В. С. ВЫХОВАНЕЦ

Теория автоматов

Учебное пособие для вузов

Рекомендовано научно-методическим советом Приднестровского государственного университета в качестве учебного пособия для студентов высших технических учебных заведений

Тирасполь: РИО ПГУ, 2001

УДК 519.71 (075.8)

В88

ББК 22.18

Рецензенты:
кандидат физико-математических наук, доцент К.М. Филлипов
(Институт математики и информатики Академии наук Республики Молдова);
кандидат технических наук Т. Д. Бордя
(Приднестровский государственный университет).

Выхованец В.С. **Теория автоматов**: Учеб. пособие для вузов. – Тирасполь, РИО ПГУ, 2001. - 87 с.: ил.

В учебном пособии излагаются основы современной теории автоматов, представляющих собой одну из основных моделей управляющих систем. Рассматриваются вопросы, связанные с формальными языками и грамматиками, общей теорией алгоритмов, магазинными и конечными автоматами. Представлен прикладной аспект проектирования дискретных устройств.

Для студентов, обучающихся по специальности “Вычислительные машины, комплексы, системы и сети”, но может оказаться полезным также и студентам технических факультетов, изучающих курс “Дискретная математика”.

ISBN

Оглавление

ПРЕДИСЛОВИЕ

В последнее время инженеры, занимающиеся прикладными исследованиями, все больше используют аппарат теории автоматов. Это объясняется необходимостью создания и эксплуатации современной вычислительной техники, средств передачи и обработки информации, автоматизированных систем управления и проектирования. Наконец, в современной математической науке исследования в областях, традиционно относящихся к дискретной математике (формальные языки и грамматики, общая теория алгоритмов, теория конечных автоматов, трудоемкость вычислений и т.д.), занимают все более заметное место.

Цель создания учебного пособия – дать теоретическое введение в теорию автоматов, где под автоматом понимается любой преобразователь дискретной информации, имеющий входной и выходной каналы данных и находящийся в каждый дискретный момент времени в одном из допустимых состояний. В связи с общностью и универсальностью моделей, рассматриваемых в теории автоматов, представляется целесообразным создание учебного пособия для студентов, где основы перечисленных разделов дискретной математики излагались бы с единых позиций и в доступной форме, но достаточно полно и строго.

Данное учебное пособие включает методы решения ряда практических задач. Рассмотрен практический аспект проектирования дискретных устройств. Материал, приведенный в учебном пособии, разбит на введение и четыре главы. Во введении рассматриваются предмет и метод теории автоматов, на неформальном уровне вводятся основные понятия, обсуждается концепция порождения и распознавания. В первой главе рассматриваются формальные языки и грамматики. Вторая глава посвящена основам общей теории алгоритмов. В третьей главе изучаются конечные автоматы, а в четвертой - теоретические основы проектирования дискретных устройств.

При написании учебного пособия в значительной мере использовался опыт чтения автором курса по теории автоматов на инженерно-техническом факультете Приднестровского государственного университета им. Т. Г. Шевченко.

. ФОРМАЛЬНЫЕ ЯЗЫКИ И ГРАММАТИКИ

«В начале было слово…»
Бытие, 1,1

Определение. ***ТЕОРИЯ АВТОМАТОВ*** – это раздел дискретной математики, изучающей математические модели дискретных преобразователей информации.

Произвольный автомат будем представлять как математическую (формальную) систему, имеющую входные и выходные каналы данных и находящийся в каждый из дискретных моментов времени в одном из конечного числа состояний.

Зададим следующие множества:

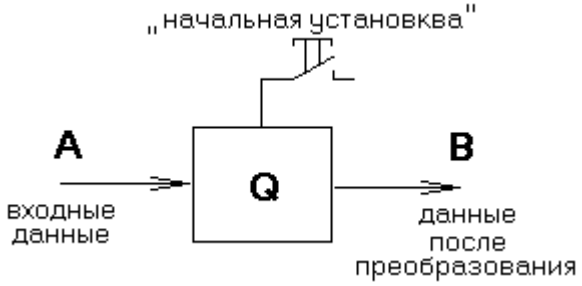
$A = \{a_1, a_2, ..., a_m\}$ - множество входных знаков (входной алфавит),

$B = \{b_1, b_2, ..., b_s\}$ - множество выходных знаков (выходной алфавит),

$Q = \{q_1, q_2, ..., q_n\}$ - множество состояний автомата.

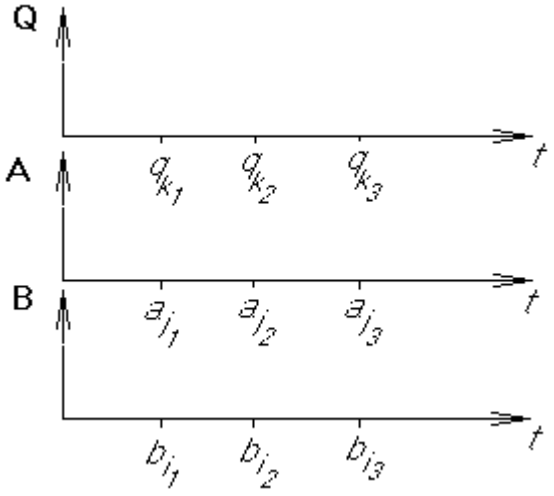
Особенность этих множеств состоит в том, что они содержат конечное число элементов.

Тогда автомат можно представить в следующем виде:



Особенностью автоматов является то, что они функционируют во времени. Работу любого автомата можно проиллюстрировать с помощью временных диаграмм.

ВРЕМЕННАЯ ДИАГРАММ РАБОТЫ АВТОМАТА.



Изначально состояние автомата неизвестно, но, подавая на вход автомата сообщение в виде последовательности знаков входного алфавита, можно понять в каком состоянии находился автомат по выходным данным, снимаемым с выхода автомата.

ПОДХОДЫ К ОПРЕДЕЛЕНИЮ И ИЗУЧЕНИЮ АВТОМАТОВ.

1 подход. ***МАКРОПОДХОД***	При макроподходе изучают внешнее поведение автомата, то есть то, как автомат перерабатывает входные данные и в каких состояниях находиться, абстрагируясь от внутренней структуры автомата. Такие автоматы называются – ***АБСТРАКТНЫМИ АВТОМАТАМИ***.
2 подход. ***МИКРОПОДХОД***	При микроподходе учитывается структура автомата, его составные части, их функционирование и связь между собой. Такие автоматы называются – ***СТРУКТУРНЫМИ АВТОМАТАМИ***.
3 подход. ***ПРАКТИЧЕСКИЙ*** или ***ПРАКТИЧЕСКИЙ АСПЕКТ***	При практическом подходе учитывается не только внешнее поведение элементарных (абстрактных) автоматов, но и особенности их реализации в виде материальных объектов (дискретных устройств).

ТЕМА 0.1. ФОРМАЛЬНЫЕ ЯЗЫКИ.

Определение. ***ЗНАК*** – это элемент конечного множества различных элементов.

Пример. +,-,*,…

Определение. ***АЛФАВИТ*** – это упорядоченное множество знаков. Порядок элементов в алфавите задает ***лексикографический порядок.***

Пример. Используется в словарях: алфавит от А до Я.

Определение. ***СИМВОЛ*** – это знак вместе с сопоставленным ему смыслом.

Пример. Код ASCII: 1 – 00110001.

Определение: ***СТРОКА*** (или ***ЦЕПОЧКА***) – это последовательность знаков некоторого алфавита. Строка характеризуется длиной, то есть числом знаков, составляющих эту строку.

Пример. Если у нас есть некоторый алфавит $A = \{0,1\}$, тогда $\alpha = 101$ - есть строка.

Определение. ***СЛОВО*** – это строка вместе с сопоставленным ей смыслом.
МНОЖЕСТВО УНИВЕРСУМ – это множество всех конечных строк в алфавите.

Определение. Если задан алфавит $A=\{a_1,a_2,...,a_n\}$, то множество универсум A^* - есть множество строк конечной длины.
Мощность множества A^* счетная, те есть все строки данного множества можно пронумеровать, хотя их число и бесконечно.

Пример. Пусть задан алфавит $B=\{0,1\}$. Определим множество универсум $B^*=\{e,0,1,00,01,10,11,000,001,...\}$.
Таким образом, проблема перечисления неразрешима, но мы можем, рассматривая некоторую строку α, сказать принадлежит она множеству универсум или нет. Следовательно, проблема распознавания алфавитов разрешима, причем за конечное число шагов.

Определение. ***МНОЖЕСТВО УНИВЕРСУМ БЕЗ ПУСТОЙ СТРОКИ***. Будем обозначать $A^+.A^+=A^*\setminus\{e\}$.

ОПЕРАЦИИ НАД СТРОКАМИ.

1. **Операция конкатенации строк (соединение строк).**

Пусть заданы строки x,y принадлежащие множеству A^* над алфавитом A.
Тогда любую строку x или y можно представить в виде последовательности знаков $x=x_1x_2...x_m, y=y_1y_2...y_s$, где $x_i,y_j\in A\ \ (i=1,...,m;\ j=1,...,s)\ \ m,s\in N$.
Тогда $x\circ y=x_1x_2...x_my_1y_2...y_s$ есть конкатенация строк x и y.

Свойства операции конкатенации.

Пусть существуют строки x,y,z принадлежащие множеству A^* над некоторым алфавитом A.

1. Конкатенация строк с пустым знаком (пустой строкой).
$$e\circ x=x\circ e=x.$$

2. Не коммутативность операции конкатенации.
$xy\neq yx$.

Пример. Пусть существуют две строки $x=ab, y=ba$, тогда $x\circ y=abba$, $y\circ x=baab$. Из примера видно что $xy\neq yx$, так как строки не равны.

3. Ассоциативность операции конкатенации.
$(x\circ y)\circ z\equiv x\circ(y\circ z)$.

Пример. Пусть существуют строки $x=ab, y=ba, z=aa$, тогда $(x\circ y)\circ z=abbaaa$, $x\circ(y\circ z)=abbaaa$. Из примера видно, что $(x\circ y)\circ z\equiv x\circ(y\circ z)$, так как строки равны.

4. Строка – есть конкатенация знаков.

$$x=x_1\circ x_2\circ...\circ x_m=\overset{m}{\underset{i=1}{\circ}}x_i.$$

Пусть задана произвольная строка x, представленная в виде конкатенации трех строк $x = \alpha \circ \beta \circ \gamma$, каждая из этих строк α, β, γ принадлежит множеству A^* над некоторым алфавитом A. Тогда:

α - **ПРЕФИКС**,
β - **ПОДСТРОКА**,
γ - **СУФИКС**.

Определение. **ФОРМАЛЬНЫМ ЯЗЫКОМ** $L(A)$ над алфавитом A называется произвольное подмножество универсального множества A^*, то есть $L(A) \subseteq A^*$.

Пример. Пусть задан алфавит $B = \{0,1\}$. Определим язык L над алфавитом B как множество двоичных чисел $0 \div 7$ и представим эти числа в следующем виде:
$L(B) = \{0,1,01,10,11,100,101,110,111\}$, тогда $L(B)$- есть язык чисел от 0 до 7.

Операции над формальными языками.

Пусть заданы три языка $L_1, L_2, L_3 \in A^*$

1. Конкатенация языка с пустым знаком.
   $\{e\} \circ L_1 = L_1 \circ \{e\} = L_1$, где операцию конкатенации для языков L_1, L_2 определим как множество строк x, y таких, что $x \in L_1, y \in L_2$, то есть
   $L_1 \circ L_2 = \{x \circ y \mid x \in L_1, y \in L_2\}$.

   Пример. Пусть существует два языка $L_1 = \{ab, a\}, L_2 = \{\alpha, \beta\alpha\}$, тогда
   $L_1 \circ L_2 = \{ab\alpha, ab\beta\alpha, a\alpha, a\alpha\beta\}$- язык заданный перечислением.

2. Некоммутативность конкатенаци языков.
   $L_1 \circ L_2 \neq L_2 \circ L_1$.

3. Ассоциативность конкатенации языков.
   $(L_1 \circ L_2) \circ L_3 \equiv L_1 \circ (L_2 \circ L_3)$.

4. Конкатенация языка с пустым множеством.
   $L_1 \circ \{\emptyset\} = \{\emptyset\} \circ L_1 = \{\emptyset\}$.

5. **Замыкание Клини.** Для произвольного языка L, заданного над некоторым алфавитом A, существует некоторый язык L^* называемый **ЗАМЫКАНИЕМ КЛИНИ**, такой, что замыкание Клини есть объединение всевозможных степеней языков L^i, где L^i - есть конкатенация языка с самим собой i раз. То есть
   $L^* = \bigcup_{i=0}^{\infty} L^i$, где $L^i = \underbrace{L \circ L \circ \ldots \circ L}_{i-раз}$, $L^0 = \{e\}$.

ТЕМА 0.2. ФОРМАЛЬНЫЕ ГРАММАТИКИ.

Формальные системы.

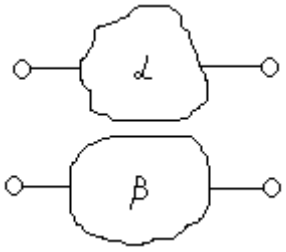
1. Зафиксируем некий конечный алфавит A. Сразу можно сказать, что данный алфавит порождает универсальное множество строк. A^*.
2. Определим **элементарные формулы,** то есть строки универсального множества A^*, которые заведомо принадлежат формальной системе.
3. Зададим **аксиомы,** то есть правила, позволяющие из элементарных формул получить произвольную строку формальной системы.
4. Зададим **правила вывода**, которые позволяют с помощью индукции получать новые строки (формулы) путем применения некоторых операций над ранее полученными строками или формулами.

Пример: Рассмотрим формальную систему – двухполюсную схему.

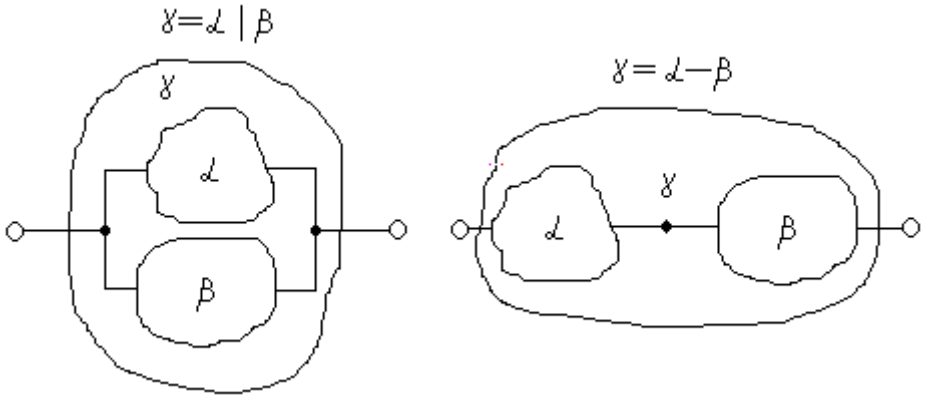
1. Определим алфавит $A = \{R, C, L, |, -, (,)\}$.
2. $\{R, C, L\}$-элементарные формулы двухполюсной системы.
3. Пусть α, β - некоторые двухполюсные схемы. Тогда можно получить двухполюсную схему как $\gamma = \{\alpha \mid \beta, \alpha - \beta\}$.

Изобразим данные схемы:

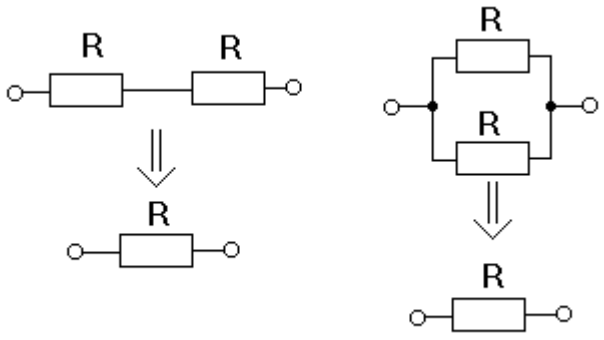
Если есть двухполюсные схемы α, β:



Тогда могут быть построены следующие схемы:



4. Смысл правил вывода не определен, но исходя из практических соображений мы можем получить следующие схемы:



$R \mid R \Leftrightarrow R, \quad L \mid L \Leftrightarrow L, \quad C \mid C \Leftrightarrow C, \quad R - C \Leftrightarrow C - R,$
$R - R \Leftrightarrow R, \; L - L \Leftrightarrow L, \; C - C \Leftrightarrow C, \quad R \mid C \Leftrightarrow C \mid R.$

Аналогично для L и C.

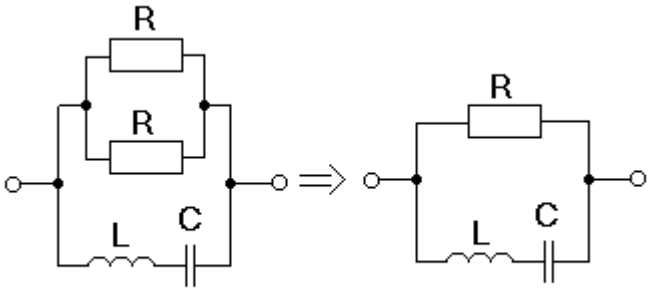
Построим произвольную двухполюсную систему $S = (R \mid R) \mid (L - C)$.

Исходя из пункта 4 получим с помощью тождественного преобразования строки

$R \to R \mid R,$
$L \to L - C.$

Получим $S = (R \mid R) \mid (L - C) \Leftrightarrow S = R \mid (L - C)$.

Изобразим эти двухполюсные схемы:



Замечание. Произвольная строка в алфавите A может не оказаться двухполюсной схемой.

Пример. $(RR-)(-L)$

Замечание. Все известные математические системы являются формальными.

Пример. Арифметика натуральных чисел, булева алгебра.

ФОРМАЛЬНЫЕ ГРАММАТИКИ.

Определение. **ФОРИАЛЬНОЙ ГРАММАТИКОЙ** G называется формальная система, состоящая из четырех объектов

$G = \langle V, W, I, P \rangle$, где

$V = \{a_1, a_2, ..., a_m\}$- алфавит терминальных знаков,

$W = \{A_1, A_2, ..., A_n\}$- алфавит нетерминальных знаков,

причем $V \bigcap W = \emptyset$,

I - аксиома или начальный нетерминальный знак $I \in W$,

P - конечное множество продукций или же правил вывода. То есть P - это всевозможные строки вида $\alpha \to \beta$ (из α выводится β), где α, β есть строки в объединенном алфавите терминальных и нетерминальных знаков (или же строки α, β принадлежат множеству универсум над объединенным алфавитом).

$P = \left\{ \alpha \to \beta \mid \alpha, \beta \in \left(V \bigcup W \right)^* \right\}$.

Есть смысл потребовать, чтобы строка α не была пустой ($\alpha \neq e$). То есть $\alpha \in \left(V \bigcup W \right)^+$, $\beta \in \left(V \bigcup W \right)^*$.

Пример. Рассмотрим грамматику симметричных строк $G = \langle V, W, I, P \rangle$.

Определим алфавиты:

$V = \{a, b\}$, $W = \{I, A, B\}$, $P = \{I \to aIa, I \to bIb, I \to e\}$. Тогда можем сделать вывод в грамматике G и убедиться что данная грамматика есть грамматика симметричных строк.

$I \to aIa \to aea = aa$,

$I \to aIa \to abIba \to abeba \to abba$,

$I \to aIa \to aaIaa \to aaaa...$

Формальные грамматики используются как конечная форма бесконечных языков.

СТРОГОЕ ОПРЕДЕЛЕНИЕ ВЫВОДА В ГРАММАТИКЕ.

Пусть есть грамматика $G = \langle V, W, I, P \rangle$ и есть вывод в грамматике $\alpha \to \beta \in P$, тогда вывод в грамматике определим индуктивно:

Шаг 0. I - есть аксиома.

Шаг k. Предположим, что в результате вывода в грамматике G получена строка $\gamma_1 \alpha \gamma_2$, где $\gamma, \gamma_1, \alpha \in \left(V \bigcup W\right)^*$, тогда $\gamma_1 \alpha \gamma_2 \to \gamma_1 \beta \gamma_2$

Шаг m. Завершаем вывод в грамматике G, когда получаем строку γ, состоящую из знаков терминального алфавита V ($\gamma \in V^*$).

Определение. **СЕНТЕНЦИАЛЬНАЯ ФОРМА ГРАММАТИКИ** G - это такая строка $\alpha \in \left(V \bigcup W\right)^*$, что существует последовательность вывода в грамматике G, оканчивающаяся строкой α.То есть $I \to \alpha_1 \to \alpha_2 \to ... \to \alpha_n \to \alpha$ ($I \xrightarrow{G} \alpha$).

Определение. **СЕНТЕНЦИЯ (ПРЕДЛОЖЕНИЕ)** грамматики называется сентенциальная форма, состоящая из знаков терминального алфавита. То есть это есть строка $\beta \mid I \xrightarrow{G} \beta, \beta \in V^*$.

Если β не есть сентенция, то мы не можем остановить вывод в грамматике G, в противном случае можем, так как в процессе вывода мы удалили все нетерминальные знаки в строке с помощью продукций грамматики G.

Определение. **ФОРМАЛЬНЫМ ЯЗЫКОМ** L, порожденным формальной грамматикой $G = \langle V, W, I, P \rangle$ $L(G)$, называется множество всех сентенций грамматики G. То есть это такие строки x в терминальном алфавите, что могут быть получены из аксиомы грамматики путем применения продукций P. $L(G) = \left\{x \in V^* \mid I \xrightarrow{G} x\right\}$.

Определение. **ЭКВИВАЛЕНТНЫМИ ГРАММАТИКАМИ** такие грамматики G_1 и G_2, которые порождают один и тот же язык, то есть множество строк равны. $L_1(G_1) = L_2(G_2)$. Обозначим как $G_1 \sim G_2$.

ФОРМЫ ЗАДАНИЯ ФОРМАЛЬНЫХ ГРАММАТИК.

1. **Формальная форма задания формальных грамматик**.

   Пример. $G = \langle V, W, I, P \rangle$.

2. **Нотация Бэкуса – Наура**.
   1. Нетерминальные знаки обозначают монятия языка и записываются в виде названия этого понятия, заключенного в скобки без пробелов $\langle ... \rangle$.
   2. Вместо знаков вывода в продукции $\rightarrow$ используется знак ::= .
   3. Знак | означает альтернативное задание правых частей продукции. Если знак | используется в алфавите языка, то альтернативное задание правых частей продукции производится на новой строке текста.

   Пример: Задание формальной грамматики целое число.

   $\langle целое \rangle ::= \langle цифра \rangle \mid \langle цифра \rangle \langle целое \rangle$

   $\langle цифра \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

ТЕМА 0.3. КЛАССИФИКАЦИЯ ГРАММАТИК

КЛАССИФИКАЦИЯ ГРАММАТИК ПО ХОМСКОМУ

Общепринятой классификацией грамматик и порождаемых ими языков является иерархия Хомского, содержащая четыре типа грамматик.

Грамматика типа 0 (произвольная) – это грамматика произвольного вида, без ограничений на вид продукций грамматики.

Продукции имеют вид $\alpha \rightarrow \beta$, где $\alpha \in (V \bigcup W)^+$, $\beta \in (V \bigcup W)^*$.

Грамматика типа 1 (контекстная грамматика) – это грамматика, все продукции которой имеют вид $\alpha A \beta \rightarrow \alpha \omega \beta$, где $\alpha, \beta \in (V \bigcup W)^*$, $A \in W$, $\omega \in (V \bigcup W)^*$. Произвольный нетерминальный знак A может быть заменен на строку ω если знак A встречается в контексте α и β, где α-левый контекст, β- правый контекст.

Пример: При анализе текста встретилось слово **коса** , понять смысл которого без начального и конечного текста невозможно (девичья коса, железная коса, речная коса).

Определение. **УКОРАЧИВАЮЩИЕ ГРАММАТИКИ** – это грамматики у которых существуют продукции вида $\alpha A \beta \rightarrow \alpha \beta$.

Определение. **НЕУКОРАЧИВАЮЩИЕ ГРАММАТИКИ** – это грамматики у которых существуют продукции вида $\alpha A\beta \to \alpha\omega\beta$, где $\omega \neq e$.

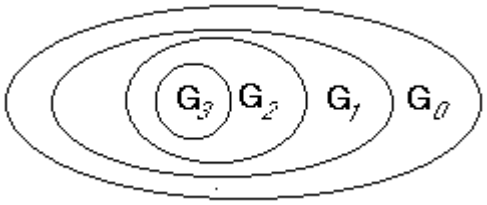
Замечание. Иногда контекстные грамматики отождествляются с неукорачивающими. Это не противоречит классификации, только приводит к тому, что укорачивающие грамматики не рассматриваются.

Грамматика типа 2 (контекстно-свободная) – это грамматика, все продукции которой имеют вид $A \to \alpha$, где $A \in W, \alpha \in \left(V \bigcup W\right)^*$.

Грамматика типа 3 (регулярная) – это грамматика, все продукции которой имеют вид $A \to aB\left(A \to Ba\right)$или $A \to a$, где $A, B \in W, a \in V$.

Язык L называется языком типа $i\left(i = 0,1,2,3\right)$,если существует порождающая его грамматика типа i.

Очевидно, что если ввести классы грамматик и обозначить всевозможные грамматики типа 0 как G_0, всевозможные грамматики типа 1 - G_1, всевозможные грамматики типа 2 - G_2, всевозможные грамматики типа 3 - G_3.Тогда получим



Предполагается, что каждый частный тип грамматики включен в состав класса граматики более общего типа.

Замечание. Один и тот же язык L может быть порожден грамматиками различных типов.

Замечание. Выразительные возможности грамматик убывают по мере увеличения номера ее типа.

ПОРАЖДЕНИЕ ЯЗЫКОВ ГРАММАТИКАМИ

Формальные языки классифицируются по типу грамматики, которая их порождает, то есть язык, порожденный грамматикой типа 0, называется **языком типа 0**. Язык, порожденный грамматикой типа 1, называется **языком типа 1**. Язык, порожденный грамматикой типа 2, называется **языком типа 2**. Язык, порожденный грамматикой типа 3, называется **языком типа 3**.

ТЕМА 0.4. КОНТЕКСТНО-СВОБОДНЫЕ ГРАММАТИКИ (КС ГРАММАТИКИ)

По определению КС грамматики могут быть как укорачивающими, так и неукорачивающими. У укорачивающих КС грамматик множество продукций P содержит

продукции вида $A \to e$. Очевидно, что никакая другая продукция не может привести к уменьшению длины строки при выводе.

Наличие у КС грамматики продукций вида $A \to e$ осложняет процедуру грамматического разбора и выводов в КС грамматике.

Продукцию вида $A \to e$ будем называть **e-ПРОДУКЦИЕЙ**, а грамматику, которая не имеет такой продукции **e-СВОБОДНОЙ (НЕУКОРАЧИВАЮЩЕЙ)**.

ТЕОРЕМА 1.1. *О e-свободной грамматике.*

Для произвольной КС грамматики $G = \langle V, W, I, P \rangle$ существует эквивалентная ей КС грамматика $G' = \langle V, W, I, P \rangle$ такая, что $L(G') = L(G) \setminus \{e\}$. То есть язык порождаемый КС грамматикой G' эквивалентен языку порождаемому КС грамматикой G с точностью до пустой строки.

Очевидно, что КС грамматика G' не содержит продукций вида $A \to e$.

ДОКАЗАТЕЛЬСТВО.

A. Пусть в продукциях P КС грамматики G имеются продукции вида $A \to e$, где $A \in W$.

B. Приведем эффективную процедуру эквивалентного преобразования КС - грамматики G в эквивалентную ей КС – грамматику G'.

Шаг 1. Представим множество продукций P КС грамматики G как объединение непересекающихся множеств P' и P'', таких что P'' содержит все e-продукции из множества продукций P.

$$P = P' \bigcup P'', P' \bigcap P'' = \emptyset, P'' = \{A \to e \mid A \in W\}.$$

Шаг 2. Фиксируем некоторую продукцию из множества P''.

Шаг 3. Добавляем в множество P' продукции из множества продукций P в которых опущены e-порождающие нетерминальные знаки A.

Шаг 4. Удаляем из множества P'' выбранную продукцию $A \to e$ и переносим из множества P' в множество P'' новые e-продукции, которые появились в множестве P' на **шаге 3**.

Шаг 5. Если все e-продукции из множества P'' просмотрены, то есть $P'' = \emptyset$, то завершаем эффективную процедуру. Иначе переходим к **шагу 2**.

C. Строим грамматику $G' = \langle V', W', I', P_1 \rangle$ следующим образом:

1. Множества V', W' содержат те терминальные и нетерминальные знаки, которые присутствуют в продукциях множества P'.
2. Аксиома $I' = I /$
3. Множество продукций P_1 содержит продукции множества P' полученного в пункте **B**.

Мы задали все четыре элемента КС грамматики G' следовательно мы полностью определили грамматику G'.

D. Приведенная в пункте **B** эффективная процедура результативна, то есть завершается за конечное число шагов так как мощность множества продукций P конечно. Может получиться, что множество $P'' = \{I \to e\}$ и на каждом **шаге 3** будет появляться новая такая же продукция в множестве P''. Это может служить еще одним условием завершения эффективной процедуры.

E. Для каждого вывода в КС грамматике G существует вывод в КС грамматике G' результатом которой будет та же строка. Верно и обратное.

$I \xrightarrow{G} \alpha, \exists I' \xrightarrow{G'} \alpha, \alpha \neq e$.

F. Если множество P'' будет состоять из продукции $I \to e$ когда мы остановили эффективную процедуру, удалив эту продукцию из P' , то тем самым мы исключили из языка пустую строку.
$L(G)\setminus\{e\}$**.** Это означает, что КС грамматика G' эквивалентна исходной КС грамматике G с точностью до пустой строки.
Окончание доказательства.

Пример. Пусть задана КС грамматика $G = \langle V, W, I, P\rangle$. $V = \{a, b\}, W = \{I, A, B\}$.

Множество продукций

$$P\begin{cases} I \to IA \mid BI \mid e, \\ A \to aBb \mid e, \\ B \to bAa \mid e. \end{cases}$$

Применим эффективную процедуру получения эквивалентной грамматики G' свободной от е-продукций.

1. $P'' = \{I \to e, A \to e, B \to e\}, P' = P \setminus P''$.
2. Зафиксируем из множества продукций P'' продукцию $I \to e$. Тогда множество $P' = P' \bigcup \{I \to A \mid B\}$. Зафиксируем из множества продукций P'' продукцию $A \to e$. Тогда множество $P' = P' \bigcup \{I \to I, B \to ba\}$. Зафиксируем из множества продукций P'' продукцию $B \to e$. Тогда множество $P' = P' \bigcup \{I \to I, A \to ab\}$.Тогда множество продукций $P'' = \{I \to IA \mid BI \mid A \mid B, A \to aBb \mid ab, B \to bAa \mid ba\}$
3. Тогда получаем эквивалентную КС грамматику $G' = \langle V', W', I', P_1\rangle$

$$V' = V, W' = W, I' = I,$$

$$P_1 = \begin{cases} I \to IA \mid BI \mid A \mid B, \\ A \to aBb \mid ab, \\ B \to bAa \mid ba. \end{cases}$$

Полученная КС грамматика эквивалентна исходной КС грамматике, так как они порождают одн и тот же язык $L(G) = L(G')$.

ЦЕПНЫЕ ПРОДУКЦИИ.

Определение. **ЦЕПНОЙ ПРОДУКЦИЕЙ** называется продукция вида $A \to B$, где $A, B \in W$.

ТЕОРЕМА 1.2. *О КС грамматике без цепных продукций.*
Для произвольной КС грамматике $G = \langle V, W, I, P\rangle$ содержащей цепные продукции вида $A \to B$, где $A, B \in W$ существует эквивалентная ей КС грамматика $G' = \langle V', W', I', P_1\rangle$ не содержащая цепных продукций.

ДОКАЗАТЕЛЬСТВО.

A. Пусть в множестве продукций P КС грамматики G имеются продукции вида $A \to B$, где $A, B \in W$.

B. Приведем эффективную процедуру эквивалентного преобразования КС грамматики G в КС грамматику G' не содержащую цепных продукций.

Шаг 1. Представим множество продукций P КС грамматики G как объединение двух непересекающихся множеств P' и P'', где множество P'' содержит все цепные продукции из множества продукций P КС грамматики G.
$P = P' \bigcup P''$, $P' \bigcap P'' = \emptyset$, где $P'' = \{A \to B\}$.

Зададим некоторое множество $P''' = \emptyset$.В множество P''' будут добавляться новые продукции, которые в последствии станут продукциями КС грамматики $G' = \langle V', W', I', P_1 \rangle$.

Шаг 2. Строим вспомогательное множество нетерминальных знаков W_A, в которое включаем все нетерминальные знаки B которые достижимы из нетерминального знака A.
$W_A = \{B \mid A \xrightarrow{G} B; B, A \in W\}$.

Шаг 3. Зафиксируем из множества P', где $P' = P \setminus P''$, продукцию вида $B \to \alpha$, где α-строка не принадлежащая нетерминальным строкам из одного знака, $B \in W$.

Шаг 4. В множество продукций P''' добавляем всевозможные продукции вида $A \to \alpha$ для всех нетерминальных знаков B принадлежащих множеству W_A.
$P''' = P''' \bigcup \{A \to \alpha \mid \forall B \in W_A\}$.

Шаг 5. Если все продукции из множества P' просмотрены, то завершаем эффективную процедуру, иначе возвращаемся к **шагу 3**.

C. Строим КС грамматику $G' = \langle V', W', I', P_1 \rangle$, эквивалентную исходной КС грамматике G, без цепных продукций, где в множества V', W' включаем те нетерминальные знаки, которые принадлежат продукциям множества $P_1 = P' \bigcup P'''$.Аксиома $I' = I$.

D. Приведенная эффективная процедура результативна, то есть завершается за конечное число шагов, так как множество продукций P исходной КС грамматики G конечно.

E. Для любого вывода в КС грамматике $G = \langle V, W, I, P \rangle$ строки α существует вывод этой строки вэквивалентной ей КС грамматике $G' = \langle V', W', I', P_1 \rangle$. Верно и обратное. То есть $L(G) = L(G')$.

Окончание доказательства.

Пример. Задана КС грамматика $G = \langle V, W, I, P \rangle$. Получить эквивалентную ей грамматику без цепных продукций.

$$P \begin{cases} I \to B \mid AB, \\ A \to B \mid C \mid bB, \\ B \to A \mid aB, \\ C \to aA \mid b \mid e. \end{cases}$$

Применем эффективную процедуру удаления цепных продукций.

Шаг 1. Разбиваем множество продукций P на множества P' и P'', таких что $P' \bigcup P'' = \emptyset$, где множество цепных продукций
$P'' = \{I \to B, A \to B \mid C, B \to A\}$.

Шаг 2. Строим вспомогательные множества W_I, куда включаем все нетерминальные знаки достижимые из нетерминального знака I.

$W_I = \{B, A, C\}$,
$W_A = \{B, C, A\}$,
$W_B = \{A, B, C\}$,
$W_C = \emptyset$.

Шаг 3. Зафиксируем продукцию $I \to AB$ из множества P'.

Шаг 4. Добавляем в множество P''' те продукции из множества продукций P у которых правая часть незменна, а левая часть является нетерминальным знаком – индексом множества, которому принадлежит исходный нетерминальный знак левой части.

Шаг 5. Если все продукции из P' просмотрены, завершаем эффективную процедуру, иначе повторяем шаги 3,4.

$A \to bB; P''' = P''' \bigcup \{I \to bB, A \to bB, B \to bB\}$,

$B \to aB; P''' = P''' \bigcup \{I \to aB, A \to aB, B \to aB\}$,

$C \to aA \mid b \mid e; P''' = P''' \bigcup \{I \to aA \mid b \mid e, A \to aA \mid b \mid e, B \to aA \mid b \mid e\}$.

В итоге получаем КС грамматику $G' = \langle V', W', I', P_1 \rangle$, где

$V' = V$,
$W' = W$,
$I' = I$,

с множеством продукций $P_1 = P' \bigcup P'''$.

$$P_1 = \begin{cases} I \to AB \mid bB \mid aB \mid aA \mid b \mid e, \\ A \to bB \mid aB \mid aA \mid b \mid e, \\ B \to aB \mid bB \mid aA \mid b \mid e, \\ C \to aA \mid b \mid e. \end{cases}$$

КС грамматика G' полностью определена и эквивалентна исходной грамматике G.

Замечание. Наличие цепных продукций в КС грамматике затрудняет вывод строк языка так как при этом возможно появление циклов, когда $A \xrightarrow{G} A$.

Замечание. При использовании цепных продукций КС грамматики выглядят более компактными, в то время как без цепных продукций мощность множества продукций P увеличивается.

ПРИВЕДЕНИЕ КС ГРАММАТИК.

Определение. **ДОСТИЖИМЫМ(ВЫВОДИМЫМ)** нетерминальным знаком называется такой нетерминальный знак A, что в грамматике $G = \langle V, W, I, P \rangle$ существует вывод $I \xrightarrow{G} \alpha A \beta$; $\alpha, \beta \in \left(V \bigcup W\right)^*$

Определение. **ПРОИЗВОДЯЩИМ** называется такой нетерминальный знак A, что в грамматике $G = \langle V, W, I, P \rangle$ существует вывод $A \xrightarrow{G} \gamma, \gamma \in V^*$.

Определение. **СУЩЕСТВЕННЫЙ** нетерминальный знак – это такой нетерминальный

знак A, который достижимый и производящий. В противном случае этот нетерминальный знак **НЕСУЩЕСТВЕННЫЙ(БЕСПОЛЕЗНЫЙ)**.

