q на классы неотличимости. Построим автомат $\bar{A}=<X,\bar{Q},Y,\bar{\psi},\bar{\varphi}>$, где

- $\bar{Q}=\ S_1,...,S_r\ ;$
- $\forall x\in X, S_i\in Q': \bar{\psi}\ x,S_i\ =S_j\ \Leftrightarrow \forall s\in S_i:\psi\ x,s\ \in S_j\ ;$
- $\forall x\in X, S_i\in \bar{Q}: \bar{\varphi}\ x,S_i\ =\varphi\ x,q\ ,\forall q\in S_i\ .$

Определение является корректным, так как столбцы таблицы выходов неотличимых состояний одинаковы, а столбцы таблицы переходов отличаются только символами неотличимых состояний, то есть все состояния из одного класса неотличимости по одному и тому же входному символу переходят в один класс. Из определения автомата $\bar{A}$ видно, что любые его два состояния отличимы, поскольку неотличимые состояния принадлежат одному из классов неотличимости. Автоматы $\bar{A}$ и A эквивалентны, так как состояние $q\in Q$ неотличимо от состояния $S_i\in\bar{Q}$, такого, что $q\in S_i$, и наоборот, $S_i\in Q'$ неотличимо от любого состояния $q\in Q$, такого, что $q\in S_i$. Окончательно получаем, что автомат $\bar{A}$ есть приведенная форма автомата A.

Пример. Рассмотрим автомат из предыдущего примера, для него найдены пары неотличимых состояний:

$$1=3,\quad 2=4,\quad 2=5,\quad 4=5,\quad 7=8\,.$$

Отсюда разбиение на классы неотличимости имеет вид:

$$1'=\ 1,3\ ,\quad 2'=\ 2,4,5\ ,\quad 3'=\ 6\ ,\quad 4'=\ 7,8\ .$$

Сгруппируем состояния в таблицах переходов и выходов по классам, получим

				ψ				
*X**Q*	1	3	2	4	5	6	7	8
a	7	8	7	8	7	5	1	3
b	6	6	2	2	4	1	8	8
c	1	3	1	3	3	1	3	1

				φ				
$X\backslash Q$	1	3	2	4	5	6	7	8
a	0	0	0	0	0	0	1	1
b	1	1	1	1	1	1	0	0
c	0	0	0	0	0	0	0	0

Построим автомат $\overline{A}$ по определению. Объединим столбцы состояний из одного класса неотличимости в один столбец, заменив номера состояний обозначениями классов неотличимости.

	$\overline{\psi}$			
$X\backslash Q$	$1'$	$2'$	$3'$	$4'$
a	$4'$	$4'$	$2'$	$1'$
b	$3'$	$2'$	$1'$	$4'$
c	$1'$	$1'$	$1'$	$1'$

	$\overline{\varphi}$			
$X\backslash Q$	$1'$	$2'$	$3'$	$4'$
a	0	0	0	1
b	1	1	1	0
c	0	0	0	0

Этот автомат является приведенной формой исходного автомата.

Таким образом, для минимизации автомата необходимо построение классов неотличимости. Поиск их с помощью графа условий неотличимости может оказаться достаточно громоздким, поэтому рассмотрим другие алгоритмы.

2.1.3. Построение классов неотличимости: алгоритм Мура

Пусть задан автомат $A=<X,Q,Y,\psi,\varphi>$ и натуральное число $k \geq 1$.

Определение. Состояния $q \in Q$ и $p \in Q$ *k–неотличимы* (обозначается $q \overset{k}{=} p$), если

$$\forall \alpha \in X^*, |\alpha| \leq k: \quad \varphi(\alpha, q) = \varphi(\alpha, p).$$

Иначе эти состояния называются *k–отличимыми* (обозначается $q \overset{k}{\neq} p$).

Лемма.

- Если $q \overset{k}{=} p$ и $l \leq k$, то $q \overset{l}{=} p$.
- Если $q \overset{k}{\neq} p$ и $l \geq k$, то $q \overset{l}{\neq} p$.
- Если $q \overset{k}{\neq} p$ и $q \neq p$, то $\exists q' \in Q, p' \in Q: \quad q' \overset{k}{=} p', q' \overset{k+1}{\neq} p'$.

Доказательство. Первые два утверждения очевидны. Докажем третье утверждение. Так как $q \neq p$, то $\exists \alpha \in X^*: \varphi(\alpha, q) \neq \varphi(\alpha, p)$.

Так как $q \overset{k}{=} p$, то $|\alpha| \geq k+1$. Из всех таких слов выберем слово α минимальной длины и представим его в виде конкатенации $\alpha = \beta\gamma$, где $|\gamma| = k+1$. Рассмотрим состояния $q' = \psi(\beta, q)$, $p' = \psi(\beta, p)$. Убедимся, что эти состояния удовлетворяют лемме. Из выбора γ следует, что $\varphi(\gamma, q') = \varphi(\alpha, q)$, $\varphi(\gamma, p') = \varphi(\alpha, p)$, откуда следует, что $\exists\gamma, |\gamma| = k+1 : \varphi(\gamma, q') \neq \varphi(\gamma, p')$, а значит, $q' \overset{k+1}{\neq} p'$.

В том, что $q' \overset{k}{=} p'$, убедимся методом от противного. Предположим, что $q' \overset{k}{\neq} p'$, тогда $\exists\delta, |\delta| \leq k : \varphi(\delta, q') \neq \varphi(\delta, p')$. Рассмотрим слово $\alpha' = \beta\delta$. Очевидно, что $|\alpha'| \leq |\alpha|$. Тогда $\varphi(\alpha', q) = \varphi(\delta, q')$ и $\varphi(\alpha', p) = \varphi(\delta, p')$, откуда $\varphi(\alpha', q) \neq \varphi(\alpha', p)$, что противоречит выбору α, как кратчайшего отличающего слова для q и p. ■

Отношение *k*–неотличимости является отношением эквивалентности на множестве Q, а значит, порождает разбиение P_k множества Q на классы *k*–неотличимости.

Свойства разбиения P_k

- Если q и p принадлежат одному и тому же классу разбиения P_k и $l < k$, то q и p принадлежат одному и тому же классу разбиения P_l.
- Если q и p принадлежат разным классам разбиения P_k и $l > k$, то q и p принадлежат разным классам разбиения P_l.
- Если $P_k \neq P$, то P_{k+1} есть собственное (не совпадающее с P_k) подразбиение P_k.

Эти свойства означают, что всякое P_k есть подразбиение всякого P_l при $l < k$, и, если $P_k = P_{k+1}$, то $P_{k+1} = P$. Исходя из этого, разбиение P строится следующим образом. Последовательно строятся разбиения P_k. Как только на очередном шаге P_k совпадет с P_{k+1}, разбиение P найдено и совпадает с P_k.

