Московский государственный технический университет
имени Н.Э. Баумана

Факультет «Фундаментальные науки»
Кафедра «Математическая физика и вычислительная математика»

В.А. Кутыркин, А.Ю. Бушуев

ЭЛЕМЕНТЫ ТЕОРИИ АВТОМАТОВ И ФОРМАЛЬНЫХ ЯЗЫКОВ

Электронное учебное издание

Методические указания к решению задач
по дисциплине «Теория автоматов и алгоритмические языки»

Москва

(С) 2013 МГТУ им. Н.Э. БАУМАНА

УДК 518.5 (075.8)
ББК 22.12

Рецензент: профессор, д.т.н. Н.И. Сидняев

Кутыркин В.А., Бушуев А.Ю.

Элементы теории конечных автоматов и формальных языков. Электронное учебное издание. - М.: МГТУ имени Н.Э. Баумана, 2014. 48 с.

Издание содержит основные понятия теории конечных автоматов и формальных языков. Предварительно приводятся необходимые сведения из теории графов, необходимые для способов компьютерного задания и наглядного представления конечных автоматов и формальных языков. Излагаются основные методы анализа и синтеза конечных автоматов. Приведён разбор типовых задач и условия типовых индивидуальных домашних заданий.

Издание рассчитано на студентов, прослушавших курс по дисциплине «Дискретная математика и математическая логика».

Для студентов МГТУ имени Н.Э. Баумана по направлению бакалавриата «Математика и компьютерные науки» и специальности "Прикладная математика".

Рекомендовано учебно-методической комиссией факультета «Фундаментальные науки» МГТУ им. Н.Э. Баумана

Электронное учебное издание

Кутыркин Владимир Андреевич

Бушуев Александр Юрьевич

ЭЛЕМЕНТЫ ТЕОРИИ АВТОМАТОВ И ФОРМАЛЬНЫХ ЯЗЫКОВ.

© 2013 МГТУ имени Н.Э. Баумана

Оглавление

Введение 4

1.Форамальные языки 6

2. Лексикографический порядок. Словари формальных языков 8

3. Элементы теории конечных графов 15

3.1. Ориентированные графы. Деревья. Логические сети 15

3.2. Неориентированные графы. Контактные сети 18

3.3. Конечные мульти-орграфы. Язык конечного мульти-орграфа 19

4. Детерминированные и недетерминированные автоматы 22

4.1. Способы задания конечных автоматов Мили 24

4.2. Минимизация конечного автомата Мили по состояниям 25

4.3 Алгоритм минимизации конечного автомата Мили по состояниям 27

5. Дискретные преобразователи 32

5.1. Логические схемы из функциональных элементов 32

5.2. Контактные схемы 35

5.2.1. Синтез контактной схемы методом каскадов 37

5.2.2. Дерево анализа контактной схемы 38

6. Вопросы для самоконтроля 41

7. Индивидуальные домашние задания 43

Приложение 45

Литература 47

Введение

В представленных методических указаниях излагаются основные понятия теории автоматов и формальных языков. Для формальных языков приводятся методы построения их словарей и поиска слова в словаре по его номеру. Такие методы полезны для работы с формальными языками в прикладных компьютерных задачах. Изложение методов анализа и синтеза конечных автоматов предваряется введением в теорию графов, основные понятия которой необходимы для компьютерных способов задания и наглядного представления конечных автоматов. После этого даны основные определения и понятия теории автоматов с описанием алгоритмов оптимального построения конечных автоматов.

Особое внимание уделено задачам анализа и синтеза для автоматов, являющихся дискретными преобразователями, поскольку на их основе реализуются способы синтеза общих конечных автоматов. Для решения задачи анализа контактной схемы вводится особый граф, называемый деревом анализа контактной схемы. Этот граф позволяет перечислить все простые пути в контактной схеме, связывающие её вход и выход, что позволяет вычислить пропускную способность контактной схемы.

Изложение методических указаний сопровождается многочисленными примерами по решению типовых задач. Для улучшения освоения материала приведены упражнения, позволяющие закрепить основные понятия и методы изложения. В заключение приведены условия задач для выполнения индивидуальных домашних заданий по первой части курса дисциплины «Теория автоматов и алгоритмические языки», изучаемой студентами по направлению бакалавриата «Математика и компьютерные науки». Для более полного изучения автоматных, регулярных языков и алгоритмических языков можно ознакомиться с учебным пособием [1] и книгой [2], для знакомства с магазинными автоматами к учебнику [3]. Кроме того, для получения практических навыков полезен задачник [4].

Кратко, представленные методические указания характеризуются следующим образом.

Цель домашних заданий – практическое освоение способов упорядочения формальных языков и методов анализа и синтеза конечных автоматов. Для этого в методических указаниях приводятся примеры создания словарей формальных языков и способов выбора слов из этих словарей. Практическому усвоению этого материала способствуют сопровождающие изложение упражнения. Для правильного освоения методов анализа и синтеза конечных автоматов излагаются предварительные сведения из

теории графов, поскольку при компьютерном задании автоматов необходимо иметь способы представления их структуры. Приводимые далее методы анализа и синтеза конечных автоматов опираются на эти сведения, что позволяет наглядно проиллюстрировать методы анализа и синтеза конечных автоматов на конкретных примерах. Для закрепления этих методов предложены соответствующие упражнения.

Краткая характеристика объекта изучения. В предлагаемых методических указаниях приводятся основные понятия и структуры формальных языков и теории автоматов, которые необходимы при компьютерной работе с этими объектами. В представленный теоретический материал позволит студенту грамотно выполнить домашние задания.

Задачи и порядок выполнения домашних заданий. Настоящие методические указания содержат раздел «Индивидуальные домашние задания», в котором перечислены условия задач, которые студент должен решить в процессе освоения дисциплины. Приведённые в настоящих методических указаниях примеры иллюстрируют способы отчёта о решении предложенных задач.

Вопросы для самоконтроля. Для самоконтроля усвоения представленного материала в настоящих методических указаниях выделен раздел «Вопросы для самоконтроля».

Форма отчета по выполнению домашних заданий. При оформлении выполненных домашних заданий студент должен представить решение задач в следующем виде.

На титульном листе приводится название: Домашние задания по дисциплине «Теория автоматов и алгоритмические языки», вариант. Ниже указывается исполнитель задания с указанием группы и факультета. В правом нижнем углу указывается преподаватель, ведущий дисциплину, и графа об оценке выполнения домашнего задания. Содержательная часть выполнения домашнего задания должна последовательно излагать решения поставленных задач с предварительной формулировкой условий задач.

1.Формальные языки

Исходным понятием для введения формальных языков является понятие математического *знака*, далее, просто – знака. Знак – это нечленимый элемент для построения строк (слов, символов) формального языки, т.е. знак является своеобразным атомом. Поэтому знак удовлетворяет закону тождества, благодаря чему возможны точные его теоретические копии и становится возможным однозначное теоретическое прочтение строк из него сконструированных (записанных). *Строка* получается при помощи конечной, последовательной, слева направо, подряд записи знаков. При записи строки каждый записываемый знак характеризуется своим местом в строке, можно сказать номером места (позиции) или просто – номером. Место начальной буквы имеет номер 1, номер места следующей буквы, если таковая имеется, равен 2 и т.д. Длина слова определяется номером последней буквы в слове.

Для обозначения строк могут использоваться знаки или другие строки, становящиеся *символами*.

Особо выделяется *пустая строка*, не имеющая в своей записи ни одной буквы. Для записи пустой строки используется символ □.

Определение 1.1. (знакового алфавита)**.** Алфавит знаков – это строка, начальный (заключительный) знак которой равен ⟨ (⟩), и между этими знаками (начальным и заключительным), последовательно, через знак запятой, перечислены попарно различные знаки этого алфавита. Такой алфавит обязан содержать хотя бы один знак.►

Пример 1.1 (знаковых алфавитов).

а) $\langle a,b,,\rangle$ – алфавит из 3-х знаков;

б) В алфавите $\langle a,_,b\rangle$ знак _ играет роль пробела.►

Замечание 1.1 (об алфавитных знаках)**.** Для обозначения алфавита используются дважды подчёркиваемые знаки-символы. Если $\underline{\underline{A}}$ – обозначение алфавита, то A – обозначение совокупности знаков этого алфавита, в частности, $\{a,b,b,b,a\}$ – совокупность из двух знаков алфавита $\underline{\underline{A}}=\langle a,b\rangle$. Алфавит устанавливает «старшинство» среди своих знаков.

Самый младший – первый и т.д. Совокупность A знаков алфавита $\underline{\underline{A}}$ будет называться *квази-алфавитом*.►

Определение 1.2 (формального языка)**.** Пусть $\underline{\underline{A}}$ – алфавит знаков. Тогда все строки, в записи которых участвуют знаки только этого алфавита, образуют позитивный язык для этого алфавита. Такой позитивный язык обозначается символом A^+. Добавив к языку A^+ пустую строку, получим универсальный язык для этого алфавита, обозначаемый A^*. Язык A^* – совокупность всех A-строк. Любая подсовокупность универсального языка называется *формальным A-языком*.►

Большие тексты состоят из последовательной записи многих строк. Поэтому при введении формальных языков определяют особую операцию присоединения строк, называемую сцеплением или конкатенацией строк.

Определение 1.3 (сцепления строк)**.** Пусть x и y – строки. Сцепить строку x со строкой y – это значит приписать к слову x справа слово y. Результат такого *сцепления* обозначается выражением $x \otimes y$, где $\otimes$ – символ-знак сцепления (*конкатенации*) строк. По определению, строки $\square \otimes x$ и $x \otimes \square$ совпадают со строкой x ($\square$ – пустая строка).►

Согласно такому определению $aab \otimes bb = aabbb$.

Замечание 1.2. Если x и y – строки, то $(x \otimes y)$ результат сцепления строк x и y, совпадающий с результатом $x \otimes y$. Для операции сцепления строк справедливо тождество ассоциативности, т.е. $(x \otimes y) \otimes z = x \otimes (y \otimes z)$. Из этого тождества следует сочетательный закон сцепления строк. Поэтому скобки при сцеплении строк можно не указывать.►

Определение 1.4 (буквенного алфавита и его слов)**.** Список из попарно различных непустых строк, в записи которых не участвует знак запятой, может образовывать буквенный алфавит, где буквы – это строки из этого списка. Пусть $\langle y_1,...,y_n \rangle$ – такой список попарно различных букв (угловая скобка $\langle$ – начало списка, скобка $\rangle$ – его окончание). Список $\langle y_1,...,y_n \rangle$ является *буквенным алфавитом*, если любое сцепление его букв допускает однозначное побуквенное чтение. Следовательно, если $u_1,...,u_k, v_1,...,v_m$

это буквы этого списка и $u_1 \otimes ... \otimes u_k = v_1 \otimes ... \otimes v_m$, то $k = m$ и $u_1 = v_1$ и $u_1 = v_1, ..., u_k = v_m$. Результат сцепления букв алфавита называется *словом* в этом алфавите.►

Замечание 1.3. Как и для знаковых алфавитов, вследствие однозначности прочтения, для буквенных алфавитов определяются соответствующие понятия универсальных и формальных языков. Далее, поскольку алфавит знаков также можно рассматривать как буквенный алфавит, для знаковых и буквенных алфавитов используется общий термин – *алфавит*.►

2. Лексикографический порядок. Словари формальных языков

На протяжении настоящего раздела используется алфавит $\underline{\underline{A}} = \langle a_1, ..., a_k \rangle$, для которого A – символ совокупности его букв. Совокупность A будет называться квази-алфавитом, число $N^{\underline{\underline{A}}}(a_i) = i$ $(1 \le i \le k)$ номером буквы a_i в алфавите $\underline{\underline{A}}$ или $\underline{\underline{A}}$-номером этой буквы. Следовательно, далее $|\underline{\underline{A}}| = |A| = k$ – размер этого алфавита.

Для пустого слова, которое является пустой строкой, как и ранее, будет использоваться символ □. Совокупность всех слов, включая пустое, записанных из букв алфавита $\underline{\underline{A}}$ (совокупность A-слов), образует *универсальный язык* A^*, любой подъязык которого называется *формальным A-языком*. В частности, если из универсального языка A^* исключить пустое слово, то такой формальный A-язык называется *позитивным A-языком* и для него используется обозначение A^+.

Если x и y – два слова, то выражение $x \otimes y$ обозначает результат сцепления слова x со словом y ($\otimes$ – знак операции сцепления или конкатенации слов). Например, $aab_ \otimes bb = aab_bb$ и $x \otimes \square = \square \otimes x = x$.

Для универсального языка A^* создаётся особый словарь $\boldsymbol{Dc}^{\underline{\underline{A}}}(A^*)$, называемый *натуральным $\underline{\underline{A}}$-словарём*. Для создания такого словаря используется *лексикографический $\underline{\underline{A}}$-порядок* «старшинства» слов в языке A^*, индуцированный (наведённый) алфавитом $\underline{\underline{A}}$. Если два A-слова имеют различные длины и $|x| < |y|$, то, согласно такому порядку, слово y «старше» слова x. Если же их длины одинаковы и,

кроме того, они имеют представление $x=\alpha\otimes b\otimes\beta$ и $y=\alpha\otimes c\otimes\gamma$, где $\alpha,\beta,\gamma\in A^*$, $b,c\in A$ и $\boldsymbol{N}^{\underline{\underline{A}}}(c)>\boldsymbol{N}^{\underline{\underline{A}}}(b)$, то, в этом случае, слово y «старше» слова x.