Определение. КС грамматика $G=\langle V,W,I,P\rangle$ называется **ПРИВЕДЕННОЙ** если она не содержит бесполезных нетерминальных знаков.

ТЕОРЕМА 1.3. *О приведении КС грамматик.*
Для любой КС грамматики $G=\langle V,W,I,P\rangle$ существует эквивалентная ей приведенная КС грамматика $G'=\langle V',W',I',P_1\rangle$.

ДОКАЗАТЕЛЬСТВО.

A. Приведем эффективную процедуру позволяющую выделить существенные нетерминальные знаки.

1. Выделим , используя индукцию, достижимые знаки.

Шаг 1. Пусть множество M_1 - множество нетеринальных знаков, содержащихся в правых частях продукций $I\to\alpha$.Очевидно, что все знаки из множества M_1 достижимы.

Шаг i. Пусть получено множество достижимых знаков M_i и показано, что все знаки множества M_i достижимы.

Шаг i+1. Строим множество $M_{i+1}=M_i\bigcup M_i'$, где M_i' - множество всех нетерминальных знаков из правых частей продукций вида $A\to\alpha$, где $A\in M_1$. Очевидно, что для всех нетерминальных знаков $B\in M_i$, если достижим нетерминальный знак A, то и достижим нетерминальный знак B.

Шаг k. Завершаем эффективную процедуру, если на k шаге множество M_k не изменилось после проведения итерации или M_k содержит все нетерминальные знаки $M_k=W$. Очевидно, что множество $M=M_k$ - множество достижимых нетерминальных знаков.

2. Получим, используя индукцию, множество производящих знаков Q.

Шаг 1. Пусть Q_1 - множество нетерминальных знаков A для которых существуют продукции вида $A\to\gamma$, где $\gamma\in V^*$. Очевидно, что нетерминальный знак A и все нетерминальные знаки из множества Q_1 -производящие.

Шаг i. Пусть имеется множество Q_i и показано, что все знаки из Q_i производящие.

Шаг i+1. Строим множество $Q_{i+1}=Q_i\bigcup Q_i'$, где множество Q_i' - это множество таких нетерминальных знаков A для которых существует продукция $A\to\alpha$, где $\alpha\in\left(V\bigcup Q_i\right)^*$. Очевидно, что все нетерминальные знаки из множества Q_{i+1} производящие.

Шаг k. Завершаем эффективную процедуру если на шаге k множество Q_k не изменилось после проведения итерации или Q_k содержит все нетерминальные знаки $Q_k=W$. Очевидно, что множество $Q=Q_k$ - множество производящих знаков.

3. Строим приведенную грамматику $G'=\langle V',W',I',P_1\rangle$.

Шаг 1. Получаем множество продукций P_1, исключая из множества продукций P те продукции, которые содержат бесполезные знаки.

$N = W \setminus (M \bigcup Q)$.

Шаг 2. Множество нетерминальных знаков КС грамматики G' есть множество существенных знаков, то есть тех знаков, которые как достижимые, так и производящие.

$W' = M \bigcap Q$.

Шаг 3. V' - это множество тех терминальных знаков, которые содержатся в правых частях продукций множества продукций P_1.

В. Для любого вывода в КС грамматике G существует вывод в грамматике G' и наоборот. $L(G) = L(G')$.

Окончание доказательства.

Пример. Задана КС грамматика $G = \langle V, W, I, P \rangle$. Получить эквивалентную ей КС грамматику $G' = \langle V', W', I', P_1 \rangle$ не содержащую бесполезных нетерминальных знаков.

$$P = \begin{cases} I \to bA \mid aB \mid aC, \\ A \to a \mid aI \mid bAA, \\ B \to b \mid bI \mid aBB \mid e, \\ C \to aD, \\ D \to CC. \end{cases}$$

1. Строим множество достижимых нетерминальных знаков M .

Шаг 1. Задаем начальное приближение множества достижимых нетерминальных знаков M_1, куда включаем аксиому, достижимую по определению грамматики и те нетерминальные знаки, которые непосредственно достижимы из аксиомы.

$W = \{I, A, B, C\}$.

Шаг 2. Строим множество достижимых знаков как объединение множеств M_1 и M_1', где в множество M_1' включаем те нетерминальные знаки, которые непосредственно достижимы из знаков множества M_1

$M_2 = M_1 \bigcup M_1'; M_1' = \{I, A, B, D, C\} \Rightarrow M_2 = \{I, A, B, D, C\}$.

Шаг 3. Завершаем эффективную процедуру, когда множество M_i на текущем шаге итерации оказалось равным множеству на предыдущем шаге итерации

$M = M_2 = W$.

2. Строим множество производящих знаков Q.

Шаг 1. Задаем начальное приближение множества производящих знаков Q_1 куда включаем те нетерминальные знаки, которые непосредственно порождают терминальные строки

$Q = \{A, B\}$.

Шаг 2. Следующее приближение множества производящих знаков строим как $Q_2 = Q_1 \bigcup Q_1'$, где в множество Q_1' включаем те нетерминальные знаки,

которые непосредственно порождают стоки, состоящие из терминальных знаков и знаков множества Q_1

$Q_2 = \{I, A, B\}$.

Шаг 3. Повторяем шаг 2 до тех пор, пока множество Q_i на текущем шаге итерации не сравняется с множеством Q_{i-1} на предыдущем шаге итерации или когда $Q = W$

$Q_3 = Q_2 \bigcup Q_2'$, $Q_2 = \{I, A, B\}$.Так как $Q_3 = Q_2$, тогда $Q = \{I, A, B\}$.

3. Строим приведенную грамматику $G' = \langle V', W', I', P_1 \rangle$.

А) Определяем множество существенных знаков как $W' = Q \bigcap M = \{I, A, B\}$.

Б) В множество продукций P_1 включаем те продукции из множества продукций P, которые не содержат бесполезных знаков $N = W \setminus W'$

$N = \{C, D\}$.

$$P_1 = \begin{cases} I \to bA \mid aB, \\ A \to a \mid aI \mid bAA, \\ B \to b \mid bI \mid aBB \mid e. \end{cases}$$

В) Определим множество терминальных знаков V' куда включаем те терминальные знаки из множества V, которые встречаются в продукциях множества P_1

$V' = \{a, b\}$.

Замечание. Если аксиома I грамматики $G = \langle V, W, I, P \rangle$ не производящий знак, то язык $L(G)$ пуст. Аксиома I является достижимым знаком по определению, так как любой вывод в грамматике G начинается с аксиомы. Следовательно, для того, чтобы язык $L(G)$ содержал строки, аксиома I должна быть производящим нетерминальным знаком.

ТЕМА 0.5. РАЗРЕШИМОСТЬ ЯЗЫКОВ.

Под **РАЗРЕШИМОСТЬЮ ЯЗЫКОВ** понимается распознавание принадлежности произвольной строки $\alpha \in V^*$ языку, заданному формальной грамматикой $G = \langle V, W, I, P \rangle$.

ТЕОРЕМА 1.4. *О разрешимости КС языков.*
Если $G = \langle V, W, I, P \rangle$ КС грамматика, то разрешима проблема определения принадлежнос-ти произвольной строки α в терминалном алфавите языку $L(G)$, порожденному этой грамматикой.

ДОКАЗАТЕЛСТВО.

А. Рассмотрим КС грамматику $G = \langle V, W, I, P \rangle$ продукции которой имеют вид $A \to \alpha$, где $A \in W, \alpha \in \left(V \bigcup W\right)^*$. Рассмотрим систему множеств X_A таких, что $X_A \subset V^*, A \in W$, которые состоят из терминальных строк, выводимых из знака A в КС грамматике G.

В. Введем операции над множествами X_A.

1. Конкатенация множеств $X_A \circ X_B = \{\alpha \circ \beta \mid \alpha \in X_A, \beta \in X_B\}$.
2. Объединение множеств $X_A \bigcup X_B = \{\alpha \mid \alpha \in X_A \text{ или } \beta \in X_B\}$.

Свойства операции над множествами X_A.

1. Некоммутативность конкатенации $X_A X_B \neq X_B X_A$.
2. Коммутативность объединения $X_A \bigcup X_B = X_B \bigcup X_A$.
3. Неидемпотентность конкатенации $X_A X_A \neq X_A$.
4. Идемпотентность объединения $X_A \bigcup X_A = X_A$.
5. Ассоциативность конкатенации и объединения $(X_A X_B)X_C = X_A(X_B X_C)$, $(X_A \bigcup X_B)\bigcup X_C = X_A \bigcup (X_B \bigcup X_C)$.
6. Дистрибутивность объединения относительно конкатенации $(X_A \bigcup X_B)X_C = X_A X_C \bigcup X_B X_C, X_C(X_A \bigcup X_B) = X_C X_A \bigcup X_C X_B$.
7. Действия с универсальным, пустым множествами и множеством, состоящим из пустой строки
   $X_A \circ \emptyset = \emptyset \circ X_A = \emptyset, X_A \circ \{e\} = \{e\} \circ X_A = X_A, X_A \circ V^* = V^*$,
   $X_A \bigcup \emptyset = X_A, X_A \bigcup \{e\} = \{e\} \bigcup X_A = X_A, X_A \bigcup V^* = V^*$.

C. Рассмотрим систему уравнений относительно введенных множеств X_A в КС грамматике $G = \langle V, W, I, P \rangle$

$$\begin{cases} X_I = f_I(X_I, X_A, ..., X_Z), \\ X_A = f_A(X_I, X_A, ..., X_Z), \\ \vdots \\ X_Z = f_Z(X_I, X_A, ..., X_Z). \end{cases}$$

Систему уравнений составим по следующим правилам: Если в множестве продукций P КС грамматики $G = \langle V, W, I, P \rangle$ существует группа продукций с терминальным знаком A вида $A \to \alpha \mid \beta$, где α, β представим как конкатенацию нетерминальных и терминальных знаков α_i, β_i

$\alpha = \alpha_1 \alpha_2 ... \alpha_n, \beta = \beta_1 \beta_2 ... \beta_m; \alpha_i \beta_j \subset V \bigcup W$, тогда получим уравнение $X_A = X_{\alpha 1} X_{\alpha 2} ... X_{\alpha n} \bigcup X_{\beta_1} X_{\beta_2} ... X_{\beta_m}$, в котором $X_\alpha = \begin{cases} X_\alpha, если \alpha \in W, \\ \{\alpha\}, если \alpha \in V. \end{cases}$

Пример: Если в множестве продукций P КС грамматики $G = \langle V, W, I, P \rangle$ есть продукция $B \to AacB \mid CB \mid Aa$, то уравнение для B имее вид $X_B = X_A\{ac\}X_B \bigcup X_C X_B \bigcup X_A\{a\}$.

Замечание. При выводе уравнения учтено, что $\{a\}\{c\} = \{ac\}$, то есть конкатенация множеств, состояших из одного знака есть множество, включающее единственную строку, равную конкатенации знаков этих множеств (следует из свойств операции конкатенации).

D. Найдем решение системы уравнений $X = F(X)$, где $X = (X_I, X_A, ..., X_Z)$ - вектор множеств с числом компонентов равным числу нетерминальных знаков.

Свойство решения системы уравнений $X = F(X)$**.** Если существует система множеств T такая, что является подмножеством другой системы множеств U, то $F(T) \subset F(U)$. Покажем это.

Так как T является подмножеством U, то $U = T \bigcup S$, где $T \bigcap S = \emptyset$.

Произвольное уравнение системы представим в виде

$f(X) = \bigcup_K X_{\alpha_1} X_{\alpha_2} ... X_{\alpha_K}$, тогда $f(U) = f(T \bigcup S) = \bigcup_K (T_{\alpha_1} \bigcup S_{\alpha_1})(T_{\alpha_2} \bigcup S_{\alpha_2})...(T_{\alpha_K} \bigcup S_{\alpha_K})$,

где учтено, что

$T = (T_I, T_A, ..., T_Z)$ и $S = (S_I, S_A, ..., S_Z)$. Следовательно

$f(U) = \left(\bigcup_K T_{\alpha_1} T_{\alpha_2} ... T_{\alpha_K} \right) \bigcup \left(\bigcup_K S_{\alpha_1} S_{\alpha_2} ... S_{\alpha_K} \right) = F(T) \bigcup F(S)$. Последнее выражение получено путем использования дистрибутивности, конкатенации и объединения и с учетом того, что $T_i S_j = \emptyset$, $\forall i \neq j$ так как множества T и S не пересекаются. Тогда $f(T) \subset f(U)$, в векторном виде $F(T) \subset F(U)$.

Получим решение системы уравнений $X = F(X)$.

Пусть решение системы $X = F(X)$ будет Y. Тогда $Y = F(Y)$. Известно, что $\emptyset \subset Y$, тогда $F(\emptyset) \subset F(Y) = Y \Rightarrow F(F(\emptyset)) \subset F(Y) = Y$. Продолжая далее получим $F^i(\emptyset) \subset Y \Rightarrow F^{i+1}(\emptyset) \subset F(Y) = Y$. Следовательно, решением уравнения $X = F(X)$ является

$$Y = \bigcup_{i=0}^{\infty} F^i(\emptyset).$$

$$F^i(\emptyset) = \underbrace{F(F(....F(\emptyset)...))}.$$

Замечание. Множество строк принадлежащих языку $L(G)$ находится в множестве X_I. Все остальные множества являются вспомогательными для построения множества X_I

E. Приведем эффективную процедуру определения принадлежности строки γ языку $L(G)$, порожденному КС грамматикой $G = \langle V, W, I, P \rangle$.

Шаг 1. Строим систему уравнений $X = F(X)$ как это было описано в пункте **C**.

Шаг 2. Задаем начальное приближение решения системы уравнений $X = F(X)$**.**

$X^{(0)} = (\emptyset, \emptyset, ..., \emptyset) = \emptyset$.

Шаг i. Пусть имеется частное решение системы уравнений $X = F(X)$

$X^{(i)} = \left(X_I^{(i)}, X_A^{(i)}, ..., X_Z^{(i)} \right)$.

Шаг i+1. Строим следующее приближение решения систем уравнений $X = F(X)$ в виде

$X^{(i+1)} = F\left(X^{(i)} \right)$**.**

Удаляем из множества $X^{(i+1)}$ все строки, которые не могу быть подстроками искомой строки γ.

Замечание. Действительно, если при выводе в КС грамматике $G = \langle V, W, I, P \rangle$ получена синтенциальная форма, состоящая из нетерминальных знаков и терминальных подстрок γ_i, которая приводит к строке γ, то γ_i является подстрокой строки γ.

$I \xrightarrow{G} \gamma_1 A_1 \gamma_1 A_2 ... \gamma_2 A_1 \gamma_{K-1} A_{K-1}; \gamma_K \xrightarrow{G} \gamma$

Шаг k. Завершаем процедуру на этом шаге, если $\gamma \in X_I$, то есть γ является строкой языка, или, когда $X^{(i+1)} = X^{(i)}$(в этом случае строка не принадлежит языку).

F. Так как число уравнений n в системе уравнений $X = F(X)$ конечно и длина строки γ ограничено и равно m ,то решение поставленной задачи о разрешимости КС языка $G = \langle V, W, I, P \rangle$ для произвольной строки γ получаем не более чем за $n \times m$ шагов. Это утверждение основано на том, что КС грамматика $G = \langle V, W, I, P \rangle$ является неукорачивающей и при проведении итераций длины строк растут. Таким образом проблема принадлежности произвольной строки γ языку $L(G)$ разрешима.

Окончание доказательства.

Пример. Определить, принадлежит ли строка $\gamma = abbaba$ языку $L(G)$, порожденному КС грамматикой $G = \langle V, W, I, P \rangle$.

$V = \{a, b\}, W = I, A, B$,

$$P = \begin{cases} I \to AI \mid BI \mid a, \\ A \to BB \mid a, \\ B \to AA \mid b. \end{cases}$$

Применим эффективную процедуру определения принадлежности строки γ языку $L(G)$ порожденному КС грамматикой G.

Шаг 1. Строим систему уравнений относительно множеств $X = (X_I, X_A, ..., X_Z)$, где X_A содержит терминальные строки, выводимые из знака A.

$$\begin{cases} X_I = X_A X_I \bigcup X_B X_I \bigcup \{a\}, \\ X_A = X_B X_B \bigcup \{a\}, \\ X_B = X_A X_A \bigcup \{b\}. \end{cases}$$

Шаг 2. Задаем начальное приближение решения системы уравнений $X^{(0)} = (\emptyset, \emptyset, \emptyset)$.

Шаг 3. Находим следующее приближение системы уравнений

$X_I^{(1)} = \{a\}$,

$X_A^{(1)} = \{a\}$,

$X_B^{(1)} = \{b\}$.

Удаляем из полученного множества X те строки, которые не являются подстроками исходной строки γ.

Шаг 4. Повторяем шаг 3 до тех пор, пока искомая строка γ не появится в множестве $X^{(i)}$, или система множеств на текущем шаге итерации не совпадет с системой множеств на предыдущем шаге итерации (строка не принадлежит языку $L(G)$.

$X_I^{(2)} = \{\sout{aa}, ba, a\}$,

A) $X_A^{(2)} = \{bb, a\}$,

$X_B^{(2)} = \{\sout{aa}, b\}$.

$X_I^{(3)} = \{\sout{bbba}, bba, aba, \sout{aa}, ba, a\}$,

B) $X_A^{(3)} = \{bb, a\}$,

$X_B^{(3)} = \{\sout{bbbb}, bba, abb, \sout{aa}, b\}$.

C) $X_I^{(4)} = \{bbbba, bbaba, bbba, bba, abba, aabaq, aba, aa, bbabba,$
$bbaaba, bbaba, bbaa, abbbba, \underline{abbaba}, abbba, baba, bba, ba, a\}$.

Строка $\gamma = abbaba$ принадлежит языку $L(G)$, порожденному КС грамматикой $G = \langle V, W, I, P \rangle$.

ТЕОРЕМА 1.5. *О разрешимости проблемы пустоты КС языка.*
Если $G = \langle V, W, I, P \rangle$ - КС грамматика, то разрешима проблема пустоты языка $L(G)$.

ДОКАЗАТЕЛЬСТВО.
Применим к КС грамматике $G = \langle V, W, I, P \rangle$ эффективную процедуру приведения КС грамматики. Если получена КС грамматика $G' = \langle V', W', I', P_1 \rangle$ у которой аксиома I' - производящий знак, то **язык** $L(G)$ **не пуст**. В противном случае язык $L(G)$ **пуст**.
Окончание доказательства.

ТЕОРЕМА 1.6. *О разрешимости проблемы бесконечности КС языка.*
Если $G = \langle V, W, I, P \rangle$ КС грамматика, то разрешима проблема определения бесконечности языка $L(G)$, порожденного этой грамматикой.

ДОКАЗАТЕЛЬСТВО.
Приведем эффективную процедуру определения бесконечности языка $L(G)$.
Достаточно построить такой вывод в КС грамматике $G = \langle V, W, I, P \rangle$ из досижимого и производящего нетерминального знака A, что
$A \xrightarrow{G} \alpha A \beta, где \alpha \neq e, \beta \neq e; \alpha, \beta \in (V \bigcup W)^*$.

Достаточность.
Так как A - производящий знак, то языку $L(G)$ принадлежат всевозможные строки γ у которых строки $\alpha^n A \beta^n$ являются подстроками. Число таких подстрок счетно. Следовательно заключаем, что число строк в языке $L(G)$ счетно или же бесконечно.

Необходимость.
Если таких знаков A нет, то ни в одном выводе в КС грамматике $G = \langle V, W, I, P \rangle$ нетерминальные знаки не повторяются. Так как число нетерминальных знаков конечно и конечно число продукций, то число строк в языке $L(G)$ ограничено.
Окончание доказательства.

ТЕОРЕМА 1.7. *О разрешимости языков, заданных контекстными грамматиками.*
Если грамматика $G = \langle V, W, I, P \rangle$ - контекстная, неукорачивающая грамматика, то язык $L(G)$ разрешим.

ДОКАЗАТЕЛЬСТВО.

A. Разрешимость языка означает, что для наперед неизвестной строки γ и грамматики $G = \langle V, W, I, P \rangle$ существует эффективная процедура определения принадлежности строки γ языку $L(G)$.

B. Так как грамматика G - неукорачивающая, то количество всевозможных выводов в грамматике G строк, с длинной, не превосходящей длину искомой строки, конечно. Это означает, что построив всевозможные выводы строк с длиной не превосходящей

длину строки γ, путем простого перебора легко обнаружить или не обнаружить строку γ.

Таким образом приведенная процедура эффективна, то есть требует конечного объема памяти и реализуется за конечное время (конечное число шагов).

С. Заметим, что для КС грамматик требование неукорачиваемости строк при выводе не предъявлялось. Для контекстных грамматик это требование необходимо так как если в контекстной грамматике сеществуют укорачивающие продукции, то число всевозможных выводов в контекстной грамматике строк заданной длины может быть счетно.

Окончание доказательства.

ТЕОРЕМА 1.8. *О неукорачивающих грамматиках.*

Если грамматика $G = \langle V, W, I, P \rangle$ - неукорачивающая, то есть продукции грамматики G имеют вид $\alpha \to \beta, где |\alpha| \le |\beta|$, то существует эквивалентная ей контекстная грамматика $G' = \langle V', W', I', P_1 \rangle$.

Если грамматика $G = \langle V, W, I, P \rangle$ - неукорачивающая, то в соответствии с теоремами 1.7 и 1.8 язык $L(G)$, порожденный этой грамматикой, разрешим.

ТЕМА 0.6. НЕРАЗРЕШИМЫЕ ПРОБЛЕМЫ ДЛЯ ГРАММАТИКИ.

Определение. **НЕРАЗРЕШИМОЙ ПРОБЛЕМОЙ** называется отсутствие эффективной процедуры решения некоторой задачи для наперед неизвестной грамматики $G = \langle V, W, I, P \rangle$.

Основным методом доказательства неразрешимостей в теории формальных систем является сведение задачи к комбинаторной проблеме Поста.

ЛЕМА 1.9. *О комбинаторной проблеме Поста.*

Комбинаторная проблема Поста неразрешима. Для двух различных списков строк $X = (\alpha_1, \alpha_2, ..., \alpha_m) и Y = (\beta_1, \beta_2, ..., \beta_m)$ длины m, в алфавите U, при достаточно больших $m (m >> 1)$ существует ли последовательность индексов $i_1, i_2, ..., i_n; n \in N$ конечной длины такая, что $\alpha_{i_1}, \alpha_{i_2}, ..., \alpha_{i_n} = \beta_{i_1}, \beta_{i_2}, ..., \beta_{i_n}$: эти строки равны – является неразрешимой проблемой.

ДОКАЗАТЕЛЬСТВО.

А. Неразрешимость означает, что для наперед нам неизвестных список строк XuY не существует эффективной процедуры, которая за конечное число шагов и при конечной памяти может построить последовательность индексов $i_1, i_2, ..., i_n; n \in N$ конечной длины.

Пример. Пусть есть два списка строк

$X = (a, aba, ab),$

$Y = (b, bb, aba).$

Проиндексируем члены этих списков и возьмем в качестве строк строки, индекс

Которых равен 3. $\alpha = ab$ и $\beta = aba$. Выберем строки с индексами 3 и 1 и объединим с помощью операции конкатенации эти строки. Получим $\alpha = aba, \beta = abab$. Выберем строки с индексами 3 и 2. Получим

$\alpha = ababba, \beta = ababb$ Строки α и β не равны, следовательно, проблема Поста не разрешима.

B. Докажем утверждение леммы 1.9 в общем случае.

Число строк вида $\{\alpha_{i_1}, \alpha_{i_2}, ..., \alpha_{i_n}\} u \{\beta_{i_1}, \beta_{i_2}, ..., \beta_{i_n}\}$ счетно. Эти множества являются множествами строк конечной длины в конечном алфавите, длины этих строк определяются конечной последовательностью индексов $i_1, i_2, ..., i_n; n \in N$. Отсюда заключаем, что для решения комбинаторной проблемы Поста необходимо выбрать и сравнить сответствующие строки из двух счетных множеств, порожденных списками X и Y.

C. Так как списки X и Y наперед неизвестны, то есть не заданы, то гарантировать, что за конечное число шагов будут найдены совпадающие строки невозможно.

Окончание доказательства.

ЛЕММА 1.10. *Комбинаторная проблема Поста для КС языков, порожденных списками строк.*

Если $X = (\alpha_1, \alpha_2, ..., \alpha_m) u Y = (\beta_1, \beta_2, ..., \beta_m)$ - списки строк конечной длины m в алфавите U, то комбинаторная проблема Поста для этих списков имеет решение. То есть, если языки $L(X) \bigcap L(Y) = \emptyset$, где $L(X) u L(Y)$ - это КС языки, включающие всевозможные строки вида $\gamma \in L(X), \gamma \in L(Y)$, где $\gamma = b_{i_n} ... b_{i_1} \alpha_{i_1} ... \alpha_{i_n}$, $\gamma = b_{i_n} ... b_{i_1} \beta_{i_1} ... \beta_{i_n}$, здесь $b_i \in B$ - знаки терминального алфавита B, состоящего из m знаков $B = \{b_1, b_2, ..., b_m\}. B \bigcap U = \emptyset$.

ДОКАЗАТЕЛЬСТВО.

A. Пусть строка γ принадлежит языку $L(X)$ и $L(Y)$, тогда $b_{i_n} ... b_{i_1} \alpha_{i_1} ... \alpha_{i_n} = b_{i_n} ... b_{i_1} \beta_{i_1} ... \beta_{i_n}$ то есть $\alpha_{i_1} ... \alpha_{i_n} = \beta_{i_1} ... \beta_{i_n}$, то есть построена последовательность индексов для произвольных списков строк XuY. Но языки $L(X)$ и $L(Y)$ не пересекаются.

B. Покажем, что языки $L(X)$ и $L(Y)$ порождаются КС грамматикой $L(X) = L(G_X)$ $(L(Y) = L(G_Y))$, где G_X, G_Y - КС грамматики. Для этого построим продукции грамматики следующего вида $P_X = \{I \to b_i I_X \alpha_i, I_X \to e \mid i = 1...m\}$, $P_Y = \{I \to b_i I_Y \alpha_i, I_Y \to e \mid i = 1...m\}$, очевидно, что эти граматики контекстно – свободные и порождают языки $L(X)$ и $L(Y)$ соответственно.

C. Грамматики G_X, G_Y являются **однозначными**, то есть для каждой строки языка γ существует единственная последовательность ее вывода.

Окончание доказательства.

ТЕОРЕМА 1.11. *О неразрешимых проблемах для КС грамматик.*

Если G_1 и G_2 КС грамматики, то неразрешимыми проблемами являются:

1. Пусто ли пересечение КС языков? То есть $L(G_1) \bigcap L(G_2) = \emptyset$, или нет?
2. Является ли КС языком пересечение КС языков? То есть если G_1 и G_2 КС грамматики и $L(G_1) \bigcap L(G_2) = L(G_3)$, G_3 -КС язык или нет?
3. Является ли КС языком дополнение КС языка? То есть если G_1 - КС грамматика, а V^* - дополнение к КС языку, порожденному грамматикой G_1 и $V^* \setminus L(G_1) = L(G_2)$, то является ли КС грамматикой грамматика G_2?
4. Тривиален ли КС язык? То есть $L(G_1) = V^*$?

5. Эквивалентны ли две КС грамматики $G_1 \sim G_2$, то есть совпадают ли языки, ими порождаемые $L(G_1) = L(G_2)$?
6. Однозначна ли КС грамматика или нет? То есть существует ли единственный вывод в КС грамматике?

ДОКАЗАТЕЛЬСТВО.

A. Если бы проблему пустоты пересечения КС языков можно было бы решить, то можно было бы определить пусто ли пересечение языков $L(X)$ и $L(Y)$ в лемме 1.10. Так как комбинаторная проблема Поста к которой сводится решение данной задачи неразрешима, то и не разрешима проблема пустоты пересечения двух КС языков. Что и требовалось доказать.

B. Пересечение КС языков не обязательно является КС языком, поскольку для КС языков проблема пустоты пересечения КС языков не разрешима (см. пункт А). Следовательно, данная проблема не разрешима. Что и требовалось доказать.

C. Для доказательства пункта 3 теоремы 1.11 предварительно покажем, что объединение двух КС языков есть КС язык.

Пусть заданы две КС грамматики $G_1 = \langle V, W_1, I_1, P_1 \rangle$ и $G_2 = \langle V_2, W_2, I_2, P_2 \rangle$ тогда язык L есть объединение языков, порожденных грамматиками G_1 и G_2. Язык L порождается грамматикой $G = \langle V_1 \bigcup V_2, \{I\} \bigcup W_1 \bigcup W_2, I, P_1 \bigcup P_2 \bigcup \{I \to I_1 \mid I_2\} \rangle$, где $\{I\}$- новая аксиома, то есть $L(G) = L(G_1) \bigcup L(G_2)$.

Обозначим дополнение языка $L(G)$ как $\overline{L(G)}$, где $\overline{L(G)} = V^* \setminus L(G)$. Из теории множеств известно, что пересечение множеств есть дополнение объединения дополнений этих множеств. В нашем случае языки, порождаемые КС грамматиками – множество строк, тогда пересечение языков есть дополнение объединения дополнений этих языков, то есть $L(G_1) \bigcap L(G_2) = \overline{\overline{L(G_1)} \bigcup \overline{L(G_2)}}$. Так как объединение языков разрешимо, то при разрешимости дополнения КС языка мы решаем проблему пустоты пересечения КС языков, которая не разрешима. Следовательно, проблема, указанная в пункте 3 теоремы 1.11 не разрешима. Что и требовалось доказать.

D. Определение тривиальности языка является неразрешимой проблемой так как универсальное множество V^* представимо в виде объединения КС языка и его дополнения $V^* = L(G) \bigcup \overline{L(G)}$. Если бы тривиальность КС языка бала бы разрешима, то разрешима была бы проблема дополнения КС языка. Но это не так, следовательно, проблема тривиальности КС языка не разрешима. Что и требовалось доказать.

E. Эквивалентность грамматик так же является неразрешимой проблемой так как в противном случае была бы разрешима проблема пустоты пересечения КС языков и проблема дополнения КС языков.

F. Доказывается аналогично пункту Е.

Окончание доказательства.

Вывод. 1. Нами найдено, что разрешимыми являются контекстные (неукорачивающие) языки. Следовательно, определение свойств нетривиального языка в общем случае является неразрешимой проблемой.

Пример: Принадлежность строки языку, порожденному грамматикой типа 0, является неразрешимой проблемой.

2. Для каждого типа грамматики существуют свои неразрешимые проблемы, в том числе и для регулярных грамматик.
3. Неразрешимость той или иной проблемы есть следствие придельной общности поставленной задачи, когда ставиться проблема нахождения свойств языка по заранее неизвестной формальной грамматике, его порождающей.
4. Для частных грамматик, то есть грамматик конкретного вида, неразрешимые в общем случае проблемы могут иметь решение. В таком случае решение задачи будет индивидуально для каждой грамматики.
5. Неразрешимость некоторой проблемы в свете вышесказанного означает, что эффективная процедура, будучи построена для каждого конкретного случая, становится неэффективной в общем случае, так как бесконечно сложна из-за счетности множества грамматик каждого типа.

ОСНОВЫ ОБЩЕЙ ТЕОРИИ АВТОМАТОВ

«Многие вещи нам не понятны не потому,
что наши понятия слабы, но потому,
что сие вещи не входят в круг наших понятий»
К. Прутков.

ТЕМА 0.7. ЭФФЕКТИВНАЯ ПРОЦЕДУРА И АЛГОРИТМ

При доказательстве неразрешимостей для формальных грамматик мы предполагали, что выполнимы некоторые действия (операции) над конечными последовательностями строк конечной длины. Если получалось доказать, что существует конечная последовательность действий над конечным числом объектов, приводящая к решению поставленной задачи, то мы говорили о том, что существует эффективная процедура.

Выполнимость элементарных операций или их последовательности не обсуждалось, а предполагалось, что они изначально выполнимы, что не очевидно.

1. Эффективная процедура становится определенной, объективной, если задается алгоритмическая модель, то есть нечто внешнее по отношению к субъективным представлениям каждого человека, которое позволяет однозначно трактовать как элементарные операции, так и их последовательности конечной длины.

ТИПЫ АЛГОРИТМИЧЕСКИХ МОДЕЛЕЙ.

1. Машина Тьюринга.
2. Детерминированные устройства (автоматы).

В рамках формальных систем используются формальные системы, являющиеся алгоритмическими моделями.

1. Формальные языки и грамматики.
2. Ассоциативное исчисление.

Определение. **АССОЦИАТИВНЫМ ИСЧИСЛЕНИЕМ** или **ПОЛУСИСТЕМОЙ ТУЭ** называется формальная система T, задаваемая парой объектов $T = \langle V, P \rangle$, где
$V = \{a_1, a_2, ..., a_m\}$- алфавит нетерминальных знаков,
$P = \{\alpha \leftrightarrow \beta \mid \alpha, \beta \in V^*\}$ - множество ассоциаций (двунаправленных продукций).

Пример. $\begin{matrix} abaa \leftrightarrow aaba \\ abaa \leftrightarrow baaa \end{matrix}$, где множество ассоциаций $P = \{ab \leftrightarrow ba\}$.

3. Цепи Маркова.

По своей сути цепи Маркова являются ассоциативным исчислением, у которого жестко задан порядок применения ассоциаций (двунаправленных продукций). Это позволяет говорить о завершении вывода преобразования строк, когда ни одна ассоциация не применима.

4. Рекурсивные функции.

Теория рекурсивных функций строится по аксиоматическому принципу. В качестве аксиомы принимаются функции, которые заведомо вычислимы. В качестве правил вывода

этой формальной системы применяется правило: рекурсии, композиции и итерации функций, которые утверждают, что функции, подставленные в схему композиции функций, приводят к вычислимой функции, если они вычислимы.

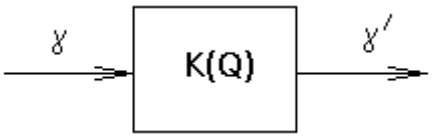
ПРЕДСТАВИМОСТЬ ФОРМАЛЬНЫХ СИСТЕМ АВТОМАТАМИ.

Пусть задана формальная грамматика $G = \langle V, W, I, P \rangle$ и некоторое устройство – автомат K, относительно которого известно, что оно может находиться в одном из конечного множества состояний $Q = \{q_0, q_1, ..., q_{n-1}\}$ в каждый момент времени, где выделяется два состояния:

q_0 - **конечное или заключительное состояние**,

q_1 - **начальное состояние.**

На вход автомата подается произвольная строка γ в алфавите V (с выхода снимается произвольная строка γ' в алфавите, совпадающем с алфавитом грамматики $G = \langle V, W, I, P \rangle$).



ение. K_P называется **АВТОМАТОМ АКЦЕПТОРОМ (РАСПОЗНАВАТЕЛЕМ)**, если произвольная строка языка $\gamma \in L(G)$, будучи поданная на вход автомата K переводит его из начального состояния в заключительное $K(q_1) \xrightarrow{\gamma} K(q_0)$ за конечное время и не переводит его в заключительное состояние, если строка γ не принадлежит языку $\gamma \notin L(G)$. Такой автомат будем называть **АВТОМАТОМ – РАСПОЗНАВАТЕЛЕМ.**

ение. $K_П$ называется **ПОРОЖДАЮЩИМ АВТОМАТОМ** для языка $L(G)$, если при изменении его состояния из состояния q_1 в состояние q_0 $K(q_1) \xrightarrow{\gamma} K(q_0)$ на выходе автомата K появляется строка языка γ', порожденная грамматикой $G = \langle V, W, I, P \rangle$.

ение. **АТОМ – ПРЕОБРАЗОВАТЕЛЕМ** K называется автомат, который для каждой распознанной строки γ языка $L(G_1)$ на выходе порождает некоторую строку γ', принадлежащую языку $L(G_2)$, где G_1, G_2 - некоторые формальные грамматики.

Автомат–преобразователь является общим случаем автомата, а автомат–акцептор и порождающий автомат могут быть рассмотрены как частные случаи или аспекты функционирования автомата общего вида.

ПРЕДВАРИТЕЛЬНАЯ КЛАССИФИКАЦИЯ АВТОМАТОВ.

Грамматика	Алгоритмическая модель (автомат)
произвольная грамматика).	ная машина Тьюринга.
контекстная грамматика).	ковый (магазинный) автомат.

контекстно-свободная грамматика).	ковый (магазинный) автомат.
регулярная грамматика).	й автомат.

Автомат представляет грамматику заданного типа если он способен распознать и породить строки, задаваемые этой грамматикой.

ТЕМА 0.8. МАШИНА ТЬЮРИНГА

Определение. **МАШИНОЙ ТЬЮРИНГА** T называется формальная система (устройство), задаваемое четверкой объектов $T = \langle V, Q, I, P \rangle$, где

$V = \{e, a_1, a_2, ..., a_m\}$- конечный алфавит ленты,

$Q = \{q_0, q_1, ..., q_{n-1}\}$ - конечное множество состояний устройства управления,

$I = q_1\alpha$ - начальная конфигурация машины Тьюринга, где $\alpha \in V^*$- строка в алфавите ленты.

P- программа машины Тьюринга. Множество P есть отображение множества декартового произведения множества состояний устройства управления на множество алфавита ленты в множество декартового произведения множеств состояния устройства управления, алфавита ленты и множества перемещения ленты D, то есть $Q \times V \to Q \times V \times D$, где $D = \langle L, E, R \rangle$.

$P = \{q_i a_j \to q_i' a_j' d_K \mid q_i q_i' \in Q; a_j a_j' \in V; d_K \in D\}$

ТЕОРЕМА 2.1. *О машинах Тьюринга и грамматиках.*
Машина Тьюринга представляет грамматику типа 0.