Алгоритм Мура построения классов неотличимости

Начало. Задан автомат $A = < X, Q, Y, \psi, \varphi >$. Требуется найти раз-

биение множества Q на классы неотличимости.

Шаг 1. Построим разбиение P_1: состояния q и p поместим в один класс разбиения P_1 тогда и только тогда, когда столбцы q и p в таблице выходов одинаковы. Положим $k=1$.

Шаг 2. Построим разбиение P_{k+1} следующим образом: состояния q и p из одного класса разбиения P_k поместим в один класс разбиения P_{k+1} тогда и только тогда, когда для всех $x \in X$ состояния $\psi(x,q)$ и $\psi(x,p)$ принадлежат одному классу разбиения P_k.

Шаг 3. Если $P_k \neq P_{k+1}$, положим $k=k+1$ и вернемся на шаг 2.

Конец. Искомое разбиение $P=P_k$.

Пример. Рассмотрим автомат из предыдущего примера

ψ

$X\backslash Q$	1	2	3	4	5	6	7	8
a	7	7	8	8	7	5	1	3
b	6	2	6	2	4	1	8	8
c	1	1	3	3	3	1	3	1

φ

$X\backslash Q$	1	2	3	4	5	6	7	8
a	0	0	0	0	0	0	1	1
b	1	1	1	1	1	1	0	0
c	0	0	0	0	0	0	0	0

Шаг 1. Разбиение P_1 имеет вид

$$1' = 1,2,3,4,5,6\ , \quad 2' = 7,8\ .$$

Шаг 2. Сгруппируем состояния в таблице переходов в классы, состояния из различных классов выделим разным шрифтом.

ψ

$X\backslash Q$	1	2	3	4	5	6	**7**	**8**
a	**7**	**7**	**8**	**8**	**7**	5	1	3
b	6	2	6	2	4	1	**8**	**8**
c	1	1	3	3	3	1	3	1

Состояния $1,2,3,4,5$ остаются в одном классе, а состояние 6 образует новый класс, поскольку состояния $1,2,3,4,5$ по символу a переходят в класс $2'$, а состояние 6 – в класс $1'$. По остальным сим-

волам состояния $1,2,3,4,5,6$ переходят в один и тот же класс. Состояния $7,8$ остаются в одном классе. Разбиение P_2 имеет вид

$$1' = 1,2,3,4,5\ ,\quad 2' = 6\ ,\quad 3' = 7,8\ .$$

Шаг 3. Поскольку $P_2 \neq P_1$, полагаем $k = 2$ и идем на шаг 2.

Шаг 2. Сгруппируем состояния в таблице переходов в классы, состояния из различных классов выделим разным шрифтом.

				ψ				
*X**Q*	1	2	3	4	5	*6*	**7**	**8**
a	**7**	**7**	**8**	**8**	**7**	5	1	3
b	*6*	2	*6*	2	4	1	**8**	**8**
c	1	1	3	3	3	1	3	1

Класс $1,2,3,4,5$ разбивается на два класса: $1,3$ и $2,4,5$, поскольку состояния $1,3$ по символу b переходят в класс $2'$, а состояния $2,4,5$ – в класс $1'$. Состояния $7,8$ остаются в одном классе. Разбиение P_3 имеет вид

$$1' = 1,3\ ,\quad 2' = 2,4,5\ ,\quad 3' = 6\ ,\quad 4' = 7,8\ .$$

Шаг 3. Поскольку $P_3 \neq P_2$, полагаем $k = 3$ и идем на шаг 2.

Шаг 2. Сгруппируем состояния в таблице переходов в классы, состояния из различных классов выделим разным шрифтом.

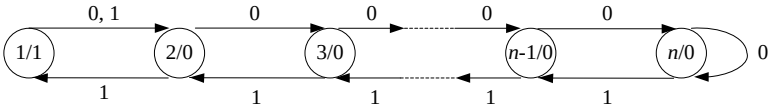
					ψ			
*X**Q*	1	3	2	4	5	*6*	**7**	**8**
a	**7**	**8**	**7**	**8**	**7**	5	1	3
b	*6*	*6*	2	2	4	1	**8**	**8**
c	1	3	1	3	3	1	3	1

Ни один класс больше не разбивается.

Шаг 3. Поскольку $P_4 = P_3$, получено искомое разбиение, которое совпадает с разбиением из подраздела 2.1.2.

Оценим вычислительную сложность алгоритма Мура. Каждый раз, кроме последнего, при выполнении шага 2 число классов неотличимости увеличивается. Так как максимальное число классов неотличимости равно n – числу состояний исходного автомата A, а на шаге 1 мы получим как минимум один класс неотличимости, то максимальное число итераций алгоритма будет $n-1$.

Пример. Рассмотрим автомат Мура, для которого число итераций алгоритма будет в точности равно $n-1$.



Разбиение P_1 состоит из двух классов – 1 и $2,\ldots,n$. На первой итерации алгоритма отдельный класс образует состояние 2, на второй – состояние 3, и т.д.

Примем за сложность алгоритма число вычислений функции $\psi\ \cdot$. На каждой итерации мы вычисляем значения $\psi(x,q)$ в худшем случае для всех x и q. Пусть $|X|=m$, $|Q|=n$, тогда сложность не превышает $mn(n-1)$. При больших n эта оценка имеет порядок n^2.

2.1.4. Построение классов неотличимости: алгоритм Хопкрофта

Рассмотрим алгоритм, более эффективный в вычислительном плане, чем алгоритм Мура, но, возможно, более сложный для понимания.

Определение. *Обратной функцией переходов* конечного автомата назовем функцию вида

$$\psi^{-1}\ x,q\ =\ p:\psi\ x,p\ =q\ ,$$

т.е. значением обратной функции переходов является множество состояний, которые переходят по заданному символу в заданное состояние.

Пример. Рассмотрим автомат из предыдущего подраздела

	ψ							
$X\backslash Q$	1	2	3	4	5	6	7	8
a	7	7	8	8	7	5	1	3
b	6	2	6	2	4	1	8	8
c	1	1	3	3	3	1	3	1

Обратная функция переходов для него имеет вид

	ψ^{-1}							
$X\backslash Q$	1	2	3	4	5	6	7	8
a	7	$\varnothing$	8	$\varnothing$	6	$\varnothing$	1,2,5	3,4
b	6	2,4	$\varnothing$	5	$\varnothing$	1,3	$\varnothing$	7,8
c	1,2,6,8	$\varnothing$	3,4,5,7	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

Обратная функция переходов очевидным образом может быть

распространена на множества:

$$\psi^{-1}(x,S) = \bigcup_{q \in S} \psi^{-1}(x,q) .$$

Идея алгоритма Хопкрофта состоит в следующем. Разбиение на классы неотличимости обладает следующими двумя свойствами:

- столбцы таблицы выходов состояний из одного блока разбиения одинаковы;
- по любому входному символу все состояния одного блока переходят в состояния, также принадлежащие одному и тому же блоку.