Упражнение 2.1. Доказать, что в формальном позитивном A-языке A^+ количество слов длиной $n\in\mathbb{Z}_+$ равно $|A|^n$.►

Замечание 2.1. Из результатов *упражнения 1.1* следует, что в универсальном A-языке количество слов, длина которых не превышает число $n\in\mathbb{Z}_+$, конечно и равно числу $|A|^0+|A|^1+...+|A|^n$. Поэтому количество слов в универсальном A-языке, которые, согласно лексикографическому $\underline{\underline{A}}$-порядку, «младше» некоторого фиксированного слова, конечно. Кроме того, за этим фиксированным словом непосредственно следует единственное слово, которое «старше» его в лексикографическом $\underline{\underline{A}}$-порядке. Следуя такому лексикографическому «старшинству», за конечное число шагов можно «добраться» до любого A-слова. Таким образом, согласно лексикографическому $\underline{\underline{A}}$-порядку, все слова формального языка A^* можно расположить в натуральном $\underline{\underline{A}}$-словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(A^*)$, согласно их «старщинству» (пустое слово – начальное слово этого словаря).►

Пример 2.1. Пусть задан алфавит $\underline{\underline{B}}=\langle a,b\rangle$. Тогда начало натурального $\underline{\underline{B}}$-словаря $\boldsymbol{Dc}^{\underline{\underline{B}}}(B^*)$ имеет вид: $\langle\square,a,b,aa,ab,ba,bb,aaa,aab,aba,abb,baa,bab,bba,bbb\rangle$.►

Если $L\subset A^*$ – непустой формальный A-язык, то для него также создаётся *натуральный* $\underline{\underline{A}}$*-словарь* $\boldsymbol{Dc}^{\underline{\underline{A}}}(L)$ следующим образом. Для этого при развёртывании словаря $\boldsymbol{Dc}^{\underline{\underline{A}}}(A^*)$ из него последовательно вычёркиваются слова, не принадлежащие языку L. В результате остаётся натуральный $\underline{\underline{A}}$-словарь $\boldsymbol{Dc}^{\underline{\underline{A}}}(L)$ языка L. Вследствие *принципа математической индукции*, каждое слово $x\in L$ получает в словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(L)$ свой естественный номер $\boldsymbol{N}_L^{\underline{\underline{A}}}(x)$, а именно: начальное слово этого словаря получает первый номер, непосредственно следующее (если таковое имеется) – второй и т.д.

Приведём способ вычисления номера слова в натуральном $\underline{\underline{A}}$-словаре позитивного языка A^+.

Теорема 2.1. Для слова $x = b_1 \otimes ... \otimes b_m \in A^+$, где $b_1,...,b_m \in A$, его номер $\boldsymbol{N}^{\underline{\underline{A}}}_{A^+}(x)$ в натуральном $\underline{\underline{A}}$-словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(L)$ вычисляется согласно формуле:

$$\boldsymbol{N}^{\underline{\underline{A}}}_{A^+}(x) = \boldsymbol{N}^{\underline{\underline{A}}}(b_1)\cdot k^{m-1} + \boldsymbol{N}^{\underline{\underline{A}}}(b_2)\cdot k^{m-2} + ... + \boldsymbol{N}^{\underline{\underline{A}}}(b_{m-1})\cdot k + \boldsymbol{N}^{\underline{\underline{A}}}(b_m), \qquad (1)$$

где $|\underline{\underline{A}}| = |A| = k$.►

Следствие 2.1. Номер $\boldsymbol{N}^{\underline{\underline{A}}}_{A^*}(y)$ непустого слова $y \in A^*$ в словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(A^*)$ вычисляется согласно формуле $\boldsymbol{N}^{\underline{\underline{A}}}_{A^*}(y) = \boldsymbol{N}^{\underline{\underline{A}}}_{A^+}(y) + 1$.►

Пример 2.2. Если $\underline{\underline{B}} = \langle a, b \rangle$ – алфавит из *примера 1.1*, то, согласно *теореме 1.1*, номера $\boldsymbol{N}^{\underline{\underline{B}}}_{B^+}(y)$ слова $y = bbb \in B^+$ равен числу:

$$\boldsymbol{N}^{\underline{\underline{B}}}_{B^+}(y) = \boldsymbol{N}^{\underline{\underline{B}}}(b)\cdot 2^{3-1} + \boldsymbol{N}^{\underline{\underline{B}}}(b)\cdot 2^{3-2} + \boldsymbol{N}^{\underline{\underline{B}}}(b) = 2\cdot 2^2 + 2\cdot 2 + 2 = 14,$$

что соответствует словарю *примера 2.1*.►

Замечание 2.2. Если какому-либо символу x присваивается значение другого символа y, то для этого используется обозначение $x := y$, где $:=$ – знак *операции присваивания значения символа*, стоящего справа от этого знака, символу, стоящему слева от этого знака. В частности, в этом случае возможно выражение $x := x + 1$, если символу x уже присвоено некоторое значение.►

Пример 2.3. Приведём алгоритм вычисления слова в натуральном словаре позитивного языка по его номеру в этом словаре.

Пусть $n = \boldsymbol{N}^{\underline{\underline{A}}}_{A^+}(x)$ – номер слова $x \in A^+$ в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(A^+)$, вычисленный согласно формуле (1). Тогда для определения этого слова x используется алгоритм, имеющий вид:

(*1-ый шаг*) $y := n$;

(*2-ой шаг*) $m := 1$;

(*3-ий шаг*) $x = \square$;

(*4-ый шаг*) если $y = 0$, перейти к *14-ому шагу*;

(*5-ый шаг*) $z := y \boldsymbol{rest} k$ (остаток от деления числа y на число k);

(6-ой шаг) если $z=0$ перейти к *10-ому шагу*;

(*6-ой шаг*) $y:=(y-z)\,\mathbf{div}\,k$ (частное от деления числа $(y-z)$ на число k);

(*7-ой шаг*) $x=a\otimes x$, где $a\in A$ и $\boldsymbol{N}^{\underline{\underline{A}}}(a)=z$;

(*8-ой шаг*) $m:=m+1$;

(*9-ый шаг*) перейти к *4-ому шагу*;

(*10-ый шаг*) $y:=(y-k)\,\mathbf{div}\,k$;

(*11-ый шаг*) $x=a_k\otimes x$;

(*12-ый шаг*) $m:=m+1$;

(13-ый шаг) если $z=0$ перейти к *4-ому шагу*;

(14-ый шаг) вывод слова $x\in A^+$ и завершение алгоритма.►

Пример 2.4. Если $\underline{\underline{B}}=\langle a,b\rangle$ – алфавит из *примера 2.2* и $\boldsymbol{N}_{B^+}^{\underline{\underline{B}}}(y)=14$ – номер слова $y\in B^+$, то, согласно алгоритму *примера 2.3*, получаем: $y=bbb$, что соответствует *примеру 1.2*.►

Упражнение 2.2. Пусть $\underline{\underline{B}}=\langle \boldsymbol{0},\boldsymbol{1}\rangle$ – алфавит размера 2, где $\boldsymbol{0}$ и $\boldsymbol{1}$ – специальные «жирные» знаки, отличные от натуральных чисел нуля 0 и единицы 1. Для слова формального языка $L=\{\boldsymbol{1}\otimes x: x\in B^*\}$ написать формулу вычисления номера этого слова в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{B}}}(L)$, используя номера букв алфавита $\underline{\underline{B}}$, составляющих это слово (L – язык двоичного представлении натуральных чисел из совокупности $\mathbb{N}$). Создать алгоритм вычисления слова по его номеру в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{B}}}(L)$.►

Упражнение 2.3. Для заданного слова формального A-языка $L=\{a\otimes x: a\in A, a\neq a_1, x\in A^*\}$ написать формулу вычисления номера этого слова в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(L)$, используя номера букв алфавита $\underline{\underline{A}}$, составляющих это слово (L – язык k-ичного представлении натуральных чисел из совокупности $\mathbb{N}$). Затем, создать алгоритм вычисления слова по его номеру в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(L)$.►

Упражнение 2.4. Пусть в алфавите $\underline{\underline{A}}$ заданы два непустых слов x и y, имеющие номера m и n, соответственно, в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(A^*)$. Используя длины $|x|$ и $|y|$ этих слов, найти номер слова $x \otimes y$ в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{A}}}(A^*)$.►

Упражнение 2.5. Пусть $\underline{\underline{B}} = \langle \blacktriangle, \blacktriangledown, / \rangle$ – алфавит для языка рациональных дробей $\mathbb{Q}_{/} = \{m/n : m \in \mathbb{Z}, n \in \mathbb{N}\}$, где $\mathbb{N} = \langle \blacktriangle \rangle^{+} = \mathbb{Z}_{(+)}$ и $\mathbb{Z} = \mathbb{N} \cup \langle \blacktriangledown \rangle^{*}$. Для заданной неотрицательной (отрицательной) рациональной дроби $x = m/n \in \mathbb{Q}_{/}$ вычислить её номер $\boldsymbol{N}^{\underline{\underline{B}}}_{\mathbb{Q}_{/}}(x)$ в натуральном словаре $\boldsymbol{Dc}^{\underline{\underline{B}}}(\mathbb{Q}_{/})$.►

Определение 2.1. Два натуральных числа m и n, одно из которых ненулевое, называются *взаимно простыми*, если их наибольший общий делитель $\boldsymbol{НОД}(m,n) = 1$. Ненулевое натуральное число называется *простым*, если оно не является единицей и делится нацело только на себя само и единицу.►

Пример 2.5. *Алгоритм Евклида* для вычисления наибольшего общего делителя $\boldsymbol{НОД}(k,n)$ двух натуральных чисел k и n, одно из которых ненулевое, имеет следующий последовательно пошаговый вид:

(*1-ый шаг*) $x_1 := \boldsymbol{min}(k,n)$ (*вычисление минимального из двух чисел*);

(*2-ой шаг*) $x_2 := \boldsymbol{max}(k,n)$ (*вычисление максимального из двух чисел*);

(*3-ий шаг*) $x_3 := \boldsymbol{min}(k,n)$;

(*4-ый шаг*) если $x_1 = 0$, то переходим к *9-ому шагу*;

(*5-ый шаг*) $x_1 := \boldsymbol{min}(x_2 - x_3, x_3)$;

(*6-ой шаг*) $x_2 := \boldsymbol{max}(x_2 - x_3, x_3)$;

(*7-ой шаг*) $x_3 := x_1$;

(*8-ой шаг*) переход к *4-ому шагу*;

(*9-ый шаг*) вывод "$\boldsymbol{НОД}(k,n) =$" x_2 и завершение алгоритма.►

Упражнение 2.6. Используя алгоритм Евклида найти $\boldsymbol{НОД}(90,36)$.►

Замечание 2.3.

а) Согласно результату *упражнения 2.5*, язык рациональных дробей допускает построение его в виде словаря $\boldsymbol{Dc}^{\underline{B}}(\mathbb{Q}_{/})$. Однако, этот словарь не является словарём рациональных чисел, поскольку каждое рациональное число представляется бесконечным словарём рациональных дробей, в частности, словарь дробей $\langle 0/n : n \in \underline{\underline{\mathbb{N}}} \rangle$ определяет одно рациональное число («нулевое»). Для того чтобы избежать такой неоднозначности в представлении рациональных чисел, в качестве единственного представителя рационального числа следует выбирать только несократимые рациональные дроби. В этом случае совокупность рациональных чисел $\mathbb{Q}$ представима в виде совокупности $\mathbb{Q} = \{m/n \in \mathbb{Q}_{/} : \boldsymbol{НОД}(m,n) = 1\}$, где каждое рациональное число представлено единственной несократимой дробью. *Алгоритм Евклида*, приведённый в *примере 2.5*, позволяет выделить из натурального словаря $\boldsymbol{Dc}^{\underline{B}}(\mathbb{Q}_{/})$ натуральный словарь $\boldsymbol{Dc}^{\underline{B}}(\mathbb{Q})$ рациональных чисел. При таком выделении необходимо при развёртывании словаря $\boldsymbol{Dc}^{\underline{B}}(\mathbb{Q}_{/})$ в словарь $\boldsymbol{Dc}^{\underline{B}}(\mathbb{Q})$ отбирать только несократимые рациональные дроби. Таким образом, для языка рациональных чисел возможно построение его словаря.

б) В математике рациональноё число $m/1 \in \mathbb{Q}$ отождествляется с целым числом $m \in \mathbb{Z}$. В частности, при таком отождествлении $0/1 = 0 \in \mathbb{Q}$. Таким образом, справедливо высказывание $\mathbb{Z} \subset \mathbb{Q}$.►

Для непустого формального A-языка $L \subseteq A^*$ могут создаваться словари, не обязательно являющиеся натуральными. В этом случае для такого словаря, как правило, может использоваться обозначение $\boldsymbol{Dc}(L)$ или $\boldsymbol{Dc}_n(L)$, где $n \in \mathbb{N}$. Кроме того, для номера слова $x \in L$ в таком словаре будет использоваться символ $\boldsymbol{N}_{\boldsymbol{Dc}(L)}(x)$.

Пусть $k, m \in \mathbb{N}$, $\underline{U} = \langle u_1, ..., u_k \rangle$ и $\underline{\underline{V}} = \langle v_1, ..., v_m \rangle$ – два словаря формальных языков. Введём два словаря $\boldsymbol{Dc}_1(L)$ и $\boldsymbol{Dc}_2(L)$ для формального языка $L = U \times V = \{u \boxtimes v : u \in U, v \in V\}$ прямого произведения языков U и V. Для этого каждому слову $x = u_i \boxtimes v_j \in L$, где $i = 1, ..., k$ и $j = 1, ..., m$, в словаре $\boldsymbol{Dc}_1(L)$ присвоим одномерный номер

$$\boldsymbol{N}_{\boldsymbol{Dc}_1(L)}(x) = (i-1) \cdot m + j. \qquad (2).$$

В словаре $\boldsymbol{Dc}_2(L)$ этому слову $x \in L$ присвоим одномерный номер

$$\boldsymbol{N}_{\boldsymbol{Dc}_2(L)}(x) = (j-1) \cdot m + i. \qquad (3).$$

Если для слова $x = u_i \boxtimes v_j \in L$ ввести символ c_j^i, то вместо словарей $\boldsymbol{Dc}_1(L)$ и $\boldsymbol{Dc}_2(L)$ для перечисления без повторов слов языка L можно использовать представление языка L в виде матрицы C из k строк и m столбцов:

$$C = \begin{pmatrix} c_1^1 \ldots c_m^1 \\ \ldots\ldots\ldots \\ c_1^k \ldots c_m^k \end{pmatrix} = (c_j^i)_m^k, \qquad (4)$$

где для так называемых немых индексов i и j указаны их пробеги ($i = 1,\ldots,k$ и $j = 1,\ldots,m$) и (i, j) – двумерный номер слова $x = u_i \boxtimes v_j \in L$, согласно матрице C. В этом случае формула (2) отражает перечисление одномерных номеров компонент матрицы (4) по строкам, когда после последней компоненты строки матрицы счёт продолжается с начальной компоненты следующей строки, если таковая имеется. Формула (3) отражает аналогичный счёт одномерных номеров компонент матрицы (4) по столбцам.

Формула (2) может использоваться и в том случае, когда, как и ранее, словарь $\underline{\underline{U}} = \langle u_1,\ldots,u_k \rangle$ конечен, но словарь $\underline{\underline{V}} = \langle v_j : j \in \underline{\underline{\mathbb{N}}} \rangle$ бесконечен. Аналогично, формула (3) может использоваться, когда словарь $\underline{\underline{U}} = \langle u_i : i \in \underline{\underline{\mathbb{N}}} \rangle$ бесконечен, но словарь $\underline{\underline{V}} = \langle v_1,\ldots,v_m \rangle$ конечен.