ДОКАЗАТЕЛЬСТВО.

A. Построим грамматику $G = \langle V, W, I, P \rangle$, которая производит те же действия, что и машина Тьюринга. Для этого отождествим терминальный алфавит грамматики и алфавит ленты машины Тьюринга, множество нетерминальных знаков грамматики и множество состояний устройства управления машины Тьюринга $W \equiv Q$. В качестве аксиомы грамматики I зададим состояние q_1' и добавим это состояние в множество Q. $W = Q \bigcup \{q_1'\}, I = q_1'$.

B. Множество продукций P грамматики G строим следующим образом:

1. Для всех команд машины Тьюринга вида $q_i a_j \to q_i' a_j' E$ в множество продукций P добавляем продукции вида $q_i a_j \to q_i' a_j'$.
2. Для всех команд машины Тьюринга вида $q_i a_j \to q_i' a_j' R$ в множество продукций P добавляем продукции вида $q_i a_j \to a_j' q_i'$.
3. Для всех команд машины Тьюринга вида $q_i a_j \to q_i' a_j' L$ в множество продукций P добавляем продукции вида $a_j'' q_i a_j \to a_j' a_j'' q_i'$, для всех $a_j'' \in V$.

C. В множество продукций P добавляем продукцию вида $q_i' \to q_i \alpha$, порождающая исходные данные α машины Тьюринга.

D. Построенная таким образом грамматика $G = \langle V, W, I, P \rangle$ порождает (распознает) те же строки, что и машина Тьюринга, при переходе из состояния q_1 с некоторым состоянием ленты α в состояние q_0.

Действительно, если применяется команда машины Тьюринга типа 1 , то может быть применена продукция грамматики G типа; если применяется команда типа 2, то может быть применена продукция типа 2; при применении команды типа 3 существует одна из m продукций, приводящая к тем же действиям, что и команда машины Тьюринга.

В Общем случае обратное не верно, но существует эквивалентное преобразование грамматики типа 0 в форму, представимую машиной Тьюринга.

Окончание доказательства.

Машина Тьюринга вычисляет функцию $Y = T(X)$, заданную на множестве строк $X, Y \in V^*$, где V^* - универсальное множество, образованное лентой машины Тьюринга, если:

1. Для каждой строки $\alpha \in X$ существует строка $\beta \in Y$ такая, что машина Тьюринга будучи запушена из начального состояния q_1 и строкой α на ленте, останавливается, когда на ленте записана строка β. $(q_1\alpha) \xrightarrow{T} (q_0\beta)$.
2. Для всех строк $\alpha \notin X$ машина Тьюринга не останавливается, то есть не переходит в состояние q_0.

Определение. **ЭКВИВАЛЕНТНЫМИ** такие машины Тьюринга, которые вычисляют одну и ту же функцию $Y = T(X)$.

ТЕЗИС ТЬЮРИНГА.

Всякий алгоритм может быть реализован машиной Тьюринга. То есть, если машины Тьюринга не реализует данный алгоритм, то этот алгоритм вовсе не алгоритм. И не существует алгоритмических моделей, не сводимых к машине Тьюринга.

Пример. Сложение двух двоичных чисел

$$\begin{array}{r} 1010 \\ +\ 0110 \\ \hline 1000 \end{array}$$

. Строим машину Тьюринга, которая сумму двоичных чисел преобразует в результат 1000. Если такая машина Тьюринга построена, то данная задача алгоритмизуема. В противном случае – нет.

Замечание. Доказать тезис Тьюринга нельзя, поскольку само понятие алгоритма является неточным. Тезис Тьюринга можно только опровергнуть примером.

Замечание. Тезис Тьюринга не теорема и не постулат математической теории, а утверждение, которое связывает математическую теорию с теми объектами, для описания которых она создана. То есть что такое алгоритм словами описать невозможно, но есть устройство, которое определяет алгоритм это или нет.

Замечание. По своему характеру тезис Тьюринга напоминает гипотезы физики об адекватности математических моделей физическим явлениям, которые они описывают.

Замечание. Подтверждением тезиса Тьюринга является:

1. Математическая и алгоритмическая практика.
2. Описание алгоритма в терминах другой алгоритмической модели, которая сводится к машине Тьюринга.

Замечание. Тезис Тьюринга позволяет заменить **неточные** утверждения о существовании эффективных процедур точными утверждениями о существовании машины Тьюринга и наоборот: утверждение о не существовании машины Тьюринга истолковать как утверждение о не существовании алгоритма вообще.

Любое доказательство является субъективным. Для объективности необходимо построит машину Тьюринга.

УНИВЕРСАЛЬНАЯ МАШИНА ТЬЮРИНГА

Систему команд машины Тьюринга можно интерпретировать как описание работы некоторого устройства или как программу. До сих пор мы воспринимали машину Тьюринга как программу, выступая в роли некоторого устройства, интерпретирующего эту программу. Все ли одинаково выполняют эту интерпретацию? Не значит ли это, что существует некоторый алгоритм (машина Тьюринга), который выполняет те же действия, что и произвольная машина Тьюринга?

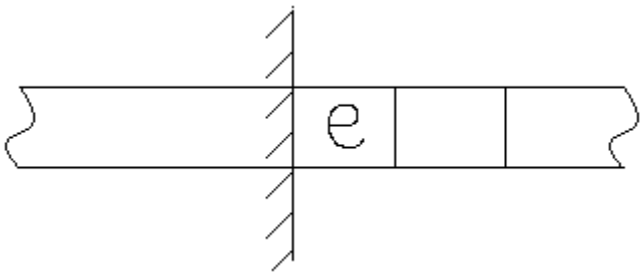
ТЕОРЕМА 2.2. *Об универсальной машине Тьюринга.*

Универсальная машина Тьюринга существует или любую вычислимую по Тьюрингу функцию можно реализовать на универсальной машине Тьюринга.

ДОКАЗАТЕЛЬСТВО.

А. ЛЕМА *О правой полуленте.*

Для произвольной машины Тьюринга существует эквивалентная ей машина Тьюринга, которая работает только на правой полуленте.



ДОКАЗАТЕЛЬСТВО.

Доказательство основано на таком преобразовании системы команд исходной машины Тьюринга при котором любая команда вида $q_i a_j \to q_i' a_j' L$ приводит к смещению всего того, что записано на правой полуленте в право на одну ячейку с занесением в образовавшуюся ячейку знака заполнителя ленты, если эта команда приводит к выходу устройства доступа за границу полуленты.

Существует машина Тьюринга, выполняющая это преобразование системы команд исходной системы команд машины Тьюринга.

Окончание доказательства.

В. Пусть $U = \langle V_U, Q_U, I_U . P_U \rangle$ есть универсальная машина Тьюринга, реализующая систему команд любой другой машины Тьюринга $T = \langle V_T, Q_T, I_T, P_T \rangle$. Мы должны не

ленте записать $I_U = q_{U_1} P_T \parallel I_T \xrightarrow{U} q_{U_0} P_T \parallel \beta$, где знак $\parallel$ показывает границу раздела полуленты, β - строка, записанная на ленте после остановки машины Тьюринга U. Если $I_T = q_{T_1}\alpha$ - начальная конфигурация машины Тьюринга T, то $T(\alpha) = \beta$ есть вычислимая по Тьюрингу функция, построенная для множества строк в алфавите ленты машины Тьюринга T.

C. В таком виде как в пункте В доказательства машина Тьюринга существовать не может. Необходимо выбрать алфавит V_U такой, что бы он содержал всевозможные знаки алфавитов любой наперед неизвестной машины Тьюринга. Так как для любого знака можно привести знак, отличный от него, и конечное множество других, то мощность алфавита V_U счетна, а следовательно универсальной машины Тьюринга не существует, так как она должна иметь конечный алфавит. Следовательно систему команд P_T машины Тьюринга T нельзя просто записать на ленту. Выход заключается в кодировании алфавитов машины Тьюринга T в алфавите универсальной машины Тьюринга $V_T, Q_T, D = \{L, E, R\}$.

D. Проведем кодирование алфавитов некоторой машины Тьюринга T в алфавите универсальной машины Тьюринга.

Пусть S есть код, задающий отображение знаков моделируемой машины Тьюринга в стоки алфавита V_U следующим образом:

$S(a_j) = a\emptyset^{m-1}1^j$, где $a_j \in V_T; m = |V_T|$,

$S(q_i) = q\emptyset^{m-1}1^i$, где $q_i \in Q_T; m_q = |Q_T|$,

$S(d_K) = d_K$, где $d_K \in D = \langle L, E, R \rangle$,

$S(\rightarrow) = \rightarrow$.

Это преобразование взаимно однозначное.

E. Уточним постановку задачи построения универсальной машины Тьюринга. Необходимо записать на ленте следующее $q_{U_1} S(P) \parallel S(\alpha) \xrightarrow{U} q_{U_0} S(P) \parallel S(\beta)$, где $T(\alpha) = \beta$. Алфавит универсальной машины Тьюринга состоит из следующих знаков $V_U = \langle 0, 1, a, q, L, E, R, \rightarrow, \| \rangle$, где система команд машины Тьюринга T предварительно преобразована для работы с правой полулентой.

F. При такой постановке задачи легко может быть построена универсальная машина Тьюринга при произвольных m и m_q, то есть построенная машина Тьюринга работает независимо от длины кодовых последовательностей m и m_q, так как поиск знака $q_i(a_j)$ на левой полуленте не зависит от числа знаков, которыми закодированы $q_i(a_j)$.

Окончание доказательства.

Замечание. Шенноном доказана теорема о том, что существует универсальная машина Тьюринга, имеющая два состояния $|Q_n| = 2$.

Замечание. Боброу и Минский показали, что не существует универсальной машины Тьюринга с двумя состояниями и двумя знаками алфавита V_U

ВЫЧИСЛИМОСТЬ И РАЗРЕШИМОСТЬ ПО ТЬЮРИНГУ

Стоит задача определения результативности алгоритма, которая является необходимым условием для реализации.

Определение. **РЕЗУЛЬТАТИВНОСТЬ** – есть получение результата за конечное время (конечное число шагов).

ПРОБЛЕМА ОСТАНОВКИ

Необходимо решить задачу: по любому алгоритму машины Тьюринга T и данным α (состояние ленты машины Тьюринга) определить приведут ли вычисления функции $T(\alpha)$ к некоторому результату β (машина Тьюринга остановится) или нет (машина Тьюринга не остановится).

Для решения поставленной задачи необходимо построить некоторый алгоритм (машину Тьюринга) T_0 такой, что T_0 в результате вычисления функции $T(\alpha)$ даст истину, если машина Тьюринга останавливается, или лож, если не останавливается.

$$T_0(T,\alpha)=\begin{cases} И, T(\alpha) - \textit{останавливается}, \\ Л, T(\alpha) - \textit{неостанавливается}. \end{cases}$$

То есть мы должны построить машину Тьюринга T_0 , которая для любой машины Тьюринга T и любых исходных данных α для этой машины Тьюринга определит останавливается ли $T(\alpha)$ или нет. Такая задача получила название **ПРОБЛЕМА ОСТАНОВКИ.**

ТЕОРЕМА 2.3. *О проблеме остановки.*

Не существует машины Тьюринга T_0, решающей проблему остановки для произвольной наперед неизвестной машины Тьюринга T.

ДОКАЗАТЕЛЬСТВО.

A. От противного. Предположим, что машина Тьюринга T_0 существует. Тогда необходимо записать на ленту $q_{01}P_T \parallel q_1\alpha \xrightarrow{T_0} q_{00}\{И, Л\}P_T \parallel q_1\alpha$, где

q_{01}, q_{00} - начальное и заключительное состояние машины Тьюринга T_0,

P_T - система команд исследуемой машины Тьюринга, закодированная как в теореме 2.2 об универсальной машине Тьюринга.

q_1 - начальное состояние машины Тьюринга.

α - входные данные для машины Тьюринга, которые так же закодированы.

B. Построим вспомогательную машину Тьюринга $T_0^/$ такую, что $T_0^/$ останавливается, если машина Тьюринга $T(\alpha)$ не останавливается и машина Тьюринга $T_0^/$ не останавливается, если машина Тьюринга $T(\alpha)$ останавливается.

Для построения машины Тьюринга $T_0^/$ используем систему команд машины Тьюринга T_0, добавив новое заключительное состояние и две команды $q_{00}Л \to q_{00}^/eR$ и $q_{00}И \to q_{00}^/ИЕ$.

Если существует машина Тьюринга T_0, то и существует машина Тьюринга $T_0^/$ так как машина Тьюринга $T_0^/$ получена из машины Тьюринга T_0 вполне конструктивным образом, то есть легко может быть построена машина Тьюринга, которое может сделать это преобразование.

C. Поставим задачу самоприменимости, то есть применим машину $T_0^/$ к самой себе. $T_0^/(T_0^/, T_0^/)$, где в качестве исходной строки α записана система команд машины Тьюринга $T_0^/$.

Машина Тьюринга $T_0^{/}$ останавливается в том случае, когда машина Тьюринга $T_0^{/}$ не останавливается наоборот.

D. В пункте С доказательства получено противоречие. В связи с тем, что машина Тьюринга $T_0^{/}$ получена из машины Тьюринга T_0 конструктивными средствами и при этом никак не связана с конкретным видом команд машины Тьюринга T_0, то можно сделать вывод о том, что никакая машина Тьюринга T_0 решающая проблему остановки невозможна.

Окончание доказательства.

Замечание. Алгоритмически неразрешимой проблемой является определение результативности алгоритмов.

Замечание. Теорема 2.3 утверждает лишь, что **отсутствует единый алгоритм** решающий проблему остановки. При этом вовсе не исключается возможность решения этой проблемы в частных случаях, когда задан конкретный алгоритм, но каждый раз решение этой задачи может быть выполнено различными средствами.

Замечание. Неразрешимость общей проблемы остановки вовсе не снимает необходимость доказывать сходимость разрабатываемых алгоритмов и показывает, что поиск таких доказательств нельзя полностью автоматизировать.

Замечание. Не существует общего алгоритма для отладки программ, который по тексту произвольной программы и данным для нее определил бы зациклится ли программа на этих данных или нет. Это не противоречит эмпирическому опыту, когда большинство программ в конце концов удается отладить. Но при этом необходимо использовать каждый раз различные средства, определяемые опытом и искусством разработчика программы.

ТЕОРЕМА 2.4. *Теорема Райса.*

Никакое нетривиальное свойство вычислимых функций не является алгоритмически разрешимым.

ДОКАЗАТЕЛЬСТВО.

A. Переформулируем условие теоремы.

Пусть C - некоторый класс вычислимых функций по Тьюрингу, нетривиалных в том смысле, что имеются функции, как принадлежащие классу C, так и функции не принадлежащие классу C.

Не существует алгоритма, который по описанию некой вычислимой функции $T(X)$ определял бы принадлежит ли эта функция некоторому классу C, или нет.

B. Предположим, что такой алгоритм T_C существует, тогда алгоритм T_C можно определить так:

$T_C(T)=\begin{cases} И, T \in C, \\ Л, T \notin C. \end{cases}$, где T - некоторая машина Тьюринга, вычисляющая функцию $T(X)$.

C. Построим вспомогательную машину $T^{/}$ для заданной машины T такую, что

$$T'(T,\alpha)=\begin{cases} f_C(\alpha)\in C, если T(\alpha) останавливается, \\ f_C(\alpha)\notin C, если T(\alpha) не\ останавливается. \end{cases}$$

$f_C(\alpha)$ нигде не определена.

Такую машину T' можно построить имея конкретную систему команд машины T.

D. Применим полученную машину T' в качестве исходных данных для машины T_C. Заметим, что T' заведомо принадлежит классу C так как вычисляет функцию $f_C(\alpha)$, то есть

$$T_C(T')=\begin{cases} И, если T(\alpha) остонавливается, \\ Л, если T(\alpha) не\ остонавливается. \end{cases}$$

E. В результате использования конструктивных средств при построении машины T' получили машину $T_C \equiv T_0$, решающую проблему остановки, что невозможно. Следовательно предположение о существовании машины T_C неверно.

Окончание доказательства.

Замечание. Из теоремы Райса следует, что по описанию вычислимой функции (алгоритма) нельзя узнать является ли эта функция постоянной, периодической, ограниченной, определима ли в конкретной точке ?

Замечание. Теорема Райса связывает такие понятия как алгоритм и вычислимая функция, где алгоритм не тождественен вычислимой функции, то есть по описанию алгоритма (где все понятно и ясно) ничего нельзя сказать о функции, которую он реализует.

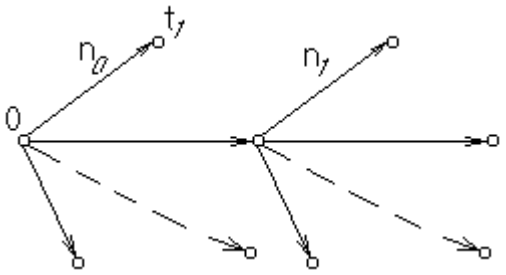
НЕДЕТЕРМИНИРОВАННАЯ МАШИНА ТЬЮРИНГА.

Определение. **ДАТЕРМЕНИРОВАННОЙ МАШИНОЙ ТЬЮРИНГА** называется такая машина Тьюринга, у которой отображение P множества декартового произведения множеств алфавита ленты и состояния устройства управления в множество декартового произведения множеств алфавита ленты, состояния устройства управления и множества перемещения ленты – функционально, то есть не существует двух одинаковых команд в множестве P с одинаковой левой частью.

Определение. **НЕДЕТЕРМЕНИРОВАННОЙ МАШИНОЙ ТЬЮРИНГА** называется такая машина Тьюринга в системе команд которой имеются команды с одинаковой левой частью.

ПРЕДСТАВЛЕНИЕ НЕДЕТЕРМЕНИРОВАННОЙ МАШИНЫ ТЬЮРИНГА В ВИДЕ ДЕТЕРМЕНИРОВАННОЙ МАШИНЫ ТЬЮРИНГА

Пусть имеется машина Тьюринга в момент времени t_0. Если существует возможность выбора одной из n_0 команд порождаем n_0 машин Тьюринга, которые будут функционировать с момента времени t_1. Если какая-то машина Тьюринга в момент времени t_1 имеет возможность использования n_1 команду с одинаковой левой частью, то размножаем эту машину Тьюринга в n_1 экземпляре и так далее. Очевидно, что число порожденных машин Тьюринга, которые приводят к остановкам машины Тьюринга, конечно и возможно бесконечно, если машина Тьюринга зацикливается.



ТЕМА 0.9. СЕТИ ПЕТРИ

Сеть Петри представляет собой модель дискретных систем.

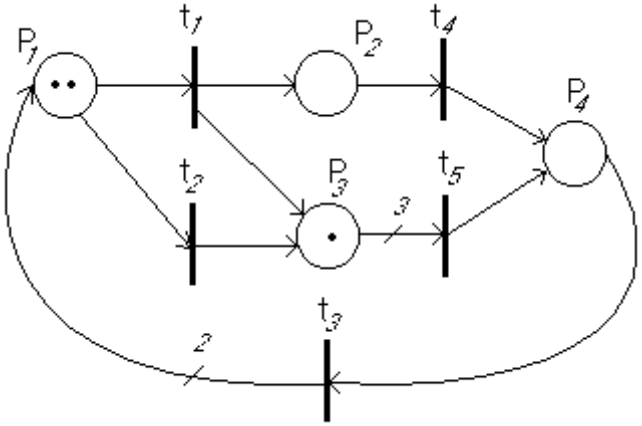
Дадим неформальное определение сети Петри.

Определение. **<u>СЕТЬ ПЕТРИ</u>** представляет собой двудольный ориентированный граф, имеющий вершины двух типов:

1. Вершина ◯ - позиция (вершина первого типа), определяет емкость ресурсов.
2. Вершина | - называется переходом. Соответствует событиям потребления или производства ресурсов заданных видов.

Количество ресурсов представляется метками – точками, располагаемыми в позициях ⊙, а дуги, соединяющие позиции и переходы показывают причинно - следственные связи.

Пример. Сеть Петри.



Вершины одного и того же типа не могут иметь непосредственные связи дугами.

ФУНКЦИОНИРОВАНИЕ СЕТИ ПЕТРИ.

1. **“Возбуждение переходов”**. Те переходы, которые имеют входящие дуги от позиций, содержащих метки, возбуждаются, то есть способны к функционированию (срабатыванию). В примере возбуждаются преходы t_1, t_2.
2. **“Срабатывание перехода”**. Только один из “возбужденных” переходов может сработать. Выбор срабатываемого перехода произвольный.
3. **“Извлечение меток”**. Из всех позиций, являющимимся входными для сработавшего перехода извлекается по одной метке.

4. **“Добавление меток”**. Во все позиции, являющиеся выходными для перехода добавляется по одной метке.
5. Процесс повторяется начиная с пункта 1. Если возбужденных переходов нет, то функционирование сети останавливается.

Замечание. Для моделирования ситуации, когда от одного перехода в одну и ту же позицию идет несколько дуг, введем кратность дуги (будем изображать одну дугу с кратностью равной натуральному числу N), тогда для возбуждения переходов необходимо наличие числа меток во входной позиции равной кратности дуги, и если кратность выходной дуги равна числу N, то при срабатывании перехода в позицию добавляется N меток.

Дадим формальное определение сети Петри.

Определение. **СЕТЬЮ ПЕТРИ** называется формальная система, заданная четверкой объектов $S = \langle P, T, E, M \rangle$, где

$P = \{P_1, ..., P_n\}$- конечное множество позиций,

$T = \{t_1, ..., t_m\}$- конечное множество переходов, $P \bigcap T = \emptyset$,

E- отношение инцидентности, заданное на алфавите позиций и переходов. $E : R \times T \to N \times N$ - есть отображение декартового произведения множества позиций на множество переходов в декартовое произведение двух множеств натуральных чисел. E задает кратность дуг, ведущих от позиций к переходам и обратно.

$M = \{\mu_1, ..., \mu_n\}$-начальная маркировка (начальные ресурсы сети Петри). Определяет число меток в каждой позиции.

Пример. Для представления сети Петри (рассмотренной в примере (см выше)) зададим ее формальное описание:

$S = \langle P, T, E, M \rangle$,

$P = \{P_1, P_2, P_3, P_4\}$,

$T = \{t_1, t_2, t_3, t_4, t_5\}$,

Отношение инцидентности E зададим следующим образом:

1. Зададим как для машины Тьюринга $P_i t_j \to \pi_{ij} \tau_{ij}$, где

   π_{ij} - кратность дуги, ведущей из позиции P_i к переходу t_j. $P_i \xrightarrow{\pi} t_j$.

   τ_{ij} - кратность дуги, ведущей от перехода t_j к позиции P_i. $t_j \xrightarrow{\pi} P_i$.

   $P_1 t_1 \to {}^{1}\!/_{0}, P_3 t_1 \to {}^{0}\!/_{1}$,

   $P_1 t_2 \to {}^{1}\!/_{0}, P_3 t_2 \to {}^{0}\!/_{1}$,

   $P_1 t_3 \to {}^{0}\!/_{2}, P_3 t_5 \to {}^{3}\!/_{0}$,

   $P_2 t_1 \to {}^{0}\!/_{2}, P_4 t_3 \to {}^{1}\!/_{0}$,

   $P_2 t_4 \to {}^{1}\!/_{0}, P_4 t_4 \to {}^{0}\!/_{1}$,

   $P_4 t_5 \to {}^{0}\!/_{1}$.

2. Зададим табличным способом.

$t \setminus P$	P_1	P_2	P_3	P_4
t_1	${}^{1}\!/_{0}$	${}^{0}\!/_{2}$	${}^{0}\!/_{1}$	-
t_2	${}^{1}\!/_{0}$	-	${}^{0}\!/_{1}$	-

t_3	$0/2$	-	-	$1/0$
t_4	-	$1/0$	-	$0/1$
t_5	-	-	$3/0$	$0/1$

Начальная маркировка $M=(2,0,1,0)$.

Число сетей Петри счетно.

ФОРМАЛИЗАЦИЯ ФУНКЦИОНИРОВАНИЯ СЕТЕЙ ПЕТРИ

Формализация функционирования сетей Петри заключается в последовательной смене ее маркировок.

Пусть задана сеть Петри $S=\langle P,T,E,M\rangle$, где $|P|=n$, $|T|=m$, с начальной маркировкой $M=\{\mu_1,...,\mu_n\}$, где n - число позиций, тогда функционирование сети Петри представляется как последовательность смены маркировок $M^{(0)}\to M^{(1)}\to ...\to M^{(l)}\to ....$

Сеть Петри останавливается, если эта последовательность конечна и не останавливается в противном случае.

Определение. Маркировка M называется **ТУПИКОВОЙ**, если является последней маркировкой в конечной последовательности срабатывания.

Функционирование сети Петри опишем рекурсивными уравнениями.

$M^{(K+1)}$ есть некая функция δ от маркировки, зависящей от маркировки на предыдущем шаге функционирования и $T^{(K)}$ - возбужденные переходы на к-ом шаге, то есть

$$\begin{cases} M^{(K+1)}=\delta\left(M^{(K)},T^{(K)}\right), \delta-\textit{функция смены маркировок},\\ T^{(K)}=\omega\left(M^{(K)}\right), \omega-\textit{функция возбуждения преходов}. \end{cases}$$

Функция δ может быть случайной, так как должна выбрать из всех возбужденных переходов только один.

Функция ω детерминированная, то есть четко определенная.

Рассмотрим первое уравнение и два вектора:

$\pi_i=(\pi_{i1},\pi_{i2},...,\pi_{in})$ - вектор возбуждения переходов,

$\tau_i=(\tau_{i1},\tau_{i2},...\tau_{in})$- вектор срабатывания переходов.

$$\pi_{ij}(\tau_{ij})=\begin{cases} 0, \textit{если от } P_i \textit{ нет дуги в } \tau_j \left(\textit{от } \tau_j \textit{ в } P_i\right),\\ n-\textit{кратность дуги ведущей от } P_i \textit{ в } \tau_j \left(\textit{от } \tau_j \textit{ в } P_i\right). \end{cases}$$

Заметим, что в таблице инцидентности находятся числа π_{ij}/τ_{ij}.

Дадим содержательную интерпретацию векторов π,τ.

Вектор π_i определяет на сколько уменьшится маркировка сети Петри при срабатывании i-ого перехода.

Вектор τ_i определяет насколько изменится маркировка сети Петри при срабатывании i - ого перехода.

Откуда находим

$M^{(K+1)}=M^{(K)}+\tau_i-\pi_i$, где π_i - числитель i - ой строки, τ_i - знаменатель i - ой строки.

$M^{(K+1)}=M^{(K)}+\Phi_i$, где Φ_i - вектор разностей $\tau_i-\pi_i$.

Рассмотрим уравнение $M^{(K+1)} = M^{(K)} + \Phi_i$. Оно задает вектор возбуждаемых переходов на k - ом шаге функционирования сети Петри

$T^{(K)} = \left(\mathrm{E}_1^{(K)}, \mathrm{E}_2^{(K)}, ..., \mathrm{E}_m^{(K)},\right)$, где

$$\mathrm{E}_i^{(K)} = \begin{cases} 0, \textit{если } t_i \textit{ переход не возбужден}, \\ 1, \textit{если } t_i \textit{ переход возбужден}. \end{cases}$$

Определение. Для определения функции возбуждения переходов необходимо ввести отношения сравнения векторов – целых чисел длинной n. Если заданы две маркировки $M^{(u)}$ и $M^{(v)}$, то будем говорить, что маркировка $M^{(u)} \geq M^{(v)}$, если для всех $M_i^{(u)} \geq M_i^{(v)}, (i = 1,...,n)$ (производится покомпонентное сравнение векторов $M^{(u)}$ и $M^{(v)}$).

Дадим содержательную интерпретацию отношения $\geq$. Если покомпонентно вектор $M^{(u)}$ больше вектора $M^{(v)}$, то выполняется соотношение $\geq$.

Пример. $(0,1,2) \geq (0,1,1)$

Аналогично выводятся соотношения $<, \leq, =, \neq, >$.

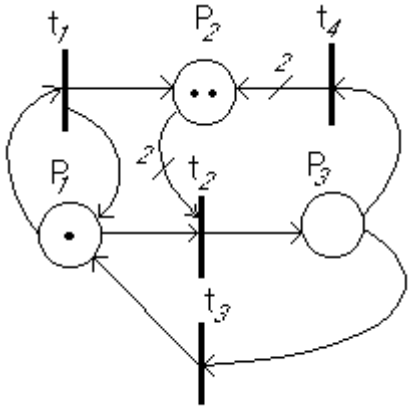
Условием возбуждения переходов является

$\mathrm{E}_i^{(K)} = \left(\mu_i^{(K)} \geq \pi_i\right), i = 1,...,n$

$T^{(K)} = \left(\mu_i^{(K)} \geq \pi_1, \mu_2^{(K)} \geq \pi_2, ..., \mu_n^{(K)} \geq \pi_n\right)$, то есть число меток в каждой позиции должно быть больше или равно кратности дуг ведущих из данной позиции в переход t_i. Для вычисления вектора $T^{(K)}$ необходимо сравнить покомпонентно текущую маркировку с числителями строк таблицы инцидентности.

Таким образом осталась не определена функция смены маркировок δ, которая случайно или по какому-то закону выбирает из возбужденных переходов срабатываемый.

Пример. Пусть задана сеть Петри двудольным графом и таблицей инцидентности.



$T \setminus P$	P_1	P_2	P_3
t_1	$1/1$	$0/1$	-
t_2	$1/0$	$2/0$	$0/1$
t_3	$0/1$	-	$1/0$
t_4	-	$0/2$	$1/0$

Вычислим вектора Φ_i.

$\Phi_1 = (0,1,0),$

$\Phi_2 = (-1,-2,1),$

$\Phi_3 = (1,0,-1),$

$\Phi_4 = (0,2,-1).$

Вычислим вектор возбуждения переходов T. Рассмотрим начальную маркировку $M^{(0)} = (1,2,0)$. Проверяем условие возбуждения переходов. Так как условие возбуждения переходов выполняется, то вектор возбуждения переходов будет $T^{(0)} = (1,2,0,0)$. Предположим сработал второй переход, тогда получим вектор маркировок

$M^{(1)} = (1,2,0) + (-1,-2,1) = (0,0,1)$, тогда

$T^{(1)} = (0,0,1,1)$ и так далее.

Определение. Маркировка $M^{(l)}$ **ДОСТИЖИМА** из маркировки $M^{(K)}$ если существует последовательность маркировок сети Петри, оканчивающаяся маркировкой $M^{(l)}$, где

$M^{(K)} \xrightarrow{t^{(K)}} M^{(K+1)} \xrightarrow{t^{(K+1)}} \ldots \xrightarrow{t^{(l-1)}} M^{(l)}$

$M^{(K)} \xrightarrow{Q} M^{(l)}$, где $Q = t^{(K)} t^{(K+1)} \ldots t^{(l-1)}$ - строка в алфавите переходов $Q \in T^*$. То есть при функционировании сети Петри порождается строка в алфавите переходов.

Определение. **МНОЖЕСТВО ДОСТИЖИМЫХ МАРКИРОВОК** R для сети Петри S из маркировки $M^{(K)}$ есть всевозможные маркировки $M^{(l)}$ такие, что из маркировки $M^{(K)}$ достижима маркировка $M^{(l)}$ при конечном числе срабатываний переходов.

$R(S, M^{(K)}) = \{M^{(l)} \mid M^{(K)} \xrightarrow{Q} M^{l}; Q \in T^*\}$

Если маркировка $M^{(K)} = M$ - начальной маркировке сети Петри, то множество достижимых маркировок $R(S, M) = R(S)$ называется **МНОЖЕСТВОМ ДОСТИЖИМЫХ МАРКИРОВОК СЕТИ ПЕТРИ**.

Определение. **СВОБОДНЫЙ ЯЗЫК СЕТИ ПЕТРИ** $L(S)$ есть множество конечных последовательностей срабатывания сети Петри S

$L(S) = \{Q \in T^* \mid M \xrightarrow{Q} M^{(l)} \subset R(S)\}$, то есть таких, что из начальной маркировки M получается достижимая маркировка $M^{(l)}$ сети Петри S.

СВОБОДНЫЙ ЯЗАК СЕТИ ПЕТРИ $L(S)$ - это множество строк в алфавите T ведущих от начальной маркировки сети к **каждой** допустимой (достижимой) маркировке $M^{(l)}$ сети Петри S.

Множество свободных языков, порождаемых всевозможными сетями Петри, образуют класс свободных языков L_s.

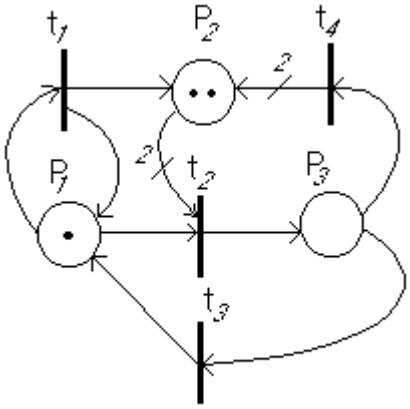
Определение. **ПОМЕЧЕННАЯ СЕТЬ ПЕТРИ** есть совокупность трех объектов $\widehat{S} = (S, A, \lambda)$, где

$S = \{P, T, E, M\}$ - сеть Петри,

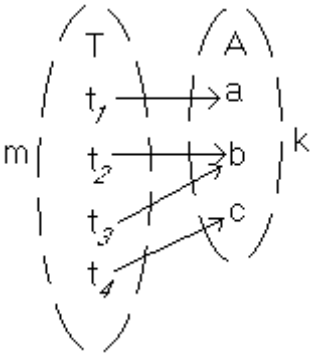
$A = \{a_1, a_2, \ldots, a_K\}$ - помечающий алфавит,

$\lambda : T \to A$ - помечающая функция (отображение), ставящее в соответствие каждому переходу знак алфавита A.

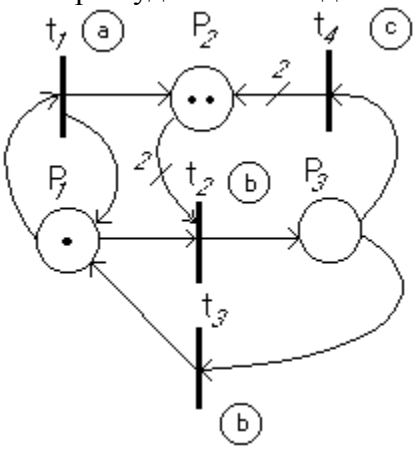
Пример. Пометим сеть Петри следующего вида



Зададим алфавит $A=\{a,b,c\}$. Помечающую функцию λ представим в следующем виде



Тогда помеченная сеть Петри будет иметь вид:



Определение. **ПРЕФИКСНЫЙ ЯЗЫК ПОМЕЧЕННОЙ СЕТИ ПЕТРИ** $\widehat{S}$ Есть всевозможные строки в алфавите A такие, что

$L(\widehat{S})=\{\lambda(Q)\,|\,Q\in L(S)\}$

Это есть множество значащих строк, принадлежащих языку $L(S)$ в помечающем алфавите A, соответствующих одной из возможных последовательностей срабатывания сети Петри S, задаваемая помечающей функцией λ.

Определение. **ТЕРМИНАЛЬНЫЙ ЯЗЫК ПОМЕЧЕННОЙ СЕТИ ПЕТРИ** $\widehat{S}$.Удобно рассматривать не свободный язык сети Петри $L(\widehat{S})$, а его подмножество терминальных сток, ведущих от начальной маркировки $M^{(0)}$ к некоторой

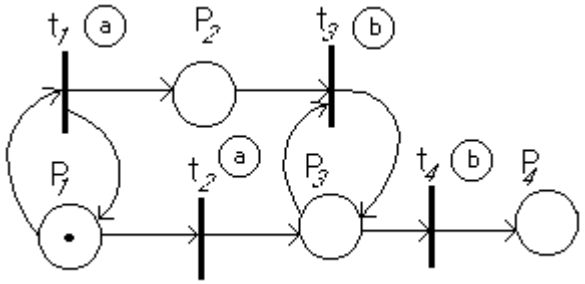
фиксированной маркировки $M^{(f)}$.

$M^{(0)} \xrightarrow{\bar{S}} M^{(f)}$.

То есть терминальный язык $L(\bar{S})$ есть всевозможные строки в алфавите A такие, что помечающая последовательность Q переводит начальную маркировку сети Петри $M^{(0)}$ в заданную маркировку $M^{(f)}$.

$L(\bar{S}, M^{(f)}) = \{\lambda(Q) \in A^* \mid M^{(0)} \xrightarrow{Q} M^{(f)}\}$.

Пример. Пусть задана сеть Петри $\bar{S} = (S, A, \lambda)$ двудольным графом



Введем помечающий алфавит $A = \{a, b\}$ и пометим данную сеть Петри. Эта сеть Петри порождает свободный язык сети Петри $L(S)$, строками которого являются следующие последовательности срабатывания переходов

$t_1 t_2 t_3 t_4$,

$t_1 t_1 t_2 t_3 t_3 t_4$,

$t_1 t_1 t_2 t_3 t_4$,...

Префиксный язык сети Петри $L(\bar{S}) = \{a^2 b^2, a^3 b^3, a^3 b^2, ...\}$.

Терминальный язык сети Петри $L(\bar{S}, (1,0,0,0)) = \{a^2 b^2, a^3 b^3, a^3 b^2, ...\}$.

В общем случае эта сеть Петри порождает следующий терминальный язык $L(\bar{S}, (1,0,0,0)) = \{a^n b^n \mid n \in N \setminus \{0\}\}$.

ТЕОРЕМА 2.5. *О мощности классов языков сети Петри.*

Введем следующие обозначения.