Начиная с некоторого стартового разбиения (например, можно взять разбиение P_1 из алгоритма Мура) строится разбиение, обладающее такими свойствами. Для этого рассматриваются поочередно пары, состоящие из входного символа и блока разбиения, и каждый блок разбивается на два по следующему правилу: в одно множество помещаются состояния, которые переходят по заданному символу в заданный блок, в другое – остальные состояния. Процесс продолжается, пока порождаются новые разбиения.

Алгоритм Хопкрофта

Начало. Задан автомат $A = < X, Q, Y, \psi, \varphi >$ и некоторое начальное разбиение $\tilde{P}_1 = \{S_1, ..., S_m\}$. Требуется найти разбиение множества Q на классы неотличимости.

Шаг 1. Для каждого входного символа x строим список $L(x)$, куда включаем все блоки разбиения $\tilde{P}_1$, для которых $\psi^{-1}(x, S_i) \neq \varnothing$. Удаляем из списка блок, для которого множество $\psi^{-1}(x, S_i)$ имеет максимальную мощность. Полагаем $k = 1$.

Шаг 2. Если все списки $L(x)$ пусты, идем на конец. Иначе выбираем пару (x, S_i), $S_i \in L(x)$, где x – любой входной символ, $\psi^{-1}(x, S_i)$ – множество минимальной мощности для всех блоков из $L(x)$. Удаляем S_i из $L(x)$.

Шаг 3. Выбираем очередной блок $S_j \in \tilde{P}_k$ и проводим его разбиение по паре (x, S_i). Блок S_j делится на два блока: S'_j и S''_j по следующему правилу: в S'_j помещаем состояния из S_j, которые по

символу x переходят в блок S_i, в S_j'' – остальные состояния из S_j:

$$S_j' = \{q \in S_j : q \in \psi^{-1}(x, S_i)\};$$

$$S_j'' = \{q \in S_j : q \notin \psi^{-1}(x, S_i)\}.$$

Шаг 4. Если S_j не разделился, добавляем его в разбиение $\tilde{P}_{k+1}$. Иначе добавляем в разбиение $\tilde{P}_{k+1}$ блоки S_j' и S_j'' и модифицируем все списки $L(z)$:

- если блок $S_j \in L(z)$, то удаляем его из $L(z)$ и добавляем в $L(z)$ блоки S_j' и S_j'';
- если блок $S_j \notin L(z)$, то добавляем в $L(z)$ блок S_j' или S_j'', а именно тот, для которого обратная функция переходов имеет меньшую мощность, но не является пустым множеством.

Если не все блоки $S_j \in \tilde{P}_k$ просмотрены, идем на шаг 3.

Шаг 5. Если $\tilde{P}_{k+1} \neq \tilde{P}_k$, полагаем $k = k+1$. Идем на шаг 2.

Конец. Искомое разбиение $P = \tilde{P}_k$.

Пример. Рассмотрим автомат из предыдущих примеров.

ψ

$X\backslash Q$	1	2	3	4	5	6	7	8
a	7	7	8	8	7	5	1	3
b	6	2	6	2	4	1	8	8
c	1	1	3	3	3	1	3	1

φ

$X\backslash Q$	1	2	3	4	5	6	7	8
a	0	0	0	0	0	0	1	1
b	1	1	1	1	1	1	0	0
c	0	0	0	0	0	0	0	0

Для него уже получена обратная функция переходов

ψ^{-1}

$X\backslash Q$	1	2	3	4	5	6	7	8
a	7	$\varnothing$	8	$\varnothing$	6	$\varnothing$	1,2,5	3,4
b	6	2,4	$\varnothing$	5	$\varnothing$	1,3	$\varnothing$	7,8
c	1,2,6,8	$\varnothing$	3,4,5,7	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

Шаг 1. Разбиение $\tilde{P}_1$ построим, как в алгоритме Мура

$$S_1 = 1,2,3,4,5,6\ ,\quad S_2 = 7,8\ .$$

Обратная функция переходов для этих блоков

ψ^{-1}

$X \backslash \tilde{P}_1$	S_1	S_2
a	6,7,8	1,2,3,4,5
b	1,2,3,4,5,6	7,8
c	1,2,3,4,5,6,7,8	$\varnothing$

Списки имеют вид:

$$L\ a\ : S_1,\quad L\ b\ : S_2,\quad L\ c\ : \varnothing.$$

Шаг 2. Выбираем пару a, S_1 , где $\psi^{-1}\ a, S_1 = 6,7,8$, удаляем S_1 из $L\ a$. Списки принимают вид

$$L\ a\ : \varnothing,\quad L\ b\ : S_2,\quad L\ c\ : \varnothing.$$

Шаг 3. Выбираем блок $S_1 \in \tilde{P}_1$ и разбиваем его по паре a, S_1 :

$$S_1' = 6\ ;$$

$$S_1'' = 1,2,3,4,5\ .$$

Шаг 4. Добавляем в разбиение $\tilde{P}_2$ блоки S_1' и S_1'' и модифицируем все списки. Находим обратную функцию переходов для блоков S_1' и S_1''

ψ^{-1}

$X \backslash \tilde{P}_2$	S_1'	S_1''
a	6,7,8	$\varnothing$
b	2,4,5,6	1,3
c	1,2,3,4,5,6,7,8	$\varnothing$

Поскольку S_1 не содержится ни в одном списке, то добавляем в список $L\ b$ блок S_1'', т.к. $\left|\psi^{-1}\ b, S_1''\right| < \left|\psi^{-1}\ b, S_1'\right|$, и не меняем остальные списки, поскольку $\psi^{-1}\ a, S_1'' = \psi^{-1}\ c, S_1'' = \varnothing$. Имеем

$$L\ a\ : \varnothing,\quad L\ b\ : S_2, S_1'',\quad L\ c\ : \varnothing.$$

Не все блоки просмотрены, идем на шаг 3.

Шаг 3. Выбираем блок $S_2 \in \tilde{P}_1$ и разбиваем его по паре a, S_1 :

$$S_2' = \{7,8\};$$
$$S_2'' = \varnothing.$$

Шаг 4. Блок не разбился, добавляем в разбиение $\tilde{P}_2$ блок S_2. Все блоки разбиения $\tilde{P}_1$ просмотрены.