Упражнение 2.7. Пусть задан номер $n = \boldsymbol{N}_{\boldsymbol{Dc}_1(L)}(u \boxtimes v)$ ($n = \boldsymbol{N}_{\boldsymbol{Dc}_2(L)}(u \boxtimes v)$) неизвестного слова $u \boxtimes v \in L$ в словаре $\boldsymbol{Dc}_1(L)$ ($\boldsymbol{Dc}_2(L)$) формального языка $L = U \times V$. Предполагается, что этот номер вычислен согласно формуле (2) ((3)). Также предполагается, что словари $\underline{\underline{U}} = \langle u_1,\ldots,u_k \rangle$ и $\underline{\underline{V}} = \langle v_j : j \in \underline{\underline{\mathbb{N}}} \rangle$ ($\underline{\underline{U}} = \langle u_i : i \in \underline{\underline{\mathbb{N}}} \rangle$ и $\underline{\underline{V}} = \langle v_1,\ldots,v_m \rangle$) заданы. Требуется написать алгоритм для вычисления номеров слов u и v в словарях $\underline{\underline{U}}$ и $\underline{\underline{V}}$, соответственно.►

Пусть $\underline{\underline{X}} = \langle x_n : n \in \underline{\underline{\mathbb{N}}} \rangle$ – словарь для непустого формального языка $X \subseteq A^*$. Рассмотрим формальный язык $L = X \times X = X^{\times 2} = \langle x_i \boxtimes x_j : i, j \in \mathbb{N} \rangle$. Если для $i, j \in \mathbb{N}$ вести обозначения $x_i \boxtimes x_j = c_j^i$, то для перечисления без повторов слов языка L можно использовать представление языка L в виде матрицы C из бесконечного числа строк и столбцов:

$$C=\begin{pmatrix} c_1^1 \ldots c_m^1 \ldots \\ \ldots\ldots\ldots \\ c_1^k \ldots c_m^k \ldots \\ \ldots\ldots\ldots \end{pmatrix}=(c_j^i)_{\underline{\underline{\mathbb{N}}}}^{\underline{\underline{\mathbb{N}}}}, \qquad (5)$$

где так называемые немые индексы i и j последовательно пробегают ряд натуральных чисел $\underline{\underline{\mathbb{N}}}$.

Упражнение 2.8. Для номеров $i, j \in \mathbb{N}$ показать, что можно создать такой словарь $\boldsymbol{Dc}(L)$ языка $L = X^{\times 2} = \langle x_s \boxtimes x_t : s,t \in \mathbb{N} \rangle$, в котором формула для вычисления номера слова $x_i \boxtimes x_j = c_j^i$ из матрицы (5) имеет вид: $\boldsymbol{N}_{\boldsymbol{Dc}(L)}(c_j^i) = ((i+j-2)\cdot(i+j-1))/2 + j$.►

3. Элементы теории конечных графов

Для наглядного представления конечных автоматов используются их диаграммы. В общем случае такие диаграммы порождены соответствующими графами, основные понятия для которых изложены в настоящем разделе.

3.1. Ориентированные графы. Деревья. Логические сети

Пусть $\underline{\underline{V}} = \langle v_1, ..., v_s \rangle$ – словарь конечного формального языка V и на нём определено бинарное отношение $\tau \subseteq V \times V = V^2$, индуцирующее матрицу $\boldsymbol{Mtr}_\tau = (m_j^i)_s^s$ с булевыми компонентами, для которой, если $1 \le i, j \le s$, то: $m_j^i = \begin{cases} 1, (v_i, v_j) \in \tau; \\ 0, (v_i, v_j) \notin \tau. \end{cases}$

Тогда пара $(\underline{\underline{V}}, \tau)$ определяет *конечный орграф* $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}}; \tau)$, в котором словарь $\underline{\underline{V}}$ задаёт список его *вершин* и бинарное отношение $\tau \subseteq V^2$ определяет совокупность его *дуг* $\vec{\boldsymbol{R}}(\underline{\underline{V}}, \tau) = \{(u,v) \in V^2 : (u,v) \in \tau\}$. В этом орграфе матрица $\boldsymbol{Mtr}_\tau = (m_j^i)_s^s$ называется *матрицей смежности* вершин, поскольку полагается, что вершина $v_j \in V$ смежна (связана дугой) с вершиной $v_i \in V$, если $m_j^i = 1$. В этом случае говорят, что вершина $v_i \in V$ является началом и вершина $v_j \in V$ концом дуги $(v_i, v_j) \in \vec{\boldsymbol{R}}(\underline{\underline{V}}, \tau)$, которая *исходит*

из вершины $v_i \in V$ и *входит в вершину* $v_j \in V$. Для наглядного представления орграфа используют его рисунок в виде *диаграммы орграфа*, на которой вершины изображаются, как правило, кружочками, помеченными соответствующими словами языка V, и дуги – стрелками, связывающими соответствующие вершины. В *упрощённой диаграмме орграфа* вершины-кружочки остаются пустыми.

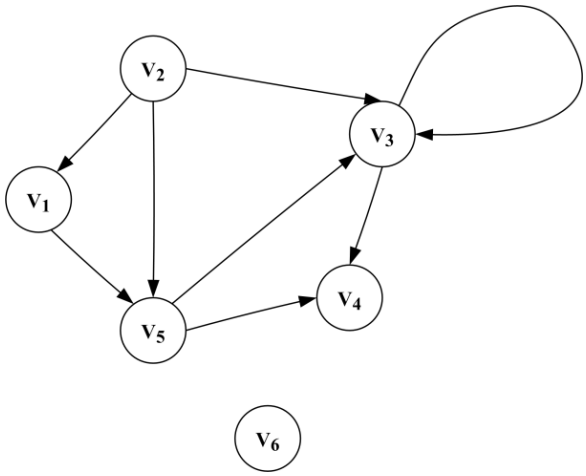


Рис.1

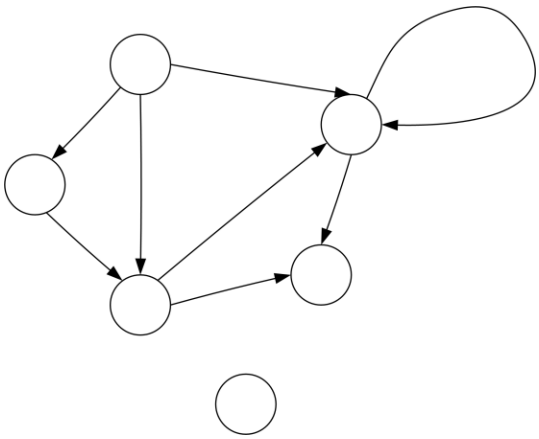


Рис.2

Пример 3.1. На рис. 1 (рис. 2) показана (упрощённая) диаграмма графа $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$ со словарём вершин $\underline{\underline{V}} = \langle v_1, v_2, v_3, v_4, v_5, v_6 \rangle$ и матрицей смежности

$$\boldsymbol{Mtr}_\tau = (m_j^i)_6^6 = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \blacktriangleright$$

Если в вершину $v \in V$ орграфа $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$ входит ровно k дуг, то говорят, что её *позитивная валентность* $val_+(v) = k$. Если из вершины $v \in V$ орграфа $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$ исходит ровно m дуг, то говорят, что её *негативная валентность* $val_(v) = m$. Число $val(v) = val_+(v) - val_(v)$ называется валентностью вершины $v \in V$. Если $val_+(v) = 0$, то вершину $v \in V$ называют *источником*, если $val_(v) = 0$ – *стоком*, если же $val_+(v) = val_(v) = 0$ – *глухим источником*.

Список $\boldsymbol{\pi}^{>} = ((u_0,u_1),(u_1,u_2),...,(u_{k-1},u_k))$ из k последовательных дуг орграфа $\overrightarrow{\boldsymbol{Gr}}(\underline{V};\tau)$ называется его *путём* с началом в вершине $u_0 \in V$ и окончанием в вершине $u_k \in V$. Этот же путь будет обозначаться и списком вершин $\boldsymbol{\pi}^{<} = (u_0,u_1,u_2,...,u_{k-1},u_k)$, в котором каждая непосредственно следующая вершина смежна с предыдущей. Число $k = |\boldsymbol{\pi}^{>}| = |\boldsymbol{\pi}^{<}|$ называется длиной этого пути. Если $u_0 = u_k$, то этот такой путь называется *контуром* или *циклом*, в котором любая вершина может считаться начальной или конечной. Если же в этом цикле все вершины $u_1,u_2,...,u_{k-1},u_k$ попарно различны, то такой цикл называется простым. В частности, если $k = 0$, то такой простой цикл называется петлёй. Путь, не являющийся циклом, при проходе которого нет повторяющихся вершин, называется *простым путём*, в противном случае – *составным путём*, в котором, в качестве его составляющей, обязательно содержится цикл.

Определение 3.1 (ациклического орграфа). Орграф, в котором нет циклов, называется ациклическим.►

Орграф, диаграмма которого показана на рис. 1 не является ациклическим, поскольку содержит петлю. Кроме того, на этой диаграмме путь $(u_1,u_2,u_4,u_3,u_2,u_5)$ является составным, т.к. содержит цикл (u_2,u_4,u_3,u_2).

Определение 3.2 (дерева). Конечный ациклический орграф с одним источником, в котором любая вершина, не являющаяся источником, имеет единичную позитивную валентность, называется *деревом*. Источник дерева называется его *корнем*, стоки – *листьями*.►

Теорема 3.1. Если орграф является деревом, то для его двух различных вершин – либо нет соединяющего их пути, либо такой путь есть и он единственен.►

На рис. 3 показана упрощённая диаграмма орграфа-дерева.

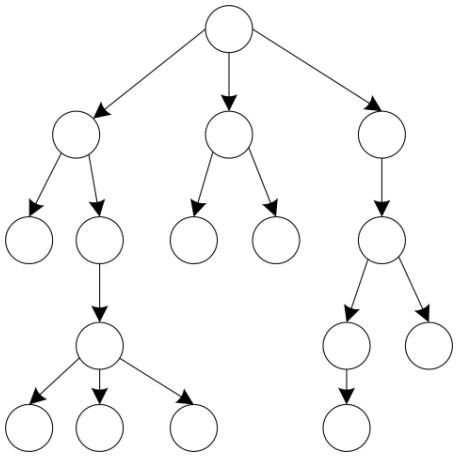


Рис.3

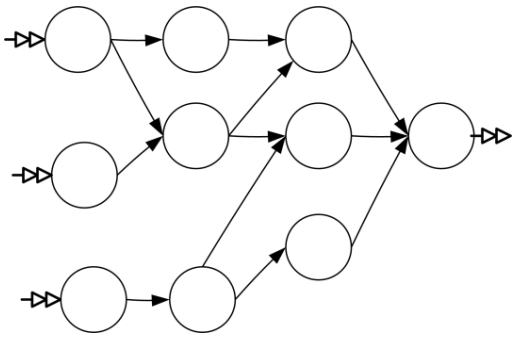


Рис.4

Определение 3.3 (логической сети). Пусть $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$ – такой конечный ациклический орграф с n не глухими источниками и одним стоком, в котором любые два пути соединяющиеся источники с его вершиной, не являющейся источником, имеют одинаковую длину. Тогда такой орграф называется логической сетью.►

На рис. 4 показана упрощённая диаграмма логической сети с тремя источниками, которые, как и сток, отмечены особыми (двойными) стрелками.

3.2. Неориентированные графы. Контактные сети

Пусть в конечном орграфе $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$ бинарное отношение τ является антирефлексивным (т.е. $(v,v)\notin\tau$ для любой вершины $v\in V$) и антисимметричным ($(v,u)\notin\tau$, если $(u,v)\in\tau$). Следовательно, орграф $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$ не имеет петель и две его вершины не могут быть взаимно смежными. В этом случае орграф $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$, определяет конечный неориентированный граф $\boldsymbol{Gr}(\underline{\underline{V}};\theta)$, называемый, кроме того, разориентацией орграфа $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$. Граф $\boldsymbol{Gr}(\underline{\underline{V}};\theta)$ имеет тот же словарь вершин и его неориентированные рёбра собраны в совокупность $\boldsymbol{R}(\underline{\underline{V}},\tau)=\{\{u,v\}=\{v,u\}:(u,v)\in\tau\}$. Следовательно, для неориентированного графа $\boldsymbol{Gr}(\underline{\underline{V}};\theta)$ бинарное отношение $\theta\subseteq V\times V$ имеет вид $\theta=\{(u,v)\in V\times V:(u,v)\in\tau \text{ или } (v,u)\in\tau\}$ и симметричная матрица $\boldsymbol{Mtr}_\theta$ с нулями на главной диагонали определяет неориентированное понятие смежности его вершин. По

аналогии с орграфом $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\tau)$ в графе $\boldsymbol{Gr}(\underline{\underline{V}};\theta)$ вводятся понятия путей и их длин, но понятие смежности вершин теряет ориентацию. Поэтому для вершин графа вводится только одно понятие (общей) валентности, но нельзя ввести позитивные и негативные валентности.

На рис. 5 показана упрощённая диаграмма орграфа, для которого на рис. 6 приведена упрощённая диаграмма его разориентации. На упрощённой диаграмме разориентации особыми стрелками указаны источник и сток исходного орграфа.

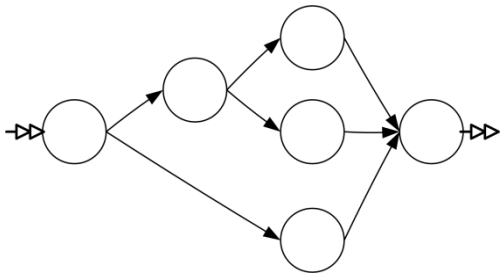


Рис.5

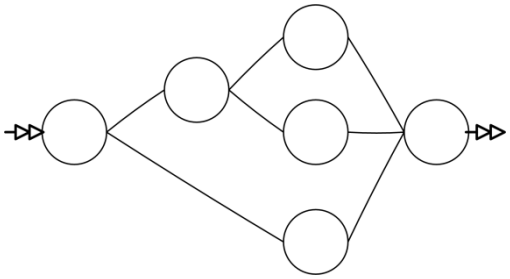


Рис.6

Определение 3.4 (контактной сети). Пусть граф $\boldsymbol{Gr}(\underline{\underline{V}};\tau)$ является разориентацией конечного ациклического орграфа с одним не глухим источником $q \in V$ и одним стоком $e \in V$. Тогда граф $\boldsymbol{Gr}(\underline{\underline{V}};\tau)$, в котором вершина $q \in V$ является *входом* и вершина $e \in V$ – *выходом*, называется *контактной сетью*. Для обозначения этой сети используется выражение: $\boldsymbol{Gr}(\underline{\underline{V}};\tau \mid q,e)$.►

На рис. 6 показана упрощённая диаграмма контактной сети, полученной разориентацией орграфа, упрощённая диаграмма которого приведена на рис. 5. На диаграмме контактной сети, показанной на рис. 6, вход и выход отмечены особыми (двойными) стрелками.