Пусть L_S - языки, которые могут быть порождены всевозможными сетями Петри.

Пусть $L_П$ - всевозможные префиксные языки, порождаемые всевозможными сетями Петри.

Пусть L_T - терминальные языки, соответствующие всевозможным маркировкам всевозможных сетей Петри.

$L_S = \{\{t_1, t_2, ...\}, \{...\}, ..., \{...\}, ...\}$ - счетно. Тогда мощность класса терминальных языков сетей Петри превосходит мощность классов префиксных языков, причем $L_П \subset L_T$.

ДОКАЗАТЕЛЬСТВО.

Класс префиксных языков $L_П$ порождается всевозможными последова-тельностями срабатывания всевозможных сетей Петри, причем только тех последовательностей, которые заканчиваются тупиковой маркировкой. Понятно, что класс терминальных языков более мощен, чем класс префиксных языков и включает часть его, так как помимо последовательностей срабатывания, приводящих к тупиковым маркировкам, содержит и последовательности срабатываний, приводящих к всевозможным заключительным маркировкам $M^{(f)}$, в том числе и к тупиковым.

Окончание доказательства.

ТЕОРЕМА 2.6. *О мощности классов языков сетей Петри, по отношению к классам языков, порождаемых формальными грамматиками.*

Класс терминальных языков сетей Петри:

1. Строго мощнее, чем класс языков, порождаемых регулярными грамматиками $L_3 \subset L_T$.
2. Не сравним с классом языков, порождаемых КС грамматиками $L_2 \neq L_T$.
3. Менее мощен, чем класс языков, порождаемых произвольными грамматиками $L_T \subset L_0$.

Таким образом $L_3 \subset L_T \neq L_2 \subset L_0$.

ДОКАЗАТЕЛЬСТВО.

А. Докажем, что $L_3 \subset L_T$.

Произвольный регулярный язык L_3 может быть представлен помеченной сетью Петри $\widehat{S}$, порождающей тот же язык в качестве терминального.

Если задана помеченная сеть Петри $\widehat{S} = \langle\langle P,T,E,M\rangle, A, \lambda\rangle$ и регулярная грамматика $G_3 = \langle V,W,I,P\rangle$, то каждому нетерминальному знаку $A \in W$ поставим в соответствие позиции $P_A \in P$.

$\forall A \in W \leftrightarrow P_A \in P$.

Каждой продукции вида $A \to aB; AB \in W, a \in V$ поставим в соответствие дугу с переходом, помеченным знаком a, идущую от позиции P_A к позиции P_B.

В качестве заключительной маркировки $M^{(f)}$ выберем маркировку, где первая позиция соответствует заключительной сентенциальной форме, когда удается единственный нетерминальный знак продукций вида $A \to a$.

$$M^{(f)} = \underbrace{(1,0,0,0,...,0)}_{|W|+1}.$$

Начальная маркировка M такова, что число меток в позиции P_I равно 1.

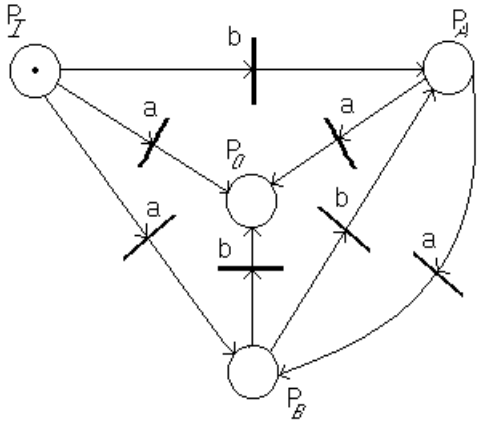
$M = (0,..,1,...)$, $M_I = 1$ в позиции P_I.

Пример. Пусть задана регулярная грамматика $G = \langle V,W,P,I\rangle$

$V = \{a,b\}$, $W = \{I, A, B\}$,

$$P = \begin{cases} I \to a \mid aB \mid bA, \\ A \to a \mid aB, \\ B \to b \mid bA. \end{cases}$$

Построим сеть Петри, порождающую тот же язык, что и регулярная грамматика G $L(\widehat{S}) = L(G)$.

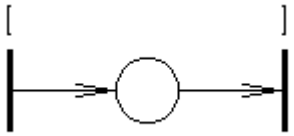


Где P_0 - заключительная или нулевая позиция.

Заключительной маркировкой будет маркировка $M^{(f)} = (1,0,0,0)$. Очевидно, что сеть Петри порождает то же язык, что и регулярная грамматика G.

Однако существуют сети Петри, которые порождают языки, не являющийся регулярными.

Пример. Пусть есть следующая сеть Петри



Данная сеть Петри порождает строки следующего вида [],[[]],[][] - правильно расставленные скобки. Тогда грамматика порождающая этот язык будет

$G = \langle V, W, I, P \rangle$

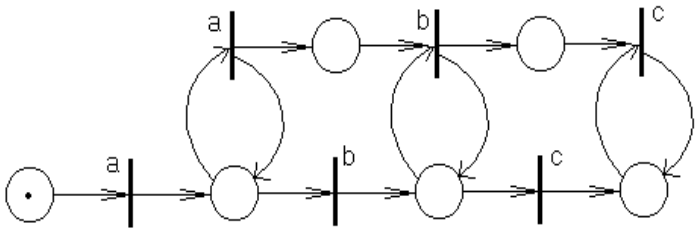
$V = \{[,]\},\ W = \{I\},\ P = \{I \to [I] \mid II \mid []\}.$

Очевидно, что грамматика G - КС грамматика. Язык $L(G)$ невозможно породить регулярной грамматикой.

B. Докажем, что L_T не сравним с L_2, то есть $L_T \neq L_2$. То есть не все КС языки могут быть представлены сетями Петри, а так же есть языки, представимые сетями Петри, но которые не могут быть порождены КС грамматиками.

Пример. Приведем пример не КС языка, порождаемого сетью Петри.

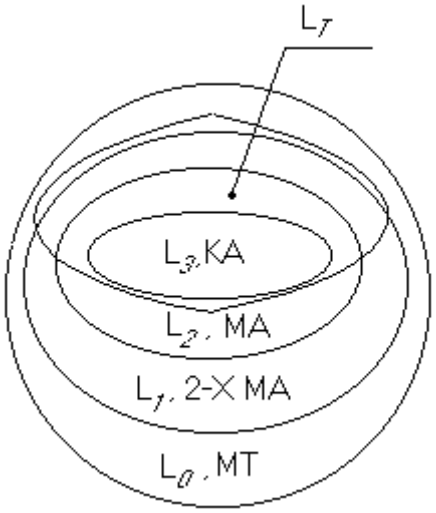
Рассмотрим язык $L(G) = (a^n b^n c^n \mid n \in N)$. Ранее доказывалось, что этот язык порождается контекстной грамматикой (см глава 1).



Однако существует КС язык, на порождаемый сетью Петри.

Пример. Рассмотрим следующую грамматику $G = \langle V, W, I, P \rangle$ с продукциями вида $P = \{I \to Ac, A \to AA \mid aAb \mid e\}$. Очевидно, что данная грамматика порождает строки вида $a^n b^n c$.

C. Таким образом получаем:



, где КА – конечный автомат,
МА – магазинный автомат,
2-Х МА – двухстековый магазинный автомат,
МТ – машина Тьюринга.
Окончание доказательства.

Вывод. Какова – бы ни была сеть Петри машина Тьюринга лучше.

ТЕМА 0.10. МАГАЗИННЫЙ АВТОМАТ

Определение.

МАГАЗИННЫМ АВТОМАТОМ называется формальная система состоящая из 5 (6) объектов $M = \langle A, Q, S, P, I[, B] \rangle$, где

$A = \{a_1, a_2, ..., a_m\}$- конечный алфавит входных знаков;

$Q = \{q_0, q_1, ..., q_{n-1}\}$ - конечное множество состояний магазинного автомата, где q_0 - заключительное состояние, q_1 - начальное состояние, если не оговорено особо;

$S = \{s_1, s_2, ..., s_l\}$ - стековый алфавит.

P - магазинное отображение множества декартового произведения множеств состояний магазинного автомата, стекового алфавита и входного алфавита в множество декартового произведения множеств состояний магазинного автомата, стекового алфавита [и выходного алфавита.]

$P : Q \times S \times A \to Q \times S^*[\times B]$.

P - есть множество команд следующего вида

$P = \{q_i s_j a_k \to q_i' \sigma[b_t] \mid q_i, q_i' \in Q; s_j \in S, a_k \in A, \sigma \in S^*[; b_t \in B]\}$, где σ - строка в стековом алфавите S^*.

$I = q_1 \sigma$, где $q_1 \in Q, \sigma \in S^*$ - начальная конфигурация магазинного автомата.

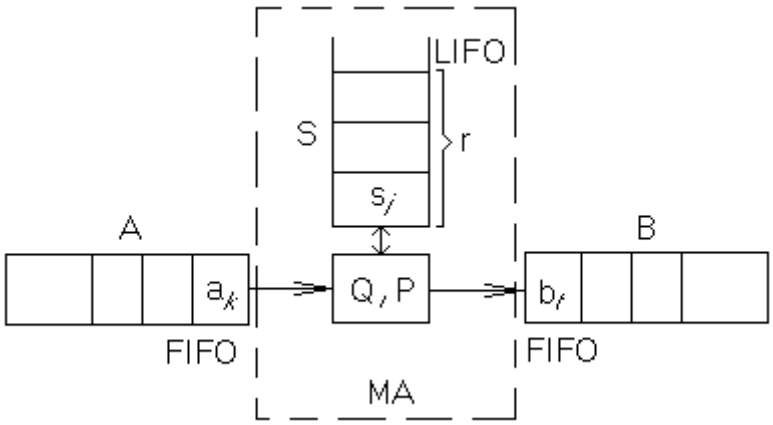
$B = \{b_1, b_2, ..., b_r\}$- выходной алфавит магазинного автомата.

Не трудно нарисовать внешний вид (структуру) магазинного автомата.

Магазинный автомат состоит из:

1. **Устройства управления, реализующего систему команд P и имеющего состояние Q.**
2. Входной очереди A.
3. Стека S.
4. Выходной очереди B.

Тогда структура магазинного автомата будут выглядеть следующим образом



Где из входной очереди A в устройство управления могут поступать по одному знаки, то есть обозревается только конец очереди. В выходную очередь B может записываться только один знак. Стек S обозревается на глубину одного знака, но запись может производиться на произвольной глубине стека.

Дисциплина доступа к очереди FIFO, к стеку LIFO.

Определение. **КОНФИГУРАЦИЕЙ МАГАЗИННОГО АВТОМАТА** K называется выражение $K = q_i\sigma$, где

q_i - состояние устройства управления,

σ - состояние стека.

Конфигурация магазинного автомата однозначно определяет его сосстояние.

Отличие магазинного автомата от машины Тьюринга в том, что вместо ленты, доступной в каждой ячейки, используется стек, доступный только на вершине. Таким образом изменено устройство чтения и записи знаков.

Процесс занесения на ленту машины Тьюринга исходных данных и чтение результата моделируется в магазинном автомате входной и выходной очередью, и в процессе функционирования, магазинный автомат последовательно извлекает знаки из входной очереди и записывает знаки в выходную очередь, возможно не вкаждый такт работы.

e-ПЕРЕМЕЩЕНИЕ МАГАЗИННОГО АВТОМАТА

e-перемещение магазинного автомата существует четырех видов:

1 рода . Осуществляется командами вида $q_i s_j e \to q_i^{/}\sigma[b_t]$.Не извлекается знак в 1 такт работы магазинного автомата.

2 рода. Осуществляется командами вида $q_i e a_k \to q_i^{/}\sigma[b_t]$. Извлекается знак из входной очереди, но не извлекается знак из стека.

3 рода. Осуществляется командами вида $q_i s_j a_k \to a_j^{/} s_j[b_t]$. Состояние стека не изменяется, то есть извлеченный знак s_j из стека тут же заносится обратно в стек.

4 рода. Осуществляется командами вида $q_i e e \to q_i^{/}\sigma[b_t]$. В стек заносится строка σ

Определение. **МНОЖЕСТВО ПРИНИМАЕМЫХ МАГАЗИННЫМ АВТОМАТОМ СТРОК** $L(M)$ - есть такое множество строк в алфавите A, при подаче которых на вход магазинного автомата, магазинный автомат перейдет из начальной конфигурации в заключительную, года состояние устройства управления q_0.

$q_1\sigma \xrightarrow{\alpha} q_0\sigma^{/}$, где $\alpha \in A^*; \sigma, \sigma^{/} \in S^*$.

В этом случае строка α называется **СТРОКОЙ, ПРИНИМАЕМОЙ (РАСПОЗНАННОЙ) МАГАЗИННЫМ АВТОМАТОМ.**

Магазинный автомат распознает (принимает) язык $L(M)$ если принимает все его строки и не принимает (не распознает) строки не принадлежащие языку $L(M)$.

Пример. Рассмотрим магазинный автомат, распознающий строки, одинаково читаемые как слева на право, так и справа налево.

$M = \langle A.Q.S.P.I \rangle$

$A = \{a, b\}$- входной алфавит,

$Q = \{q_0, q_1, q_2\}$- множество состояний,

$S = \{*, A, B\}$- алфавит стека, где * - ограничитель стека,

система команд

$$P = \begin{cases} q_1 * a \to q_1 A*, \\ q_1 * b \to q_1 B*, \\ q_1 Aa \to q_1 AA \mid q_2 e, \\ q_1 Ba \to q_1 AB \mid q_2 e, \\ q_1 Ab \to q_1 BA \mid q_2 e, \\ q_1 Bb \to q_1 BB \mid q_2 e, \\ q_2 Aa \to q_2 e, \\ q_2 Bb \to q_2 e, \\ q_2 * e \to q_0 e. \end{cases}$$

$I = q_1 *$ - начальная конфигурация (* - есть аксиома языка).

Рассмотрим строку $abba$. Подадим ее на вход магазинного автомата $M = \langle A.Q.S.P.I \rangle$, тогда результат обработки этой строки представим в виде таблицы.

A	конфигурация	
	Q	S
$abba$	q_1	*
abb	q_1	$A*$
ab	q_1	$BA*$
a	q_2	$A*$
e	q_2	*
e	q_0	

Таким образом видно , что данный автомат принял заданную строку.

Очевидно, что рассмотренный автомат принимает все “зеркальные строки” и является недетерминированным так как P не является функцией и переход из состояния q_1 в состояние q_2 должен произойти на середине строки.

Очевидно, что какой - нибудь из автоматов, полученный в результате размножения при альтернативном выборе команд остановится, если строка “ зеркальна ” и ни один из автоматов не остановится в противном случае.

ТЕОРЕМА 2.7. *О КС грамматике и магазинном автомате.*

Для любого языка, порождаемого КС грамматикой существует магазинный автомат, возможно недетерминированный, распознающий (принимающий) строки этого языка.

То есть для любого $G_2 = \langle V, W, I, P \rangle$ существует такой $M = \langle A, Q, S, P, K \rangle$, что $L(G_2) = L(M)$.

ДОКАЗАТЕЛЬСТВО.

A. Положим, что входной алфавит магазинного автомата совпадает с алфавитом терминальных знаков грамматики
$A = V, L(G_2) \subset A^*$.

B. Определим магазинный автомат $M = \langle A, Q, S, P, K \rangle$ следующим образом

$A = V$ - входной алфавит;

$Q = \{q_0, q_1\}$ - множество состояний магазинного автомата;

$S = V \bigcup W \bigcup \{*\}$ - алфавит стека – объединение терминального, нетерминального алфавитов и знака – ограничителя стека;

Множество команд

$$P = \begin{cases} q_1 Be \to q_1 \alpha, для\ любых\ продукций\ вида\ B \to \alpha \in P_2, \\ q_1 aa \to q_1 e, для\ любых\ знаков\ \alpha \in V = A, \\ q_1 * e \to q_0 e. \end{cases}$$

Конфигурация $K = q_1 I$, где I - аксиома грамматики.

C. Определение автомата $M = \langle A, Q, S, P, K \rangle$ гарантирует, что для любого вывода в грамматике $G_2 = \langle V, W, I, P \rangle$

$I \xrightarrow{G_2} \alpha\beta; \alpha \in V^*; \beta \in (V \bigcup W)^*$

Из входной очереди магазинного автомата удаляется строка α, то есть автомат реализует левосторонний вывод.

D. **ЛЕММА** *О левостороннем выводе в КС грамматике*

Любая строка языка , порожденная КС грамматикой может быть получена в результате левостороннего вывода, когда в сентенциальной форме грамматики заменяется (раскрывается) самый левый нетерминальный знак.

ДОКАЗАТЕЛЬСТВО.

Достаточно показать что в грамматике G_2 из сентенциальной формы $\gamma_1 b \gamma_2 c \gamma_3$, где $\gamma_1, \gamma_2, \gamma_3 \in (V \bigcup W)^*$ результат вывода не зависит от последовательности применения продукций $B \to \beta, c \to \delta$.

$\gamma_1 b \gamma_2 c \gamma_3 \to \gamma_1 b \beta_2 \delta \gamma_3$.

Учитывая, что $\gamma_1, \gamma_2, \gamma_3$ - произвольные подстроки сентенциальной формы, то тем самым мы показали, что результат вывода в грамматике не зависит от последовательности раскрытия нетерминальных знаков.

E. Магазинный автомат $M = \langle A, Q, S, P, K \rangle$

1. Удаляет из входной очереди A совпадающие с вершиной стека S нетерминальные знаки (видно из системы команд).
2. Если на вершине стека S находится нетерминальный знак, то он заменяется на строку, ему соответствующую, произвольным недетерминированным образом. $q_1 eA \to q_1 \alpha$, $A \to \alpha$.
3. Когда стек S исчерпан, то есть найден знак *, магазинный автомат переходит в заключительное состояние, выбрав предварительно из входной очереди A последовательность терминальных знаков, которая может быть порождена грамматикой $q_1 e* \to q_0 e$.

4. Если выбрана продукция в пункте 2, которая не использовалась для порождения строки во входной очереди, то автомат не остановится, то есть этот экземпляр автомата строку не распознает, но будет существовать экземпляр, который распознает эту строку.

Окончание доказательства.

Вывод. Построенный автомат $M = \langle A, Q, S, P, K \rangle$ распознает все строки языка $L(G_2)$, то есть $L(M) = L(G_2)$.

Замечание. Нетрудно показать, что существует магазинный автомат $M = \langle \{e\}, Q, S, P, I, B \rangle$, который аналогичным образом может породить в выходном алфавите B любую строку языка $L(G_2)$.

ТЕОРЕМА 2.8. *О магазинном автомате и КС языке.*

Если $L(M)$ - язык, принимаемый некоторым, возможно недетерминированным магазинным автоматом $M = \langle A, Q, S, P, I \rangle$, то этот язык может быть порожден КС грамматикой, то есть существует грамматика $G_2 = \langle V, W, P_G, I_G \rangle$, такая что $L(M) = L(G_2)$.

ДОКАЗАТЕЛЬСТВО.

A. Заметим, что теорема 2.7 была теоремой синтеза, когда, имея грамматику, мы строим устройство, распознающее строки КС языка, порожденного этой грамматикой. Рассмотрим теорему 2.8. Эта теорема анализа, когда задано устройство и ставится задача найти формальное описание языка, принимаемого этим устройством.

B. Построим грамматику G_2 следующим образом:

1. Поставим в соответствии каждому знаку s стекового алфавита нетерминальный знак A_{ij} грамматики таким образом, что из нетерминального знака A_{ij} могут быть выведены (порождены) строки в результате вывода которых автомат M переходит из состояния q_i в состояние q_j и одновременно из стека извлекается знак s.

   $B_{ij} \xrightarrow{G_2} \alpha$, $B_{ij} \in W, \alpha \in A^*$, такие, что $q_i \xrightarrow{\alpha} q_j$.

   Пример. Если стековый алфавит $S = \{C, D\}$, а множество состояний $Q = \{q_0, q_1\}$, тогда алфавит нетерминальных знаков будет
   $W = \{C_{01}, C_{10}, C_{11}, C_{00}, D_{01}, D_{10}, D_{11}, D_{00}\}$.

2. Множество продукций P_G зададим следующим образом:
   - Обозначим заключительное состояние q_0 как q_n, если $Q = \{q_0, q_1, ..., q_{n-1}\}$.
   - Множество нетерминальных знаков определим как $W = \{B_{ij} \mid B \in S, 1 \le i, j \le n\}$.
   - Если имеется команда $q_i B a \to q_j B^{(1)} B^{(2)} ... B^{(k)}$, где $q_i, q_j \in Q$, $B, B^{(1)} ... B^{(k)} \in S$, $a \in A$, то в множество продукций P_G добавляем всевозможные продукции вида $B_{ij} \to a B^{(1)}_{in_1} B^{(2)}_{n_1 n_2} ... B^{(k)}_{n_{K-1} j}$, где $1 \le n_1, n_2, ..., n_{k-1} \le n$, где $n = |Q|$ - число состояний автомата M.
   - Аксиомой грамматики объявим нетерминальный знак $I = B_{1n}, B \in S$, где B_{1n} содержит множество строк, которые принимаются магазинным автоматом при переходе из начального состояния в конечное состояние, когда из стека извлекается знак B.

     Аксиому грамматики строим таким образом, что при извлечении из стека любого префикса начальной конфигурации, магазинный автомат переходит из

начального состояния в заключительное состояние, что эквивалентно принятию такого множества строк, которое будет соответствовать построенной нами аксиоме.

С. Вывод. Видно, что при переходе из начального состояния в заключительное состояние магазинный автомат принимает такое множество строк, которые описываются построенными продукциями P_G, которые, очевидно, определяют КС грамматику.

То есть произвольный магазинный автомат, если принимает какие–то строки, то их множество может бать порождено КС грамматикой.

ТЕМА 0.11. ДВУХСТЕКОВЫЙ АВТОМАТ

Определение. **ДВУХСТЕКОВЫЙ АВТОМАТ** это формальная система, задаваемая 6 объектами $D = \langle A, Q, S, F, P, I[, B] \rangle$, где

$A = \{a_1, a_2, ..., a_m\}$- конечный входной алфавит,

$B = \{b_1, b_2, ..., b_t\}$- конечный выходной алфавит,

$Q = \{q_0, q_1, ..., q_{n-1}\}$- множество состояний двухстекового автомата,

$S = \{s_1, s_2, ..., s_k\}$- стек просмотра назад,

$F = \{f_1, f_2, ..., f_w\}$- стек просмотра вперед,

P – система команд. Есть отображение множества декартового произведения множеств состояний двухстекового автомата, стека просмотра назад, входного алфавита и стека просмотра вперед в множество декартового произведения множеств состояний двухстекового автомата, стека просмотра вперед и стека просмотра назад.

$P : Q \times S \times A \times F \to Q \times F \times S^*$

Команды двухстекового автомата имеют вид

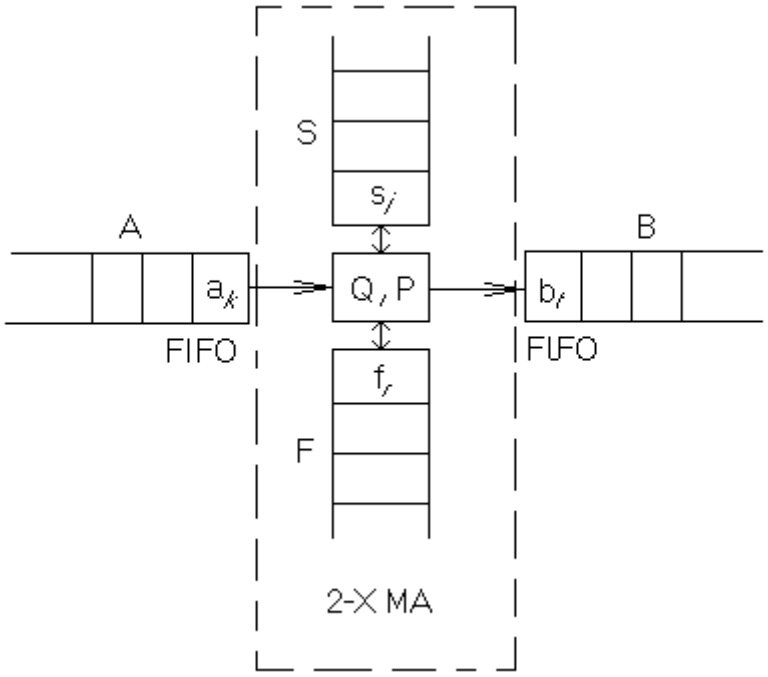
$P = \{q_i f_j a_k s_l \to q_i' f_j' \sigma \mid \sigma \in S^*; q_i, q_i' \in Q; f_j, f_j' \in F; s_l \in S; a_k \in A\}$.

Если P есть функция, то двухстековый автомат детерменированный.

$I = \varphi q_i \sigma$-начальная конфигурация двухстекового автомата, где

$\varphi \in F^*, \sigma \in S^*$, а F^* - есть начальное состояние стека.

Изобразим структуру двухстековаго автомата.



ТЕОРЕМА 2.9. *О двухстековом автомате и контекстных грамматиках.*

Для любого языка, порожденного контекстной грамматикой, существует двухстековый автомат, возможно недетерминированный, такой, что множество принимаемых им строк совпадает с множеством строк этого языка.

Верно и обратное: язык, принимаемый некоторым двухстековым автоматом может быть порожден контекстной грамматикой.

ТЕМА 0.12. КОНЕЧНЫЙ АВТОМАТ

Определение. **КОНЕЧНЫЙ АВТОМАТ** есть формальная система, задаваемая 5 объектами $K = \langle A, Q, B, I, P \rangle$, где

$A = \{a_1, a_2, ..., a_m\}$- конечный входной алфавит,

$B = \{b_1, b_2, ..., b_t\}$- конечный выходной алфавит,

$Q = \{q_0, q_1, ..., q_{n-1}\}$- конечное множество состояний конечного автомата,

P - система команд конечного автомата. P - есть отображение множества декартового произведения множеств состояний конечного автомата и входного алфавита в множество декартового произведения множеств состояний конечного автомата и выходного алфавита

$P : Q \times A \to Q \times B$.

P - есть множество всевозможных команд вида

$P = \{q_i a_j \to q_i' b_k \mid q_i, q_i' \in Q; a_j \in A; b_k \in B\}$,

$I = q_0$- начальная конфигурация конечного автомата, где $q_0 \in Q$.

Замечание. Иногда конечный автомат представляют как 5 объектов $K = \langle A, Q, B, \delta, \lambda \rangle$, где

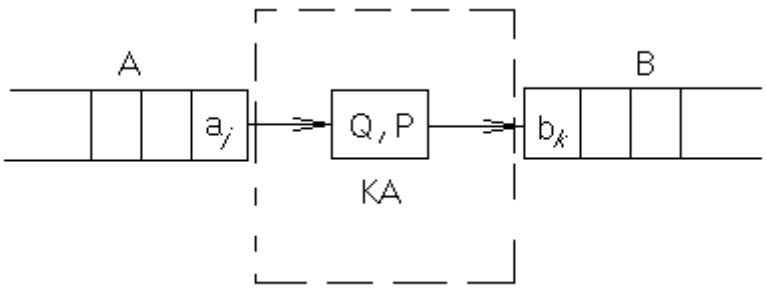
$\delta : Q \times A \to Q$ - отображение (функция) переходов,

$\lambda : Q \times A \to B$ - отображение (функция) выходов.

Очевидно, что это определение, с точностью до начального состояния, эквивалентно предыдущему. Действительно, отображение P декомпозируется на два отображения δ и λ. $P : Q \times A \to Q \times B$.

Заметим, что если P - функция, то задание автомата во втором виде может привести к недетерминированности.

Изобразим структуру конечного автомата.



Пример. Пусть есть конечный автомат $K = \langle A, Q, B, I, P \rangle$.

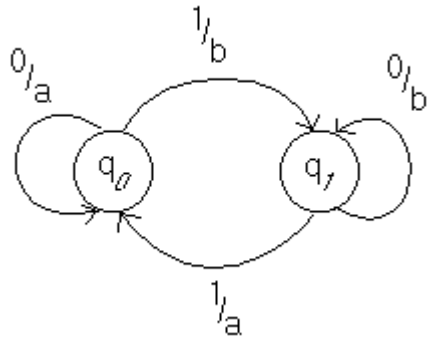
$A = \{0,1\},\ B = \{a, b\},\ Q = \{q_0, q_1\}$,

$$P=\begin{cases} q_0 0 \to q_0 a, \\ q_0 1 \to q_1 b, \\ q_1 0 \to q_1 b, \\ q_1 1 \to q_0 a. \end{cases}$$

Представим данный конечный автомат в виде автоматной таблицы.

$A \setminus Q$	q_0	q_1
0	q_0 / a	q_1 / b
1	q_1 / b	q_0 / a

Представим данный конечный автомат в виде графа переходов конечного автомата.



В этом случае функции

$$\delta=\begin{cases} q_0 0 \to q_0, \\ q_0 1 \to q_1, \\ q_1 0 \to q_1, \\ q_1 1 \to q_0. \end{cases} \qquad \lambda=\begin{cases} q_0 0 \to a, \\ q_0 1 \to b, \\ q_1 0 \to b, \\ q_1 1 \to a. \end{cases}$$

ТЕОРЕМА 2.10. *О представимости регулярных языков конечными автоматами.*

Для любого непустого регулярного языка $L(G)$, порождаемого регулярной грамматикой G, существует конечный автомат K, возможно недетерминированный, представляющий (порождающий и распознающий) язык $L(G)$.

Верно и обратное: для любого конечного автомата K существует регулярный язык $L(G)$, строки которого принимаются этим автоматом, то есть конечный автомат есть форма представления регулярных языков.

ДОКАЗАТЕЛЬСТВО.

А. Пусть задана регулярная грамматика $G=\langle V, W, I, P\rangle$, где

$V=\{a_1, a_2, ..., a_m\}$,

$W=\{w_1, w_2, ..., w_n\}$,

$I=w_1$,

$P=\{w_i \to a_j w_i^{/}, w_i \to a_j \mid w_i, w_i^{/} \in W; a_j \in V\}$.

B. Введем дополнительный нетерминальный знак w_0. Заменим в множестве продукций P все продукции вида $w_i \to a_j$ на продукции вида $w_i \to a_j w_0$ и добавим продукцию вида $w_0 \to e$. Очевидно, что мы получили грамматику, эквивалентную исходной, порождающую тот же язык $L(G)$.

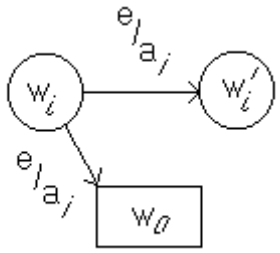
Обоснование этого факта заключается в том, что произвольный вывод в исходной грамматике имеет соответствующий ему вывод в преобразованной грамматике, отличающейся от исходной применением продукции $w_0 \to e$ в конце вывода.

Верно и обратное. Следовательно грамматики совпадают.

C. Построим конечный автомат K порождающий язык $L(G')$, где $G' = \langle V, W', I, P' \rangle$ - грамматика построенная в пункте B.

Тогда конечный автомат есть $K_П = \langle \{e\}, W', V, I, P'' \rangle$, где система команд конечного автомата P'' получена из множества продукций P' грамматики G' следующим образом: если имеется продукция вида $w_i \to a_j w_i'$, то в систему команд P'' добавляется команда $w_i e \to w_i' a_j$

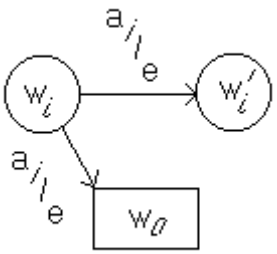
D. Покажем, что построенный автомат $K_П$ порождает тот же язык, что и грамматика G'



Показывается аналогично тому, как это делалось в пункте B.

E. **Синтез конечных автоматов.** Построим конечный автомат, распознающий (принимающий) строки языка $L(G')$, вида $K_P = \langle V, W', \{e\}, I, P''' \rangle$, где P''' построим следующим образом: для каждой продукции грамматики вида $w_i \to a_j w_i'$ в систему команд P''' конечного автомата добавляем продукцию вида $w_i a_j \to w_i' e$.

F.



Доказательство того, что построенный автомат K_P распознает (принимает) строки языка $L(G')$ выполняем аналогично тому, как это сделано в пункте D.

G. **Анализ конечных автоматов.** Покажем обратное утверждение теоремы.

Пусть задан конечный автомат $K = \langle A, Q, B, I, P \rangle$, где

$A = \{a_1, a_2, ..., a_m\}$,

$B = \{b_1, b_2, ..., b_t\}$,

$Q = \{q_0, q_1, ..., q_{n-1}\}$,

$P = \{q_i a_j \to q_i' b_k \mid q_i, q_i' \in Q; a_j \in A; b_k \in B\}$.

Н. Построим грамматику $G_П = \langle B, Q, I, P_П \rangle$, где аксиома $I \in Q$.

Множество продукций $P_П$ построим следующим образом: обозначим заключительное состояние как q_0. Для каждой команды конечного автомата K вида $q_i a_j \to q_i' b_k$ в множество продукций $P_П$ грамматики $G_П$ добавляем продукцию вида $q_i \to b_k q_i'$ и одну продукцию вида $q_0 \to e$.

Очевидно, что построенная грамматика $G_П = \langle B, Q, I, P_П \rangle$ порождает тот же язык, что и конечный автомат $K = \langle A, Q, B, I, P \rangle$.

I. Построим грамматику $G_P = \langle B, Q, I, P_P \rangle$, которая порождает язык, принимаемый конечным автоматом $K = \langle A, Q, B, I, P \rangle$ следующим образом: для каждой команды конечного автомата K вида $q_i a_j \to q_i b_k$ в множество продукций P_P грамматики G_P добавляем продукцию вида $q_i \to a_j q_i'$ и одну продукцию вида $q_0 \to e$.

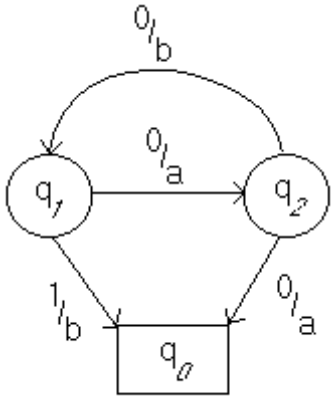
Очевидно, что полученная грамматика $G_P = \langle B, Q, I, P_P \rangle$ порождает язык, принимаемый автоматом $K = \langle A, Q, B, I, P \rangle$.

Вывод. Таким образом произвольный конечный автомат $K = \langle A, Q, B, I, P \rangle$ описывается двумя регулярными грамматиками, имеющими общий алфавит нетерминальных знаков и аксиому.

Окончание доказательства.

Пример. Произведем анализ конечного автомата.

Пусть задан конечный автомат K графом переходов.



$A = \{0,1\}$, $B = \{a, b\}$.

Автомат недетерминированный.

Используя теорему 2.10 запишем продукции грамматики, порождающей язык, строки которого распознаются данным конечным автоматом $G_P = \langle A, Q, I = q_1, P_P \rangle$, где множество продукций

$P_P = \{q_1 \to 0q_2, q_1 \to 1q_0, q_2 \to 0q_1, q_2 \to 0q_0, q_0 \to e\}$.

Построим грамматику, которая порождает язык, порождаемый конечным автоматом.

$G_П = \langle A, Q, I = q_1, P_П \rangle$, где множество продукций

$P_П = \{q_1 \to aq_2, q_1 \to bq_0, q_2 \to bq_1, q_2 \to aq_0, q_0 \to e\}$.

РАЗДЕЛ 3: Абстрактный синтез автоматов

«Tempora mutantur, et nos mutamur in illis”
(Времена меняются, и мы меняемся вместе с ними)
Латинский афоризм

Устройство управления автоматом – есть конечный автомат

1. Конечный автомат: $K =< A,Q,B,P_k,I_k >$ где $P_k : Q\times A \to Q\times B$ пусть есть два алфавита $\alpha_к = \{q_i a_j \mid q_i \in Q, a_j \in A\}$ и $\omega_к = \{q_i b_k \mid q_i \in Q, b_k \in B\}$, в результате продукцию можно выразить в виде $P_k : \alpha_k \to \omega_k$
2. Магазинный автомат: $M =< A,Q,S,B,P_m,I_m >$ где $P_m : Q\times S\times A \to Q\times S^*\times B$
   пусть есть два алфавита
   $\alpha_m = \{q_i s_j a_k \mid q_i \in Q, a_k \in A, s_j \in S\}$ и $\omega_к = \{q_i \sigma_j b_k \mid q_i \in Q, b_k \in B, \sigma_j \in S^*, \sigma_j - \text{из команд автоматов}\}$
   , в результате продукцию можно выразить в виде $P_m : \alpha_m \to \omega_m$
3. Двухстековый автомат: $D =< A,Q,S,F,B_D,P_D,I_D >$ где $P_D : Q\times F\times S\times A \to Q\times F\times S^*\times B$ поступаем аналогично магазинному автомату: $P_D : \alpha_D \to \omega_D$, где $|\alpha_D|,|\omega_D| \in N$
4. Сеть Петри: $S =< P_S,T,E,M >$, где $E : P_S\times T \to N\times N$, где заведомо известно, что число меток, конечно.
5. Машина Тьюринга: $T =< A,Q,P_T,I_T >$, где $P_T : Q\times A \to Q\times A\times D$, где $D = \{L,E,R\}$

Вывод: устройством управления всех известных автоматов есть конечный автомат. Это значит, что для изучения автоматов необходимо изучить запоминающие устройства, различные для каждого типа автомата, и устройства управления, являющегося конечным автоматом.