Шаг 5. Разбиение $\tilde{P}_2 \neq \tilde{P}_1$, далее рассматриваем это разбиение

$$S_1 = \{1,2,3,4,5\}, \quad S_2 = \{6\}, \quad S_3 = \{7,8\}.$$

Перепишем в новых обозначениях списки

$$L(a) : \varnothing, \quad L(b) : S_3, S_2, \quad L(c) : \varnothing.$$

Получим обратную функцию переходов

	ψ^{-1}		
$X \setminus \tilde{P}_2$	S_1	S_2	S_3
a	6,7,8	$\varnothing$	1,2,3,4,5
b	2,4,5,6	1,3	7,8
c	1,2,3,4,5,6,7,8	$\varnothing$	$\varnothing$

Идем на шаг 2.

Шаг 2. Выбираем пару (b, S_2), где $\psi^{-1}(b, S_2) = \{1,3\}$, удаляем S_2 из $L(b)$. Списки принимают вид

$$L(a) : \varnothing, \quad L(b) : S_3, \quad L(c) : \varnothing.$$

Шаг 3. Выбираем блок $S_1 \in \tilde{P}_2$ и разбиваем его по паре (b, S_2):

$$S_1' = \{1,3\};$$
$$S_1'' = \{2,4,5\}.$$

Шаг 4. Добавляем в разбиение $\tilde{P}_3$ блоки S_1' и S_1'' и модифицируем все списки. Находим обратную функцию переходов для блоков S_1' и S_1''

	ψ^{-1}	
$X \setminus \tilde{P}_2$	S_1'	S_1''
a	7,8	6
b	6	2,4,5
c	1,2,3,4,5,6,7,8	$\varnothing$

Новые списки принимают вид

$$L(a) : S_1'', \quad L(b) : S_3, S_1', \quad L(c) : \varnothing.$$

Не все блоки просмотрены, идем на шаг 3.

Шаг 3. Блок $S_2 \in \tilde{P}_2$ состоит из одного символа, поэтому дальше не разбивается. Выбираем блок S_3, разбиваем его по паре b, S_2 :

$$S_3' = \varnothing;$$
$$S_3'' = \{7,8\}.$$

Шаг 4. Добавляем в разбиение $\tilde{P}_3$ блоки S_2 и S_3. Все блоки просмотрены.

Шаг 5. Разбиение $\tilde{P}_3 \neq \tilde{P}_2$, далее рассматриваем это разбиение

$$S_1 = \{1,3\}, \quad S_2 = \{2,4,5\}, \quad S_3 = \{6\}, \quad S_4 = \{7,8\}.$$

Перепишем в новых обозначениях списки

$$L(a): S_2, \quad L(b): S_4, S_1, \quad L(c): \varnothing.$$

Получим обратную функцию переходов

ψ^{-1}

$X \setminus \tilde{P}_3$	S_1	S_2	S_3	S_4
a	7,8	6	$\varnothing$	1,2,3,4,5
b	6	2,4,5	1,3	7,8
c	1,2,3,4,5,6,7,8	$\varnothing$	$\varnothing$	$\varnothing$

Нетрудно убедиться, что никакие пары больше не порождают новых разбиений. Предоставляем это читателю.

Полученное разбиение совпадает с разбиением на классы неотличимости из предыдущих подразделов.

2.2. Минимизация частичных автоматов

2.2.1. Совместимость состояний

Пусть $f(x)$ – некоторая частичная функция. Введем обозначение $!f(a)$, которое будем использовать, если значение $f(a)$ определено.

Рассмотрим частичный автомат $A = \langle X, Q, Y, \psi, \varphi \rangle$. Распространим функции $\psi(\cdot)$ и $\varphi(\cdot)$ на пары $(\alpha, q) \in X^* \times Q$ следующим образом. Значения функций $\psi(x,q)$, $\varphi(x,q)$, где $x \in X$, возьмем из таблиц переходов и выходов автомата A. Для слов вида $\alpha = \beta x$, где $x \in X, \beta \in X^*$ имеем

$$!\psi(\beta x,q)\cap\psi(\beta x,q)=\psi(x,\psi(\beta,q)) \quad\Leftrightarrow\quad !\psi(\beta,q)\cap!\psi(x,\psi(\beta,q));$$
$$!\varphi(\beta x,q)\cap\varphi(\beta x,q)=\varphi(x,\psi(\beta,q)) \quad\Leftrightarrow\quad !\psi(\beta,q)\cap!\varphi(x,\psi(\beta,q)).$$

Другими словами, эти функции для частичного автомата определяются так же, как и для полного, там, где переходы частичного автомата определены: если автомат под воздействием слова пробегает цепочку состояний, то значения функций переходов и выходов на данном слове – соответственно последнее состояние этой цепочки и последний генерируемый автоматом выходной символ. Если же какой-либо переход в цепочке не определен, то значения функций переходов и выходов на данном слове не определены.

Пример. Рассмотрим автомат, заданный таблицами переходов и выходов

ψ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	5	2	6	1	1	–	7
b	–	–	–	2	6	–	4	–
c	–	–	7	–	–	8	3	1
d	4	–	4	–	–	–	4	4

φ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	0	1	1	0	1	–	1
b	–	–	–	1	1	–	0	–
c	–	–	0	–	–	0	1	1
d	1	–	1	–	–	–	0	0

Здесь $\psi\ dba,1=5$, $\varphi\ dba,1=0$. Значения $\psi\ dbc,1$ и $\varphi\ dbc,1$ не определены, поскольку $\psi\ db,1=2$, а $\psi\ c,2$ не определено.

Рассмотрим два частичных автомата $A_1=<X,Q_1,Y,\psi_1,\varphi_1>$ и $A_2=<X,Q_2,Y,\psi_2,\varphi_2>$.

Определение. Состояния $q_1\in Q_1$, $q_2\in Q_2$ называются *совместимыми* (обозначается $q_1\approx q_2$), если

$$\forall\alpha\in X^*: \quad !\varphi_1(\alpha,q_1)\cap!\varphi_2(\alpha,q_2) \quad\Rightarrow\quad \varphi_2(\alpha,q_2)=\varphi_2(\alpha,q_2).$$

Иными словами, значения функций переходов автоматов в совместимых состояниях должны быть одинаковы там, где они определены. В частном случае, при $A_1=A_2=A$, говорят о совместимых состояниях автомата A.

Из определения совместимости состояний нетрудно вывести правила проверки состояний на совместимость:

1. Столбцы матрицы выходов совместимых состояний совпадают в тех строках, где обе функции выходов определены.
2. Соответствующие этим состояниям столбцы матрицы переходов должны быть равны или отличаться символами совместимых состояний в тех строках, где обе функции выходов определены.