3.3. Конечные мульти-орграфы. Язык конечного мульти-орграфа

Пусть задан алфавит $\underline{\underline{A}} = \langle a_1,...,a_k \rangle$ с совокупностью A его букв и словарь $\underline{\underline{V}}$ конечного формального языка V, на котором определён список бинарных отношений $\underline{\hat{A}} = (\hat{a}_1,...,\hat{a}_k)$, индуцируемых их матрицами. Тогда определён набор из k орграфов

$\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\underline{\hat{A}}) = (\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\hat{a}_1),...,\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\hat{a}_k))$, называемый конечным *мульти-орграфом*. В этом мульти-орграфе $\underline{\underline{V}}$ – словарь его вершин и $\vec{\boldsymbol{R}}(\underline{\underline{V}},\underline{\hat{A}}) = \{u \xrightarrow{a_j} v : (\exists j = \overline{1,k} \,|\, (u,v) \in \hat{a}_j\}$ – совокупность его ориентированных рёбер-дуг. Если $\hat{a}_j \in \underline{\hat{A}}$ и $u \xrightarrow{a_j} v \in \vec{\boldsymbol{R}}(\underline{\underline{V}},\underline{\hat{A}})$, то $u \in V$ – начало и $v \in V$ конец этого ребра и, кроме того, вершина $v \in V$ является $\hat{a}_j$-смежной с вершиной $u \in V$. Следовательно, в отличие от обычного орграфа, вершина мульти-орграфа может иметь несколько типов смежности с другой вершиной или сама с собой. Для наглядного представления мульти-орграфа, как и для обычного орграфа, используется его диаграмма. Для иллюстрации, на рис.7 показан пример диаграммы конечного мульти-орграфа. Как и для обычного орграфа, в мульти-орграфе $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\underline{\hat{A}})$ определяется путь с началом в вершине $u_0 \in V$ и концом в вершине $u_m \in V$ в виде набора его дуг $\boldsymbol{\pi}^{>} = (u_0 \xrightarrow{b_1} u_1, u_1 \xrightarrow{b_2} u_2, ..., u_{m-1} \xrightarrow{b_m} u_m)$, где $b_1,...,b_m \in A$. Этот путь длиной $|\boldsymbol{\pi}^{>}| = m$ связывает вершину $u_0 \in V$ с вершиной $u_m \in V$.

В мульти-орграфе $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\underline{\hat{A}})$ могут выделяться непустые совокупности $V_0 \subseteq V$ и $V_1 \subseteq V$ начальных и заключительных его вершин, соответственно. В этом случае для такого мульти-орграфа используется обозначение $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\underline{\hat{A}} \,|\, V_0, V_1)$, вершины $v_0 \in V_0$ и $v_1 \in V_1$ называются начальной и заключительной, соответственно. Кроме того, для обозначения совокупности путей, связывающих его начальные и заключительные вершины, используется выражение $\Pi(V_0, V_1)$.

Определение 3.5 (языка мульти-орграфа). Для совокупности путей $\Pi(V_0, V_1)$ мульти-орграфа $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\underline{\hat{A}} \,|\, V_0, V_1)$ определено отображение $\varphi : \Pi(V_0, V_1) \to A^*$ в универсальный A-язык A^* следующим образом. Если $u_0 \in V_0$, $u_m \in V_1$ и $\boldsymbol{\pi}^{>} = (u_0 \xrightarrow{b_1} u_1, u_1 \xrightarrow{b_2} u_2, ..., u_{m-1} \xrightarrow{b_m} u_m) \in \Pi(V_0, V_1)$, то $\varphi(\boldsymbol{\pi}^{>}) = b_1...b_m \in A^*$. Формальный A-язык $\{\varphi(\boldsymbol{\pi}^{>}) : \boldsymbol{\pi}^{>} \in \Pi(V_0, V_1)\} \subseteq A^*$, дополненный пустым словом, если $V_0 \cap V_1 \neq \varnothing$, называется *языком мульти-орграфа* $\overrightarrow{\boldsymbol{Gr}} = \overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\underline{\hat{A}})$ и для него используется обозначение $Lng(\overrightarrow{\boldsymbol{Gr}})$.►

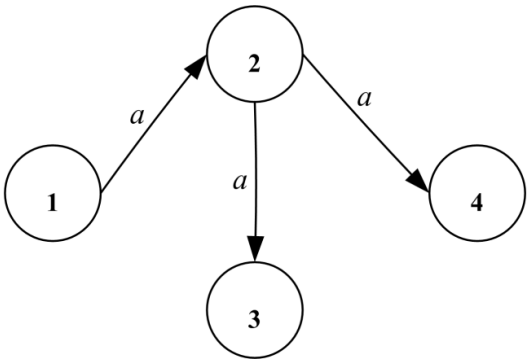


Рис.7

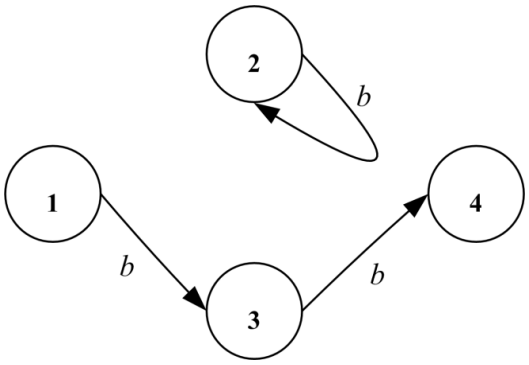


Рис.8

Пример 3.2 (мульти-орграфа и его языка). На рис. 7 и рис. 8 показаны диаграммы орграфов $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\hat{a})$ и $\overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\hat{b})$, соответственно. Из этих орграфов образован мульти-орграф $\overrightarrow{\boldsymbol{Gr}} = \overrightarrow{\boldsymbol{Gr}}(\underline{\underline{V}};\hat{a},\hat{b}\,|\,\langle 1\rangle,\langle 3,4\rangle)$, диаграмма которого приведена на рис. 9. Входные и заключительные вершины этого мульти-орграфа отмечены на диаграмме особыми (двойными) стрелками.

Из анализа диаграммы мульти-орграфа $\overrightarrow{\boldsymbol{Gr}}$ следует, что его язык $Lng(\overrightarrow{\boldsymbol{Gr}})$ имеет вид:

$$Lng(\overrightarrow{\boldsymbol{Gr}}) = \{ab^m a : m \in \mathbb{Z}_+\} \cup \{ab^m ab : m \in \mathbb{Z}_+\} \cup \{bb\} \cup \{b\} \in \langle a,b\rangle^*,$$

где $\mathbb{Z}_+ = \{j \in \mathbb{Z} : j \geq 0\}$, $b^{\otimes m} = \underbrace{b\ldots b}_{m\ раз}$ для $m \in \mathbb{Z}_+$ и $b^{\otimes 0} = \square$ – пустое слово.▶

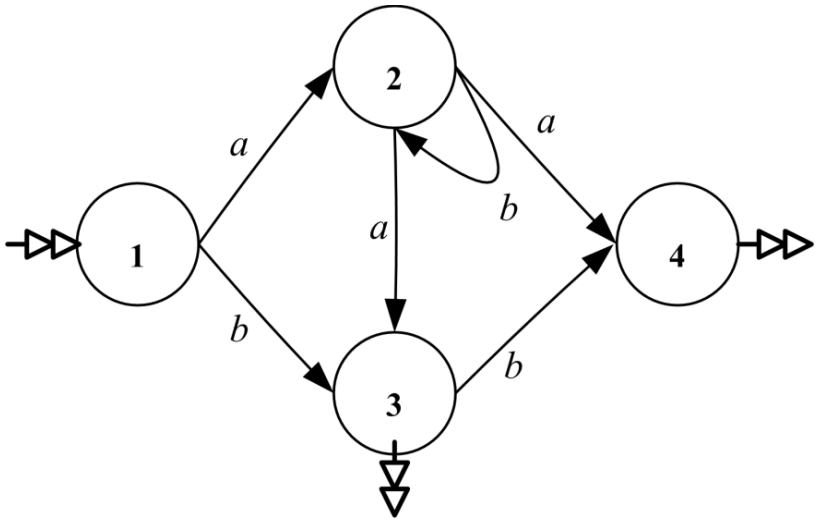


Рис.9

Замечание 3.1 (об искусственных языках). Формальные языки, порождённые конечными мульти-орграфами, относятся к типу искусственных языков, используемых в языках

программирования. Среди формальных языков искусственные языки образуют более узкий класс. Основное требование к языкам этого узкого класса состоит в том, чтобы они были разрешимы, т.е. распознаваемы с помощью соответствующих алгоритмов. Такое требование обусловлено тем, что тексты практических программ достаточно велики и, фактически, недоступны для проверки на корректность (синтаксическую правильность) человеком. Следовательно, анализ корректности текстов таких программ должен осуществляться автоматически с помощью тех или иных алгоритмов, реализуемых на современных компьютерах.

Следует отметить, что среди искусственных языков языки, порождённые мульти-орграфами и названные регулярными, являются самыми простыми и используются для создания простейших конструкций языков программирования. Распознавания таких языков происходит с помощью соответствующих автоматов, о которых будет рассказано в последующих разделах.►

4. Детерминированные и недетерминированные автоматы

Пусть A и B – два квази-алфавита, т.е. два неупорядоченных «алфавита», и $\underline{\underline{V}}$ – словарь (конечного) формального языка V. Кроме того, заданы отображения $\delta: A\times V \to V$ и $\lambda: A\times V \to B$. Тогда пятерка $\Gamma = (A, B, V, \delta, \lambda)$ определяет (конечный) автомат Мили, в котором квази-алфавиты A и B называются его алфавитами входа и выхода, соответственно, язык V – совокупностью его состояний, отображения $\delta: A\times V \to V$ и $\lambda: A\times V \to B$ – его функциями перехода по состояниям и выхода, соответственно. В этом случае автомат Мили называют (конечным) детерминированным автоматом. Если же функция δ не определена для некоторых пар из совокупности $A\times V$ или на некоторых парах этой совокупности – многозначна, то такой автомат Γ называется (конечным) недетерминированным автоматом.

Автомат Мили Γ называется *автоматом Мура*, если есть такое отображение $\mu: V \to B$, что $\lambda(a,v) = \mu(\delta(a,v))$ для любых $a \in A$ и $v \in V$.

Если в автомате Мили Γ всего одно состояние, т.е. $|V| = 1$, то автомат Γ называется *дискретным преобразователем*.

Автомат Мили Γ, в котором алфавит входа состоит из единственной буквы, называется *автономным автоматом Мили*.

Автомат Мили Γ, в котором выделено одно состояние, с которого он всегда начинает работу, называется *инициальным автоматом*.

Если алфавит выхода автомата (детерминированного или нет) состоит из единственной буквы, то такой автомат называется *автоматом без выхода*.

В автомате Мили Γ функции δ и λ можно рассматривать как бинарные многосортные алгебраические бинарные операции с сигнатурой $*$ и $\bullet$, соответственно. Следовательно, в этом случае для $a \in A$ и $v \in V$ можно использовать обозначения:

$$\begin{cases} \delta(a,v) = a * v; \\ \lambda(a,v) = a \bullet v. \end{cases} \qquad (1)$$

Кроме того, для обозначения автомата Мили Γ можно использовать выражение $\Gamma = (A,B,V,*,\bullet)$.

В рамках обозначений (1) работу автомата Мили $\Gamma = (A,B,V,*,\bullet)$ можно описать следующим образом. Эта работа осуществляется потактово. Если в начальный такт работы автомат Γ находился в состоянии $v(0) \in V$ и на его вход поступила буква $b(0) \in A$, то он переходит в состояние $v(1) = a(0) * v(0)$ и на его выходе появляется буква $b(1) = a(0) \bullet v(0)$. Если автомат Γ на текущем такте t находился в состоянии $v(t) \in V$ и на его вход поступила буква $a(t) \in A$, то он переходит в состояние $v(t+1) = a(t) * v(t)$ и на его выходе появляется буква $b(t+1) = a(t) \bullet v(t)$. Такая потактовая работа автомата Γ описывается уравнениями:

$$\begin{cases} v(t+1) = b(\mathrm{t}) * v(\mathrm{t}); \\ b(t+1) = b(t) \bullet v(t). \end{cases} \qquad (2)$$

Уравнения (2) называются *каноническими уравнениями автомата* Γ. Таким образом, если автомат Γ в начале работы находился в состоянии $v(0) \in V$ и на его вход последовательно поступали буквы $b(0), b(1), ..., b(t-1) \in A$, то он переходит в состояние $v(t) \in V$ и на его выходе последовательно появляются буквы $c(1), ..., c(t) \in B$. Поэтому функции δ и λ (операции $*$ и $\bullet$) можно расширить до отображений $\delta^* : A^* \times V \to V$ и $\lambda^* : A^* \times V \to B^*$, полагая, что для слова $x = b(0)b(1)...b(t-1) \in A^*$ и состояния $v = v(0) \in V$ справедливы равенства: $\delta^*(x,v) = x * v = v(t)$ и $\lambda^*(x,v) = x \bullet v = c(1)...c(t)$. Такое определение можно описать следующим индуктивным образом.

Определение 4.1. В автомате Мили $\Gamma = (A,B,V,\delta,\lambda) = (A,B,V,*,\bullet)$ для любого его состояния $v \in V$ и пустого слова $\square$, по определению, полагается, что $\delta^*(\square, v) = \square * v = v$ и

$\lambda^*(\square, v) = \square \bullet v =$. Если для слова $x \in A^*$ и состояния $v \in V$ уже определены значения $\delta^*(x,v) = x * v$ и $\lambda^*(x,v) = x \bullet v$, то, по определению, для буквы $a \in A$ полагается, что:

$$\begin{cases} \delta^*(x \otimes a, v) = (x \otimes a) * v = \delta(a, \delta^*(x,v)) = a * (x * v); \\ \lambda^*(x \otimes a, v) = (x \otimes a) \bullet v = \lambda^*(x,v) \otimes \lambda(a, \delta^*(x,v)) = (x \bullet v) \otimes (a \bullet (x * v)). \end{cases} \blacktriangleright \qquad (3)$$

Для определённых в (3) отображений $\delta^* : A^* \times V \to V$ и $\lambda^* : A^* \times V \to B^*$ (операций $*$ и $\bullet$) справедлив следующий результат, легко выводимый из соотношений (3).