Т.3.1. Понятия и определения

определение: 1. Конечный автомат будем представлять как пятерку объектов: $K =< A,Q,B,\delta,\lambda >$ **-** , где A входной алфавит $A = \{a_1, a_2, \ldots, a_m\}$ **;** Q - алфавит (множество) состояний содержит **n** элементов, начиная с нуля $Q = \{q_0, \ldots, q_{n-1}\}$; B - выходной алфавит $B = \{b_1, \ldots, b_e\}$; δ - отображения (функция) переходов $\delta : Q\times A \to Q$ **;** λ - отображения (функция) выходов $\lambda : Q\times A \to Q$ **;**

Если не оговорено особо, начальное состояние автомата является q_1, а заключительное q_0. Если отображения δ и λ - функции, то автомат детерминированный, в противном случае автомат недетерминированный.

определение: 2. Конечный автомат называется **полным**, если для всех упорядоченных пар $q_i a_j$ имеется команда вида $q_i a_j \to q_i' a_j' \in P_K$**.** Из автоматного отображения (функции) δ **,** λ определены во всех точках. Это означает, что в автоматной таблице не существует пустых клеток, кроме разве что столбца соответствующего заключительному состоянию q_0**.**

определение: 3. Автоматное отображение – это соответствие, отображающее входные строки конечного автомата в выходные.

Пусть задан конечный автомат $K =< A,Q,B,\delta,\lambda >$. Подавая на его вход всевозможные строки в алфавите A, на выходе получаем строки в алфавите B т.е. конечный автомат K есть отображение универсального множества над алфавитом A в универсальное множество над алфавитом B.

$$K : A^+ \to B$$

Если задана строка $\alpha \in A^+$ то, будучи поданная на вход конечного автомата, приводит к появлению на выходе строки $\beta \in B^+$, что записывается так: $K(q_i,\alpha)$. K - будем называть **автоматным оператором.**

Пусть строка α есть $\alpha = a_{j1}, a_{j2}, \ldots, a_{jk}$

(последовательностью a_j) $K(q_i,\alpha): \delta(q_i,a) = \delta(\delta(\ldots \delta q_i, a)\ldots, a_{jk-1})a_{jk})$.

Индуктивное определение автоматного оператора, т.е. когда длина строки равна:1.

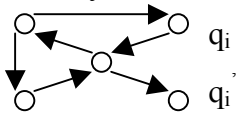
$K(q_i,a_j) = \lambda(q_i,a_{j1})$; K . $K(q_i,a_{j1}\ldots a_{jk}) = \beta_k$; K+1.

$K(q_i,a_{j1}\ldots a_{jk}a_{jk+1}) = \beta_k \lambda(\delta(q_i,a_{j1}\ldots a_{jk})a_{jk+1})$;

Свойства автоматного отображения

1. Свойство сохранения длины: пусть $\beta = K(q_i,\alpha)$, где $|\alpha| = |\beta|$.
2. Отсутствие предвосхищения: если строку α мы представили, как $\alpha = \alpha_1\alpha_2$ и нам известно, что $K(q_i,\alpha_1\alpha_2) = \beta_1\beta_2$ и если $|\alpha_1| = |\beta_1|$, то $K(q_i,\alpha_2) =_1 \beta_2$, иначе говоря, образ отрезка длины K равен отрезку образа той же длины, т.е. автоматный оператор перерабатывает строку слева на право, не заглядывая вперед или же K - ый знак выходной строки зависит только о первых K - знаков входной строки.

определение: 4. Достижимое состояние. Состояние q_k называется достижимым из состояния q_i, если существует такая входная строка α , что $\delta(q_i,\alpha) = q_k$, $\alpha \in A^+$.



определение: 5. Сильно связный автомат – это такой автомат, когда из любого его состояния достижимо его любое другое состояние, т.е. на графе автомата существует путь соединяющий любые его две вершины.

определение: 6. Автоматный автомат (генератор) – это автомат, у которого входной алфавит состоит из одного знака $A = \{\alpha\}$, иногда говорят, что вход A это вход синхронизаций, задающий такты работы автомата.

Гомоморфизм и изоморфизм

Пусть заданы два конечных автомата $K_1 =< A_1, Q_1, B_1, \delta_1, \lambda_1 >$ и $K_2 =< A_2, Q_2, B_2, \delta_2, \lambda_2 >$

определение: 7. Гомоморфизмом автомата K_1 в автомат K_2 - называется тройка функций:

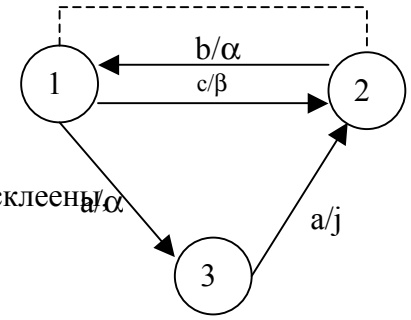
$f: A_1 \to A_2 \mid a \in A_1, q \in Q_1, b \in B_1$;

$g: Q_1 \to Q_2 \mid \delta(\mathrm{g(q)}, \mathrm{f(a)}) = \mathrm{g}(\delta_1(q_1,a))$;

$h: B_1 \to B_2 \mid \lambda_2(gq), f(b)) = h(\lambda_1,(q,a))$; Гомоморфизм означает, что существует такое переименование алфавитов, которое переводит функцию δ_1 в δ_2, а λ_1 в λ_2.

определение: 8. Изоморфизмом называется взаимно однозначный гомоморфизм. При изоморфизме происходит замена одного знака алфавита, на знак другого алфавита, это взаимно однозначно, в то время как при гомоморфизме происходит склеивание знаков (совмещение).

Пример:

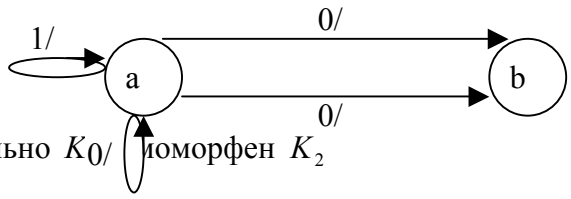


$f: a \to 0, a \to 1, a \to 0$

$g: 1 \to a, 2 \to a, 3 \to b$

$h: \alpha \to a, \beta \to a, \gamma \to b$

состояния 1 и 2 склеены.



следовательно $K_{0/}$ изоморфен K_2

определение: 9. Неотличимые состояния. **Два состояния** q **и** q' называются неотличимыми, если для всех строк $\alpha \in A^+$ $K(q,\alpha) = K(q',\alpha)$.

определение: 10. Автоматы K_1 и K_2 называются неотличимыми, если для каждого состояния автомата $q_1 \in Q$ существует единственное состояние $q_2 \in Q$, такое что A, $\alpha \in A^+$, $K_1(q_1,\alpha) = K_2(q_2,\alpha)$.

определение: 11. Эквивалентные автоматы – это такие автоматы K_1 и K_2, когда для любого состояния автомата K_1 найдется неотличимое от него состояние автомата K_2.

Т.3. 2. Минимизация конечных автоматов

Теорема 3. 1. О минимизации конечных автоматов

Формулировка: Для любого конечного автомата $K =< A, Q, B, \delta, \lambda >$ существует неотличимый от него минимальный автомат $K_0 =< A_0, Q_0, B_0, \delta_0, \lambda_0 >$ единственный с точностью до изоморфизма, имеющий $|Q_0| = l$ состояний, если множество состояний автомата Q разбивается на l классов эквивалентности

$$C_1 = \{q_{11}, q_{12}, \ldots, q_{1l}\};$$
$$C_1 = \{q_{21}, q_{22}, \ldots, q_{2l}\};$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$C_1 = \{q_{l1}, q_{l2}, \ldots, q_{ll_l}\}.$$

Доказательство: **A)** постановка задачи

Среди автоматов эквивалентных K (неотличимых от K) найти автомат K_0 с наименьшим числом состояний. По определению неотличимости: если исходный автомат $K(q_i,\alpha) = K_0(c_j,\alpha)$, где $q_i \in C_j$.

B) Пусть q_{j1} и q_{j2} принадлежат одному классу эквивалентности $C_k (q_{j1}, q_{j2} \in C_k$, тогда для $\forall a \in A$ из входного алфавита A, следующие состояния должны принадлежать одному и тому же классу эквивалентности (возможно другому) $\delta(q_{j1},a)$, $\delta(q_{j2},a) \in C_k'$. Докажем это: $\forall \alpha \in A^+$, $K(q_{j1},\alpha) = K(q_{j2},\alpha)$. Предположим, что $\delta(q_{j1},a)$ и $\delta(q_{j2},a)$ - отличимы. Это означает, что найдется строка α, такая что $\exists \alpha \in A^+ \mid K(\delta(q_{j1},a),\alpha) \neq K(\delta(q_{j2},a),\alpha)$. Но мы знаем, что $\delta(q_{j1},a_\alpha)$ можно представить, как $\delta(q_{j1},a) = \delta(\delta(q_{j1},a),\alpha)$, это означает, что $K(q_{j1},a_\alpha) = K(q_{22},a_\alpha)$ т.е. мы показали, что состояния q_{j1} и q_{j2} различны.

C) Минимальный автомат K_0 определим так: $K_0 =< A_0, Q_0, B_0, \delta_0, \lambda_0 >$, где $Q_{01} = \{c_1, c_2, \ldots, c_l\}$ (множество классов эквивалентности) $\delta_0 : \forall \alpha \in A$ $\delta_0(c_j,a)$ будут равны тому же состоянию C_j', где C_j' вычисляется так: $\delta(c_j,a) = q_j'$, где $q_i \in c_j$, $q_j' \in c_j'$.

Физический смысл: δ_0 - это функция, заданная на двух множествах: на множестве входных знаков и множестве состояний, которое мы ввели. Результат функции новое состояние. λ_0 - определим аналогично: $\forall \alpha \in A \quad \lambda_0(c_j, a) = \lambda(q_i, a), q_j \in c_j$ таким образом, мы строим отображение (функцию) выхода минимального автомата.

D) Построенный автомат K_0 неотличим от автомата K, причем автомат K_0 не имеет неотличимых состояний.

E) Докажем, что K_0 минимален. Т.е. не имеет неотличимых состояний:

a) Предположим, что K_0 не минимален, т.е. существует $K_0^{'}$ с меньшим числом состояний $|Q_0^{э'}| < l$.

b) По определению неотличимости состояний, для каждого состояния K_0 найдется неотличимое состояние $K_0^{'}$, но так как $K_0^{'}$ имеет меньшее число состояний, то какие, то два состояния K_0 неотличимы от одного состояния $K_0^{'}$.

c) В силу транзитивности неотличимости т.е. если q_{j1} и q_{j2} неотличимы от какого-то состояния q, то q_{j1} и q_{j2} неотличимы между собой ($q_{j1} \approx q, q_{j2} \approx q \Rightarrow q_{j1} \approx q_{j2}$) откуда следует, что в автомате K_0 имеются неотличимые состояния, что противоречит нашему допущению (пункт а).

F) Докажем, что любой минимальный автомат $K_0^{'}$ автомата K изоморфен автомату K_0

a) Пусть $K_0^{'}$ любой другой минимальный автомат от автомата K т.е. в соответствии с пунктом Е, K_0 и $K_0^{'}$ имеют одинаковое число состояний, причем K_0 и $K_0^{'}$ неотличимы.

b) Тогда различные состояния $K_0^{'}$ неотличимы от различных состояний K_0, поэтому между K_0 и $K_0^{'}$ можно установить изоморфизм.

Замечание 1. Для того чтобы воспользоваться теоремой, для построения минимального автомата необходимо уметь находить классы эквивалентности состояний.

Замечание 2. Определение неотличимости не содержит конструктивной процедуры проверки неотличимости двух состояний, так как предполагает перебор по бесконечному множеству входных строк.

Алгоритм Мили (алгоритм эквивалентных состояний)

a) Пусть задан автомат $K =< A, Q, B, \delta, \lambda >$ и известно, что число его состояний $n = |Q|$.

b) Зададим начальное приближение классов эквивалентности $C^{(0)} = \{c_1^{(0)}, c_2^{(0)}, \ldots, c_{l0}^{(0)}\}$ и сделаем это следующим образом: два состояния q и $q^{'}$ относим в один класс эквивалентности $c_j^{(0)}$, если и только если для всех входных знаков $\forall a \in A \quad \lambda(q_1, a) = \lambda(q_1^{'}, a)$.

c) Пусть на шаге i получено разбиение на классы эквивалентности $C^{(i)} = \{c_1^{(i)}, c_2^{(i)}, \ldots, c_{li}^{(i)}\}$, тогда на шаге $i+1$ два состояния q и $q^{'}$ из одного класса эквивалентности $c_j^{(i+1)}$ относим в один класс $c_k^{(i+1)}$ если и только если $\forall \alpha \in A \ \delta(q, a); \delta(q^{'}, a) \in C_k^{(i)}$ иначе образуем новый класс $c_s^{(i+1)}$ куда заносим состояние q и все другие $q^{"}$ состояния, для которых $\delta(q, a); \delta(q^{"}, a) \in C_k^{(i)}$

d) Завершаем процедуру на шаге к, если этот шаг не изменяет разбиение на классы, иначе повторяем с).

Замечание: Алгоритм Мили конструктивен, так как число шагов не более чем n-1. По построению видно, что состояние из одного множества неотличимо.

Пример: Конечный автомат K задан в виде автоматной таблицы.

A\Q	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
a_1	q_2/0	q_1/1	q_1/1	q_8/0	q_6/1	q_8/0	q_6/1	q_4/1	q_7/0
a_2	q_4/1	q_1/0	q_6/0	q_1/1	q_1/1	q_9/1	q_1/1	q_4/0	q_9/0
a_3	q_4/1	q_5/0	q_5/0	q_1/1	q_3/0	q_6/1	q_3/0	q_7/0	q_7/1

Задаем начальное приближение:

0. $\{q_1, q_4, q_6, q_9\}$ $\{q_2, q_3, q_8\}$ $\{q_5, q_7\}$

шаг 1: два состояния относим в один класс (по числителю)

1. $\{q_1, q_4, q_6\}$ $\{q_9\}$ $\{q_2, q_3, q_8\}$ $\{q_5, q_7\}$
2. $\{q_1, q_4\}$ $\{q_6\}$ $\{q_9\}$ $\{q_2, q_3, q_8\}$ $\{q_5, q_7\}$
3. $\{q_1, q_4\}$ $\{q_6\}$ $\{q_9\}$ $\{q_2, q_3, q_8\}$ $\{q_5, q_7\}$

Так как разбиение на классы не изменилось, можно сделать вывод, что в минимальном автомате будет 6 состояний -–это C_1, C_2, C_3, C_4, C_5, C_6

$A\backslash Q_0$	C_1	C_2	C_3	C_4	C_5	C_6
a_1	C_4/0	C_4/0	C_6/0	C_1/1	C_1/1	C_2/1
a_2	C_1/1	C_3/1	C_3/1	C_1/0	C_2/0	C_1/1
a_3	C_1/1	C_2/1	C_6/1	C_6/0	C_6/0	C_5/0

Т.3.3. Автомат Мура

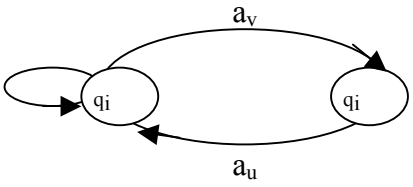
Ранее данное определение конечному автомату описывает автомат, который называется автоматом Мили. В теории автоматов рассматривается другая форма представления конечного автомата, и такой автомат называется автоматом Мура.

определение 12:Автоматом Мура называется такой конечный автомат, у которого функция входов λ не зависит от входного знака и зависит только от состояния.

Пусть $K =< A,Q,B,\delta,\lambda >$. Если для любого состояния $\forall q_i \in Q$ и для всех пар знаков $\forall a_j, a_k \in A \quad \lambda(q_i,a_j) = \lambda(q_i,a_k)$ то это автомат Мура.

Замечание: функция выходов является одноаргументной функцией μ.

μ-функция отметок, так как в автомате Мура выходной знак зависит только от состояния, в котором находится автомат, то на графе выходной знак пишется не на дуге, а внутри вершины состояний.



Теорема 3.2. о существовании автомата Мура

Утверждение: Для любого автомата Мили $K =< A,Q,B,\delta,\lambda >$ существует неотличимый от него автомат Мура $K_m =< A_n,Q_n,B_m,\delta_m,\mu >$.

Доказательство:

А)Пусть задан автомат Мили $K =< A,Q,B,\delta,\lambda >$ такой что: $A = \{a_1, a_2, \ldots, a_m\}$

$Q = \{q_1, \ldots, q_n\}$

$B = \{b_1, \ldots, b_e\}$

В)Определим автомат Мура K_m

а) $Q_m = \{q_{10}, q_{20}, \ldots, q_{n0}, q_{11}, q_{21}, \ldots, q_{1m}, q_{21}, q_{22}, \ldots, q_{2m}, q_{n1}, q_{n2}, \ldots, q_{nm}\}$ $|Q_m| = n + nm = (m+1)$, где q_{i0} соответсвует состоянию $q_i, i = \overline{1,n}$

q_{ij} соответствует упорядоченным парам $(q_i a_j)$таким что

$q_{ij} \leftrightarrow (q_i, a_j) q_i \in a, a_j \in \alpha \ (i = \overline{1,n}, j = \overline{1,m})$

б) Определим функцию переходов автомата Мура:

$\delta_m(q_{i0}, a_k) = q_{ik}$

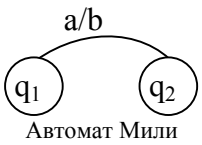
$\delta_m(q_{ij}, a_k) = q_{lk}$, где l такова, что $\delta(q_i, a_j) = q_l$

в) Функция ошибки $P(q_{i0})$ - неопределенна, можно принимать любые значения $\mu(q_{ij}) = \alpha(q_i, a_j)$.

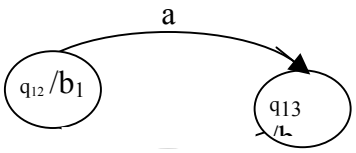
г) начальное состояние автомата Мура q_{i0} т.е. $I_m = q_0$, если начальное состояние автомата Мили - q_i.

С) Покажем, что эти автоматы неотличимы, т.е. $K(q_i, \alpha) = K_m(q_{i0}, \alpha) \ \forall \alpha \in A^+$. Для доказательства используем индукцию по длине строки α

Шаг 1: базис индукции: пусть длина строки $|\alpha| = 1$ тогда $K(q_i, a_j) = \alpha(q_i, a_j) = \mu(q_{ij}) = \lambda(q_i, a_j)$



Автомат Мили



Шаг p: предположим, что $K(q_i, \alpha_p) = K_m(q_{i0}, \alpha_p)$ где длина строки $|\alpha_p| = P$, $K(q_i, a_j, \ldots, a_k) = \alpha(q_i, a_j) \ldots \alpha(q_s, a_k)$,пусть $\delta(q_s, a_k) = q_t$.

Шаг p+1: $|\alpha_p + 1| = p + 1$ очевидно, что $K_m(q_{i0}, a_j, \ldots, a_k) = \mu(q_{ij}) \ldots \mu(q_{sk}) = \alpha(q_i, a_j) \ldots \alpha(q_s, a_k)$

Вывод: результат обработки этих строк – одинаков.

Пример: Задан автомат Мили. Построить неотличимый от него автомат Мура.

A\Q	q_1	q_2	q_3
a_1	q_2/a	q_3/c	q_1/b
a_2	q_1/b	q_2/a	q_2/a

$B = \{a, b, c\}$

Строим неотличимый от него автомат Мура $K_m = < A_n, Q_n, B_m, \delta_m, \mu >$,

$Q_m = n(m+1) = 3 \cdot (2+1) = 9$

$Q_m = \{q_{10}, q_{20}, q_{30}, q_{11}, q_{12}, q_{21}, q_{22}, q_{31}, q_{32}\}$

A\Q	q_{10}/-	q_{20}/-	q_{30}/-	q_{11}/a	q_{12}/b	q_{21}/c	q_{22}/a	q_{31}/b	q_{32}/a
a_1	q_{11}	q_{21}	q_{31}	q_{21}	q_{11}	q_{31}	q_{21}	q_{11}	q_{21}
a_2	q_{12}	q_{22}	q_{32}	q_{22}	q_{12}	q_{32}	q_{22}	q_{12}	q_{22}

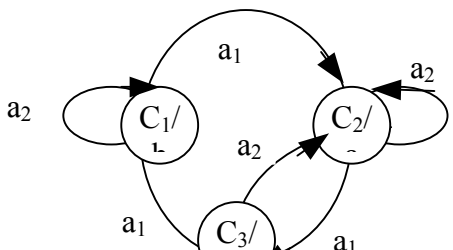
Минимизируем полученный автомат Мура

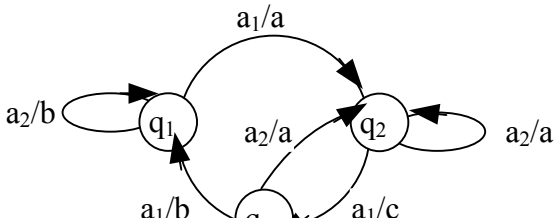
$\{q_{11}, q_{22}, q_{32}\}, \{q_{12}, q_{31}\}, \{q_{21}, q_{30}\}$

C_2 C_1 C_3

Рисуем итоговую таблицу

$A\backslash Q_m$	C_1/b	C_2/a	C_3/c
a_1	C_2	C_3	C_1
a_2	C_1	C_2	C_2





Т. 3.4. Детерминация автоматов

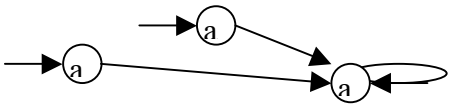
Алгебра регулярных событий. Пусть заданы два языка L_1 и L_2 над некоторым алфавитом V^*. Введем три операции:

1. объединение языков $L_1 \cup L_2 = \{\alpha \mid \alpha_1 \in L_1 \text{ или } \alpha_2 \in L_2\}$
2. конкатенация языков $L_1 \circ L_2 = L_1 L_2 = \{\alpha_1 \in L_1 \alpha_2 \in L_2\}$

3. итерация языков $L_1^* = e \cup L_1 \cup L_1 L_1 \cup \ldots = \bigcup_{i=0}^{\infty} L_i^*$ (объединение степеней языка, где i изменяется от 0 до i,(итерация языка из которой исключена пустая строка $L_1^* = L_1^* \setminus \{e\}$)).

Определение13: Элементарные события (выражения): всевозможные знаки алфавита V - есть элементарные события т.е. $a_i \mid a_i \in V$, таким образом **алгеброй регулярных событий -** называется совокупность объектов , где используется $R =< V, \{\cup, \circ, *, +\} >$, которая позволяет получить всевозможные выражения результатом вычисления которых является множество строк.

Пример: построим регулярное выражение: пусть задан алфавит V который состоит из двух знаков, $V = \{a, b\}$. Очевидно, что регулярным выражением являются сами знаки a и b. На их основе сделаем более сложные элементарные выражения: например, $(a \cup b)a^+$. Это элементарное выражение можно представить в виде графа, где вершинами являются знаки алфавита, $(a \cup b)a^+ = \{aa, aaa, \ldots, ba, ba\}$



Теорема 3.3. Об алгебре регулярных событий и регулярн6ой грамматике

Утверждение: Любой язык, построенный в алгебре регулярных событий (заданный регулярным событием) является языком порождаемым регулярной грамматикой.

Доказательство: **A)** Пусть заданы два события L_1 и L_2 над алфавитом V^*. Элементарные события - это всевозможные знаки a_i принадлежащие V.

B)Представим основные операции над событием в виде продукций регулярной грамматики, зная или предполагая, что существуют регулярные грамматики порождающие L_1 и L_2. $\exists G_1, G_2, L_1 = L(G_1)\ L_2 = L(G_2)$ В случае элементарных языков это очевидно, что $L_1 = \{0\}$, тогда грамматикой порождающей этот язык является $G = \langle V, \{I_2\}, \{I_1 \to \alpha\}, I_1 \rangle$.

C) Операция объединения языков $L_1 \cup L_2$. Зная грамматики содержащие L_1 и L_2, где $G_1 = \langle V, W_1, P_1, I_1 \rangle$ и $G_2 = \langle V, W_2, P_2, I_2 \rangle$. Построим грамматику $G = \langle V, W, P, I \rangle$ такую что $L(G) = L_1 \cup L_2$, очевидно, что для этой грамматики $P = \{I \to I_1 \mid I_2\} \cup P_1 \cup P_2$, а множество нетерминальных знаков $W = \{I\} \cup W_1 \cup W_2$.

D) Конкатенация двух языков $L_1 \circ L_2$.

Существует ли регулярная грамматика, которая порождает конкатенацию двух языков, если известно, что каждый язык порождается регулярной грамматикой. В этом случае P определим как $P_1 \circ P_2$, где продукции вида $\{A \to a\} \in P_1$ такие что $A \in W_1$, $a \in V$, изменяем и приводим к виду $\{A \to aI_2\} \in P_1'$ и получим грамматику G которая содержит объединение нетерминальных знаков первой и второй грамматик. Построим множество продукций P и аксиому I_1. В результате видно, что любой вывод строки начинается с строки языка L_1, а по завершению вывода в G_1 начинается вывод в G_2 и порождается суффикс строки $L_1 \circ L_2$.

E) Итерация языков L_1^*.

Строим грамматику G которая имеет тот же алфавит терминальных знаков (они остаются те же), а множество продукций будет следующим: $P = \{I \to e \mid I_1\} \cup P_1'$, где P_1' - это множество продукций преобразованных аналогично пункту D, используя вместо I_2 аксиому $I_1, G = < V, W \cup \{I\}, P, I >$.

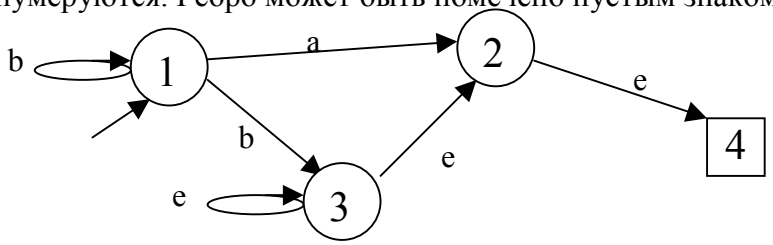
F) Если выбрать в качестве исходных языков элементарные события, которые порождаются регулярной грамматикой, то любое регулярное выражение представляет собой регулярный язык.

Теорема 3.4. Теорема Клини

Утверждение: Для любого регулярного выражения существует конечный автомат представляющий (распознающий, порождающий) язык, задаваемый этим выражением.
Доказательство: Очевидно, что в соответствии с теоремой 3.3. язык задаваемый регулярным выражением можно представить регулярной грамматикой. В соответствии с теоремой о регулярной грамматике и конечном автомате (2.?) мы знаем, что для каждой регулярной грамматики существует конечный автомат представляющий ее, следовательно, эта теорема справедлива.

Источники

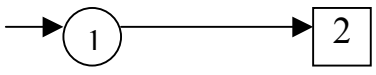
Для описания произвольного множества строк в некотором алфавите можно использовать графы , ребра, которого помечены знаками этого алфавита. Такие графы называются источниками.
Источником называется ориентированный граф, в котором выделены начальные и заключительные (состояния) вершины, причем начальные помечаются стрелочкой из неоткуда → ○ а заключительные квадратом □ . Ребра помечаются знаками некоторого алфавита V, вершины нумеруются. Ребро может быть помечено пустым знаком e.
Пример:



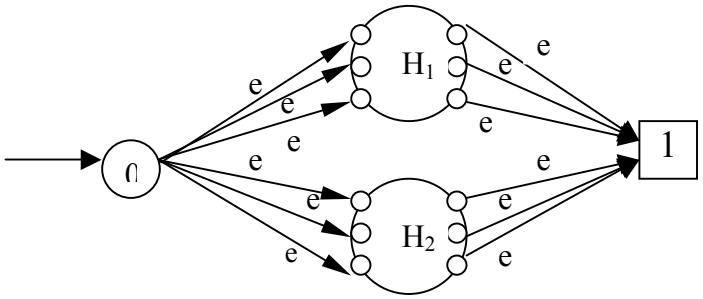
Теорема 3.5. Об источниках и регулярных выражениях

Утверждение: Для любого регулярного выражения R существует источник представляющий тот же язык .
Доказательство: **A)**пусть задано регулярное событие R образованное операциями U,•,*,+ . Рассмотрим два регулярных события. Предположим , что для представления этих двух событий нужно построить графы H_1 и H_2. Очевидно, что элементарные события представлены источниками.

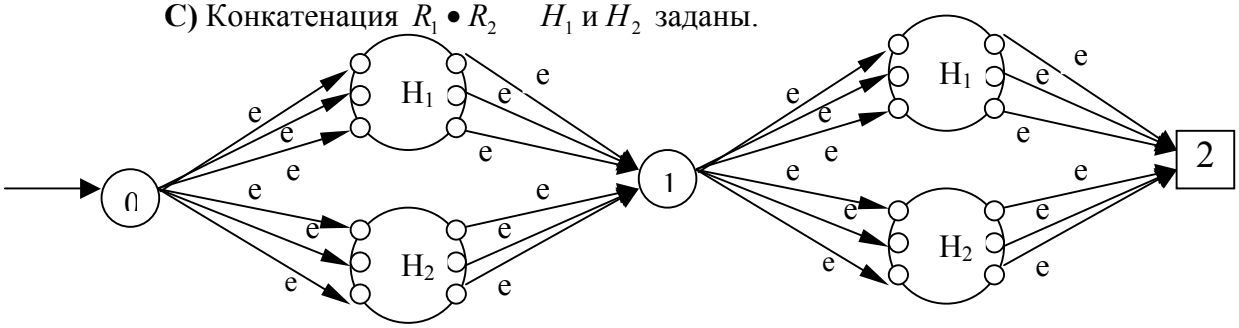
H_1, $R_1 = a$



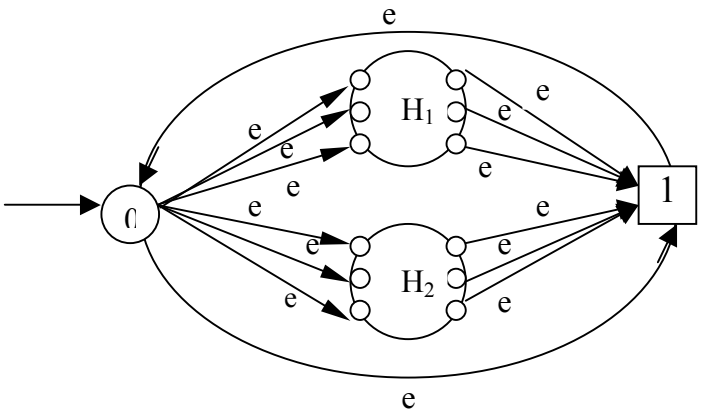
B) Покажем, что объединение элементарных событий , так же представимо источником $R_1 \cup R_2$ H_1 и H_2 заданы.



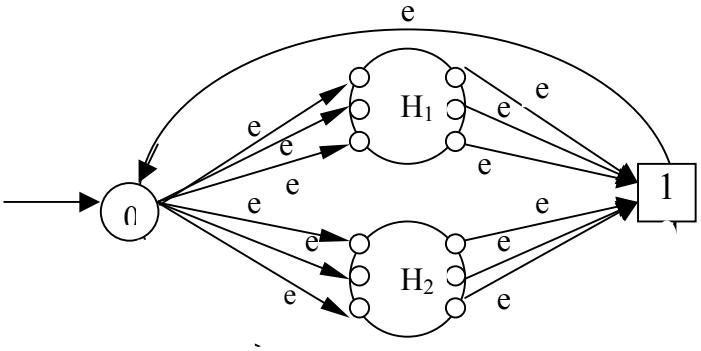
C) Конкатенация $R_1 \bullet R_2$ H_1 и H_2 заданы.



D) Итерация R_1^* H_1 известно

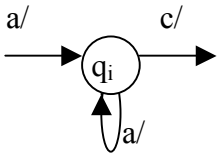


E) R_1^+ H_1 известно



Теорема 3.6. О детерминации источников

определение 2:асинхронный автомат (источник) - называется такой конечный автомат (источник) у которого все состояния (вершины) устойчивы по каждому входному знаку.



Физический смысл: если есть конечный автомат и есть входной канал А, то такты работы автомата определяются сменой входных знаков.

aaaabbbaa

↓ ↓ ↓

$t_1 \quad t_2 \quad t_3 - такты$

Математическое определение: для всех знаков $\forall q_i \in Q$ и всех знаков a_j из входного алфавита A должно выполнятся $\delta(q_i, a_j a_j) = \delta(q_i, a_j)$.

Такое же определение действует для источника , где автомат общего вида - синхронный.

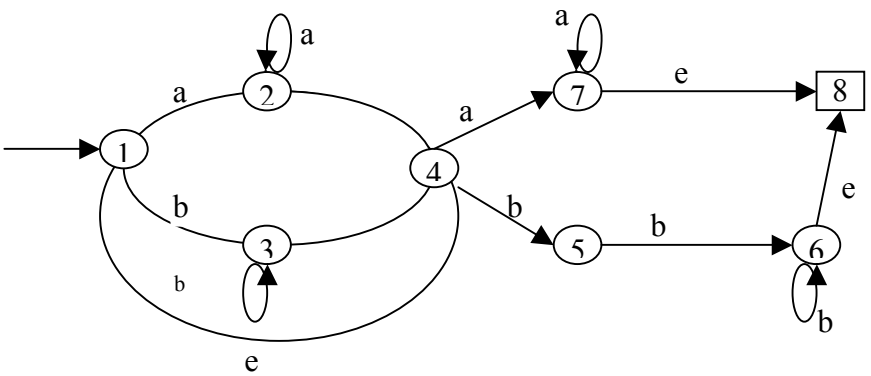
определение 13:Достижимая вершина. Для входного знака a_j из состояния q_i вершина $q_i^{'}$ называется достижимой , если существует путь в графе. Помеченный входными знаками $ee\ldots ea_j ee\ldots e$. Различают синхронную и асинхронную достижимость. При синхронной достижимости в этом пути существует единственный знак a_j. При асинхронной достижимости - число знаков a_j произвольное(>=1) т.е. помечающий путь представлен так: $ee\ldots ea_j ee\ldots ea_j ee\ldots e$.

определение 14: множество достижимых вершин(Q_i^j)- то множество вершин , которое достижимо из состояния q_i при входном знаке a_j.

Пример: рассмотрим выражение (регулярное) $(a \cup b)^* (bb \cup a)$

Очевидно, не каждый язык является асинхронным. Для того чтобы представить любой язык асинхронным , необходимо чтобы регулярное выражение имело вид $(aa^* \cup bb^*)^* (bbb^* \cup aa^*)$. в таком случае мы можем, упростив $(a^+ \cup b^+)^* (b^+ b^+ \cup a^+)$. Из него видно, что после преобразования мы получили асинхронный язык.

Изобразим этот источник:

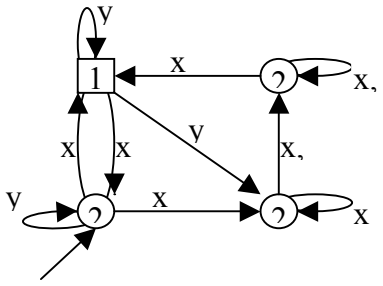


Это асинхронный источник, так как все вершины устойчивы к каждому входному знаку.
Проверим достижимость.
$Q_1^a = \{2,4,1\}, Q_1^b = \{3,4,1\}$- это синхронная достижимость
$Q_1^a = \{2,4,1,7,8\}, Q_1^b = \{3,4,1,5,6,8\}$- это асинхронная достижимость
Утверждение: Для любого недетерминированного источника с n вершинами существует эквивалентный ему детерминированный источник, имеющий не более чем 2^n состояний
Доказательство: **A)** Пусть задан источник $H =< A, Q, \delta >$, где A - набор входных знаков; Q - множество вершин; δ -множество дуг - это отображение, которое задает дуги этого графа. Построим неотличимый от него источник H_D .Очевидно $H_D =< A_D = A, Q_D, \delta_D >$, где H_D - детерминированный источник.

B)

шаг1:
для множества начальных вершин (состояний) и всех знаков a_j строим систему подмножеств вершин достижимых из вершин множества Q_1 для всех входных знаков. Используем синхронную или асинхронную достижимость. Объявляем полученную систему множеств $m =| A |$ множеством вершин источника H_D .
$Q_D^{(1)} = \left\{ Q_1^{a_j} \mid \forall a_j \in A, Q_1 - \text{множество начальных вершин} \right\}$
шаг к:
пусть получено множество вершин источник $Q_D^{(k)}$.
шаг к+1:
для каждой полученной ранее вершины источника H_D -(а это множество вершин исходного источника) $Q_D^{(k)} = \left\{ Q_1^{a_j} \mid \forall a_j \in A \right\}$
Строим множество вершин источника H , которые достижимы из вершин $Q_i^{a_j}$ (полученных на предыдущем шаге) для всех знаков. Считаем полученные множества $Q_D^{(k+1)}$ новыми вершинами источника H_D т.е. новое множество вершин будет $Q_D^{(k)} \cup Q_D^{(k+1)}$
шаг s:
завершаем построение, если разбиение на множество вершин не изменилось.