Алгоритм проверки состояний на совместимость полностью аналогичен алгоритму проверки состояний полного автомата на неотличимость, поэтому здесь не приводится. Рассмотрим лишь пример поиска всех пар совместимых состояний.

Пример. Рассмотрим автомат из предыдущего примера

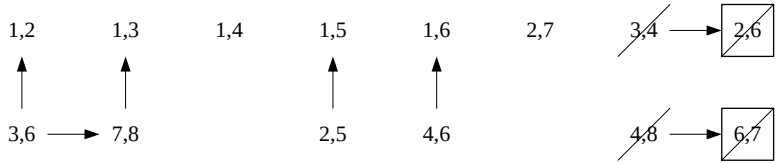
ψ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	5	2	6	1	1	–	7
b	–	–	–	2	6	–	4	–
c	–	–	7	–	–	8	3	1
d	4	–	4	–	–	–	4	4

φ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	0	1	1	0	1	–	1
b	–	–	–	1	1	–	0	–
c	–	–	0	–	–	0	1	1
d	1	–	1	–	–	–	0	0

Граф условий совместимости имеет вид



Пары (2,6) и (6,7), обведенные в рамку, не удовлетворяют первому условию, поэтому вычеркиваются. Далее вычеркиваются пары (3,4) и (4,8), производными от которых являются вычеркнутые пары. В результате получаем пары совместимых состояний:

(1,2), (1,3), (1,4), (1,5), (1,6), (2,5), (2,7), (3,6), (4,6), (7,8).

Отношение совместимости рефлексивно, симметрично, но не транзитивно, то есть не является отношением эквивалентности. Поэтому, в отличие от проверки отношения неотличимости для полного

автомата, необходимо проводить проверку отношения совместимости для всех пар, удовлетворяющих первому условию.

Пример. В предыдущем примере $1 \approx 2$ и $1 \approx 3$, но состояния 2 и 3 несовместимы.

Определение. Любое подмножество множества Q, в котором все состояния попарно совместимы, называется *множеством совместимости*.

Определение. Любое подмножество множества Q, в котором нет ни одной пары совместимых состояний, называется *множеством несовместимости*.

Далее будет показано, что для решения задачи минимизации частичного автомата требуются эти множества.

Алгоритм построения множеств совместимости

Начало. Пусть $A =< X, Q, Y, \psi, \varphi >$ – частичный автомат. Требуется построить все множества совместимости.

Шаг 1. Если $Q = \{1, ..., n\}$, то $\{1\}, ..., \{n\}$ суть множества совместимости. Положим $k = 1$.

Шаг 2. Предположим, что все множества совместимости мощности $k \geq 1$ построены. Построим множества совместимости мощности $k+1$. Пусть $S_1, ..., S_l$ суть множества совместимости и $|S_i| = k, i = \overline{1, l}$. Для всякого S_i найдем все такие состояния $q \notin S_i$, которые по отдельности совместимы с любым состоянием из S_i. Множества $S_i \cup \{q\}$ будут новыми множествами совместимости.

Шаг 3. Если список множеств совместимости мощности $k+1$ не пуст, положим $k = k+1$ и перейдем на шаг 2.

Конец.

Пример. Для автомата из предыдущего примера найдены совместимые пары. Список множеств совместимости для него

$k = 1$: 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 ;

$k = 2$: 1,2 , 1,3 , 1,4 , 1,5 , 1,6 , 2,5 , 2,7 , 3,6 , 4,6 , 7,8 ;

$k = 3$: 1,2,5 , 1,3,6 , 1,4,6 .

Множества несовместимости могут быть найдены аналогично.

2.2.2. Сохраняемое правильное покрытие

Идея минимизации частичного автомата состоит в том, что со-

вместимые состояния объединяются в одно состояние. Главное отличие задачи минимизации частичного автомата от такой же задачи для полного автомата состоит в том, что отношение совместимости, в отличие от отношения неотличимости, не является транзитивным. Это приводит к тому, что это отношение не является отношением эквивалентности, а значит, не порождает разбиения. Поэтому для минимизации частичного автомата строится сохраняемое правильное покрытие множества состояний. Отличие покрытия от разбиения состоит в том, что одно и то же состояние может входить в несколько блоков покрытия.

Определение. Для любых функций $f\cdot$ и $g\cdot$ будем говорить, что $f\cdot$ есть *доопределение* $g\cdot$, если

$$!g(a) \Rightarrow !f(a) \cap f(a) = g(a).$$

Рассмотрим два в общем случае частичных автомата: $A =< X, Q, Y, \psi, \varphi >$ и $A' =< X, Q', Y, \psi', \varphi' >$.

Определение. Автомат A' *продолжает* автомат A, если для любого состояния $q \in Q$ автомата A существует состояние $q' \in Q'$ в автомате A', такое, что $\varphi'\ \alpha, q'$ является доопределением функции $\varphi\ \alpha, q$.

Задача минимизации частичного автомата ставится следующим образом: найти автомат A' с наименьшим числом состояний, продолжающий A.

Замечание. Для всюду определенных автоматов понятие продолжимости совпадает с понятием эквивалентности.

Определение. *Покрытием* множества Q называется всякая система подмножеств $S_1, ..., S_m \in Q$ такая, что $\bigcup_{i=1}^{m} S_i = Q$. Сами подмножества $S_1, ..., S_m$ называются *блоками покрытия*.

Определение. Покрытие $\Pi = S_1, ..., S_m$ множества Q называется *сохраняемым* в $A =< X, Q, Y, \psi, \varphi >$, если

$$\forall x \in X,\ \forall S_i \in \Pi:\ \ \exists S_j \in \Pi : !\psi(x, S_i) \Rightarrow \psi(x, S_i) \in S_j.$$

Здесь считаем, что $!\psi(x, S_i) \Leftrightarrow \exists q \in S_i : !\psi(x, q)$.

Это означает, что все состояния из блока S_i, функции переходов которых по символу x определены, переходят по этому символу в

один и тот же блок покрытия.

Определение. Сохраняемое покрытие $\Pi = S_1,...,S_m$ множества Q называется *правильным*, если

$$\forall x \in X,\ \forall S_i \in \Pi,\ \forall q, p \in S_i: \quad !\varphi(x, p) \cap !\varphi(x, q) \Rightarrow \varphi(x, p) = \varphi(x, q).$$

Это означает, что значения функции выходов состояний одного блока по любому символу x одинаковы для всех состояний, для которых они определены.