Теорема 4.1. Для любого состояния $v \in V$ автомата Мили $\Gamma = (A, B, V, \delta, \lambda) = (A, B, V, *, \bullet)$ и слов $x, y \in A^*$ справедливы равенства:

$$\begin{cases} \delta^*(x \otimes y, v) = (x \otimes y) * v = \delta^*(y, \delta^*(x,v)) = y * (x * v); \\ \lambda^*(x \otimes y, v) = (x \otimes y) \bullet v = \lambda^*(x,v) \otimes \lambda^*(y, \delta^*(x,v)) = (x \bullet v) \otimes (y \bullet (x * v)). \end{cases} \blacktriangleright$$

4.1. Способы задания конечных автоматов Мили

Для использования в компьютерной практике автоматов необходимо предложить способы их задания, которые были бы удобны для их теоретического анализа и реализации на компьютере. Такие способы задания и рассматриваются в настоящем разделе на соответствующих наглядных примерах.

Пример 4.1 (табличный способ задания конечного автомата Мили). Рассмотрим конечный автомат Мили $\Gamma = (A, B, V, \delta, \lambda)$, где $\underline{V} = \langle v_1, v_2 \rangle$ – словарь его состояний, $A = \langle a, b, c \rangle$ и $B = \langle 0, 1 \rangle$ – алфавиты его входа и выхода, соответственно. Его функции перехода по состояниям $\delta : A \times V \to V$ и выхода $\lambda : A \times V \to B$ имеют вид:

$$\begin{cases} \delta(a, v_1) = v_2, \delta(b, v_1) = v_1, \delta(c, v_1) = v_2, \delta(a, v_2) = v_1, \delta(b, v_2) = v_2, \delta(c, v_2) = v_1; \\ \lambda(a, v_1) = 1, \lambda(b, v_1) = 1, \lambda(c, v_1) = 1, \lambda(a, v_2) = 0, \lambda(b, v_2) = 0, \lambda(c, v_2) = 0. \end{cases}$$

Работа такого автомата Мили $\Gamma = (A, B, V, \delta, \lambda)$ определяется таблицей:

	a	b	c
v_1	$v_2, 1$	$v_1, 1$	$v_2, 0$
v_2	$v_1, 0$	$v_2, 0$	$v_1, 1$

Такая таблица удобна для её использования в компьютерной программе.►

Пример 4.2 (графический способ представление автомата его диаграммой).

Работа автомата Мили $\Gamma = (A, B, V, \delta, \lambda)$ из *примера 4.1* наглядно представляется в виде его диаграммы, показанной на рис. 10. Из этой диаграммы видно, что она является диаграммой мульти-орграфа, который можно задавать с помощью набора матриц смежности, индуцирующих соответствующий набор его бинарных отношений.►

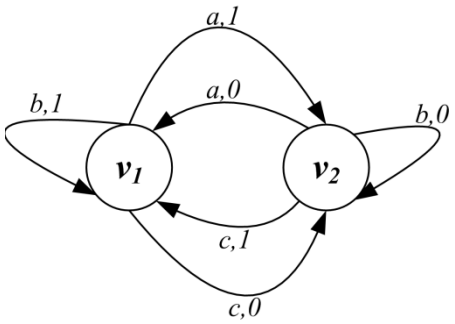


Рис.10

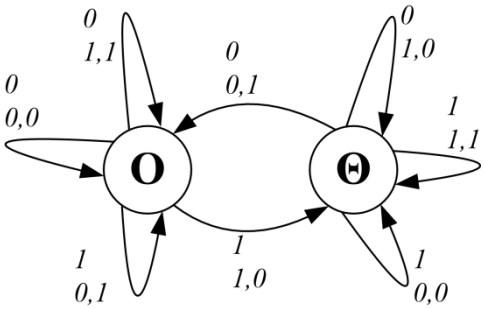


Рис.11

Пример 4.3. На рис. 11 представлена диаграмма конечного автомата Мили $\Gamma = (A, B, V, \delta, \lambda)$, реализующего *двоичный сумматор*, в котором: $A = \left\langle \begin{matrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{matrix} \right\rangle$, B =< 0,1 >, V=<O, Θ> . Протокол работы этого автомата на входное слово

$\begin{matrix} 0\,1\,0\,1\,1\,0\,1 \\ 0\,1\,1\,0\,1\,0\,1 \end{matrix}$ имеет вид:

$$\begin{array}{ccccccc} \Theta & O & \Theta & \Theta & \Theta & \Theta & O \quad -состояния \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \quad -вход \\ \hline 1 & 0 & 1 & 1 & 0 & 0 & 0 \quad -выход \end{array} .►$$

4.2. Минимизация конечного автомата Мили по состояниям

В силу *определения 4.1*, каждое состояние $v \in V$ автомата Мили $\Gamma = (A, B, V, \delta, \lambda) = (A, B, V, *, \bullet)$ индуцирует отображение $\Gamma_v : A^* \to B^*$, для которого:

$$\Gamma_v(x) = \lambda^*(x, v) = x \bullet v, \text{ если } x \in A^*. \qquad (4)$$

Такое отображение $\Gamma_v : A^* \to B^*$ называется *автоматным отображением*, индуцированным состоянием $v \in V$ автомата Γ.

Определение 4.2 (автоматно-эквивалентных состояний, приведённого автомата). Пусть $\Omega = (A, B, W, \Delta, \Lambda) = (A, B, W, *, \bullet)$ – ещё один автомат Мили с той же сигнатурой алгебраических операций, что и автомат Мили Γ. Для этого автомата не исключается и ситуация, когда $\Gamma = \Omega$. Как и для автомата Γ, для автомата Ω, его состояния $w \in W$ и буквы $a \in A$ используются обозначения:

$$\begin{cases} \Delta(a, w) = a * w; \\ \Lambda(a, w) = a \bullet w. \end{cases} \tag{5}$$

Если состояния $v \in V$ и $w \in W$ автоматов Γ и Ω таковы, что автоматные отображения $\Gamma_v : A^* \to B^*$ и $\Omega_w : A^* \to B^*$ совпадают, т.е. $\Gamma_v = \Omega_w$, то состояния $v \in V$ и $w \in W$ называются *автоматно-эквивалентными состояниями* и используется обозначение: $v \sim^a w$. Если в автомате Γ для каждого его состояния нет других состояний, ему автоматно-эквивалентных, то автомат Γ называется *приведённым автоматом*.►

Лемма 4.1. В автомате Γ отношение автоматной эквивалентности на совокупности его состояний является бинарным отношением эквивалентности, разбивающее совокупность его состояний на непересекающиеся классы автоматно-эквивалентных состояний.►

Замечание 4.1 (об автоматной эквивалентности). Минимизировать автомат Γ по состояниям это значит создать такой приведенный автомат $\Omega = (A, B, W, \Delta, \Lambda) = (A, B, W, *, \bullet)$, что для каждого состояния автомата Γ в автомате Ω есть автоматно-эквивалентное ему состояние и, наоборот. Следовательно, такой приведённый автомат Ω имеет тот же набор автоматных отображений, что и автомат Γ, т.е. $\{\Omega_w : w \in W\} = \{\Gamma_v : v \in V\}$, но в автомате Ω нет автоматно-эквивалентных состояний.►

Лемма 4.2 (о конгруэнтности автоматной эквивалентности). Если состояния $v_1, v_2 \in V$ автомата Γ автоматно-эквивалентны, т.е. $v_1 \sim^a v_2$, то для любой любого слова $x \in A^*$ состояния $\delta^*(x, v_1) = x * v_1$ и $\delta^*(x, v_2) = x * v_2$ также автоматно-эвивалентны, т.е. $x * v_1 \sim^a x * v_2$.►

Замечание 4.2. Согласно *лемме 4.2* при любом входе автоматно-эквивалентные состояния автомата Γ переходят в автоматно-эквивалентные. Следовательно, можно корректно определить автомат $\Omega = (A, B, W, \tilde{\delta}, \tilde{\lambda}) = (A, B, W, *, \bullet)$, в котором совокупность состояний $W = V/\sim^a$ состоит из классов автоматной эквивалентности состояний автомата Γ и функции $\tilde{\delta}$ и $\tilde{\lambda}$ для входной буквы $a \in A$ и состояния $\tilde{v}^a \in W = V/\sim^a$ ($\tilde{v}^a$ – класс автоматной эквивалентности с представителем $v \in V$) принимают значения:

$$\begin{cases} \tilde{\delta}(a, \tilde{v}^a) = a * \tilde{v}^a = \delta(a, v); \\ \tilde{\lambda}(a, \tilde{v}^a) = a \bullet \tilde{v}^a = \lambda(a, v). \end{cases}$$

Такой автомат $\Omega = (A, B, W, \tilde{\delta}, \tilde{\lambda}) = (A, B, W, *, \bullet)$ будет приведенным, для него используется обозначение: $\Omega = \Gamma/\sim^a$, и он называется фактор-автоматом для автомата Γ относительно автоматной эквивалентности состояний автомата Γ. Поскольку такой фактор-автомат $\Omega = \Gamma/\sim^a$ является приведённым, этот автомат можно рассматривать в качестве результата решения задачи минимизации автомата Γ по состояниям.►

4.3 Алгоритм минимизации конечного автомата Мили по состояниям

Для конечного автомата $\Gamma = (A, B, V, \delta, \lambda) = (A, B, V, *, \bullet)$ алгоритм его минимизации формулируется следующим образом.

Первый шаг. На совокупности состояний V вводим автомата Γ отношение эквивалентности $\sim^e$, полагая, что $(v_1 \sim^e v_2 \Leftrightarrow (\forall a \in A \Rightarrow a * v_1 = a * v_2))$. Это отношение индуцирует разбиение $\boldsymbol{\pi}$ совокупности состояний V на классы $\sim^e$-эквивалентности вида: $\boldsymbol{\pi} = (\pi_1, ..., \pi_{k(\boldsymbol{\pi})})$.

Второй шаг. Для компоненты $\pi_j \in \boldsymbol{\pi}$, где $j = \overline{1, k(\boldsymbol{\pi})}$, вводим отношение эквивалентности $\sim$, полагая, что состояния $v_1, v_2 \in \pi_j$ $\sim$-эквивалентны только в том случае, когда для любой буквы $a \in A$ состояния $a * v_1$ и $a * v_2$ принадлежат некоторой одной компоненте разбиения $\boldsymbol{\pi}$, зависящей от буквы $a \in A$. Тем самым индуцируется разбиение $\pi^{(j)}$ компоненты $\pi_j \in \boldsymbol{\pi}$ на классы $\sim$-эквивалентности. В результате образуется разбиение $\Pi = (\pi^{(1)}, ..., \pi^{(k(\boldsymbol{\pi}))})$ совокупности V.

Третий шаг. Если $\pi \neq \Pi$, то присваиваем разбиению π значение разбиения Π ($\pi := \Pi$) и переходим ко *второму шагу* алгоритма. Если $\pi = \Pi$, то присваиваем разбиению π значение разбиения Π ($\pi := \Pi$) и переходим к *четвёртому шагу* алгоритма.

Четвёртый шаг. Создаём автомат $\Omega = \Gamma / \sim^a = (A, B, \pi, \tilde{\delta}, \tilde{\lambda}) = (A, B, \pi, *, \bullet)$, где для состояния $v \in \pi_j$ из компоненты $\pi_j \in \pi$:

$$\begin{cases} \tilde{\delta}(a, \tilde{v}^a) = a * \tilde{v}^a = \delta(a, v); \\ \tilde{\lambda}(a, \tilde{v}^a) = a \bullet \tilde{v}^a = \lambda(a, v). \end{cases}$$

Пятый шаг. Выходим из алгоритма с результатом в виде приведённого автомата $\Omega = \Gamma / \sim^a$.

Замечание 4.3. Поскольку автомат Γ конечен, описанный выше алгоритм результативен.►

Пример 4.4. Конечный автомат Мили $\Gamma = (\langle A, B, C \rangle, \langle A, B, C \rangle, \langle 1, 2, 3, 4, 5, 6, 7, 8 \rangle, \delta, \lambda)$ задан *таблицей 1*. В этом автомате входной и выходной алфавиты совпадают и имеют вид: $\langle A, B, C \rangle$. Кроме того, $V = \langle 1, 2, 3, 4, 5, 6, 7, 8 \rangle$ – словарь его состояний.

Таблица 1

	A		B		C	
1	3	B	4	C	7	A
2	3	B	5	C	6	A
3	1	A	2	B	5	C
4	6	B	1	C	7	A
5	3	B	5	C	8	A
6	5	A	2	B	1	C
7	4	A	6	B	2	C
8	1	A	3	B	2	C

Далее для минимизации рассматриваемого автомата Γ по состояниям применяется указанный выше алгоритм минимизации.

Первый шаг. Отношение эквивалентности $\sim^e$ индуцирует разбиение $\pi = (\pi_1, \pi_2)$ совокупность состояний $V = \langle 1, 2, 3, 4, 5, 6, 7, 8 \rangle$ автомата Γ на классы $\sim^e$-эквивалентности $\pi_1 = \langle 1, 2, 4, 5 \rangle$ и $\pi_2 = \langle 3, 6, 7, 8 \rangle$, в каждом из которых буква выхода автомата определяется входной буквой, не зависящей от состояния автомата из рассматриваемого класса эквивалентности. *Таблица 2* иллюстрирует полученное разбиение $\pi = (\pi_1, \pi_2)$.

Таблица 2

	A		B		C		$\boldsymbol{\pi}$
1	3	B	4	C	7	A	π_1
2	3	B	5	C	6	A	
4	6	B	1	C	7	A	
5	3	B	5	C	8	A	
3	1	A	2	B	5	C	π_2
6	5	A	2	B	1	C	
7	4	A	6	B	2	C	
8	1	A	3	B	2	C	

Второй шаг. Отношение эквивалентности ~ индуцирует разбиения $\boldsymbol{\pi}^{(1)} = (\Pi_1)$ и $\boldsymbol{\pi}^{(2)} = (\Pi_2, \Pi_3)$ на каждом из классов $\pi_1 = \langle 1,2,4,5 \rangle$ и $\pi_2 = \langle 3,6,7,8 \rangle$, соответственно, где в каждом из классов $\Pi_1 = \langle 1,2,4,5 \rangle$, $\Pi_2 = \langle 3,6 \rangle$ и $\Pi_3 = \langle 7,8 \rangle$ буква входа автомата переводит состояние автомата из этого класса в состояние одного из этих классов π_1, или π_2, зависящего только от этой входной буквы. *Таблица 4* иллюстрирует эти переходы.