C) Докажем, что полученное множество вершин является вершинами детерминированного источника, неотличимого от исходного. Может быть построено не более чем 2^n вершин, где $n =| Q |$ - число вершин исходного источника $| Q_D^{(s)} |= 2^n$
Пример: пусть задан источник или же автомат в контексте распознавания входных знаков



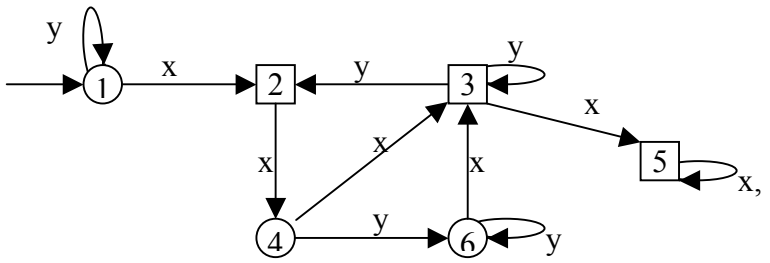
1. строим таблицу

A\Q	X	Y
{0}	{1,2}	{0}
{1,2}	{0,2,3}	{1,2,3}
{0,2,3}	{1,2,3}	{0,3}
{1,2,3}	{0,1,2,3}	{1,2,3}
{0,3}	{1,2,3}	{0,3}

{0,1,2,3}	{0,1,2,3}	{0,1,2,3}

2. строим множество достижимых вершин, используя синхронную достижимость.
3. строим неотличимый от исходного детерминированного источника гомоморфный ему, задавая гомоморфизм таблицей.

Q_D	{0}	{1,2}	{0,2,3}	{1,2,3}	{0,3}	{0,1,2,3}
Q_D`	1	2	3	4	5	6



заключительными являются те вершины, которые содержат хотя бы одну заключительную вершину исходного источника (в данном случае 1).

Проверка: $H(0, yxyyx); H(0, xx)$

$H_D(1, yxyyx); H(1, xx)$

РАЗДЕЛ 4. СТРУКТУРНЫЙ СИНТЕЗ АВТОМАТОВ

Самый отдаленный пункт… к чему-нибудь да близок, а самый близкий – от чего-нибудь да отдален.

Козьма Прутков.

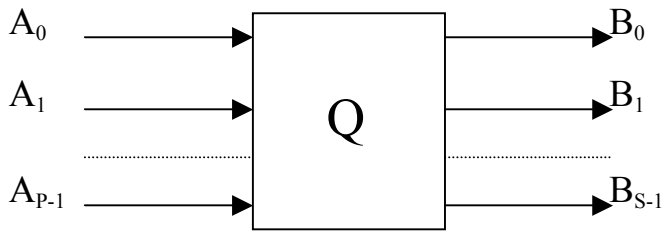
В данном разделе в отличие от предыдущего мы займёмся теоретической реализацией конечных автоматов при помощи логических элементов.

Тема 4.1. Сети из автоматов

Определение1: Автоматной сетью называется соединение автоматов, осуществляемое таким образом, что они функционируют согласованно, и образуют новый автомат. Такой подход, при котором автомат рассматривается как совокупность соединенных автоматов, мы раньше называли микроподходом. До сих пор автоматы рассматривались нами макроподходом, т.е. мы описывали его не вникая, из чего он состоит.

Пусть дан автомат, на входных каналах которого поступают алфавиты $A = A_0, A_1, ..., A_{p-1}$, а на выходе получаем алфавиты $B = B_0, B_1, ..., B_{s-1}$,и множеством состояний Q.

Количество входных и выходных каналов у КА можно расширить.



Представим такой автомат как конечный автомат:

$$K = < A, Q, B, \delta, \lambda >,$$

Входные алфавиты являются декартовым произведением алфавитов $A = A_0, A_1, ..., A_{p-1}$:

Error! Objects cannot be created from editing field codes.,

а выходные алфавиты также являются декартовым произведением алфавитов $B = B_0, B_1, ..., B_{s-1}$:

$$B = B_0 \times B_1 \times ... \times B_{s-1} = \prod_{i=0}^{s-1} B_i .$$

$a \in A$,*то есть* $a = (a^{(1)}, a^{(2)} ... a^{(p-1)})$,

$a^{(i)} \in A_i$.

Определение2: *Сетью из автоматов (автоматной сетью)* называют соединения конечных автоматов таким образом, что выход одного автомата служит входом для другого и они функционируют согласованно, то есть изменяют свои состояния в одни и те же дискретные моменты времени.

Основные виды соединения автоматов

Возможны различные соединения абстрактных автоматов, когда на вход одного из них подаются знаки с выхода другого, а выход, служит входом для других автоматов, при этом функционирование автоматов согласованно во времени (синхронизировано).

Существует три вида соединения автоматов:

Тема 1.1 Параллельное;

Тема 1.2 Последовательное;

Тема 1.3 С обратной связью.

Произвольную автоматную сеть можно получить путем этих соединений. Рассмотрим каждое из них.

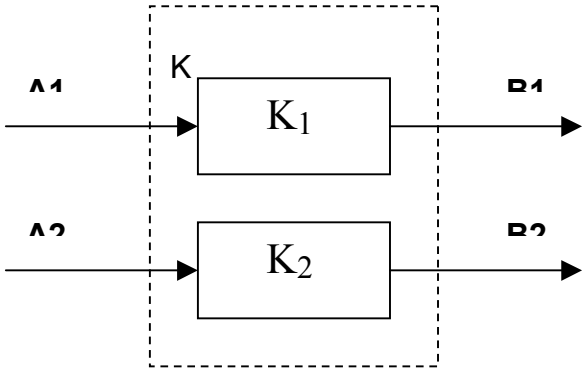
Тема 1.4 *Параллельное соединение автомата с раздельными входами.*

Пусть заданы два конечных автомата K_1, K_2.

$$K_1 =< A_1, Q_1, B_1, \delta_1, \lambda_1 >$$

$$K_2 =< A_2, Q_2, B_2, \delta_2, \lambda_2 >$$

Соединим их попарно с раздельными входами и получим новый автомат *K*, представленный в следующем виде:



У полученного автомата входной алфавит будет являться декартовым произведением входных алфавитов $A = A_0, A_1, ..., A_{p-1}$.Также поступаем с множеством состояний $A = Q_0, Q_1, ..., Q_{n-1}$ и выходными алфавитами $B = B_0, B_1, ..., B_{s-1}$ и представим в следующем виде:

$$K =< A_1 \times A_2, Q_1 \times Q_2, B_1 \times B_2, \delta, \lambda >$$

Тогда функция переходов примет вид:

$$\delta(q,a) = \left\langle \delta_1\left(q^{(1)}, a^{(1)}\right), \delta_2\left(q^{(2)}, a^{(2)}\right)\right\rangle$$

где: $q \in Q_1 \times Q_2$ - состояние автомата *K*,

$q^{(1)} \in Q_1$ - состояния первого исходного автомата

$q^{(2)} \in Q_2$ - состояния второго исходного автомата

$a \in A_1 \times A_2$

$a^{(1)}$ - входные знаки первого канала, где $a^{(1)} \in A_1$

$a^{(2)}$ - входные знаки второго канала, где $a^{(2)} \in A_2$

Учитывая предыдущее, функция отметок примет вид:

$$\lambda(q,a)=\left\langle \lambda_1\left(q^{(1)},a^{(1)}\right),\lambda_2\left(q^{(2)},a^{(2)}\right)\right\rangle$$

где: $\lambda_1\left(q^{(1)},a^{(1)}\right)$ - функция отметок первого исходного автомата, а

$\lambda_2\left(q^{(2)},a^{(2)}\right)$ - функция отметок второго исходного автомата.

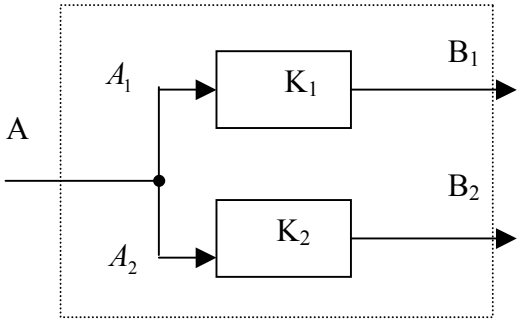
Также получаем произведение состояний и произведение знаков:

$$q=\left\langle q^{(1)},q^{(2)}\right\rangle;$$

$$a=\left\langle a^{(1)},a^{(2)}\right\rangle$$

a. параллельное соединение с общим входом

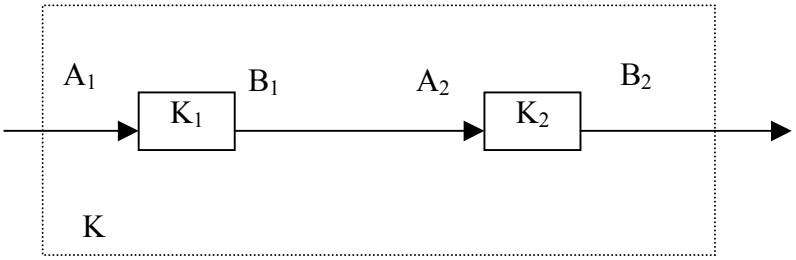
Автомат из пункта 1 можно упростить, если на входы автоматов K_1 и K_2 подаются одинаковые знаки.



$(A_1 = A_2)$

В результате получили автомат с общим входом.

b. последовательное соединение



$B_1=A_2$

Можно ослабить условие и потребовать, чтобы $B_1 \subset A_2$, в этом случае из автоматного графа надо удалить все дуги из множества $C = A_2 \setminus B_1$. Если окажутся изолированные вершины, то и они могут быть удалены из описания автомата.

а) $K_1 = K_2$ - автомат Мили, т.е. выходной знак появляется быстрее, чем меняется состояние.

$$K =< A_1, Q_1 \times Q_2, B_2, \delta, \lambda >$$

Пусть состояние автомата K - $q \in Q_1 \times Q_2$ есть $\left\langle q^{(1)},q^{(2)}\right\rangle$, где $q^{(1)} \in Q_1, q^{(2)} \in Q_2$, тогда для K получим:

$$\delta\left(q(t),a^{(1)}(t)\right)=\left\langle q^{(1)}(t+1),q^{(2)}(t+1)\right\rangle$$

где $a^{(1)} \in A_1$ - входной знак автомата $K(K_1)$,

а $q^{(1)}(t+1)$ и $q^{(2)}(t+1)$ состояние автоматов K_1 и K_2 в следующий момент времени.

$$q^{(1)}(t+1)=\delta_1\left(q^{(1)}(t),a^{(1)}(t)\right)$$

$$q^{(2)}(t+1)=\delta_2\left(q^{(2)}(t),\lambda_1\left(q^{(1)}(t),a^{(1)}(t)\right)\right)$$
$$\lambda\left(q(t),a^{(1)}(t)\right)=\lambda_2\left(q^{(2)}(t),\lambda_1\left(q^{(1)}(t),a^{(1)}(t)\right)\right)$$

а) Пусть K_1 - автомат Мура.

K_2 - автомат Мили.

$$K_2 =< A_1, Q_1 \times Q_2, B_2, \delta, \lambda >$$

A,Q,B – тоже, что и в пункте (а).

$$\lambda_1\left(q^{(1)}(t),a^{(1)}(t)\right)=b^{(1)}(t+1) \qquad *$$

Состояние:

$$q^{(1)}(t+1)=\delta_1\left(q^{(1)}(t),a^{(1)}(t)\right)$$

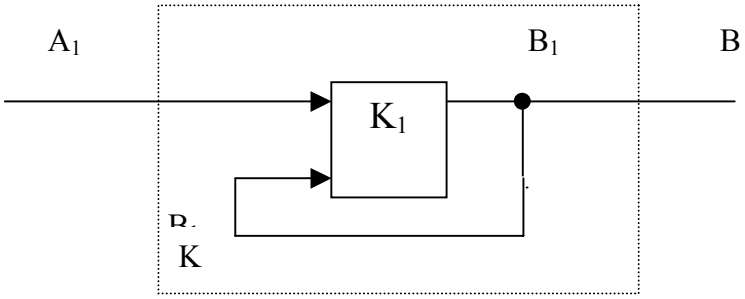
Подставим вместо $t \to t-1$

$$q^{(2)}(t+1)=\delta_2\left(q^{(2)}(t),\lambda_1\left(q^{(1)}(t-1),a^{(1)}(t-1)\right)\right)$$

Следующее состояние зависит как от предыдущего, так и от состояния предшествующего предыдущему . $\lambda_2\left(q^{(2)}(t),\lambda_1\left(q^{(1)}(t-1),a^{(1)}(t-1)\right)\right)$

c. с обратной связью:

Соединение с обратной связью предполагает, что выход автомата соединяется с его входом. Предполагается, что есть автомат имеющий следующий вид:



Входным алфавитом у этого автомата будет являться декартовым произведением входного алфавита A_1 и выходного алфавита B_1.

$$K_1 =< A_1 \times B_1, Q_1, B_1, \delta, \lambda >$$

Если обратную связь поместить внутрь автомата *K*, то входной алфавит будет являться входным алфавитом A_1, а множество состояний увеличивается до декартового произведения $B_1 \times Q_1$.

$$K =< A_1, B_1 \times Q_1, \delta, \lambda >$$

Функция отметок K_1 будет изменена и примет вид:

$$\delta(q,a)=\left\langle \delta_1\left(q^{(1)},\langle a^{(1)},b^{(1)}\rangle\right),b^{(1)}\right\rangle$$

$$q \in Q_1 \times B_1, \text{где } q \in A_1$$

Здесь возможны два случая, когда автомат K_1 - синхронный и K_0 -асинхронный автоматы.

Синхронный автомат – когда существуют состояния не устойчивые по входным знакам. Т.е. для синхронного автомата K_1 следующее состояние будет зависеть как от входного знака A_1, так и от вх.(вых.) знака B_1, с задержкой на один такт.

$$\delta\left(q(t),a^{(1)}(t)\right)=\left\langle \delta^{(1)}\left(q^{(1)}(t),\langle a^{(1)}(t),b^{(1)}(t-1)\rangle\right),\lambda\left(q^{(1)}(t),\langle a^{(1)}(t)\right),b^{(1)}(t-1)\rangle\ \right)\rangle$$

Асинхронный автомат – в случае асинхронного автомата, моменты функционирования задаются изменением выходных знаков. В этом случае изменение знака на входе *A*, при заданном состоянии выхода *B*, переводит автомат в следующее

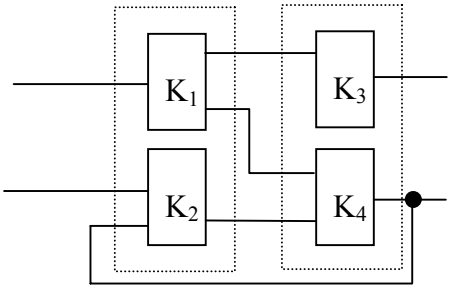
состояние, когда вычислен новый знак на выходе B. Если это новое состояние не устойчиво по паре знаков A и B, то автомат начинает изменять свои состояния, хотя не должен. При устойчивом состоянии по этой паре входных знаков, автомат остаётся в том состоянии, в котором он переменен.

Для асинхронного автомата автомат с обратной связью имеет вид:

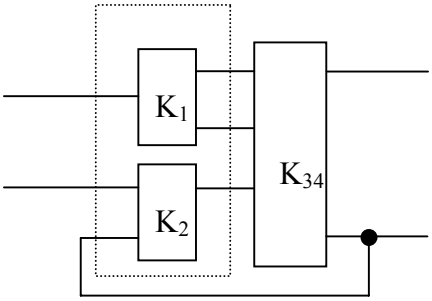
$$K =< A_1, Q_1, B_1, \delta, \lambda >$$

Для представления любой автоматной сети достаточно трех видов соединения.

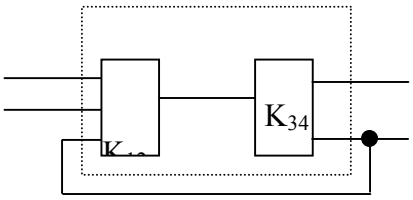
Пример : Попытаемся упростить автоматную сеть при помощи трёх видов соединений и привести её к одному конечному автомату:



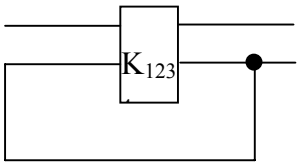
Параллельное соединение автоматов



Параллельное соединение



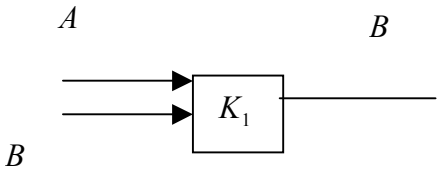
Последовательное



Соединение автомата с

Пример: Синхронный автомат с обратной связью.

Имея абстрактный автомат, построить описание синхронного автомата с обратной связью.



Пусть дан автомат $K_1 =< A \times B, Q, \delta, \lambda >$

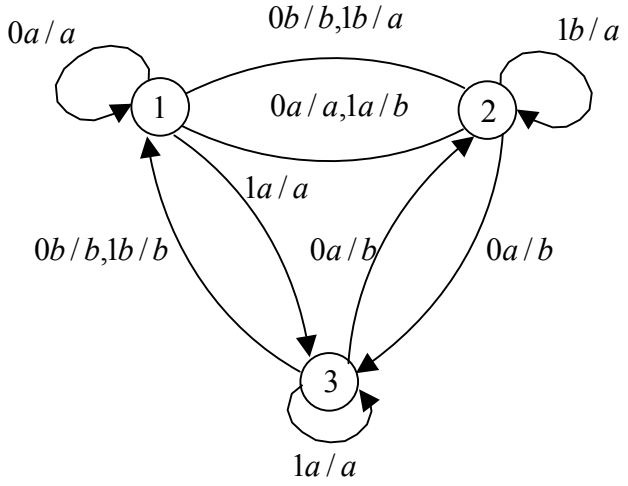
Где входной алфавит- A = { 0,1 },

а выходной алфавит B = *{* 0,1 *)*.

Построим исходную автоматную таблицу:

$A \times B$		1	2	3
A	B			
0	a	1/a	1/a	2/b
0	b	2/b	3/a	1/b
1	a	3/a	1/b	3/a
1	b	2/a	2/a	1/b

Изобразим граф исходного автомата:



Состояние2–не устойчиво по входному знаку 0b (переходит из. **Error! Objects cannot be created from editing field codes.**, и из**Error! Objects cannot be created from editing field codes.**).

Построим итоговую таблицу для автомата с обратной связью:

B / *A*	1*a*	1*b*	2*a*	2*b*	3*a*	3*b*
0	1*a*/ *a*	2*b*/*b*	1*a*/*a*	3*a*/*a*	2*b*/*b*	1*b*/*b*
1	3*a*/ *a*	2*a*/*a*	1*b*/*b*	2*a*/*a*	3*a*/*a*	1*b*/*b*

Пример: Рассмотрим асинхронный автомат:

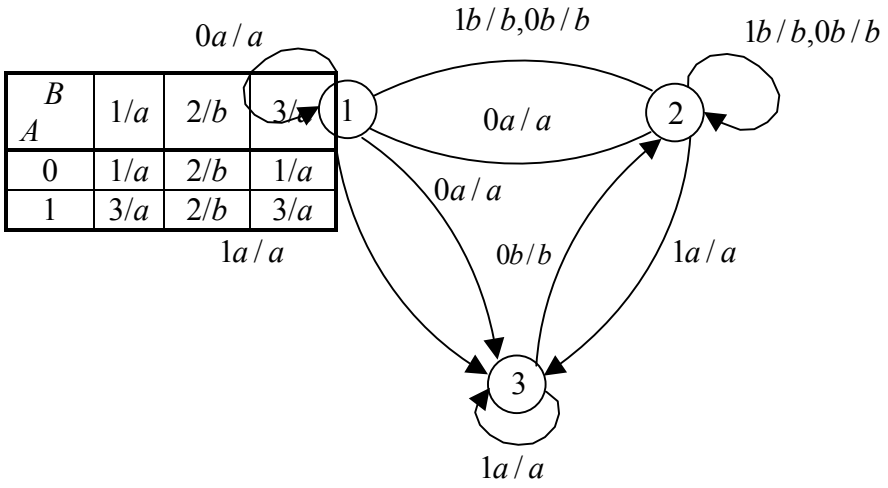
Построим исходную автоматную таблицу:

A×*B*				
A	*B*	1	2	3
0	*a*	1/*a*	1/*a*	1/*a*
0	*b*	2/*b*	2/*b*	2/*b*
1	*a*	3/*a*	3/*a*	3/*a*
1	*b*	2/*a*	2/*b*	2/*b*

Построим итоговую таблицу для автомата с обратной связью:

Изобразим граф исходного автомата:

A \ B	1/*a*	2/*b*	3/*a*
0	1/*a*	2/*b*	1/*a*
1	3/*a*	2/*b*	3/*a*



<u>Вывод</u>: в результате мы построили асинхронный автомат т.к. все состояния устойчивы по входным знакам.

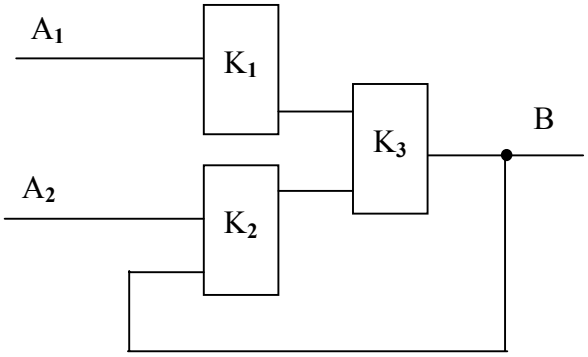
Способы введения времени в автоматную сеть

Существует три способа введения времени в автоматную сеть:

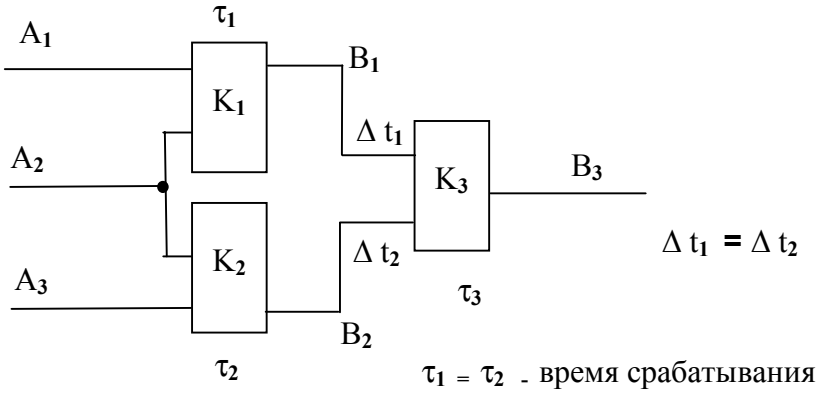
a) *<u>синхронный способ</u>* – когда вводится шкала времени, которая делится на отрезки одинаковой длины (такты), также нумеруют, начиная с 0.

- Длина такта (длительность такта) принимается за единицу времени. Входная строка рассматривается как последовательность знаков сменяемых с каждым тактом.
- Автоматные функции реализуются с задержкой в один такт. $\delta(q(t),a(t)) = q(t+1)$ в свою очередь λ зависит от вида автомата (Мура или Мили) будет иметь $\lambda(q(t),a(t)) = \begin{cases} b(t) & \text{- Мили} \\ \text{b(t+1)} & \text{- Мура} \end{cases}$.
- Синхронизация работы автомата в автоматной сети достигается за счет внешних синхронизирующих часов (источник синхронизации). Согласованная и одновременная работа автоматов в сети достигается за счёт использования одного и того же источника синхронизации.

b) *<u>Асинхронный способ.</u>* При асинхронном способе согласованная работа автоматов достигается изменением входных знаков, т.е. моменты функционирования автомата задаются изменением знака на входе. Мы ввели понятие асинхронного автомата.



c) *Апериодический способ синхронизации* – реализуется за счет подбора временных характеристик среды распространения знаков между автоматами и времени функционирования автоматов.



Каждый из этих автоматов функционирует, какое то время Δt и вносит задержку. Очевидно, для того чтобы автоматы работали синхронно, нужно подобрать временные характеристики так, чтобы добиться согласованности работы системы(т.е. в данном случае для согласованной правильной работы сети необходимо равенство времени задержки автоматов K_1 и K_2).

Тема 4.2. Комбинационный автомат

Определение: *Комбинационный автомат- это конечный автомат, у которого выходной знак зависит от входного, и не зависит от состояния автомата.*

Формализуем определение: если задан автомат $K =< A, Q, B, \delta, \lambda >$, то для всех знаков $a \in A$ и любой пары $q, q' \in Q$, должно выполнятся следующее соотношение $\lambda(q,a) = \lambda(q',a)$, т.е. выходной знак не зависит от состояния, а зависит только от входных знаков. Это свойство автомата касается функции выхода. Это ограничение приводит к следующим выводам, которые приводится в теореме 4.1.

Теорема 4. 1. О комбинационном автомате

Утверждение: любой комбинационный автомат эквивалентен автомату Мили с одним состоянием или автомату Мура с m состояниями, где $m = |A|$ - количество знаков входного алфавита.

Доказательство:

А) $K =< A, Q, B, \delta, \lambda >$ пусть задан произвольный автомат.

$m = |A|$, $n = |Q|$, (где $n, m \geq 1$) такой, что $\forall a \in A$ и $\forall q, q' \in Q$ выполняется равенство $\lambda(q,a) = \lambda(q',a)$, т.е. является комбинационным.

B) Доказательство осуществим путем минимизации произвольного автомата. Для этого рассмотрим некоторую строку $\alpha_K \in A^*$ - строка входного алфавита, длины $|\alpha|_K = k$ Покажем, что любые два состояния у комбинационного автомата неотличимы друг от друга по индукции:

Шаг 1:*Базис индукции*.

Пусть задана строка:

$$\alpha_1 = a \in A, n = 1$$

Тогда получим:

$$K(q,a) = \lambda(q,a) = b ;$$
$$K(q',a) = \lambda(q',a) = b .$$

Шаг 2::*Предположение*.

Пусть длины строки α_K равна k и известно что,

$$K(q,\alpha_K) = K(q',\alpha_K)$$

Шаг 3: *Индуктивный переход*.

Покажем, что $K(q,\alpha_{K+1}) = K(q',\alpha_{K+1})$, где $\forall \alpha_{K+1} \in A^*$, но таких, что выполняется предположение индукции:

$$\alpha_{K+1} = \alpha_K \cdot a$$

$K(q,\alpha_{K+1}) = K(q,\alpha_K \cdot a) = K(q,\alpha_K) \cdot \lambda(q'',a)$, где q'' - то состояние, в которое перейдёт автомат, будучи запущен из состояния q с входной строкой α_K ;

$$K(q',\alpha_{K+1}) = K(q',\alpha_K \cdot a) = K(q',\alpha_K) \cdot \lambda(q''',a)$$

Исходя из предположения индукции (*шаг 2*) и свойства комбинационного автомата заключаем, что $K(q,\alpha_{K+1}) = K(q',\alpha_{K+1})$.

C) Из пункта **B,** заключаем, что все состояния комбинационного автомата попарно неразличимы, что если они попадают в один и тот же класс эквивалентности при минимизации автомата, то алгоритм Мили не расщепляет состояния или же классы эквивалентности, при минимизации не изменяются.

D) Таким образом, чтобы узнать число состояний комбинационного автомата необходимо знать начальное приближение классов эквивалентности.

Т.к. автомат комбинационный, т.е. для $\forall a_j \in A$, $\forall q', q \in Q$

$$\lambda(q,a) = \lambda(q',a)$$

То для каждого состояния столбец выходных знаков будет одинаков.

1.1.1.1	q_1	...	q_i	...	q_n
a_1	$/b_j$	...	$/b_2$	...	$/b_1$
⋮	⋮	⋮	⋮	⋮	⋮
a_j	$/b_j$	...	$/b_j$	...	$/b_j$
...	...	...	...	...	...

Последнее означает, что при использовании алгоритмов Мили, полученное начальное приближение классов эквивалентности состоящее из одного множества, куда включены все состояния автоматов.

Вывод: Комбинационный автомат может быть минимизирован; в результате чего получаем неотличимый от него автомат с одним состоянием.

Представим автомат $K' =< A, Q', B, \delta, \lambda >$ в виде таблицы минимизированного комбинационного автомата:

$A\backslash Q$	q
a_1	q/b_1
⋮	⋮
a_j	q/b_j
⋮	⋮
a_m	q/b_m

Множество состояний сократилось до одного состояния q:

$$K' =< A, \{q\}B, \lambda >$$

Тогда значение q можно опустить:

$$K' =< A, B, \lambda >$$

В результате получаем таблицу истинности дискретной функции.

$A\backslash Q$	q
a_1	b_1
a_2	b_2
a_j	b_j
⋮	⋮
a_m	b_m

E) Покажем сколько будет иметь состояний неотличимый от автомата K, автомата Мура.

$$K'_m =< A, Q'_m, B, \delta, \mu >$$

Из теоремы (О существовании автомата Мура) получим:

$$|Q_m| = n(m+1) = m+1$$

, где n – число состояний, а m – число входных знаков:

$$m = |A|$$

Доопределим состояние выхода автомата для начального состояния q_{10} таким образом, чтобы совпало с выходным знаком в некотором состоянии q_{1B} в результате минимизации автомата Мура, будем иметь не более чем m состояний $|Q'_m| \le m$.

Вывод: В общем случае комбинационный автомат с задержкой на один такт эквивалентен автомату Мура не более, чем на m состояний.

F) Определим более точное число состояний автомата Мура.

Таблица автомата Мура имеет следующий вид:

A \ Q_m	q_0 /–	q_1 / b_1	q_2 / b_2	…..	q_m / b_m
a_1	q_1	q_1	q_1	**…..**	q_1
a_2	q_2	q_2	q_2	**…..**	q_2
⋮	⋮	⋮	⋮	⋮	⋮
a_m	q_m	q_m	q_m	…..	q_m

Несложно заметить из приведённой таблицы, что число состояний автомата Мура будет равно числу выходных знаков т .е. $|Q_m| = |B|$ **.**

Поясним теорему при помощи примера:

Пример:

Задан исходный комбинационный автомат при помощи таблицы:

A \ *Q*	1	2	3
a	2/0	1/0	2/0
b	3/1	3/1	2/1
c	1/0	2/0	1/0

Так как знак на выходе автомата не зависит от его состояния и определяется только входным знаком, то в результате минимизации автомата получим:

A	*Q/B*
a	1/0
b	1/1
c	1/0

Если удалить состояние из таблицы получим:

A	*B*
a	0
b	1
c	0

Построим неотличимый от исходного минимальный автомат Мура:

A \ Q_m	1/0	2/1
a	1	1
b	2	2
c	1	1

Откуда видно, что ожидаемое количество состояний автомата не 3 – количество входных знаков, а 2 – количество выходных знаков исходного комбинационного автомата.

Тема 4.3. Теорема о структурном синтезе

Пусть задан некоторый конечный автомат K :

$$K =< A, Q, B, \delta, \lambda >$$

Вопрос : Как представить автомат в виде автоматной сети, возможно состоящий из более простых автоматов, чем исходных ?

Здесь Теория автоматов разделяется на две части:

1. *алгебраическая;*
2. *комбинаторная.*

1. *Алгебраическая теория* решения задач структурного синтеза в самой общей постановке , когда в качестве элементарного автомата используются автоматы произвольного вида (общего вида).

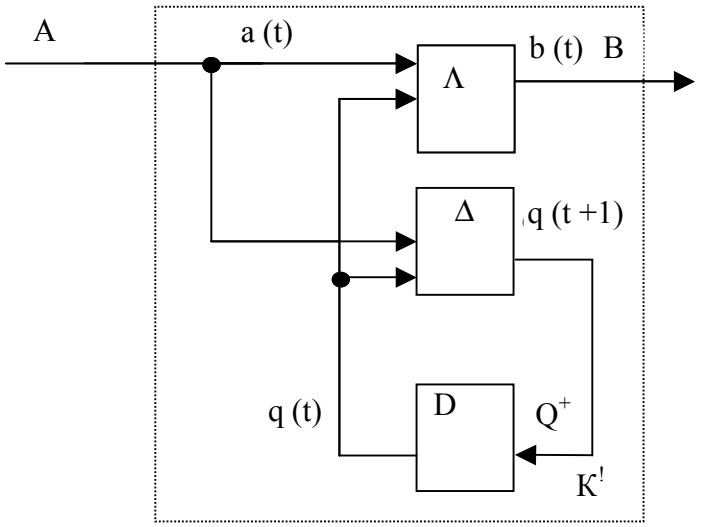
Доказано, что имея конечный автоматный базис т.е. конечное число элементарных автоматов, невозможно построить любой произвольный наперёд неизвестного вида автомат.

Вывод: При алгебраическом подходе нет функциональных автоматных базисов.

2. *В комбинаторной теории* в качестве автоматного базиса используется конечное число комбинационных автоматов. Позже будет доказано, что производный автомат может быть представлен в виде схемы из комбинационных автоматов и конечного автоматного базиса.

Теорема 4.2. О структурном синтезе

Утверждение: произвольный конечный автомат $K =< A ,Q ,B ,\delta ,\lambda >$ может быть реализован автоматной сетью, состоящей из двух комбинационных автоматов Δ , Λ и элемента задержки D на один такт с $n =\mid Q \mid$ состояниями. D - это такой автомат, который повторяет входные знаки с задержкой в один такт, а n - число состояний начального автомата.



Доказательство:

A) Определим автоматы Δ , V , D:

Автомат - $\Lambda =< A \times Q ,B ,\lambda >$ реализует функцию $\lambda(q,a)$

Автомат - $\Delta =< A \times Q ,Q,\lambda >$ реализует функцию $\delta(q,a)$

Линия задержки на один такт D, повторяет входные знаки алфавита Q на выходе Q с задержкой на один такт.

$D =< Q,Q ,Q,\delta_0,\mu >$ - автомат Мура. , где δ_0 - функция переходов повторителя входных знаков.

Очевидно комбинационный автомат срабатывает мгновенно и упорядоченность работы автомата K во времени определяется элементом задержки на один такт.

B) Покажем неотличимость исходного автомата от автоматной сети индукцией по длине входной строки.

шаг 1:Базис индукции.

Пусть задана строка:

$$\alpha_1 \in A^* , \mid \alpha_1 \mid =1 , \ \alpha_1 = a , \text{где } a \in A$$

Тогда для автоматов K и K', запущенных из состояния q и при подаче входного знака a, получим:

$$\begin{array}{l} K\ (q\ ,\alpha) = \lambda(q\ ,\alpha) \\ K'(q\ ,\alpha) = \lambda(q\ ,\alpha) \end{array} \Bigg| \quad \textbf{Error!} \quad \textbf{Objects}$$

т.е. следующее состояние у автоматов одинаково.

шаг 2*:Предположение*.

Предположим, что $\left|\ \alpha_K\ \right| = k$

Пусть $K(q,\alpha_k) = K'(q,\alpha_k)$

Тогда очевидно, что и следующее состояние у автоматов одинаково.

Допустим $q(t+k)$ у обоих автоматов будет одинаково.

шаг 3: *Индуктивный переход*.

Пусть длина строки $\alpha_{K+1} = \alpha_K a$

$$K(q,\alpha_k a) = K(q,\alpha_k)\cdot\lambda(q(t+k),a)$$
$$K'(q,\alpha_k a) = K(q,\alpha_k)\cdot\lambda(q(t+k),a)$$

Таким образом мы показали, что выходные строки одинаковы.
Следующее состояние после приёма знака у автоматной сети будет соответствовать (совпадать) со следующим состоянием исходного автомата.

C) *Вывод: Т.к. для любого состояния исходного автомата мы нашли неотличимые от него состояния автоматной сети, то исходный автомат неотличим от автомата сети, если автоматная сеть и исходная сеть запускается из соответствующих состояний , что и требовалось доказать.*

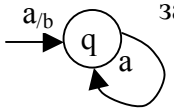
Замечание: При доказательстве теоремы использовался общий вид автомата (автомат Мили). Последнее означает, что теорема справедлива для автоматов частного вида:

синхронного и автомата Мура.

Если исходный автомат асинхронный, то элемент памяти (элемент задержки D) тоже должен быть асинхронным.

Действительно, если исходный автомат K – асинхронный, то как и для автоматной сети - изменение знака на входе приводит к изменению состояния на входе и выходе автомата Δ. Происходит изменение входного знака происходит на элементе задержки, который через время задержки τ повторяет входной знак на выходе, что является новым состоянием автомата. Это состояние может в свою очередь привести к изменению как выходного знака, так и следующего состояния, но т.к. автомат асинхронный следующее состояние совпадает с ранее вычисленным.

В этом случае элемент задержки на один такт может быть не использован в схеме.



Для реализации произвольного автомата в виде схемы из комбинационных автоматов, необходим в общем случае элемент задержки на один такт, который по своему определению является автоматом с несколькими состояниями. Из предыдущего замечания следует, что такие автоматы могут быть реализованы, как асинхронные автоматы, не содержащие элемента задержки.

Вывод: для реализации произвольного автомата необходимо уметь реализовать, только комбинационный автомат т.к. элемент памяти может быть представлен в виде асинхронного автомата, не содержащего элемента задержки

Тема 4.4. Структурный синтез комбинационных автоматов

Как было показано ранее, комбинационный автомат может быть представлен в виде автомата с одним состоянием и реализует некоторую дискретную функцию. В связи с тем, что для произвольного автомата, его комбинационный автомат может быть довольно сложным и в общем случае отличаться от комбинационных автоматов, полученных при декомпозиции других автоматов, то ставится задача выделить некоторое множество комбинационных автоматов, составляющий автоматный базис и с их помощью реализовать комбинационный автомат любой сложности.

Теорема 4.3. О структурном синтезе комбинационных автоматов.

Утверждение: Для того чтобы произвольная дискретная функция могла быть реализована схемой из элементарных комбинационных автоматов, образующих автоматный базис Ω, необходимо и достаточно, чтобы множество функций реализуемых автоматным базисом было функционально полным.