Теорема. Автомат A' с m состояниями продолжает автомат A тогда и только тогда, когда в автомате A есть сохраняемое правильное покрытие мощности m.

Доказательство.

Необходимость. Пусть автомат $A' =< X, Q', Y, \psi', \varphi' >$ продолжает автомат $A =< X, Q, Y, \psi, \varphi >$. Разобьем множество состояний автомата A на блоки $S_1,...,S_m$, так, что $q \in S_i$ тогда и только тогда, когда $\varphi'(\alpha, q_i')$ является доопределением $\varphi(\alpha, q)$. Покрытие $\Pi = S_1,...,S_m$ является правильным, поскольку для любого слова α, а значит и для любого символа x и состояния $q \in S_i$

$$!\varphi(\alpha, q) \Rightarrow !\varphi'(\alpha, q_i') \cap \varphi(\alpha, q) = \varphi'(\alpha, q_i'),$$

т.е. значения функции выхода в состояниях блока S_i одинаковы для тех состояний, для которых они определены. Покажем сохраняемость покрытия. Рассмотрим два состояния из одного блока $q, p \in S_i$ и любой символ x, для которого определены функции переходов $\psi(x, q)$ и $\psi(x, p)$. Пусть $\tilde{q} = \psi(x, q)$ и $\tilde{p} = \psi(x, p)$. Рассмотрим любое слово α, для которого определены функции выходов $\varphi(\alpha, \tilde{q})$ и $\varphi(\alpha, \tilde{p})$. Очевидно, что

$$\varphi(\alpha, \tilde{q}) = \varphi(x\alpha, q) = \varphi(x\alpha, p) = \varphi(\alpha, \tilde{p}).$$

Следовательно, состояния $\tilde{q}$ и $\tilde{p}$ имеют общее доопределение, а значит, принадлежат одному и тому же блоку S_j покрытия Π. Таким образом, любые состояния из одного блока покрытия Π по любому символу переходят в состояния из одного блока, если функции переходов определены, и покрытие является сохраняемым.

Достаточность. Пусть $\Pi = S_1,...,S_m$ есть сохраняемое правильное покрытие в автомате $A =< X, Q, Y, \psi, \varphi >$. Построим по по-

крытию Π автомат $A' =< X, Q', Y, \psi', \varphi' >$, продолжающий A, следующим образом. Положим

- $Q' = \Pi = \{S_1, ..., S_m\}$;
- $!\psi'(x, Q_i) \Leftrightarrow !\psi(x, Q_i)$, $\psi'(x, S_i) = S_j \supseteq \psi(x, S_i)$;
- $!\varphi'(x, S_i) \Leftrightarrow \exists q \in S_i : !\varphi(x, q), \varphi'(x, S_i) = \varphi(x, q)$.

В силу сохраняемости и правильности Π такое построение корректно. Так как $\forall S_i \in \Pi$ и $\forall q \in S_i$ функция $\varphi'\ x, S_i$ является доопределением $\varphi\ x, q$, то A' продолжает A. ■

Определение. Правильное сохраняемое в автомате A покрытие наименьшей мощности называется *кратчайшим*.

Автомат A', продолжающий автомат A, имеет наименьшее число состояний из всех автоматов, продолжающих A, если он построен по кратчайшему сохраняемому в A правильному покрытию. Таким образом, задача минимизации частичного автомата A сводится к задаче нахождения кратчайшего сохраняемого в A правильного покрытия.

Лемма. Всякий блок в сохраняемом правильном покрытии есть множество совместимости.

Доказательство. Рассмотрим $\alpha \in X^*$, $S_i \in \Pi$ и $q, p \in S_i$ такие, что $!\varphi(\alpha, q)$ и $!\varphi(\alpha, p)$. Пусть $\alpha = \beta x$, тогда $!\psi(\beta, q) = q'$ и $!\psi(\beta, p) = p'$, а также $!\varphi(x, q')$ и $!\varphi(x, p')$. Из сохраняемости Π следует, что $q', p' \in S_j \in \Pi$, а из правильности – $\varphi(x, q') = \varphi(x, p')$, то есть $\varphi(\alpha, q) = \varphi(\alpha, p)$, то есть состояния p, q совместимы. ■

Лемма. Мощность кратчайшего сохраняемого правильного покрытия не превосходит мощности максимального множества несовместимости.

Доказательство. Несовместимые состояния находятся в разных блоках покрытия. ■

На этих леммах основан следующий алгоритм.

Алгоритм построения кратчайшего сохраняемого правильного покрытия

Начало. Задан частичный автомат $A =< X, Q, Y, \psi, \varphi >$. Требуется построить кратчайшее сохраняемое правильное покрытие.

Шаг 1. Строим все множества совместимости и максимальное множество несовместимости. Если максимальное множество несовместимости построить трудно, то строим любое множество несо-

вместимости. Пусть r – мощность построенного множества несовместимости.

Шаг 2. Из множеств совместимости образуем последовательно покрытия мощности r, $r+1$, ..., $n-1$ и каждое из них проверяем на сохраняемость. Первое в порядке получения сохраняемое покрытие есть искомое кратчайшее сохраняемое правильное покрытие.

Конец.

Объем перебора при работе данного алгоритма довольно велик: он задается величиной $C_l^r + C_l^{r+1} + ... + C_l^{n-1}$, где l – мощность списка множеств совместимости.

Пример. Рассмотрим частичный автомат $A = < X, Q, Y, \psi, \varphi >$, заданный таблицами переходов и выходов

ψ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	5	2	6	1	1	–	7
b	–	–	–	2	6	–	4	–
c	–	–	7	–	–	8	3	1
d	4	–	4	–	–	–	4	4

φ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	0	1	1	0	1	–	1
b	–	–	–	1	1	–	0	–
c	–	–	0	–	–	0	1	1
d	1	–	1	–	–	–	0	0

Для него в подразделе 2.2.1 уже частично выполнен шаг 1 – найдены пары совместимых состояний. Список множеств совместимости имеет вид

$k = 1$: 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 ;

$k = 2$: 1,2 , 1,3 , 1,4 , 1,5 , 1,6 , 2,5 , 2,7 , 3,6 , 4,6 , 7,8 ;

$k = 3$: 1,2,5 , 1,3,6 , 1,4,6 .

Множество несовместимости, например, имеет вид 2,3,4,7 .

Рассмотрим покрытие

$$1' = 1,2,5 , \quad 2' = 3,6 , \quad 3' = 4 , \quad 4' = 7,8 .$$

Проверим покрытие на сохраняемость. Для этого сгруппируем в таблице переходов состояния по блокам. состояния, принадлежащие

разным блокам, выделим разным шрифтом.