Таблица 3

	A	B	C	$\boldsymbol{\Pi}$
1	π_2	π_1	π_2	Π_1
2	π_2	π_1	π_2	
4	π_2	π_1	π_2	
5	π_2	π_1	π_2	
3	π_1	π_1	π_1	Π_2
6	π_1	π_1	π_1	
7	π_1	π_2	π_1	Π_3
8	π_1	π_2	π_1	

Таким образом, на *втором шаге* алгоритмы индуцируется разбиение $\boldsymbol{\Pi} = (\Pi_2, \Pi_2, \Pi_3)$ совокупности состояний $V = \langle 1,2,3,4,5,6,7,8 \rangle$ автомата Γ.

Третий шаг. Поскольку $\boldsymbol{\pi} \neq \boldsymbol{\Pi}$, то, согласно алгоритму, присваиваем символу $\boldsymbol{\pi}$ значение разбиения $\boldsymbol{\Pi} = (\Pi_2, \Pi_2, \Pi_3)$. Следовательно, полагаем, что $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3)$, где $\pi_1 = \Pi_1$, $\pi_2 = \Pi_2$ и $\pi_3 = \Pi_3$, и повторяем *второй шаг* алгоритма.

Второй шаг. Отношение эквивалентности ~ индуцирует разбиения $\pi^{(1)} = (\Pi_1, \Pi_2)$, $\pi^{(2)} = (\Pi_3)$ и $\pi^{(3)} = (\Pi_4)$ на каждом классов $\pi_1 = \langle 1,2,4,5 \rangle$, $\pi_2 = \langle 3,6 \rangle$ и

$\pi_2 = \langle 7,8 \rangle$, соответственно, где в каждом из классов $\Pi_1 = \langle 2 \rangle$, $\Pi_1 = \langle 1,4,5 \rangle$, $\Pi_3 = \langle 3,6 \rangle$ и $\Pi_4 = \langle 7,8 \rangle$ буква входа автомата переводит состояние автомата из этого класса в состояние одного из этих классов π_1, π_2 или π_3, зависящего только от этой входной буквы. *Таблица 4* иллюстрирует эти переходы.

Таблица 4.

	A	B	C	**Π**
2	π_2	π_1	π_2	Π_1
1	π_2	π_1	π_3	Π_2
4	π_2	π_1	π_3	
5	π_2	π_1	π_3	
3	π_1	π_1	π_1	Π_3
6	π_1	π_1	π_1	
7	π_1	π_2	π_1	Π_4
8	π_1	π_2	π_1	

Таким образом, на этом *втором шаге* алгоритмы индуцируется разбиение $\mathbf{\Pi} = (\Pi_2, \Pi_2, \Pi_3, \Pi_4)$ совокупности состояний $V = \langle 1,2,3,4,5,6,7,8 \rangle$ автомата Γ.

Третий шаг. Поскольку $\boldsymbol{\pi} \neq \mathbf{\Pi}$, то, согласно алгоритму, присваиваем символу $\boldsymbol{\pi}$ значение разбиения $\mathbf{\Pi} = (\Pi_2, \Pi_2, \Pi_3, \Pi_4)$. Следовательно, полагаем, что $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3, \pi_4)$, где $\pi_1 = \Pi_1$, $\pi_2 = \Pi_2$, $\pi_3 = \Pi_3$ и $\pi_4 = \Pi_4$, и повторяем **второй шаг** алгоритма.

Второй шаг. Отношение эквивалентности $\sim$ индуцирует разбиения $\pi^{(1)} = (\Pi_1)$, $\pi^{(2)} = (\Pi_2)$, $\pi^{(3)} = (\Pi_3)$ и $\pi^{(4)} = (\Pi_4)$ на каждом классов $\pi_1 = \langle 2 \rangle$, $\pi_2 = \langle 1,4,5 \rangle$, $\pi_3 = \langle 3,6 \rangle$ и $\pi_4 = \langle 7,8 \rangle$, соответственно, где в каждом из классов $\Pi_1 = \langle 2 \rangle$, $\Pi_1 = \langle 1,4,5 \rangle$, $\Pi_3 = \langle 3,6 \rangle$ и $\Pi_4 = \langle 7,8 \rangle$ буква входа автомата переводит состояние автомата из этого класса в состояние одного из этих классов π_1, π_2, π_3 или π_4, зависящего только от этой входной буквы. *Таблица 5* иллюстрирует эти переходы.

Таблица 5.

	A	B	C	**Π**
2	π_3	π_2	π_3	Π_1
1	π_3	π_2	π_4	Π_2
4	π_3	π_2	π_4	

5	π_3	π_2	π_4	
3	π_2	π_1	π_2	Π_3
6	π_2	π_1	π_2	
7	π_2	π_3	π_1	Π_4
8	π_2	π_3	π_1	

Таким образом, на этом *втором шаге* алгоритмы индуцируется разбиение $\boldsymbol{\Pi} = (\Pi_2, \Pi_2, \Pi_3, \Pi_4)$ совокупности состояний $V = \langle 1,2,3,4,5,6,7,8 \rangle$ автомата Γ. Поскольку $\boldsymbol{\pi} = \boldsymbol{\Pi}$, то, согласно алгоритму, присваиваем символу $\boldsymbol{\pi}$ значение разбиения $\boldsymbol{\Pi} = (\Pi_2, \Pi_2, \Pi_3, \Pi_4)$, следовательно, полагаем, что $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3, \pi_4)$, где $\pi_1 = \Pi_1$, $\pi_2 = \Pi_2$, $\pi_3 = \Pi_3$ и $\pi_4 = \Pi_4$, и переходим к **<u>четвёртому шагу</u>** алгоритма.

<u>Четвёртый шаг</u>. Создаём автомат $\Omega = \Gamma / \sim^a = (A, B, \boldsymbol{\pi}, \tilde{\delta}, \tilde{\lambda}) = (A, B, \boldsymbol{\pi}, *, \bullet)$, где для состояния $v \in \pi_j$ ($j = \overline{1,4}$) из компоненты $\pi_j \in \boldsymbol{\pi}$:

$$\begin{cases} \tilde{\delta}(a, \tilde{v}^a) = a * \tilde{v}^a = \delta(a, v); \\ \tilde{\lambda}(a, \tilde{v}^a) = a \bullet \tilde{v}^a = \lambda(a, v). \end{cases}$$

<u>Пятый шаг</u>. Выходим из алгоритма с результатом в виде приведённого автомата $\Omega = \Gamma / \sim^a$, работу которого иллюстрирует *Таблица 6*.

Таблица 6

	A		B		C	
π_1	π_3	B	π_2	C	π_3	A
π_2	π_3	B	π_2	C	π_4	A
π_3	π_2	A	π_1	B	π_2	C
π_4	π_2	A	π_3	B	π_1	C

Согласно *таблицам 1* и *6*, иллюстрирующим работу автоматов Γ и $\Omega = \Gamma / \sim^a$ настоящего примера, соответственно, на входное слово *AABCCABBC* эти автоматы, начиная с начальных состояний $1 \in V$ и π_2 ($1 \in \pi_2$ и $1,4,5 \sim^a \pi_2$), отвечают одинаковым выходным словом *BACACBBCA*. Аналогично подтверждается автоматная эквивалентность и других состояний автоматов Γ и $\Omega = \Gamma / \sim^a$ ($2 \sim^a \pi_1$, $3,6 \sim^a \pi_3$ и $7,8 \sim^a \pi_4$).►

5. Дискретные преобразователи

К одним из простейших конечных детерминированных автоматов относятся дискретные преобразователи, т.е. конечные автоматы Мили с одним состоянием. Анализу и синтезу таких дискретных преобразователей и посвящён настоящий раздел. Важность дискретных преобразователей обусловлена тем, что на их основе разрабатываются принципы синтеза конечных детерминированных автоматов. К типу дискретных преобразователей относятся логические схемы из функциональных элементов и контактные (переключательные) схемы.

Диаграммы логических схем из функциональных элементов и контактных (переключательных) схем строятся на основе орграфов, называемых логическими и контактными сетями, соответственно. Кроме того, анализ контактной схемы осуществляется на основе специального орграфа, называемого деревом анализа контактной схемы. Краткое изложение теории графов, необходимое для описания логических и контактных сетей приведено в *разделе 3*.

5.1. Логические схемы из функциональных элементов

Логическая схема из функциональных элементов реализует дискретный преобразователь, поскольку она является автоматом с одним состоянием. Входным алфавитом такой логической схемы является словарь n-мерных булевых кортежей $A = \mathbb{B}_2^n = \{(\alpha_1,...,\alpha_n) = \alpha : \alpha_1,...,\alpha_n \in \mathbb{B}_2 = \langle 0,1\rangle\}$, где $\mathbb{B}_2 = (\langle 0,1\rangle, \langle \vee, \wedge, \neg, 0,1\rangle)$ – стандартная булева алгебра (булев 1-куб). Алфавит её выхода $B = \mathbb{B}_2$.

Диаграммой логической схемы представляется в виде логической сети (см. *определение 3.3*) с n источниками и одним стоком. Каждый из n источников логической сети помечен одной из булевых переменных $x_1,....,x_n$ так, что эти пометки попарно различны для разных источников. Каждая из остальных вершин этой сети помечена одним из логических функциональных знаков: $\vee$, $\wedge$, $\neg$, $=$. При этом вершины, помеченные знаком дизъюнкции $\vee$ (конъюнкции $\wedge$), имеют позитивную валентность, равную двум; вершины, помеченные знаком логического отрицания $\neg$ (равенства $=$), имеют единичную позитивную валентность. Пометки вершин диаграммы логической схемы, иллюстрируют работу этой схемы. На первом такте работы схемы на её вход подаётся булев n-мерный кортеж $(\alpha_1,...,\alpha_n) = \alpha \in \mathbb{B}_2^n$ и булевым переменным, помечающим источники сети

логической схемы, присваиваются значения: $x_1 = \alpha_1, ..., x_n = \alpha_n$. Пометки других вершин диаграммы иллюстрируют работу функциональных элементов логической схемы: дизъюнктора (помечен знаком $\vee$), конъюнктора (помечен знаком $\wedge$), инвертора (помечен знаком $\neg$) и дублятора-задержки (помечен знаком $=$).

С помощью диаграммы логической схемы описывается её работа по реализации булевой функции $f = f(x_1, ..., x_n) : \mathbb{B}_2^n \to \mathbb{B}_2$, называемой *пропускной способностью* этой логической схемы.

На рис. 12 для логической схемы с тремя источниками на примере её диаграммы проиллюстрирована потактовая работа этой схемы, реализующей булеву функцию $f = (x_1 \vee \bar{x}_2) \wedge \bar{x}_2 \bar{x}_3$.

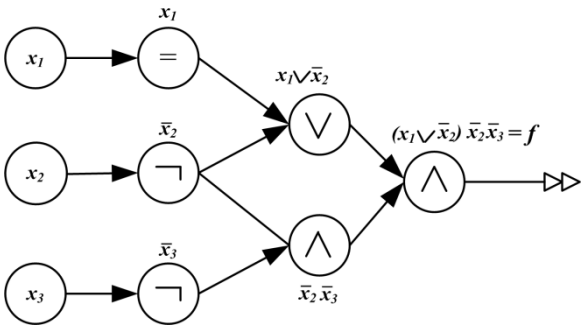


Рис.12

Замечание 5.1. Логическая схема из функциональных элементов, диаграмма которой приведена на рис. 12, работает потактово. Для соблюдения этого требования на этой диаграмме указан дублятор, иллюстрирующий задержку на один такт работы в этом функциональном элементе схемы. Кроме того, диаграмма на рис. 12 иллюстрирует методы анализа и синтеза логической схемы из функциональных элементов.►

Пример 5.1. В настоящем примере демонстрируется описание канонических уравнений конечного автомата Мили на основе булевых функций, с помощью которых работу автомата возможно реализовать логическими схемами из функциональных элементов. Для этого рассмотрим автомат $\Gamma = (A, B, V, \Delta, \Lambda)$, в котором $V = \langle v_1, v_2, v_3, v_4 \rangle$ – словарь его состояний и, кроме того, алфавиты входа и выхода совпадают и имеют вид: $A = \langle a, b \rangle = B$. Работу этого автомата описывает *таблица 7*.

Таблица 7

	a		b	
v_1	v_3	a	v_4	b
v_2	v_1	b	v_2	a
v_3	v_1	a	v_1	b
v_4	v_2	b	v_3	a

Сменим обозначения состояний и букв алфавита входа и выхода автомата $\Gamma = (A, B, V, \Delta, \Lambda)$ следующим образом. Буквы a и b заменим на символы 0 и 1, соответственно, получив в качестве алфавитов входа и выхода алфавиты $A_1 = \langle a, b \rangle = B_1$. Словарь $V = \langle v_1, v_2, v_3, v_4 \rangle$ состояний автомата $\Gamma = (A, B, V, \Delta, \Lambda)$ заменим на соответствующий словарь $V_1 = \langle (0,0), (0,1), (1,0), (1,1) \rangle$. В результате получится автомат $\Gamma_1 = (A_1, B_1, V_1, \delta, \lambda)$, фактически, реализующий автомат $\Gamma = (A, B, V, \Delta, \Lambda)$, если работа автомата $\Gamma_1 = (A_1, B_1, V_1, \boldsymbol{\delta}, \lambda)$ описывается, в соответствии с *таблицей 7*, *таблицей 8*.