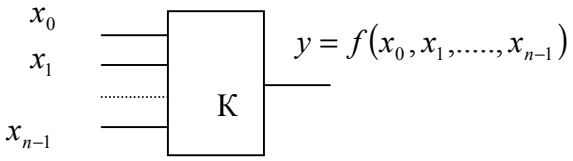
Замечание: доказано, что не существует конечного автоматного базиса, позволяющего представить произвольный КА в виде схемы, состоящей из конечных автоматов с памятью.

Доказательство:

A) Если $K =< A, B, \lambda >$ и Ω - функционально полная система дискретных функций, то производная дискретная функция $f(x_0, x_1, \ldots, x_{n-1})$ может быть представлен в виде формулы Ф над множеством функций $\Omega = (\varphi_1, \varphi_2, \varphi_3, ..., \varphi_t, ..)$. (Доказательство из дискретной математики.)

B) Пусть получена такая формула $\Phi(x_0, x_1, ..., x_{n-1}) = f(x_0, x_1, \ldots, x_{n-1})$, что для всех комбинационных автоматов независимые переменные значения функции на этих наборах переменных совпадают с результатами вычисления по формуле.

Очевидно, что переменные входящие в это выражение принадлежат различным множествам.



$x_j \in A_j$;

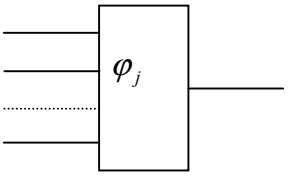
$y \in B$, при $A = A_0 \times A_1 \times ..... \times A_{n-1}$

Чтобы отвлечься от конкретного вида множеств A_j и B установим изоморфизм между подмножеством натуральных чисел и множествами A_j, B.

A_j	a_1	a_2	…..	a_{Kj}
K_{Kj}	0	1	…..	k_{j-1}

$x_j \in N_{Kj} = \{0,1,....., k_{j-1}\}$, где $k_j \in |A_j|$ и говорить, что переменная имеет значность k_j.

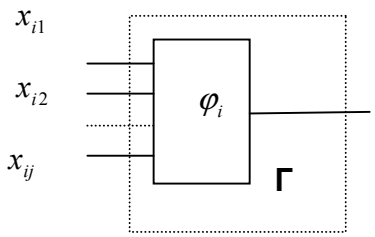
C) Предположим, что функции входящие в функционально полный базис $\Omega = (\varphi_1, \varphi_2, \varphi_3, ..., \varphi_t, ..)$ реализован в виде элементарных комбинационных автоматов,



заданных на алфавите K_{Kj} , K_{Kf} , где K_f - значность функции f .

D) Построим Γ с использованием автоматного базиса Ω . Построение будем осуществлять индукцией по глубине формулы. В формуле Φ поставим в соответствие схему Γ из элементарных комбинационных автоматов, реализующих функции базиса Ω следующим образом:

a) Если окажется, что $\Phi = \varphi_i(x_{i1}, x_{i2} \ldots, x_{iK})$, то схема Γ состоит из одного элемента φ_i , входы которого отождествлены с входными параметрами, а выходом которого является выход нашей схемы, таким образом где x_{if} - выходные переменные, а φ_i - k-местная функция автоматного базиса $\varphi_i \in \Omega$.



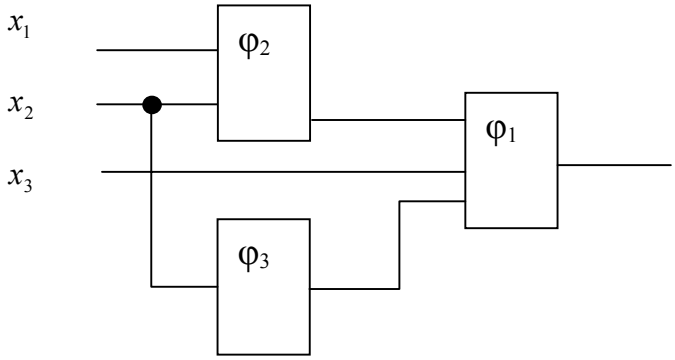
b) Если формула Φ представлена как $\Phi = \varphi_i(\Phi_{i1},...,\Phi_{iK})$, , где Φ_{ij} или переменные x_{ij} из множества входных переменных , $x_{ij} \in X$ или функции уже реализованные схемами Γ_{ij}. То схема Γ для функции Φ строится так:
К $j-$тому входу элемента φ_i подсоединяется выход схемы Γ_{ij} , если Φ_{ij} - формула или же переменная x_{ij} если $\Phi_{ij} = x_{ij}$. Выходом схемы Γ_i объявляется выход элемента φ_i. Входами схемы являются входы схемы Γ_j, где $j = \overline{1,t}$

<u>Вывод</u>: Схема полученная описанным способом имеет вид ориентированного дерева. Ее входы соответствуют концевым вершинам (переменным функции), а выход - корню дерева (значению функций). Для того чтобы произвольная функция могла быть реализована схемой над автоматным базисом Ω , необходимо и достаточно, чтобы множество функций из Ω образовывало функционально полную систему.

<u>Замечание:</u> для доказательства достаточности необходимо по произвольной схеме над автоматным базисом построить формулу функции, которую эта схема реализовывает (осуществляется в обратном порядке).

Пример:

Дана произвольная схема над автоматным базисом $\Omega = \{\varphi_1, \varphi_2, \varphi_3\}$



Построим формулу функции, которую эта функция реализовывает:

$\Phi = \varphi_1(\varphi_2(x_1, x_2), x_3, \varphi_3(x_2))$

Сведения из многозадачной логики

Определение: *Многозадачная логика* $L_K = \langle N_K, V, \&, \overline{\ } \rangle$ *- есть алгебра, заданная на множестве* N_K *с операциями &, V, — , где* $N_K = \{0,1,2,.....k-1\}$

Для любых ∀ x, y ∈ N_K вводятся три операции:

Дизъюнкция - x ∨ y = max (x, y) ;

Конъюнкция - x & y = min (x, y) ;

$\bar{x}$ = (x + 1) mod k (по Лукасевичу)

Постом было доказано, что &, V, — , является функционально полная система.

≡}к

Для *k* – значной логики, закон двойного отрицания будет выглядеть:

$\bar{\bar{x}} = x$

$\bar{=} x = x$ (в булевой)

Для представления дискретной функции с различными значностями переменных поступают так:

Выбирают алгебру логики с *K* равным:

$K = \max(k_0, k_1, ...k_{n-1}, k_f)$, где

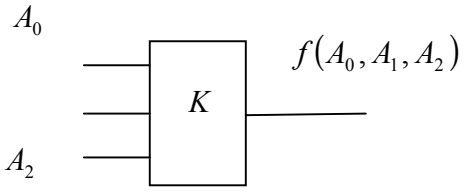
$k_j \left(j = \overline{0, n-1}\right)$ - значности переменных функции.

k_f - значность самой функции.

Тогда в соответствии с результатами Поста, произвольная дискретная функция с указанными значностями представима в виде формулы конечной длины в алгебре логики.

Пример: Синтез комбинационного автомата.

Пусть задан комбинационный автомат со входными алфавитами A_0, A_1, A_2, представим его виде логических элементов:



$k_0 = |A_0|, k_1 = |A_1|, k_2 = |A_2|$

$k_0 = 2$, $k_1 = 3$, $k_2 = 2$

$k_f = 3$

где, k_f - число знаков выходного алфавита,

а k_j - число знаков в алфавите A_j.

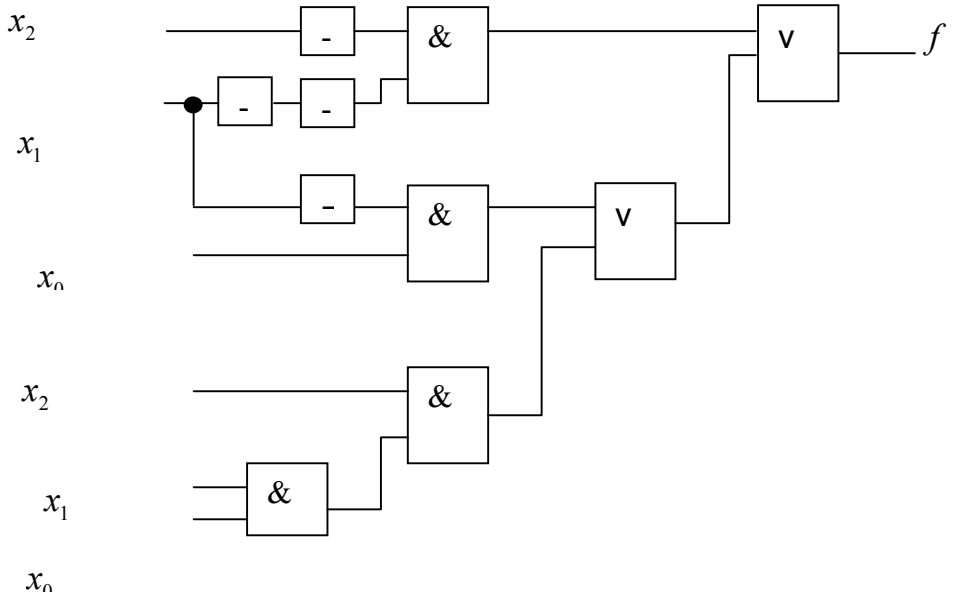
Переходим от алфавитов A_j к алфавитам или многочленам $N_{KJ} = \{0,1,2,.....k_J - 1\}$ с помощью изоморфизма полученного автомата *K*.

X_2	X_1	X_0	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
0	2	0	1
0	2	1	1
1	0	0	2
1	0	1	2
1	1	0	0
1	1	1	1
1	2	0	1
1	2	1	1

Из автоматной таблицы после применения алфавита получаем дискретную функцию, в *k*-значной логике, где *k* равно:

$$K = \max(2,3,2,3) = 3$$

Формула в автоматном базисе $\Omega = \langle V,\&,- \rangle$ будет иметь вид:



$$\Phi = \bar{x}_2 \overline{\overline{x_1}} \vee \bar{x}_1 x_0 \vee x_2 x_1 x_0$$

При помощи логических элементов реализуем комбинационный автомат:

Кодирование алфавитов

Т.к. самым распространённым автоматным базисом является базис Ω :

x_2	x_1		x_0	f	
	x_{11}	x_{10}		f_1	f_0
0	0	1	1	0	1
0	1	0	0	0	1
0	1	0	1	0	1
1	0	0	0	1	0
1	0	0	1	1	0
1	0	1	0	0	0

Ω = { , , }, в булевой алгебре над двоичным алфавитом, то необходимо так представить исходную дискретную функцию, чтобы она могла быть реализована над этим алфавитным базисом. Для этого используют кодирование алфавита в виде элементов декартового произведения нескольких булевых множеств.

Общий вид:

$$A_j\left(j=\overline{0,n-1}\right)$$
$$\left|A_j\right|=K_j$$

Тогда каждый знак любого из алфавитов A_j кодируется в виде последовательности знаков двоичного алфавита длиной не менее чем $\rceil\log_2 K_j\lceil$, где] x [- это наименьшее целое, превосходящее или равное *x***.**

Установим изоморфизм:

$$A_j \leftrightarrow \underbrace{N_2 \times N_2 \ldots\ldots \times N_2}_{\rceil\log_2 K_j\lceil}$$

В результате получаем $\rceil\log_2 K_f\lceil$ булевых функций, зависящих от $\sum_{j=0}^{n-1}\rceil\log_2 K_j\lceil$.

Пример: Используем дискретную функцию из предыдущего примера .

Закодируем знак x_1 при помощи двух знаков x_{11} и x_{10} ,чтобы прийти к двоичному алфавиту. $\rceil\log_2 3\lceil = 2$ - для кодирования 3х-значного алфавита хватает двух разрядов.

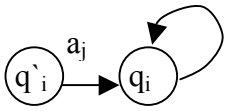
Представим функцию в СДНФ.

$$f_1 = x_2\bar{x}_{11}\bar{x}_{10}\bar{x}_0 \vee x_2\bar{x}_{11}\bar{x}_{10}x_0$$
$$f_0 = \bar{x}_2\bar{x}_{11}x_{10}x_0 \vee \bar{x}_2x_{11}\bar{x}_{10}\bar{x}_0 \vee \bar{x}_2x_{11}\bar{x}_{10}x_0$$
$$x_1 = (x_{11}, x_{10})$$
$$f = (f_1, f_0)$$

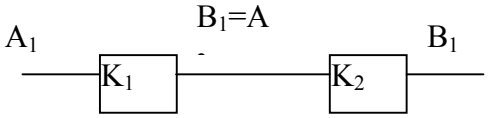
Тема 4.5. Синтез асинхронных автоматов

Определение: *Асинхронным автоматом называется конечный автомат все составляющие которого, устойчивы по всем входным знакам*. Это означает, что для всех состояний $q_i \in Q$ и $\forall a_j \in A$ $\delta(q_i a_j) = \delta(q_i a_j a_j)$. На графе, если есть дуга, ведущая в какую ни будь вершину автомата, то существует дуга, которая идет из этой же вершины, помеченной a_j и не существует дуги, ведущей из этой вершины со знаком a_j, причем,

не существование этой дуги говорит о том, что автомат может быть недетерминированным.

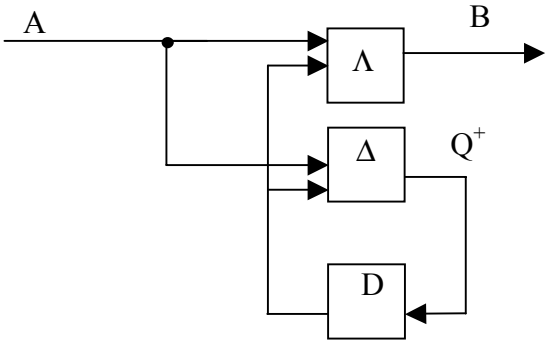


Только применение знака на входе может привести к смене состояния автомата. Это значит, что синхронизация работы автомата в автоматной сети, состоящей из асинхронных автоматов, выполняется автоматически, без специальных мер.



(Изменение входного знака приводит к изменению состояния, которое может привести к изменению выходного знака, т.е. автоматы взаимно согласуются. Такты работы автоматов задаются сменной входного знака.)

Проинтерпретируем теорему 4.2. для этого автомата:



- реализует функцию $\lambda = \lambda\ (q,a)$

- реализует функцию $\delta = \delta\ (q,a)$

$q(t+1) = q^{+}(t)$

Если исходный автомат $K = < A, Q, B, \delta, \lambda >$ асинхронный, то входной знак, поданный на его вход, приводит к функционированию автомата не в дискретные моменты времени, а в моменты смены (изменения) этого знака. Т.е. такты синхронизации или моменты функционирования автомата определяются по изменению состояния входа (асинхронный автомат)

Если асинхронный автомат представлен в виде схем по теореме о структурном синтезе, то мы имеем:

1) Изменение входного знака приводит к изменению входного знака на входе δ, а это приводит к тому, что изменяется выход у автомата δ;
2) Изменение выходного знака δ приводит к изменению входного знака у линии задержки. Через какое-то время это изменение появится на выходе *D* и приведёт к изменению знака на втором входе δ;
3) Это изменение может привести в общем случае к повторному изменению состояния выхода автомата δ_2 . Но так как исходный автомат асинхронный, то имеем:

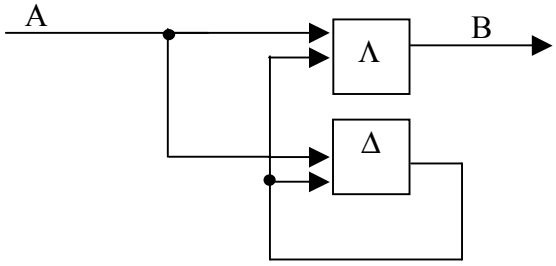
$$\delta(a, a) = q\ ;\ \delta(q`, a) = q\ ,\ \forall q`, \text{где } \forall q \in Q\ \forall a \in A$$

Последнее означает, что автомат δ, после появления задержанного знака на втором входе не может, не может изменить своё состояние выхода, если исходный автомат асинхронный.

Это означает, что при синтезе асинхронных автоматов, элемент задержки *D* можно использовать.

Теорема 4.4. О синтезе асинхронных автоматов

Определение: Произвольный асинхронный автомат может быть представлен (реализован) автоматной сетью, состоящей из двух комбинационных автоматов Δ и Λ (без линии задержки).



$\Delta : \lambda(q , a)$
$\Lambda : \delta(q , a)$

Замечание: Как мы видели ранее, мы легко можем реализовать комбинационный автомат. Реализация же элемента задержки на один такт, без дополнительных элементарных элементов задержки в автоматном базисе невозможно. Однако мы, представим исходный автомат с несколькими состояниями как комбинационный, и имеем возможность реализовать в том числе и элементы задержки, которые имеют более одного состояния. При этом мы обходимся без расширенного автоматного базиса - элементарными элементами задержки.

Теорема 4.5. Об элементе задержки на один такт

Определение: Элемент задержки на один такт, используемый в теореме о структурном синтезе автомата КА, - является асинхронным автоматом.

Доказательство:

А) В теореме 4.2. мы показали, что для автомата $K =< A , Q , B , \delta , \lambda >$, элемент задержки может быть представлен следующей автоматной таблицей:

Q \ Q	q_1/q_1	q_2/q_2	…	q_i/q_i	…	q_n/q_n
q_1	q_1	q_1	…	q_i	…	q_n
q_2	q_2	q_2	…	q_i	…	q_n
…	…	…	…	…	…	…
q_j	q_j	q_j	…	q_i	…	q_n
…	…	…	…	…	…	…
q_n	q_n	q_n	…	q_n	…	q_n

В) Из таблицы видно что, все состояния автомата устойчивы по всем входным знакам, ведущим в это состояние.

Рассмотрим некоторые состояния q_i, в это состояние ведут дуги от всех других состояний q_j, очевидно помеченные знаком q_i. Перейдя в состояние q_i , при получении того же входного знака, автомат переходит в это же состояние, т.е. каждое состояние элемента памяти устойчиво по всем входным знакам, и представляет собой асинхронный автомат, функционирующий в дискретные моменты времени. Т.е. знак на входе появляется на выходе в следующие моменты времени функционирования автомата.

Теорема доказана.

Пример: Синтез асинхронного автомата.

*Замечание к теореме***.** В соответствии с последними двумя теоремами получили вывод:

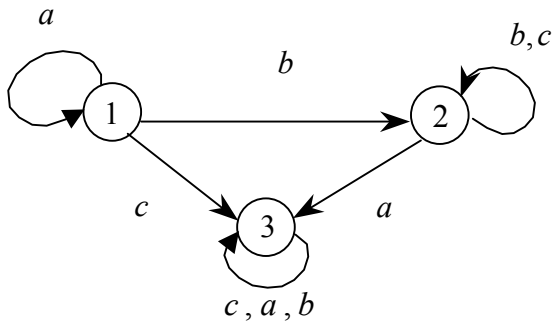
Для синтеза произвольного автомата необходимо уметь реализовать комбинационный и асинхронный автоматы. Ни тот, ни другой для своей реализации не требует элемента задержки на один такт.

Дан асинхронный автомат .

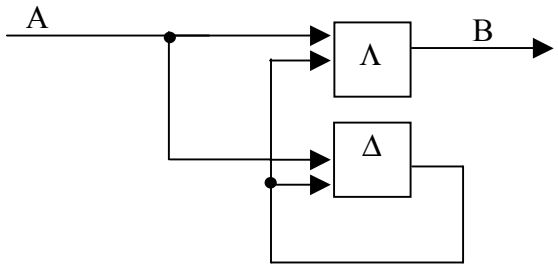
Представим его в виде автоматной сети:

A \ *Q*	1	2	3
a	1/0	3/1	3/0
b	2/1	2/1	3/0
C	3/0	2/0	3/1

Представим в виде графа:



На этом рисунке представлен вид автоматной сети.



Задаём Δ и ∧ в виде комбинационного автомата:

A	*Q*	*B*	Q^+
a	1	0	1
a	2	1	3
a	3	0	3
b	1	1	2
b	2	1	2
b	3	0	3
c	1	0	3
c	2	0	2
c	3	1	3

Выбираем автоматный базис, реализующий функционально полную систему операций в булевой алгебре: Ω = *{ &, V, – }* N_2 *= { 0, 1 }*

Задаём код

A	a	b	c
N_2^2	00	01	10

Q, Q^+	1	2	3
N_2^2	00	01	10

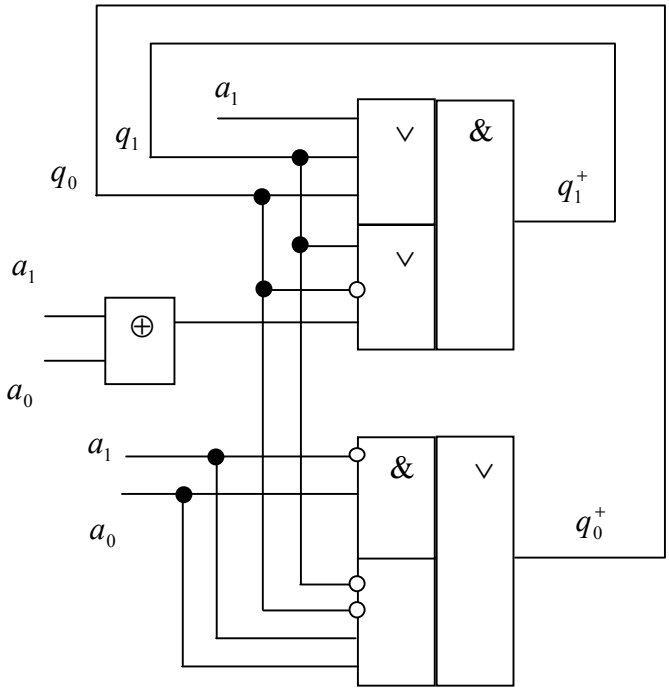
Задаём таблицу истинности для булевых функций, описывающих работу автомата с учётом кодирования алфавитов.

A		Q		B	Q^+	
a_1	a_0	q_1	q_0	b	q_1^+	q_0^+
0	0	0	0	0	0	0
0	0	0	1	1	1	0
0	0	1	0	0	1	0
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	0	1	0
1	0	0	0	0	1	0
1	0	0	1	0	0	1
1	0	1	0	1	1	0

Представляем функции из таблицы в виде выражений в функционально полном базисе Ω.

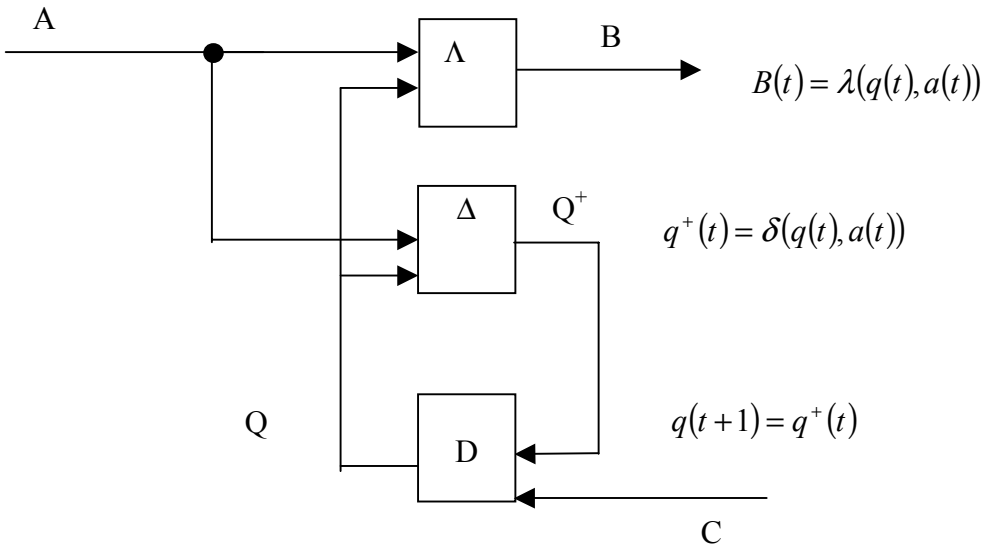
$$q_1^+ = (a_1 \vee a_0 \vee q_1 \vee q_0)(a_1 \vee \overline{a}_0 \vee q_1 \vee q_0)(a_1 \vee \overline{a}_0 \vee q_1 \vee \overline{q}_0)(\overline{a}_1 \vee a_0 \vee q_1 \vee \overline{q}_0)$$
$$= (a_1 \vee q_1 \vee q_0)(q_1 \vee \overline{q}_0)(a_1 \vee \overline{a}_0)(\overline{a}_1 \vee a_0) = (q_1 \vee \overline{q}_0)(a_1 \oplus a_0)(a_1 \vee q_1 \vee q_0)$$
$$= (a_1 \vee q_1 \vee q_0)(a_1 \vee q_0) \vee (a_1 \oplus a_0)$$
$$q_0^+ = \overline{a}_1 a_0 \overline{q}_1 \overline{q}_0 \vee \overline{a}_1 a_0 \overline{q}_1 q_0 \vee a_1 \overline{a}_0 \overline{q}_1 q_0 = \overline{q}_1 q_0 \overline{q}_1 \vee a_1 \overline{a}_0 \overline{q}_1 q_0$$

Рисуем схему искомого автомата, используя логические элементы выбранного автоматного базиса.



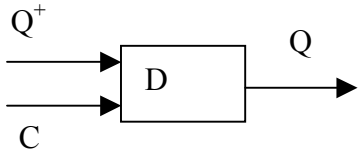
ТЕМА 4.6. Синтез синхронных автоматов

Представим схему, полученную в результате структурного синтеза синхронного автомата.



C – вход синхронизации(автомат работает, когда изменяется синхронизирующий знак).

1) Необходимо обеспечить следующее функционирование синхронного элемента задержки *D***.** Знак на входе Q^+ появляется на выходе элемента только в дискретные моменты времени t = 0, 1, 2,….
2) После появления знака на выходе Q , автомат Δ может вычислить новый входной знак Q^+, но он уже не должен воздействовать на выход элемента задержки. (мы считаем, что автомат работает мгновенно).
3) Необходимо каким-то образом задавать моменты функционирования автомата, обеспечив согласованную и одновременную работу автомата в этой автоматной сети. Это можно сделать только с использованием дополнительного синхронизированного входа C . Только изменение знака q на входе, должно приводить к функционированию как элемента задержки, так и автомата в целом.
4) Из описания 1,2,3 пунктов функционального синхронного элемента задержки следует, что он должен быть асинхронным автоматом по входу C . Ранее мы показали, что элемент задержки произвольного автомата - есть асинхронный автомат. Отсюда следует, что поиск реализации синхронного элемента задержки следует искать в классе асинхронных автоматов.
5) Докажем отличие синхронного элемента задержки от синхронного автомата. В итоге получим следующий автомат:



Если автомат D – асинхронный, то любое изменение знака на входе или Q^+ или C , должен привести к функционированию автомата D, но по описанию синхронного элемента задержки следует, что автомат D должен функционировать только при изменении знака на входе синхронизации C . Изменение знака на входе Q^+ , не должно приводить к функционированию автомата.

6) Будем в дальнейшем рассматривать элементы задержки, входные и выходные знаки которых принадлежат двоичному алфавиту. Это не уменьшает общности рассмотрения, т.к. использование кодирование знаков не двоичного алфавита

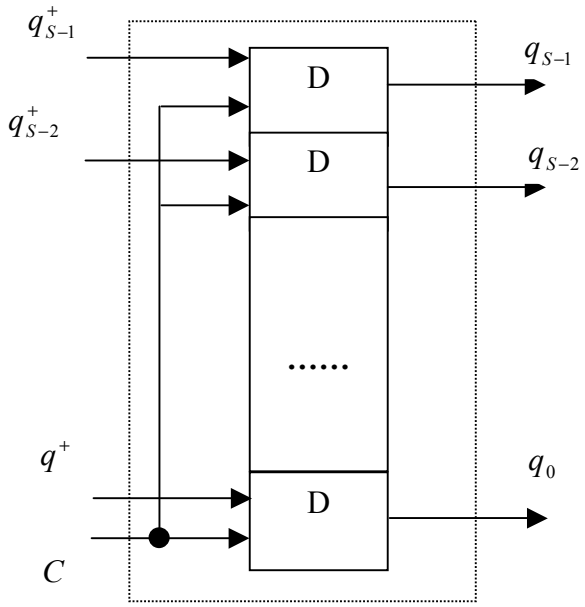
знаками двоичного алфавита, мы можем реализовать элементы задержки, когда входные и выходные знаки принадлежат алфавиту с произвольным числом знаков. К тому же техническую реализацию находят только автоматы работающие с двоичным алфавитом .(у нас нет других распространённых автоматных базисов).

Если $Q^+ = Q = \{q_1, q_2, q_3, ..., q_n\}$, то используется параллельное соединение.

С использованием S – элементов задержки $S =]\log_2 |Q|[$ мы можем реализовать элемент задержки для заданного алфавита Q.

Заметим, что для синхронизированного входа достаточно только двух знаков.

Синхронизированный элемент задержки из n – элементов.



Выходной алфавит для элемента задержки :

$$Q = \{q_{S-1}, q_{S-2}, q_{S-3}, ..., q_0\}$$

Выходной алфавит для элемента задержки :

$$Q^+ = \{q^+_{S-1}, q^+_{S-2}, q^+_{S-3}, ..., q^+_0\}$$

т.к. всё делается для 2х-значной логики, то:

$$q_t, q^+_t \{0,1\}$$

где:

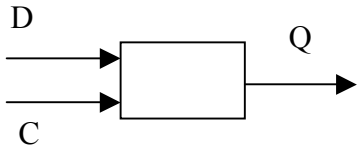
$$t = \overline{0, s-1}$$

Тема 4.7. Примеры синтеза

DL – триггер.

Рассмотрим пример синтеза асинхронного автомата, который называется DL – триггером.

DL – триггером называется такой автомат, который имеет 2-входа и один выход.



Если на вход C подаётся 0, то на выходе Q повторится значение которое будет на входе D, если на входе C будет подана 1, то триггер запомнит значение которое было подано на вход D.

$$D = \left\langle C \times Q^{+}, Q', Q, \delta, \mu \right\rangle$$

$C=\{0,1\}$, $Q^{+}=\{0,1\}$, $Q=\{0,1\}$

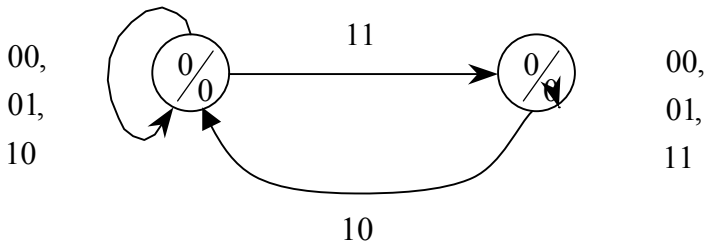
Автомат функционирует при C=1, а при C=0 – нет.

В соответствии с приведённым описанием работы триггера определим его формальное описание в виде автомата Мура.

Этот автомат асинхронный $Q'=\{0,1\}$

$C \times Q^{+}$		0/0	1/1
C	Q^{+}		
0	0	0	1
1	1	0	1
1	0	0	0
	1	1	1

то асинхронный автомат устойчив по состояниям.

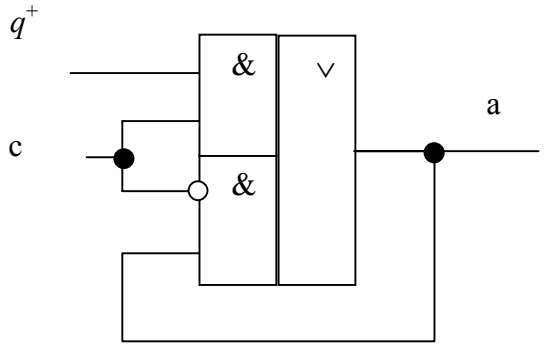


C	q^{+}	q'	q'	q
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	1	1
1	0	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	1	1

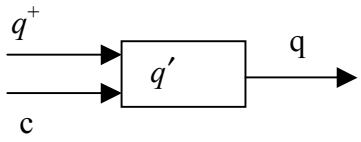
$q' = q$ (частный случай)

$$q' = \overline{c}\,\overline{q}^{+}q' \vee \overline{c}q^{+}q' \vee cq^{+}\overline{q}' \vee cq^{+}q' = \overline{c}q' \vee cq^{+}$$

Получим триггер-защёлку:



Этот триггер не подходит под описание синхронного элемента памяти.



Если с низкого уровня то сигнал не изменен, если с высокого уровня то изменяется на то, что на входе и при положении *C* оно намертво запоминается в памяти.

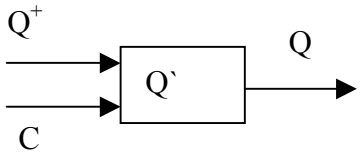
Т.о. этот триггер не может быть использован как синхронный элемент памяти т.к. при активном (потенциальном) значении сигнала C, он повторяет на выходе все изменения на входе D, это может привести к появлению ошибочных процессов т.к. у конечных автоматов этот триггер используется как синхронный элемент памяти.

Двухступенчатый – триггер

Рассмотрим пример синтеза триггера, который может быть использован в качестве синхронного элемента памяти.

Рассмотрим 4-е состояния элемента задержки:

$$Q' = \{0, 0^+, 1, 1^+\}$$



0 – синхронизующий триггер запоминает 0, выдаёт на выходе 0.

0^+ - запоминает 0, выдаёт на выходе 1.

1 - запоминает 1, выдаёт на выходе 0.

1^+ - запоминает 1, выдаёт на выходе 1.

q_1	q_0
0	0
0	1
1	0
1	1

Полученное ранее словесное описание синхронного элемента задержки формализуем в виде автоматной таблицы:

$C \times Q^+$		0/0	$0^+/1$	1/0	$1^+/1$
C	Q^+				
0	0	0	0^+	0	0^+
1	1	1	1^+	1	1^+
1	0	0	0	1^+	1^+
	1	0	0	1^+	1^+

Если **c** = 0 – запомнит входной знак.

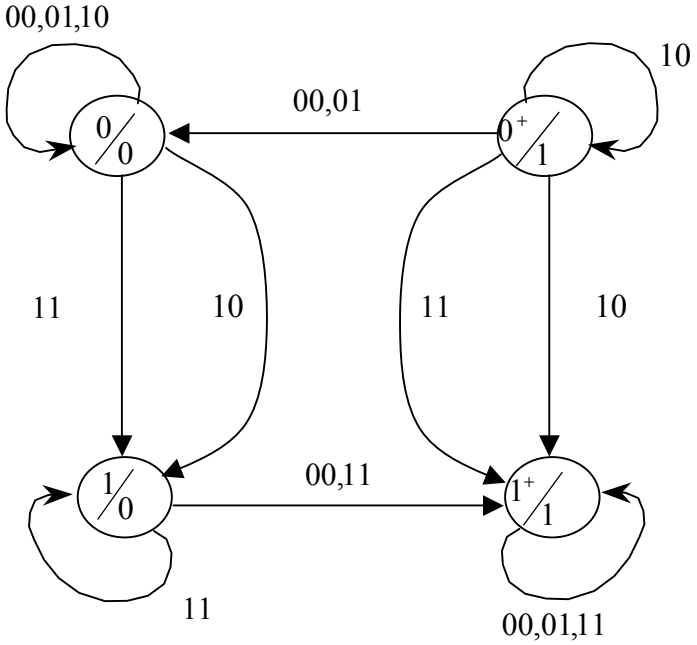
Если **c** = 0 – игнорирование входного знака и изменение выхода т.е.выдаёт то, что запомнил.

Нижнюю часть таблицы (2 сроки) определяем таким образом, чтобы автомат был асинхронным, хотя у нас имеется возможность заменить любой **0** на **1,** а любой 1^+ на 0^+**.** В пользу этого определения говорит ещё и то, что автомат не изменяет ранее запомненный входной знак.

Пусть синтезируемый триггер при пассивном значении тактового сигнала C = 0, запоминает состояние входа D. Если состояние входа C = 1, то запомненное состояние входа передаётся на выход триггера. После перехода сигнала C с 1 на 0, состояние выхода не изменится, а триггер опять запоминает сигнал на входе D.