				ψ				
	1′			2′		3′	4′	
*X**Q*	1	2	5	*3*	*6*	4	7	8
a	–	5	1	2	1	*6*	–	7
b	–	–	*6*	–	–	2	4	–
c	–	–	–	7	8	–	*3*	1
d	4	–	–	4	–	–	4	4

Как видно из таблицы, покрытие не является сохраняемым, поскольку по символу c состояния 7 и 8 переходят в разные блоки покрытия.

Рассмотрим другое покрытие

$$1' = 1,2,5\ ,\quad 2' = 1,3,6\ ,\quad 3' = 4\ ,\quad 4' = 7,8\ .$$

Проверим его на сохраняемость.

					ι				
	1′			2′			3′	4′	
*X**Q*	1	2	5	*1*	*3*	*6*	4	7	8
a	–	5	1	–	2	1	*6*	–	7
b	–	–	*6*	–	–	–	2	4	–
c	–	–	–	–	7	8	–	*3*	*1*
d	4	–	–	4	4	–	–	4	4

Покрытие является сохраняемым. Построим по нему автомат, объединив столбцы матриц переходов и выходов для каждого блока покрытия, как показано в доказательстве достаточности теоремы.

		ψ					φ		
*X**Q*	1′	2′	3′	4′	*X**Q*	1′	2′	3′	4′
a	1′	1′	2′	4′	*a*	0	1	1	1
b	2′	–	1′	3′	*b*	1	–	1	0
c	–	4′	–	2′	*c*	–	0	–	1
d	3′	3′	–	3′	*d*	1	1	–	0

Данный пример еще раз подтверждает, сколь велик перебор при построении кратчайшего сохраняемого правильного покрытия, поэтому на практике часто используются приближенные алгоритмы. Рассмотрим один из них.

2.2.3. Метод последовательных сокращений

Определение. Двухэлементное множество совместимости называется *совместимой парой*.

Определение (индуктивное). *Цепь* $C(q,p)$*, порождаемая парой* (q,p) – это множество пар состояний, построенной следующим образом:

База индукции: пара $(q,p)\in C(q,p)$.

Индуктивный переход: если пара $(r,s)\in C(q,p)$, $x\in X$ и $!\psi(x,r)\cap!\psi(x,s)$, и если при этом $\psi(x,r)\neq\psi(x,s)$, то пара $(\psi(x,r),\psi(x,s))\in C(q,p)$.

Заключительная фраза: других пар в $C(q,p)$ нет.

Пример. Рассмотрим частичный автомат из предыдущих примеров $A=<X,Q,Y,\psi,\varphi>$, заданный таблицами переходов и выходов

ψ

$X\backslash Q$	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	5	2	6	1	1	–	7
b	–	–	–	2	6	–	4	–
c	–	–	7	–	–	8	3	1
d	4	–	4	–	–	–	4	4

φ

$X\backslash Q$	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	0	1	1	0	1	–	1
b	–	–	–	1	1	–	0	–
c	–	–	0	–	–	0	1	1
d	1	–	1	–	–	–	0	0

Рассмотрим пару состояний (7,8) и построим цепь $C(7,8)$. Из таблицы переходов видно, что условие $!\psi(x,7)\cap!\psi(x,8)$ выполняется при $x=c$ и $x=d$. При этом условие $\psi(x,7)\neq\psi(x,8)$ верно только при $x=c$: $\psi(c,7)=3$, $\psi(c,8)=1$, то есть пара $(1,3)\in C(7,8)$. Рассмотрим пару (1,3). Условие $!\psi(x,1)\cap!\psi(x,3)$ выполняется только при $x=d$, при этом $\psi(d,1)=\psi(d,3)=4$, то есть мы получили пару одинаковых состояний. Следовательно, $C(7,8)=\{(7,8),(1,3)\}$.

Замечание. Цепь $C(q,p)$ можно получить из графа условий совместимости – в нее включаются все пары, достижимые из (q,p).

Определение. Состояния q,p называются 1–*совместимыми*, если и только если

$$\forall x \in X: \quad !\varphi(x,q) \cap !\varphi(x,p) \quad \Rightarrow \quad \varphi(x,q) = \varphi(x,p).$$

Пример. В предыдущем примере состояния (3,4) являются 1–совместимыми, так как условие $!\varphi(x,3) \cap !\varphi(x,4)$ выполнено только для $x = a$, и при этом $\varphi(a,3) = \varphi(a,4) = 1$, а состояния (2,3) не являются, так как $\varphi(a,2) = 0$, а $\varphi(a,3) = 1$.

Определение. Цепь называется *правильной*, если все пары в ней 1-совместимы.

Пример. Цепь $C(7,8) = \{(7,8),(1,3)\}$ является правильной, а цепь $C(3,4) = \{(3,4),(2,6),(1,5)\}$ – не является, так как состояния 2 и 6 несовместимы.

По цепи можно построить покрытие. Его блоками являются все пары, вошедшие в цепь, и состояния, не вошедшие ни в одну из пар, поодиночке.

Пример. Покрытие, построенное по цепи $C(7,8) = \{(7,8),(1,3)\}$

$1' = 1,3$, $2' = 7,8$, $3' = 2$, $4' = 4$, $5' = 5$, $6' = 6$.

Лемма. Покрытие, построенное по цепи, является сохраняемым правильным покрытием.

Доказательство. Сохраняемость следует из построения цепи – если пара состояний переходит в другую пару, то эта другая пара добавляется в цепь, а значит, является блоком покрытия. Правильность следует из того, что все пары состояний в правильной цепи 1-совместимы. ■

Из леммы следует приближенный алгоритм построения кратчайшего сохраняемого правильного покрытия. Алгоритм не гарантирует, что покрытие будет кратчайшим, но является несложным в вычислительном плане, поскольку не требует проверки покрытий на сохраняемость, а значит, и перебора покрытий.

Алгоритм последовательных сокращений

Начало. Задан частичный автомат $A = < X, Q, Y, \psi, \varphi >$. Требуется построить кратчайшее сохраняемое правильное покрытие. Положим $l = n = |Q|$

Шаг 1. Выберем очередную пару 1-совместимых состояний $q, p \in Q$. Если такие пары исчерпаны, перейдем на шаг 3. Иначе построим цепь $C(q,p)$. Если она неправильная, повторим шаг 1.

Шаг 2. Построим покрытие $\prod$ по цепи $C(q,p)$. Пусть мощность

покрытия $|\prod| = m$. Если $m < l$, положим $l = m$ и запомним покрытие $\prod$. Вернемся на шаг 1.

Шаг 3. Если $l < n = |Q|$, то по покрытию $\prod$ построим автомат $A =< X, Q', Y, \psi', \varphi' >$ и вернемся с ним на шаг 1.