Таблица 8

A_1	V_1		$\boldsymbol{\delta}$		λ
x_1	x_2	x_3	δ_1	δ_2	λ
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	0	0	1
1	1	1	1	0	0

Таким образом, работа автомата $\Gamma_1 = (A_1, B_1, V_1, \boldsymbol{\delta}, \lambda)$ описывается двумерной булевой функцией перехода по состояниям $\boldsymbol{\delta}(x_1, x_2, x_3) = (\delta_1(x_1, x_2, x_3), \delta_2(x_1, x_2, x_3))$ и булевой функцией $\lambda(x_1, x_2, x_3)$, которые определены *таблицей 8*. Из *таблицы 8* получаем:

$$\delta_1 = \overline{x}_1\overline{x}_2\overline{x}_3 \vee x_1\overline{x}_2\overline{x}_3 \vee x_1x_2x_3 = x_1x_2x_3 \vee \overline{x}_2\overline{x}_3,$$

$$\delta_2 = \overline{x}_1x_2x_3 \vee x_1\overline{x}_2\overline{x}_3 \vee x_1\overline{x}_2x_3 = x_1\overline{x}_2 \vee \overline{x}_1x_2x_3,$$

$$\lambda = \overline{x}_1\overline{x}_2x_3 \vee \overline{x}_1x_2x_3 \vee x_1\overline{x}_2\overline{x}_3 \vee x_1x_2\overline{x}_3 = \overline{x}_1x_3 \vee x_1\overline{x}_3.$$

Следовательно, автомат $\Gamma = (A, B, V, \Delta, \Lambda)$ с помощью автомата $\Gamma_1 = (A_1, B_1, V_1, \delta, \lambda)$ можно синтезировать, используя логические схемы из функциональных элементов.►

Упражнение 5.1. Синтезировать работу автомата $\Gamma_1 = (A_1, B_1, V_1, \delta, \lambda)$ из *примера 5.1* с помощью логических схем из функциональных элементов, построив для этого соответствующие диаграммы логических схем.►

5.2. Контактные схемы

Пусть $Gr(\underline{\underline{V}}; \tau \,|\, q, e)$ – контактная сеть (см. *определение 3.4*) и $\underline{\mathrm{X}} = (x_1, \overline{x_1}, x_2, \overline{x_2}, ..., x_n, \overline{x_n})$ – список булевых переменных и их отрицаний. Как и ранее, $R(V; \tau)$ – совокупность рёбер сети $Gr(\underline{\underline{V}}; \tau \,|\, q, e)$ и задано отображение $\varphi : R(V, \tau) \to \mathrm{X}$. Тогда говорят, что определена *контактная* (*переключательная*) *схема* $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$.

Рассмотрим совокупность путей $\Pi(q; e)$ в сети $Gr(\underline{\underline{V}}; \tau \,|\, q, e)$ с началом во входной вершине $q \in V$ и с окончанием в заключительной вершине $e \in V$. Путь $\pi \in \Pi(q, e)$ представляем в виде списка его последовательных ребер:

$$\pi = (\{q, v_1\}, \{v_1, v_2\}, ..., \{v_{k-2}, v_{k-1}\}, \{v_{k-1}, e\}).$$

Такому пути π поставим в соответствие конъюнкцию:

$$\varphi(\pi) = \varphi(\{q, v_1\})\varphi(\{v_1, v_2\})...\varphi(\{v_{k-1}, e\}) \in \mathbb{B}_2[x_1, x_2, ..., x_n],$$

где $\mathbb{B}_2[x_1, ..., x_n]$ – булева алгебра булевых функций от n булевых переменных $x_1, ..., x_n$. Тем самым, отображение φ расширяется до отображения $\varphi : \Pi(q, e) \to \mathbb{B}_2[x_1, x_2, ..., x_n]$.

Определение 5.1 *Пропускной способностью контактной схемы* $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$ называется булева функция $f \in \mathbb{B}_2[x_1, ..., x_n]$ вида $f = \bigvee\limits_{\pi \in \Pi(q,e)} \varphi(\pi)$.►

Но, для того чтобы, согласно *определению 5.1*, вычислить пропускную способность $f = \bigvee\limits_{\pi \in \Pi(q,e)} \varphi(\pi)$ контактной схемы $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$, необходимо рассмотреть бесконечную совокупность путей $\Pi(q, e)$. Оказывается, что для вычисления этой пропускной способности достаточно ограничиться подсовокупностью $\Pi_0(q, e)$ – простых путей

(путей, не содержащих циклы) среди путей совокупности $\Pi(q,e)$. Совокупность $\Pi_0(q,e)$, вследствие конечности контактной схемы, также является конечной и допускает, как будет показано далее, алгоритм её перебора без повторений.

Теорема 5.1 (о вычислении пропускной способности контактной схемы). Пропускная способность контактной схемы $Gr_\varphi(\underline{\underline{V}};\tau\,|\,q,e)$ вычисляется согласно формуле:

$$f = \bigvee_{\pi\in\Pi_0(q,e)} \varphi(\pi).\blacktriangleright$$

На практике возникает и другая задача. По заданной пропускной способности создать контактную схему, её реализующую. Такая задача называется *задачей синтеза контактной схемы*.

Пример 5.2 (плоско-параллельной контактной схемы). Требуется синтезировать контактную схему с пропускной способностью $f = \bar{x}_1x_2x_3 \vee x_1\bar{x}_2x_3 \vee x_1x_2\bar{x}_3 \in \mathbb{B}_2[x_1,x_2,x_3]$. На рис. 13 показана диаграмма контактной схемы с такой пропускной способностью. Эта схема называется плоско-параллельной.►

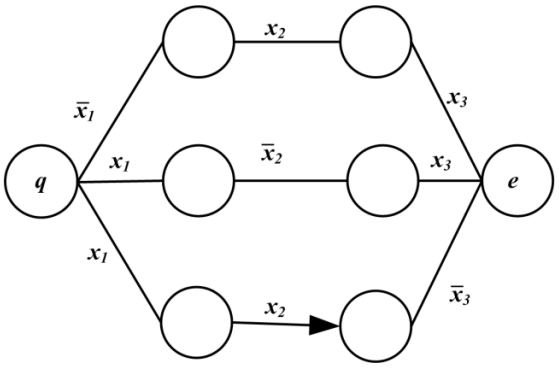


Рис.13

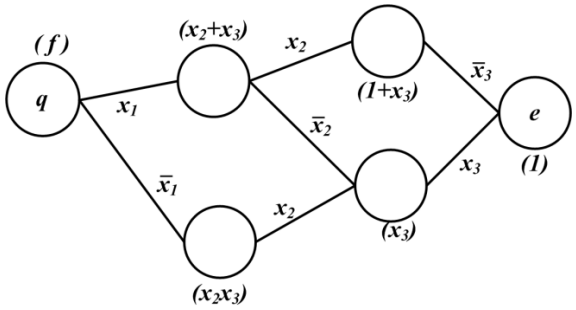


Рис.14

Для любой булевой функции, представленной её дизъюнктивной нормальной формой (ДНФ), в частности, – совершенной дизъюнктивной нормальной формой (СДНФ), плоско-параллельная контактная схема легко конструируется. Однако, такая контактная схема, как правило, не является оптимальной с точки зрения минимизации количества её вершин и рёбер. Более оптимальным синтезом контактной схемы является метод каскадов, изложению которого посвящён следующий раздел.

5.2.1. Синтез контактной схемы методом каскадов

Метод каскадов для синтеза неизвестной контактной схемы с заданной пропускной способности осуществляется индуктивно.

Пусть $f \in \mathbb{B}_2[x_1,...,x_n]$ – пропускная способность неизвестной контактной схемы $Gr_\varphi(\underline{\underline{V}};\tau\,|\,q,e)$, которую необходимо построить.

Далее предполагается, что функция f представлена полиномом Жегалкина, зависящим от всех булевых переменных x_1, x_2, …, x_n. Тогда алгоритм метода каскадов для синтеза неизвестной контактной схемы $Gr_\varphi(\underline{\underline{V}};\tau\,|\,q,e)$ имеет следующий вид.

Первый шаг. Создаем входную вершину $q \in V$ и две другие различные вершины $v_1, v_2 \in V$. Новому ребру $\{q, v_1\} \in R(V,\tau)$ ставим в соответствие перемененную x_1, новому ребру $\{q, v_2\} \in R(V,\tau)$ – переменную $\overline{x}_1$. После этого вершину v_1 помечаем полиномом Жегалкина функции $f_1(x_2,...,x_n) = f(1,x_2,...,x_n) = p_1$, вершину v_2 – полиномом Жегалкина функции $f_0(x_2,...,x_n) = f(0,x_2,...,x_n) = p_2$. Если $p_1 \equiv 0$ ($p_2 \equiv 0$), то вершину v_1 (v_2) и входящее в неё новое ребро уничтожаем. Если $p_1 \equiv 1$ ($p_2 \equiv 1$), то вершину v_1 (v_2) отождествляем с заключительной вершиной $e \in V$, т.е. полагаем $v_1 = e$ ($v_2 = e$).

……

Текущий шаг. Пусть для переменных $x_1, \overline{x}_1, ..., x_{k-1}, \overline{x}_{k-1}$, построена часть схемы $Gr_\varphi(\underline{\underline{V}};\tau\,|\,q,e)$, в которой есть вершина $v \in V$, помеченная полиномом Жегалкина p от переменных $x_k,...,x_n$ и зависящим от переменной x_k – существенно. Тогда для вершины v создаем две новые вершины $w_1, w_2 \in V$. Новое ребро $\{v, w_1\} \in R(v;\tau)$ помечаем переменной x_k, новое ребро $\{v, w_2\}$ – переменной $\overline{x}_k$. Кроме того, вершину w_1 (w_2) помечаем полиномом p_1 (p_2), полученным подстановкой в переменную x_k полинома p числа 1 (числа 0). Если оказалось, что $p_1 \equiv 0$ ($p_2 \equiv 0$), то вершину w_1 (w_2) и входящее в неё новое ребро уничтожаем. Если же $p_1 \equiv 1$ ($p_2 \equiv 1$), то вершину w_1 (w_2) отождествляем с заключительной вершиной $e \in V$, т.е. полагаем $w_1 = e$ ($w_2 = e$). Кроме того, если среди уже ранее построенных вершин есть такая вершина w, что $p_1 \equiv q$ ($p_2 \equiv q$), где q –

полином, помечающий вершину w, то в этом случае вершину w_1 (w_2) отождествляем вершиной w, т.е. полагаем $w_1 = w$ ($w_2 = w$).

……

Заключительный шаг. Процесс метода каскадов заканчивается тогда, когда становится невозможным построения новых вершин для синтезируемой контактной схемы.

Пример 5.3 (синтеза контактной схемы методом каскадов). Требуется синтезировать методом каскадов контактную схему с пропускной способностью f из *примера 5.2*. Следовательно:

$$f = \overline{x}_1 x_2 x_3 \vee x_1 \overline{x}_2 x_3 \vee x_1 x_2 \overline{x}_3 = x_1 x_2 + x_1 x_3 + x_2 x_3 + x_1 x_2 x_3,$$

где $+$ – знак операции сложения по модулю два. На рис. 14 проиллюстрирован процесс синтеза искомой контактной схемы методом каскадов. Сравнение диаграмм на рис. 13 и рис. 14 показывает, что метод каскадов является более оптимальным, чем синтез плоско-параллельной контактной схемы.►

5.2.2. Дерево анализа контактной схемы

В настоящем разделе рассматривается *задача анализа* заданной *контактной схемы* $Gr_{\varphi}(\underline{\underline{V}}; \tau \,|\, q, e)$ с неизвестной пропускной способностью $f \in \mathbb{B}_2[x_1, ..., x_n]$, которую требуется вычислить. Решение этой задачи осуществляется на основе алгоритма построения *дерева анализа контактной схемы*, диаграмма которого позволяет перечислить все простые пути этой схемы, связывающие её вход и выход и, следовательно, вычислить искомую пропускную способность заданной контактной схемы $Gr_{\varphi}(\underline{\underline{V}}; \tau \,|\, q, e)$. Алгоритм построения такого дерева анализа описывается следующим образом.

Первый шаг. Создаём корень текущего дерева анализа Drv, помечая его вершину входной вершиной q схемы $Gr_{\varphi}(\underline{\underline{V}}; \tau \,|\, q, e)$. Кроме того, из этого корня выпускаем ориентированные рёбра $q \xrightarrow{y_1} v_1, ..., q \xrightarrow{y_k} v_k$, где $v_1, ..., v_k$ – список всех попарно различных вершин, смежных в схеме $Gr_{\varphi}(\underline{\underline{V}}; \tau \,|\, q, e)$ с её входной вершиной q, и каждое ребро $\{q, v_j\} \in R(V, \tau)$ этой схемы для $j = \overline{1, k}$ помечено булевой переменной

$y_j \in \underline{X} = (x_1, \overline{x}_1, ..., x_n, \overline{x}_n)$. Кроме того, окончанием ориентированного ребра $\vec{r}_j = q \xrightarrow{y_j} v_j$ в создаваемом текущем дереве Drv является лист, помеченный вершиной v_j схемы $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$ для $j = \overline{1,k}$. Такой лист называется заключительным листом, если он помечен вершиной выхода e этой контактной схемы. Другие листы так построенного текущего дерева Drv анализа называются свободными.

Второй шаг. Если в текущем дереве анализа Drv нет свободных листов, то выходим из алгоритма с результатом в виде диаграммы этого текущего дерева. В противном случае переходим к следующему шагу алгоритма.

Третий шаг. Рассматривается свободный лист текущего дерева анализа Drv, помеченный вершиной v схемы $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$. Из этого листа выпускаем новые ориентированные рёбра $v \xrightarrow{y_1} v_1, ..., v \xrightarrow{y_k} v_k$, где $v_1, ..., v_k$ – список всех попарно различных вершин, смежных в схеме $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$ с рассматриваемым листом, и каждое ребро $\{v, v_j\} \in R(V, \tau)$ этой схемы для $j = \overline{1,k}$ помечено булевой переменной $y_j \in \underline{X} = (x_1, \overline{x}_1, ..., x_n, \overline{x}_n)$. Кроме того, для $j = \overline{1,k}$ окончанием ориентированного ребра $\vec{r}_j = q \xrightarrow{y_j} v_j$ в дополняемом текущем дереве Drv является новый лист, помеченный вершиной v_j схемы $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$. От корня дерева до этого нового листа существует единственный путь $\boldsymbol{\pi} = (q \xrightarrow{u_1} w_1, w_1 \xrightarrow{u_2} w_2, ..., v \xrightarrow{u_m} v_j)$. Если конъюнкция $\varphi(\boldsymbol{\pi}) = u_1 \wedge u_2 \wedge ... \wedge u_m \equiv 0$ или путь $(q, w_1, w_2, ..., w_{m-1}, v, v_j)$ в схеме $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$ не является простым, т.е. содержит цикл, то этот новый лист называется запрещённым и на диаграмме дополненного текущего дерева фон кружочка этого листа затемняется. Если среди оставшихся новых листов есть листы, помеченные вершиной выхода e контактной схемы $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$, то такие новые листы называются заключительными листами дополненного текущего дерева. Новый лист, не являющийся заключительным и запрещённым, называется свободным листом дополненного текущего дерева. После этого так дополненное дерево объявляется текущим деревом анализа Drv и происходит переход ко **второму шагу** алгоритма.

Пример 5.4 (анализа контактной схемы). На основе приведённого выше алгоритма для контактной схемы $Gr_\varphi(\underline{\underline{V}}; \tau \,|\, q, e)$, диаграмма которой показана на рис. 14, построено

дерево анализа этой контактной схемы. Диаграмма этого дерева анализа показана на рис. 15.