Конец.

Пример. Пример. Рассмотрим частичный автомат из предыдущих примеров $A =< X, Q, Y, \psi, \varphi >$, заданный таблицами переходов и выходов

ψ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	5	2	6	1	1	–	7
b	–	–	–	2	6	–	4	–
c	–	–	7	–	–	8	3	1
d	4	–	4	–	–	–	4	4

φ

X\Q	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	–	0	1	1	0	1	–	1
b	–	–	–	1	1	–	0	–
c	–	–	0	–	–	0	1	1
d	1	–	1	–	–	–	0	0

Построим все возможные цепи, порождаемые 1–совместимыми состояниями автомата. Выделим пары состояний в цепях, которые не являются 1-совместимыми.

- $C(1,2) = \{(1,2)\}$;
- $C(1,3) = \{(1,3)\}$;
- $C(1,4) = \{(1,4)\}$;
- $C(1,5) = \{(1,5)\}$;
- $C(1,3) = \{(1,3)\}$;
- $C(1,6) = \{(1,6)\}$;
- $C(2,5) = \{(2,5),(1,5)\}$;
- $C(2,7) = \{(2,7)\}$;
- $C(3,4) = \{(3,4),\mathbf{(2,6)},(1,5)\}$;
- $C(3,6) = \{(3,6),(1,2),(7,8),(1,3)\}$;
- $C(4,6) = \{(4,6),(1,6)\}$;

- $C(4,8)=\{(4,8),\mathbf{(6,7)},\mathbf{(3,8)},(2,7),\mathbf{(1,7)}\}$;
- $C(7,8)=\{(7,8),(1,3)\}$.

Неправильными здесь являются цепи $C(3,4)$ и $C(4,8)$. Любая из остальных цепей подойдет для построения правильного сохраняемого покрытия. Наименьшая мощность покрытия получится, если взять, например, цепь $C(7,8)=\{(7,8),(1,3)\}$, тогда покрытие, построенное по этой цепи, имеет вид

$$1'=1,3\ ,\quad 2'=7,8\ ,\quad 3'=2\ ,\quad 4'=4\ ,\quad 5'=5\ ,\quad 6'=6\ .$$

Возможны и другие варианты выбора цепи.

Построим по этому покрытию новый автомат. Рассмотрим исходную таблицу переходов и выходов, сгруппировав столбцы в соответствии с блоками покрытия.

ψ

	1′		2′		3′	4′	5′	6′
*X**Q*	*1*	*3*	*7*	*8*	*2*	*4*	*5*	*6*
a	–	2	–	7	5	6	1	1
b	–	–	4	–	–	2	6	–
c	–	7	3	1	–	–	–	8
d	4	4	4	4	–	–	–	–

φ

	1′		2′		3′	4′	5′	6′
*X**Q*	*1*	*3*	*7*	*8*	*2*	*4*	*5*	*6*
a	–	1	–	1	0	1	0	1
b	–	–	0	–	–	1	1	–
c	–	0	1	1	–	–	–	0
d	1	1	0	0	–	–	–	–

Таблицы переходов и выходов нового автомата имеют вид.

ψ

*X**Q*	1′	2′	3′	4′	5′	6′
a	3′	2′	5′	6′	1′	1′
b	–	4′	–	3′	6′	–
c	2′	1′	–	–	–	2′
d	4′	4′	–	–	–	–

	φ					
$X\backslash Q$	$1'$	$2'$	$3'$	$4'$	$5'$	$6'$
a	1	1	0	1	0	1
b	–	0	–	1	1	–
c	0	1	–	–	–	0
d	1	0	–	–	–	–

Строим для этого автомата все возможные цепи, порождаемые 1–совместимыми состояниями. При этом прекращаем построение цепи, как только в ней появилась не 1-совместимая пара, поскольку такая цепь является неправильной. В этом случае не 1-совместимую пару выделяем жирным шрифтом, а вместо пар, следующих за ней, пишем многоточие. Состояния для простоты записи будем обозначать цифрами без штрихов.

- $C(1,4) = \{(1,4), \mathbf{(3,6)}, ...\}$;
- $C(1,6) = \{(1,6), \mathbf{(1,3)}, ...\}$;
- $C(3,5) = \{(3,5), \mathbf{(1,5)}, ...\}$;
- $C(4,6) = \{(4,6), (1,6), \mathbf{(1,3)}, ...\}$.

В новом автомате нет ни одной правильной цепи, значит, дальнейшая минимизация его невозможна. Пример подчеркивает приближенность метода – в предыдущем подразделе нам удалось сократить число состояний до четырех.

2.3. Задачи

Задача 1. Минимизировать полный автомат с помощью графа условий неотличимости, алгоритмов Мура и Хопкрофта.

а)

	ψ							
$X\backslash Q$	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	8	8	8	1	2	3	3	3
b	2	2	6	5	4	8	1	4
c	2	1	8	3	3	8	8	1
d	5	4	7	6	7	5	5	7

	φ							
$X\backslash Q$	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	0	0	0	1	1	1	1	0
b	1	1	1	0	0	0	0	1
c	0	0	0	0	0	0	0	0
d	1	1	1	1	1	1	1	1

b)

ψ

*X**Q*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	3	4	5	5	4	3	4	3
b	5	5	2	1	2	7	6	8
c	6	8	7	8	3	1	2	1
d	8	7	1	1	6	6	8	8

φ

*X**Q*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	0	0	1	1	0	1	1	1
b	0	0	0	0	0	0	0	0
c	1	1	1	1	1	1	1	1
d	1	1	0	0	1	0	0	0

Задача 2. Минимизировать частичный автомат точным методом и методом последовательных сокращений, сравнить результаты.

a)

ψ

*X**Q*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	4	–	2	6	3	2	–	7
b	–	–	–	–	4	3	5	–
c	5	5	–	–	–	–	5	5
d	7	–	8	–	–	–	1	2

φ

*X**Q*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	1	–	1	0	1	0	–	1
b	–	–	–	–	1	1	0	–
c	1	1	–	–	–	–	0	0
d	0	–	0	–	–	–	1	1

b)

ψ

*X**Q*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	5	8	8	4	7	7	5	5
b	2	1	–	–	8	–	–	7
c	3	–	–	5	7	4	–	2
d	3	6	1	2	–	–	–	–

φ

*X**Q*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
a	0	0	1	1	0	1	0	0
b	1	1	–	–	1	–	–	1
c	1	–	–	1	0	0	–	0
d	0	0	0	1	–	–	–	–

Задача 3. Минимизировать автоматы из задач раздела 1.8.