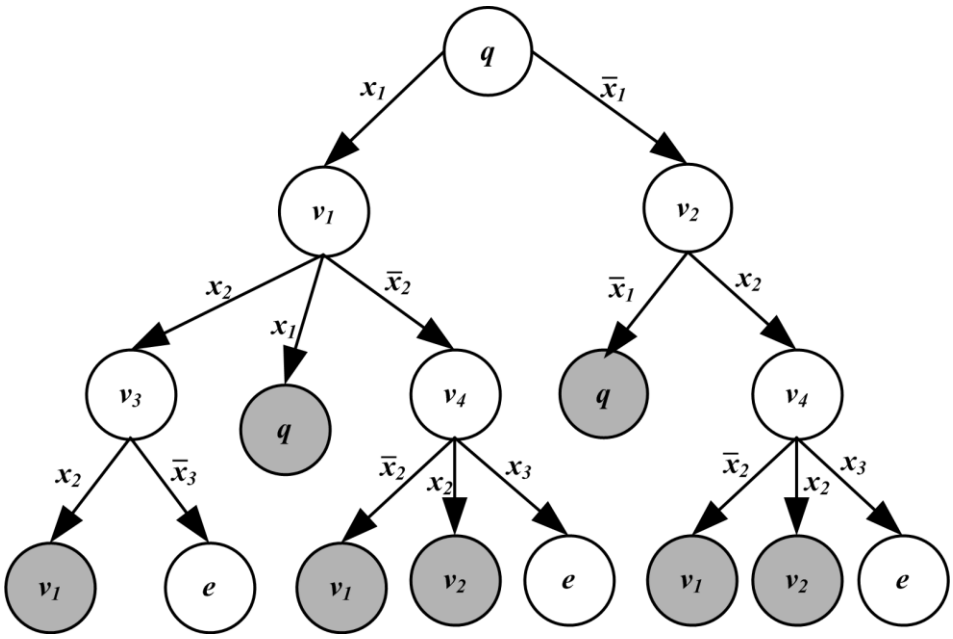


Рис.15

На диаграмме дерева анализа, показанной на рис. 15, трём заключительным листам соответствуют три пути, связывающие корень дерева с этими листьями:

$$\pi_1 = (q \xrightarrow{\overline{x_1}} v_2, v_2 \xrightarrow{x_2} v_4, v_4 \xrightarrow{x_3} e),$$

$$\pi_2 = (q \xrightarrow{x_1} v_1, v_1 \xrightarrow{\overline{x_2}} v_4, v_4 \xrightarrow{x_3} e),$$

$$\pi_3 = (q \xrightarrow{x_1} v_1, v_1 \xrightarrow{x_2} v_3, v_3 \xrightarrow{\overline{x_3}} e).$$

Эти пути можно интерпретировать и как пути в рассматриваемой контактной схеме $Gr_{\varphi}(\underline{\underline{V}}; \tau \,|\, q, e)$. Тогда $\Pi_0(q,e) = \{\pi_1, \pi_2, \pi_3\}$ – совокупность всех простых путей в этой контактной схеме. Следовательно, согласно *теореме 5.1*, её пропускная способность $f = \varphi(\pi_1) \vee \varphi(\pi_2) \vee \varphi(\pi_3) = \overline{x}_1 x_2 x_3 \vee x_1 \overline{x}_2 x_3 \vee x_1 x_2 \overline{x}_3$ и найдена правильно, согласно *примеру 5.3.*►

Упражнение 5.2. Для контактной схемы, показанной на рис. 16, построить дерево анализа и найти её пропускную способность. Затем, используя метод каскадов синтезировать контактную схему с этой пропускной способностью, сравнив её с заданной.►

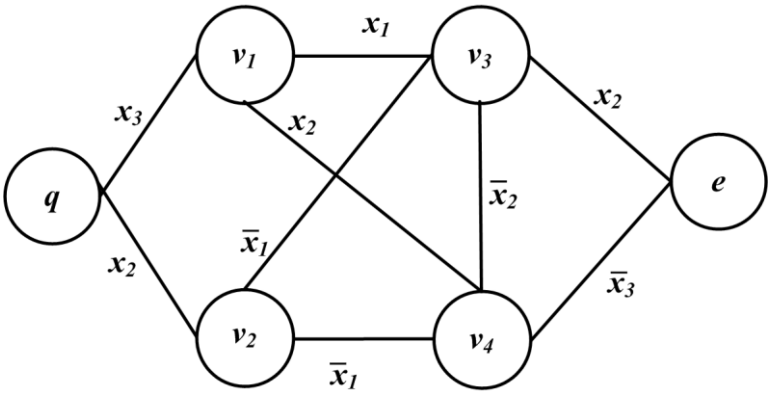


Рис.16

6. Вопросы для самоконтроля

1. Универсальный и позитивный языки алфавита. Понятие формального языка и его лексикографического словаря.
2. Вычисление номера слова в словаре позитивного языка и алгоритм определения слова по его номеру.
4. Понятие ориентированного графа и способы его задания, классификация вершин и путей в орграфе.
5. Классификация орграфов.
6. Деревья и логические сети.
7. Понятие неориентированного графа и способы его задания, классификация вершин и путей в орграфе.
8. Понятие контактной сети.
9. Понятия автоматов Мили и Мура. Классификация автоматов.
10. Способы представления конечных автоматов и описание их работы.
11. Понятия автоматного отображения и эквивалентности состояний для автоматов Мура.
12. Понятие приведённого автомата. Алгоритм минимизации конечного автомата Мура по состояниям.
13. Понятие логической схемы из функциональных элементов, её анализ и синтез.
14. Понятия контактной схемы и её пропускной способности.
15. Плоскопараллельные контактные схемы.
16. Алгоритм синтеза контактной схемы на основе метода каскадов.
17. Понятие анализа контактной схемы и вычисление её пропускной способности только на основе простых путей, связывающих её вход и выход.

18. Алгоритм построения дерева анализа контактной схемы и вычисление её пропускной способности.

7. Индивидуальные домашние задания

Задача 1

По номеру $№(x) = 1450 - 15 \cdot N$ слова x (N – номер фамилии студента в журнале группы) в лексикографическом словаре позитивного языка над упорядоченным алфавитом $A = \langle a, b, c, d, t, f \rangle$ определить это слово и проверить результат вычисления.

Задача 2

Найти СДНФ, СКНФ и полином Жегалкина булевой функции f от четырёх переменных, заданной таблицей:

X_1	X_2	X_3	X_4	f
0	0	0	0	B_1
0	0	0	1	B_2
0	0	1	0	B_3
0	0	1	1	B_4
0	1	0	0	B_5
0	1	0	1	B_6
0	1	1	0	B_7
0	1	1	1	B_8
1	0	0	0	B_9
1	0	0	1	B_{10}
1	0	1	0	B_{11}
1	0	1	1	B_{12}
1	1	0	0	B_{13}
1	1	0	1	B_{14}
1	1	1	0	B_{15}
1	1	1	1	B_{16}

В таблице $B_1B_2B_3B_4B_5B_6B_7B_8B_9\ B_{11}B_{12}B_{13}B_{14}B_{15}B_{16}$ – двоичная запись натурального числа $2^{15}+19N$, где N – номер фамилии студента в журнале группы. Если возможно, сократить запись совершенной нормальной формы и сделать проверку, используя таблицу истинности.

Задача 3

Синтезировать булеву функцию из предыдущей ***задачи 2*** с помощью логической схемы из стандартных функциональных элементов (дублятора, дизъюнктора, конъюнктора и инвертора). Провести анализ результата и представить его в виде соответствующего орграфа.

Задача 4

Методом каскадов синтезировать контактную (переключательную) схему для булевой функции из ***задачи 2***. Провести проверку результата, используя диаграмму дерева анализа, и представить его в виде соответствующего графа.

Задача 5

Для таблично заданного, инициального детерминированного автомата, с начальным состоянием 1 (см. Приложение, N – номер фамилии студента в журнале группы и номер его автомата), по входному слову AABCBBACCB найти соответствующее выходное слово и цепочку переходов состояний автомата при переработке этого входного слова (переходы по состояниям представить графически, отмечая дуги переходов по состояниям соответствующим входным символом).

Задача 6

Для таблично заданного автомата (см. Приложение, N – номер фамилии студента в журнале группы и номер его автомата), используя соответствующие булевы функции, написать канонические уравнения, его определяющие. Реализовать выход автомата в виде контактных схем из функциональных элементов, проиллюстрировав их графически и сделав их анализ с помощью дерева анализа.

Задача 7

Минимизировать по состояниям таблично заданный детерминированный автомат (см. Приложение, N – номер фамилии студента в журнале группы и номер его автомата). Результат проиллюстрировать графически, приведя диаграммы исходного и минимизированного автомата. Кроме того, выбрав входное слово, длины не менее 10, показать совпадение выходов исходного и минимизированного автомата, выбрав соответствующие начальные состояния.

Приложение

1.

	A		**B**		**C**	
1	6	A	5	B	3	B
2	1	C	4	A	6	A
3	7	A	3	B	1	B
4	5	C	2	A	7	A
5	8	A	1	B	5	B
6	2	C	8	A	3	A
7	4	C	8	A	5	A
8	4	C	6	A	1	A

2.

	A		**B**		**C**	
1	3	B	6	C	1	A
2	8	C	7	B	2	A
3	5	B	8	C	3	A
4	8	C	1	B	5	A
5	3	B	8	C	4	A
6	8	C	7	B	3	A
7	5	B	2	C	1	A
8	8	A	6	B	3	C

3.

	A		**B**		**C**	
1	2	B	5	B	2	A
2	3	C	6	A	7	B
3	5	B	8	B	8	A
4	6	A	4	C	1	C
5	1	C	4	A	7	B
6	4	A	4	C	3	C
7	1	B	2	B	4	A
8	3	C	4	A	7	B

4.

	A		**B**		**C**	
1	8	C	2	B	4	B
2	5	A	1	C	8	A
3	4	B	8	A	3	A
4	7	B	1	A	5	A
5	3	B	1	A	4	A
6	7	A	8	C	1	A
7	5	B	8	A	7	A
8	1	C	6	B	3	B

5.

	A		**B**		**C**	
1	4	C	5	B	3	A
2	5	A	7	A	8	B
3	1	A	4	A	2	B
4	7	C	2	B	1	A
5	7	C	1	B	8	A
6	4	C	6	B	2	A
7	4	C	8	B	5	A
8	6	A	7	A	3	B

6.

	A		**B**		**C**	
1	3	A	4	C	7	B
2	3	A	5	C	6	B
3	1	B	2	A	5	C
4	6	A	1	C	7	B
5	3	A	5	C	8	B
6	5	B	2	A	1	C
7	4	B	6	A	2	C
8	1	B	3	A	2	C

7.

	A		**B**		**C**	
1	6	B	5	A	3	A
2	1	C	4	B	6	B
3	7	B	3	A	1	A
4	5	C	2	B	7	B
5	8	B	1	A	5	A
6	2	C	8	B	3	B
7	4	C	8	B	5	B
8	4	C	6	B	1	B

8.

	A		**B**		**C**	
1	3	C	6	B	1	A
2	8	B	7	C	2	A
3	5	C	8	B	3	A
4	8	B	1	C	5	A
5	3	C	8	B	4	A
6	8	B	7	C	3	A
7	5	C	2	B	1	A
8	8	A	6	C	3	B

9.

	A		**B**		**C**	
1	6	A	5	C	3	C
2	1	B	4	A	6	A
3	7	A	3	C	1	C
4	5	B	2	A	7	A
5	8	A	1	C	5	C
6	2	B	8	A	3	A
7	4	B	8	A	5	A
8	4	B	6	A	1	A

10.

	A		**B**		**C**	
1	3	B	6	A	1	C
2	8	A	7	B	3	C
3	5	B	8	A	2	C
4	8	A	1	B	5	C
5	3	B	8	A	4	C
6	8	A	7	B	3	C
7	5	B	2	A	1	C
8	8	C	6	B	3	A

11.

	A		**B**		**C**	
1	2	C	5	C	2	B
2	3	A	6	B	7	C
3	5	C	8	C	8	B
4	6	B	4	A	1	A
5	1	A	4	B	7	C
6	4	B	4	A	3	A
7	1	C	2	C	4	B
8	3	A	4	B	7	C

12.

	A		**B**		**C**	
1	8	B	2	A	4	A
2	5	C	1	B	8	C
3	4	A	8	C	3	C
4	7	A	1	C	5	C
5	3	A	1	C	4	C
6	7	C	8	B	1	C
7	5	A	8	C	7	C
8	1	B	6	A	3	A

13.

	A		**B**		**C**	
1	2	A	5	A	2	B

2	3	C	6	B	7	A
3	5	A	8	A	8	B
4	6	B	4	C	1	C
5	1	C	4	B	7	A
6	4	B	4	C	3	C
7	1	A	2	A	4	B
8	3	C	4	B	7	A

14.

	A		**B**		**C**	
1	8	A	2	B	4	B
2	5	C	1	A	8	C
3	4	B	8	C	3	C
4	7	B	1	C	5	C
5	3	B	1	C	4	C
6	7	C	8	A	1	C
7	5	B	8	C	7	C
8	1	A	6	B	3	B

15.

	A		**B**		**C**	
1	4	A	5	B	3	C
2	5	C	7	C	8	B
3	1	C	4	C	2	B
4	7	A	2	B	1	C
5	7	A	1	B	8	C
6	4	A	6	B	2	C
7	4	A	8	B	5	C
8	6	C	7	C	3	B

16.

	A		**B**		**C**	
1	3	B	4	C	7	A
2	3	B	5	C	6	A
3	1	A	2	B	5	C
4	6	B	1	C	7	A
5	3	B	5	C	8	A
6	5	A	2	B	1	C
7	4	A	6	B	2	C
8	1	A	3	B	2	C

Литература

1. Бояринцева Т.Е., Щетинин А.Н., Краснов И.К. Формальные языки и конечные автоматы: Учебное пособие/Под ред. Хомича. – М.: Изд. МГТУ им. Н.Э. Баумана, 2002. – 35 сс.

2. Джон Хопкрофт, Раджив Мотвани, Джеффри Ульман. Введение в теорию автоматов и вычислений, 2-е изд.: Пер. с англ. – М.: Издательский дом «Вильямс», 2008. –528 сс.

3. Белоусов А.И., Ткачев С.Б. Дискретная математика: Учеб. для вузов/под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н. Э. Баумана, 2006. – 744 сс. (Сер. Математика в техническом университете, вып. XIX).

4. Гаврилов Г.П., Сапоженко А.А. Задачи и упражнения по дискретной математике: Учеб. Пособие. – М.:ФИЗМАТЛИТ, 2005. – 416 сс.