Построим таблицу истинности:

C	Q^+	Q'		Q^+		Q
$\boldsymbol{c}$	q^+	q_1	q_0	q_1	q_0^+	q



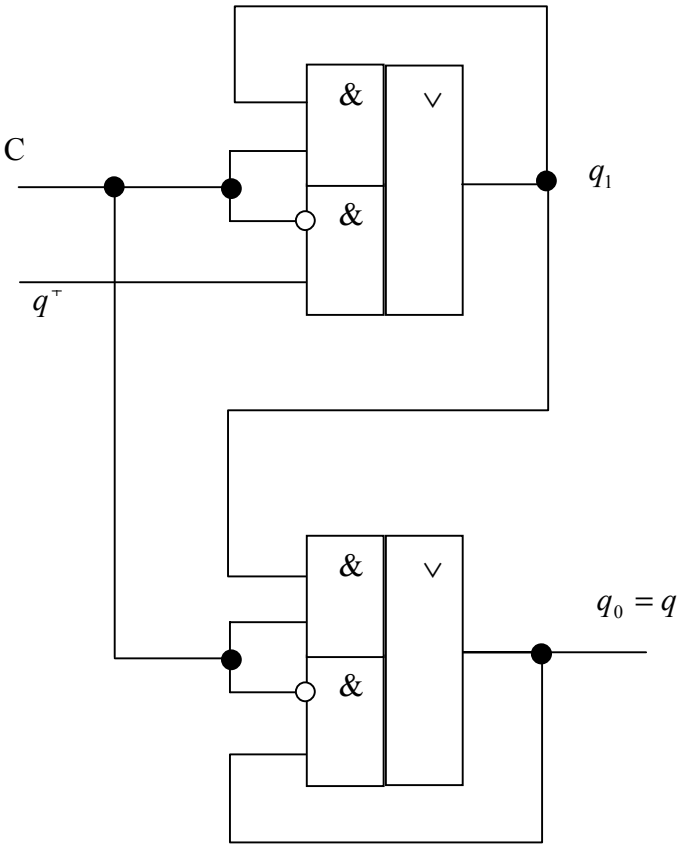
0	0	0	0	0	0	0
0	0	0	1	0	1	1
0	0	1	0	0	0	0
0	0	1	1	0	1	1
0	1	0	0	1	0	0
0	1	0	1	1	1	1
0	1	1	0	1	0	0
0	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	0	0	0
1	1	0	1	0	0	0
1	1	1	0	1	1	1
1	1	1	1	1	1	1

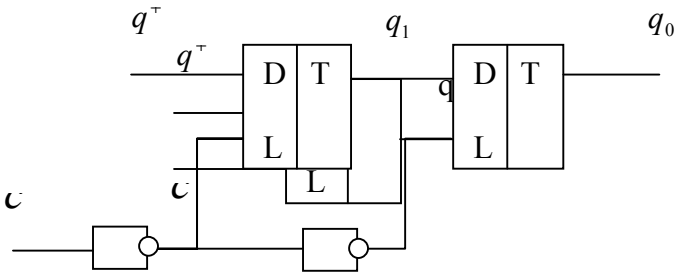
Где : $q = q_0$

$$q_0 = \bar{c} q_0 \vee c q_1$$

$$q_1 = \bar{c} q^+ \vee c q_1$$

Это последовательное соединение двух **DL** - триггеров.





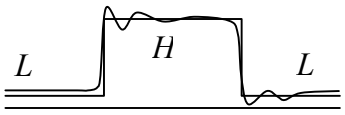
Раздел 5. Проектирование дискретных устройств

«Но да будет слово ваше «да, да», «нет, нет», а что сверх этого, то от лукавого».
Матфей 5,37.

Тема 5.1. Дискретные устройства

Дискретное устройство – это устройство, функционирующее в дискретные моменты времени и обрабатывающее дискретные сигналы.

Дискретный сигнал – это абстракция реального сигнала, который рассматривается как изменяющийся по закону дискретной функции.

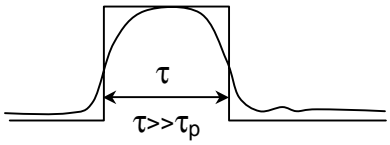


Сигнал – это параметр или числовая характеристика физического процесса, изменяющийся во времени и служащий для передачи данных.

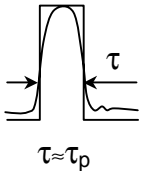
Электрический сигнал – это изменение во времени тока или напряжения в электрической сети.

Абстракции дискретных сигналов:

Потенциальный сигнал – это такой дискретный сигнал, соседние изменения которого значительно превосходят время реакции дискретного устройства, на которое он воздействует.



Импульсный сигнал– это такой дискретный сигнал, у которого время между соседними изменениями соизмеримо с временем реакции дискретного устройства на него.



Для описания изменений потенциальных сигналов и порождаемых ими импульсных сигналов удобно использовать математический аппарат, основанный на операторах переходов d и ∇.

Оператор переходов d. Пусть имеется некотроый дискретный сигнал $x(t)$.

Оператор переходов определяется соотношением:

$$dx(t) = x(t-\tau_p)\ \&\ \bar{x}(t)$$
$$d\bar{x}(t) = \bar{x}(t-\tau_p)\ \&\ \bar{\bar{x}}(t), \qquad (1)$$

где $dx(t)$ – импульсный сигнал, порождаемый изменением потенциального сигнала с 1 на 0; $x(t)$ – значение потенциального сигнала в данный момент времени; $x(t-\tau_p)$ – значение потенциального сигнала в предыдущий момент времени.

Очевидно, что $dx = 1$ только при изменении потенциального сигнала с 1 на 0, а $d\bar{x} = 1$ только при изменении потенциального сигнала с 0 на 1.

Введем обозначения сигналов $x(t) = x$, $x(t-\tau_p) = x'$, тогда, по формуле (1) имеем:

$$dx = \bar{x}\ \&\ x'$$

Можно показать, что

$$dx \,\&\, d\bar{x} = 0$$

Действительно, потенциальный сигнал не может одновременно изменяться с 1 на 0 и с 0 на 1.

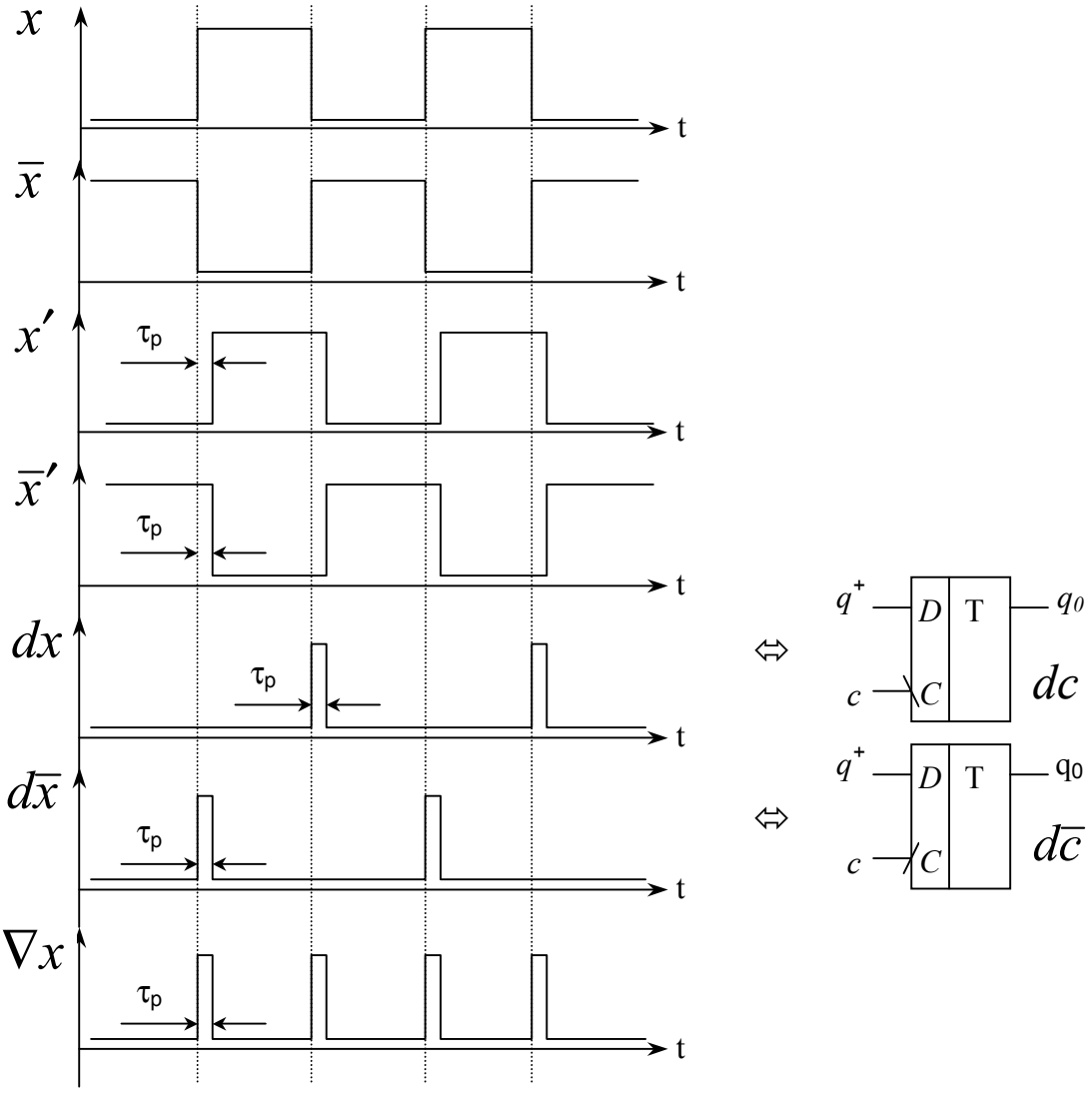
Оператор переходов ∇ определяется соотношением:

$$\nabla x = dx \cup d\bar{x} = x \oplus x' \qquad (2)$$

Очевидно, что $\nabla x = 1$ при изменении потенциального сигнала x как с 1 на 0, так и с 0 на 1.

Представим модель оператора переходов в виде временных диаграмм..

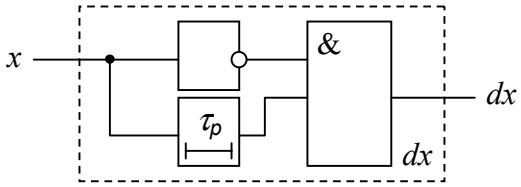
Будем считать, что для заданного устройства этот сигнал потенциальный.



Реализация оператора переходов

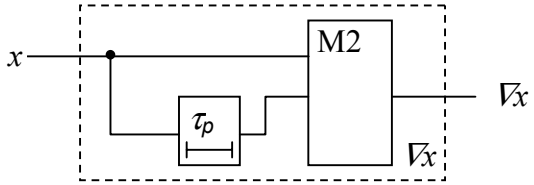
1. Построим схему оператора переходов d в соответствии с выражением (1).

$$dx(t) = x(t - \tau_p) \,\&\, \overline{x}(t)$$



2. Построим схему оператора переходов ∇ в соответствии с выражением (2).

$$\nabla x = dx \cup d\overline{x} = x \oplus x'$$

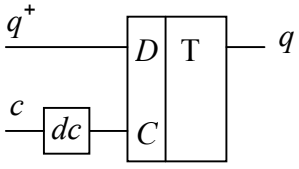


3. Синхронный элемент памяти на базе DC-триггера.

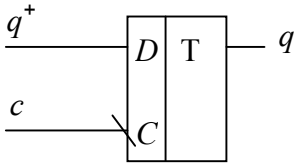
Синхронный D-триггер имеет:

- вход D для подачи информационного сигнала.
- вход C для подачи тактового сигнала.
- выход Q.

Обозначение D-триггера имеет вид:



Или можно обозначать:



Причем значение выходного сигнала *Q* определяется значением сигнала *D* в момент изменения тактового сигнала *C* с 1 на 0 (*dc*=1). При *dc*=0 выходной сигнал триггера не изменяется.

По данному словесному описанию закона функционирования синхронного D-триггера можно составить таблицу истинности:

dc	*D*	Q^+	*Q*
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

По таблице истинности составим функцию переходов, получим:

$$Q = Q^{+} dc \cup Q\overline{dc} \qquad (3)$$

Рассмотрим DC-триггер в цепи обратной связи.

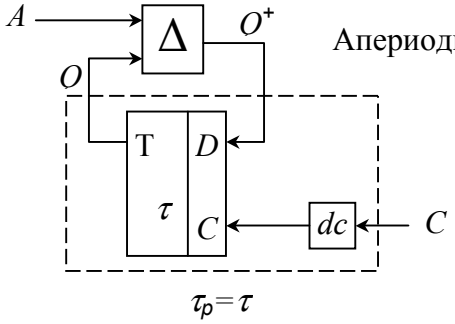


Схема называется апериодической, когда для ее правильной работы необходимо подобрать времена задержки.

В этой схеме DC триггер используется в качестве синхронного элемента памяти. Можно показать, что если функция переходов синхронного D-триггера содержала бы не импульсный сигнал *dc*, а потенциальный сигнал *C*, то такой триггер не может быть применим в синхронных схемах, т.к. после появления нового состояния на выходе автомата Δ это состояние опять воздействует на элемент памяти, т.к. сигнал C еще активный. Что может привести к повторной, неправильной, смене состояния на выходе элемента памяти. Если использовать схему оператора переходов, через которую пропускать синхросигнал, и выбрать τ_p достаточным для срабатывания триггера, то DC-триггер может быть использован, как синхронный элемент памяти.

Тема 5.2. Комбинационные схемы

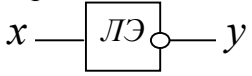
Комбинационной схемой называется сеть из логических элементов, реализующая некоторый комбинационный автомат. Следовательно, комбинационная схема реализует однозначное соответствие между значениями входных и выходных сигналов. В отличие от сети из элементарных комбинационных автоматов – эта схема построена на реальных логических элементах.

Под *Логическим Элементом (ЛЭ)* будем понимать реализацию элементарного комбинационного автомата из некоторого автоматного базиса.

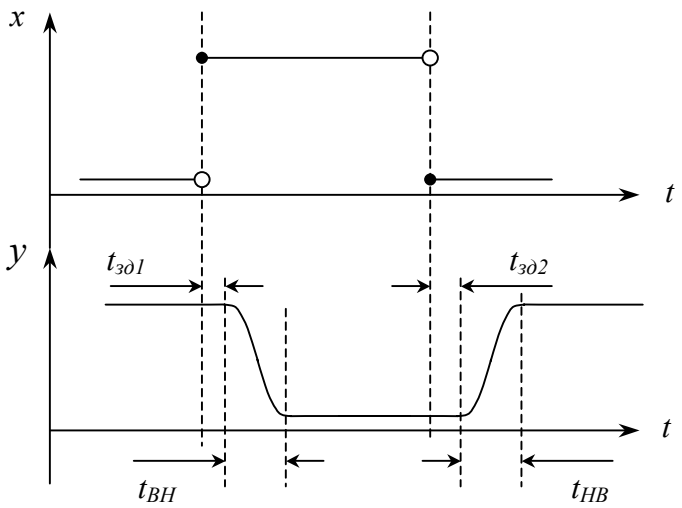
Логические элементы, используемые при синтезе комбинационных автоматов, по своим характеристикам отличаются от абстрактных элементарных комбинационных автоматов тем, что:

1. Каждый ЛЭ функционирует во времени, чем вносит задержку в распространение сигнала в схеме.
2. Каждый ЛЭ вызывает не мгновенное изменение сигнала на выходе, как это принято считать для дискретных функций, а изменение выходного сигнала занимает некоторое время.

Рассмотрим модель ЛЭ инвертора.

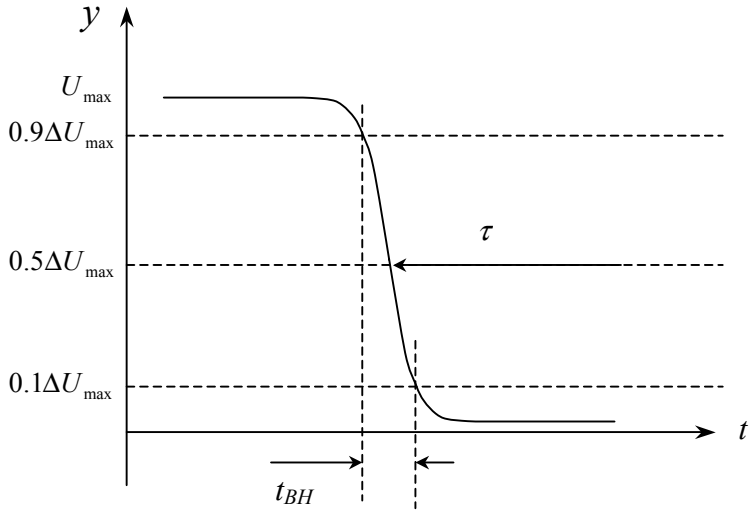


Представим модель этого ЛЭ в виде временных диаграмм.



где $t_{зд1,2}$ – время от смены сигнала на входе, до начала смены сигнала на выходе; $t_{ВН,НВ}$ – время смены выходного сигнала.

Рассмотрим подробнее участок перехода выходного сигнала от высокого уровня к низкому уровню:



Принято считать, что сигнал высокого уровня, если он больше $0,9\Delta U_{max}$ и сигнал низкого уровня, если он меньше $0,1\Delta U_{max}$.Знак Δ означает, что имеется ввиду амплитуда выходного сигнала.

Например, U_{max}=1В, но сигнал никогда не достигает этого уровня, а принимает значение не больше 0,75В. Тогда, такой сигнал, считается сигналом высокого уровня, если его значение больше 0,75*0,9=0,675В.

Величина τ – длительность нахождения сигнала в состоянии низкого уровня.

Иногда принято использовать *интегральные параметры*:

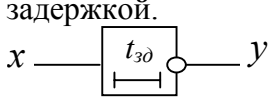
A. *Среднее время распространения задержки сигнала*

$$t_{зд} = \frac{t_{зд1} + t_{зд2}}{2}$$

B. *Среднее время перехода*

$$t_{пер} = \frac{t_{ВН} + t_{НВ}}{2}$$

Мы можем абстрагироваться от процессов перехода ЛЭ между состояниями и рассматривать модель как элемент с задержкой.

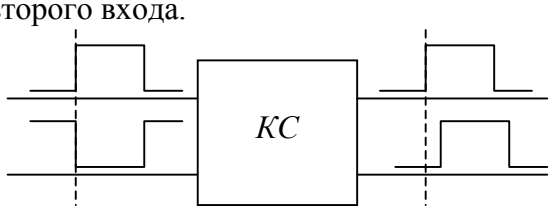


Переходные процессы в комбинационных схемах

Переходные процессы – это явление изменения во времени состояния выхода комбинационной схемы при установившихся значениях входных сигналов.

Состязания ЛЭ (гонки, явление риска) – это переходный процесс в комбинационной схеме, при котором наблюдаются не одновременные изменения выходных сигналов при одновременном изменении входных сигналов.

Например, рассмотрим комбинационную схему с двумя входами и двумя выходами. Эта схема должна пропускать сигнал с первого входа без изменений и инвертировать сигнал, поступающий со второго входа.

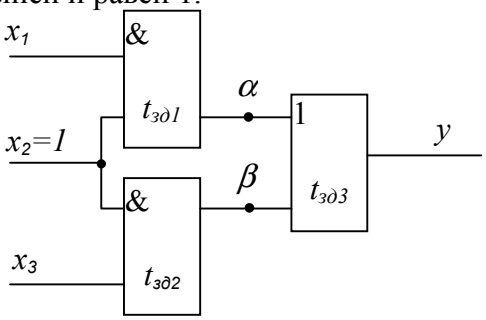


Как видно из рисунка, сигналы на входе изменяются одновременно, а на выходе второй сигнал запаздывает на некоторое время. Очевидно, что в это время схема возвращает ложное значение сигнала.

Рассмотрим причину возникновения состязания ЛЭ.

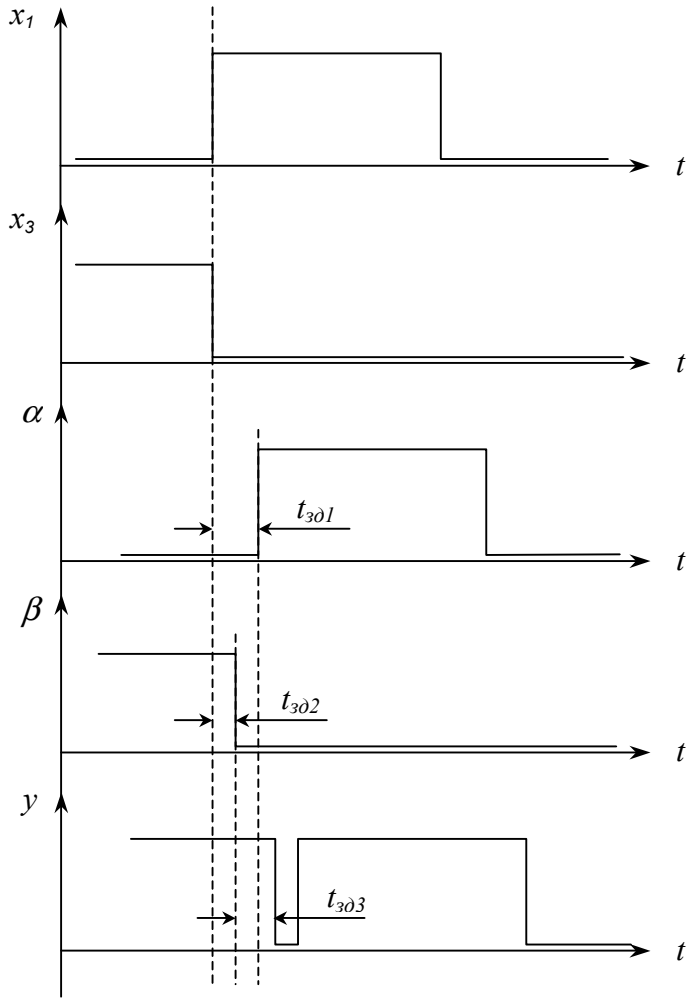
Пример. Схема имеет три входа и один выход.

Пусть сигнал x_2 постоянен и равен 1.



Будем считать, что все логические элементы имеют разные времена задержки, т.е. $t_{зд1} \neq t_{зд2} \neq t_{зд3}$.

Представим модель схемы в виде временных диаграмм.



Очевидно, что состязания наступают когда $t_{зд1}>t_{зд2}$.

Представим модель схемы в виде булевой функции:

$$y = x_1 x_2 \cup x_3 x_2$$

Если $x_2=1$, то

$$y = x_1 \cup x_3$$

Критическими состязаниями называются состязания ЛЭ при которых на выходе комбинационной схемы появляются кратковременные ложные значения выходного сигнала. (Например, на выходе схемы возникает ситуация, когда y=0, хотя в это время x_1=1.)

Синтез комбинационных схем, свободных от критических состязаний.

Ранее была показана возможность появления на выходах комбинационных схем ложных значений сигналов, что приводит к неправильной работе устройства.

Комбинационная схема называется *свободной от состязаний*, если в ней при соседних изменениях состояний входа отсутствуют критические состязания ЛЭ.

Соседние изменения состояния входа – это только те изменения входных сигналов, которые встречаются при функционировании дискретного устройства. (Например, для ранее рассмотренной схемы критического состязания может не произойти, если $x_1=0$, а $x_3=1$.)

Одним из способов борьбы с критическими состязаниями является использование такого кодирования алфавита автомата, при котором соседнее изменение состояния входа реализуется в виде изменения одного сигнала. Например, можно использовать код Грея.

Например:

$A = \{a_1, a_2, a_3\}$

$a_1 \to 010$

$a_2 \to 100$

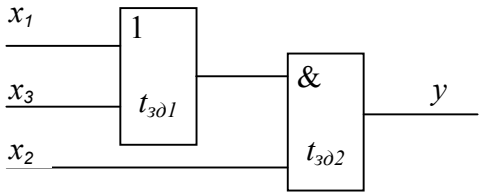
$a_3 \to 001$

Очевидно, для схем имеющих один выход, обеспечив изменение одного сигнала на входе, мы обеспечим избавление от ложных сигналов на выходе. Но если выходов несколько, то могут возникнуть критические состязания.

Теорема 5.1. О комбинационных схемах, свободных от критических состязаний

Идея теоремы состоит в том, что одни и те же схемы можно организовать по-разному:

$$y = x_1 x_2 \cup x_3 x_2 \Leftrightarrow y = (x_1 \cup x_3) x_2$$



Очевидно, если изменить одновременно x_1 и x_3, то критических состязаний не возникнет, а если x_3 и x_2, то могут возникнуть.

Утверждение. Для того, чтобы комбинационная схема была свободна от критических состязаний необходимо чтобы для соседних изменений входных сигналов только один входной сигнал x_t изменялся, а сама дискретная функция была представлена в виде выражения, свободного от критических состязаний, и такое выражение существует.

Доказательство.

- Пусть задана дискретная функция $f(x_{n-1}, x_{n-2}, \ldots x_{t+1}, x_t, x_{t-1}, \ldots, x_1, x_0)$, $x_j \in B$, которая должна быть реализована в виде комбинационной схемы, свободной от критических состязаний. Предположим, что это булева функция $B=\{0,1\}$

  В соответствии с теоремой Шеннона о разложении булевых функций запишем:

  Пусть

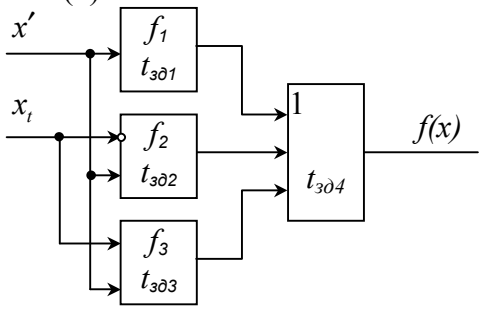
  $$x = (x_{n-1}, x_{n-2}, \ldots, x_{t+1}, x_t, x_{t-1}, \ldots, x_1, x_0)$$

  $$x' = (x_{n-1}, x_{n-2}, \ldots, x_{t+1}, x_{t-1}, \ldots, x_1, x_0)$$

  Тогда

  $$f(x) = f(x', x_t) = f_1(x') \cup \bar{x}_t f_2(x') \cup x_t f_3(x') \qquad (4)$$

- Представим произвольную комбинационную схему, сгруппируем ЛЭ которой в соответствии с выражением (4).



Так как по условию теоремы изменяется какой-то сигнал x_t, а величины задержек $t_{зд2}$ и $t_{зд3}$ не равны, то можно рассматривать комбинационную схему, зависящей от изменения двух сигналов: x_t и $\bar{x}_t$, которые могут изменяться не одновременно, т.е. может существовать период времени, когда они оба равны 0 или 1.

Очевидно, можно подобрать такую форму функции f^* чтобы неодновременные изменения x_t и $\bar{x}_t$ не внесли критических состязаний.

Т.о. выражение (4) следует рассматривать как:

$$f^*(x', x_t, \bar{x}_t) = f_1(x') \cup \bar{x}_t f_2(x') \cup x_t f_3(x') \qquad (5)$$

Представим функцию *f(x)* зависящей от *(n+1)* переменной: $x', x_t, \bar{x}_t$ и переменную x_t не будем однозначно связывать с $\bar{x}_t$. Это позволит смоделировать ситуацию, когда возникают состязания элементов, только введя опережение или запаздывание сигнала x_t относительно $\bar{x}_t$. Тогда функции f_1, f_2, f_3. можно рассматривать как не вносящие задержку в распространение сигнала.

- Возможны четыре случая, когда изменяется только один входной сигнал x_t:

a) $f(x',0) = 0$, $f(x',1) = 0$ - функция сохраняющая 0;

b) $f(x',0) = 0$, $f(x',1) = 1$ - функция не сохраняющая ни 0, ни 1;

c) $f(x',0) = 1$, $f(x',1) = 0$ - функция не сохраняющая ни 0, ни 1;

d) $f(x',0) = 1$, $f(x',1) = 1$ - функция сохраняющая 1;

Рассмотрим случай a)

$$f(x',0) = 0, \quad f(x',1) = 0$$

Из формулы (4) $f(x) = f(x', x_t) = f_1(x') \cup \bar{x}_t f_2(x') \cup x_t f_3(x')$ получаем:

$$\begin{cases} f(x',0) = f_1(x') \cup f_2(x') = 0 \\ f(x',1) = f_1(x') \cup f_3(x') = 0 \end{cases} \Rightarrow f_1(x') = f_2(x') = f_3(x') = 0$$

Тогда формула (5) $f^*(x', x_t, \bar{x}_t) = f_1(x') \cup \bar{x}_t f_2(x') \cup x_t f_3(x')$ 0принимает вид:

$$f^*(x', x_t, \bar{x}_t) = 0$$

Вывод: В случае a), при изменении сигнала x_t, состязания ЛЭ в комбинационной схеме не возникает, т.к. значение функции f^* не зависит от этого сигнала.

Рассмотрим случай b)

$$f(x',0) = 0, \quad f(x',1) = 1$$

Из формулы (4) получаем:

$$\begin{cases} f(x',0) = f_1(x') \cup f_2(x') = 0 \\ f(x',1) = f_1(x') \cup f_3(x') = 1 \end{cases} \Rightarrow \begin{cases} f_1(x') = f_2(x') = 0 \\ f_3(x') = 1 \end{cases}$$

Тогда формула (5) принимает вид:

$$f^*(x', x_t, \bar{x}_t) = x_t$$

В соответствии с этим выражением, при изменении переменной x_t с 1 на 0 или с 0 на 1, значение функции зависит только от переменной x_t и не зависит от $\bar{x}_t$.

Вывод: Т.к. в случае b) выходное значение комбинационной системы зависит только от сигнала x_t, то возникают состязания ЛЭ, но они не критические.

Рассмотрим случай c)

$$f(x',0)=1,\ f(x',1)=0$$

Из формулы (4) получаем:

$$\begin{cases} f(x',0)=f_1(x')\cup f_2(x')=1 \\ f(x',1)=f_1(x')\cup f_3(x')=0 \end{cases} \Rightarrow \begin{cases} f_1(x')=f_3(x')=0 \\ f_2(x')=1 \end{cases}$$

Тогда формула (5) принимает вид:

$$f^*(x',x_t,\bar{x}_t)=\bar{x}_t$$

Вывод: Т.к. в случае c) выходное значение комбинационной системы зависит только от сигнала $\bar{x}_t$, то возникают состязания ЛЭ, но они не критические.

Рассмотрим случай d)

$$f(x',0)=1,\ f(x',1)=1$$

Из формулы (4) получаем:

$$\begin{cases} f(x',0)=f_1(x')\cup f_2(x')=1 \\ f(x',1)=f_1(x')\cup f_3(x')=1 \end{cases} \qquad (6)$$

Система (6) имеет несколько решений:

- Если $f_1(x')=1$, то значение функции не зависит от переменной x_t и результат её вычисления схемой не зависит от переменных x_t и $\bar{x}_t$, т.е. состязания ЛЭ не возникает.
- Очевидно, состязания могут возникнуть при $f_1(x')=0$, тогда решение системы (6):

$$\begin{cases} f_2(x')=1 \\ f_3(x')=1 \end{cases}$$

Тогда формула (5) принимает вид:

$$f^*(x',x_t,\bar{x}_t)=\bar{x}_t\cup x_t$$

Если x_t и $\bar{x}_t$ некоторое время одновременно равны 1, то $f^*(x',x_t,\bar{x}_t)=1$, т.е. критические состязания не наступают. А если x_t и $\bar{x}_t$ некоторое время одновременно равны 0, то $f^*(x',x_t,\bar{x}_t)=0$, т.е. на выходе появляется ложное значение.

Вывод: Т.о. критические состязания возникают только при условии:

$$\begin{cases} f_1(x')=0 \\ f_2(x')=1 \\ f_3(x')=1 \\ f(x',0)=1 \\ f(x',1)=1 \end{cases} \qquad (7)$$

Кроме того, эти условия должны выполняться одновременно.

- Синтез формального представления булевой функции свободной от критических состязаний.

Условия (7) определяют, когда возможны критические состязания, приводящие к кратковременному появлению 0 на выходе комбинационной схемы.

Найдем все значения переменных из множества x', которые обращают уравнение системы (7) в тождества. Очевидно, если система (7) имеет решение, то могут возникнуть критические состязания. А если система (7) не имеет решения, то критические состязания не возникают.

Представим решения системы (7) в виде:

x'	Решения		
	0	…	S-1
x_{n-1}	$e_{n-1}^{(0)}$	…	$e_{n-1}^{(S-1)}$
…	…	…	…
x_{t+1}	$e_{t+1}^{(0)}$	…	$e_{t+1}^{(S-1)}$
x_{t-1}	$e_{t-1}^{(0)}$	…	$e_{t-1}^{(S-1)}$
…	…	…	…
x_0	$e_0^{(0)}$	…	$e_0^{(S-1)}$

Где e_j^i - значение переменной x_j из x' при котором уравнение системы (7) обращается в тождество, а i номер решения, т.к. их может быть несколько.

Другими словами перед нами стоит задача вычислить, на каких наборах переменных возникают критические состязания.

Определим корректирующую функцию $q(x')$ следующим образом:

$$q(x') = \bigcup_{i=0}^{S-1} x_{n-1}^{e_{n-1}^{(i)}} \dots x_{t+1}^{e_{t+1}^{(i)}} x_{t+1}^{e_{t+1}^{(i)}} \dots x_0^{e_0^{(i)}} \tag{8}$$

где

$$x^e = \begin{cases} \bar{x}, & e = 0 \\ x, & e = 1 \end{cases} \tag{9}$$

Свойства функции $q(x')$:

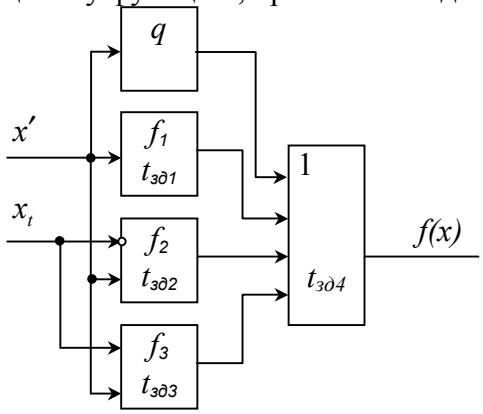
- $q(x') = 1$, если возникают критические состязания.
- $q(x') = 0$, если критические состязания не возникают.

Корректирующая функция $q(x')$ используется для реализации комбинационной схемы, свободной от критических состязаний.

С учетом функции $q(x')$ выражение (5) принимает вид:

$$f^*(x', x_t, \bar{x}_t) = f_1(x') \cup \bar{x}_t f_2(x') \cup x_t f_3(x') \cup q(x') \tag{10}$$

Тогда схема, реализующая эту функцию, принимает вид:



Критические состязания наступают только при значении функции равном 1, т. е. возможно кратковременное появление 0 на выходе, но в этот момент функция $q(x')$ принимает значение =1, поэтому *f(x)*=1.

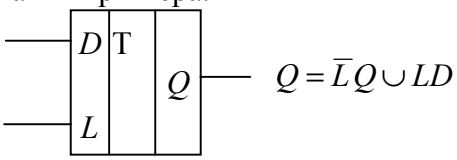
Очевидно, что функция $f(x', x_t)$ тождественно равна исходной функции $f(x)$

$$f(x', x_t) = f_1(x') \cup \bar{x}_t f_2(x') \cup x_t f_3(x') \cup q(x') \equiv f(x)$$

- Если допускается изменение более одного входного сигнала, то в общем случае невозможно синтезировать комбинационную схему, свободную от критических состязаний.

ч.т.д.

Пример 1. Реальная схема DL триггера.



Как видно, функция DL триггера зависит от *L,Q,D*, т.е. *f(L,Q,D)*

$$Q = \bar{L}Q \cup LD \qquad (11)$$

Рассмотрим случай, когда изменяется переменная *Q*, по формуле (11) и (5) имеем:

$f_1 = LD$, т.к. именно *LD* независимо от переменной *Q*.

$f_2 = 0$, т.к. нет такой части формулы (11), которая зависит от переменной $\bar{Q}$.

$f_3 = \bar{L}$, т.к. именно $\bar{L}$ зависит от переменной *Q*.

Q:
$$f = \underset{f_1}{(LD)} \cup \bar{Q} \underset{f_2}{(0)} \cup Q \underset{f_3}{(\bar{L})}$$

Очевидно, при изменении переменной *Q* критические состязания не возникают, т.к. $f_2=0$

Рассмотрим случай, когда изменяется переменная *D*, по формуле (5) имеем:

D:
$$f = \underset{f_1}{(\bar{L}Q)} \cup \bar{D} \underset{f_2}{(0)} \cup D \underset{f_3}{(L)}$$

Очевидно, при изменении переменной *D* критические состязания не возникают, т.к. $f_2=0$

Рассмотрим случай, когда изменяется переменная *L*, по формуле (5) имеем:

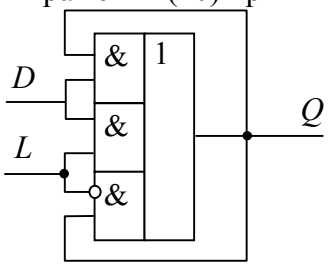
L:
$$f = \underset{f_1}{(0)} \cup \bar{L} \underset{f_2}{(Q)} \cup L \underset{f_3}{(D)}$$

Очевидно, при изменении переменной *L* возникают критические состязания, т.к. $f_1=0$, $f_2\neq0$ и $f_3\neq0$.

Тогда по выражениям (9) и (8) имеем:

$$\begin{cases} Q = 1 = e_Q^{(0)} \\ D = 1 = e_D^{(0)} \end{cases} \Rightarrow q(Q, D) = QD$$

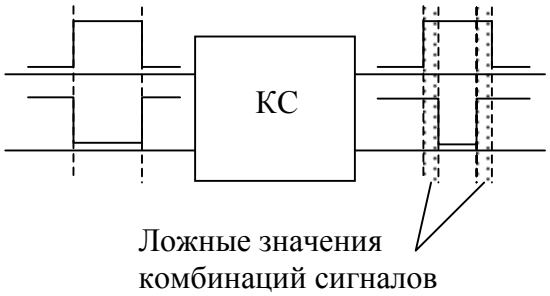
Схема DL триггера с учетом выражения (10) принимает вид:



Этот триггер работает без критических состязаний даже при различных временах задержки ЛЭ.

Синтез комбинационных схем, свободных от состязаний, с несколькими выходами

Рассмотрим комбинационную схему, повторяющую входные сигналы. Очевидно, если времена смены выходного сигнала $t_{ВН} \neq t_{НВ}$, то могут возникнуть критические состязания.



Основной метод синтеза таких комбинационных схем заключается в использовании кода Грея при кодировании выходных алфавитов, когда только один выходной сигнал меняется для каждой смены входных сигналов.

Вторым методом синтеза является синтез комбинационных схем апериодическим методом, когда задержки, вносимые ЛЭ, подбираются. Причем точность подбора задержек должна соответствовать времени реакции схемы, на которую эти сигналы воздействуют.

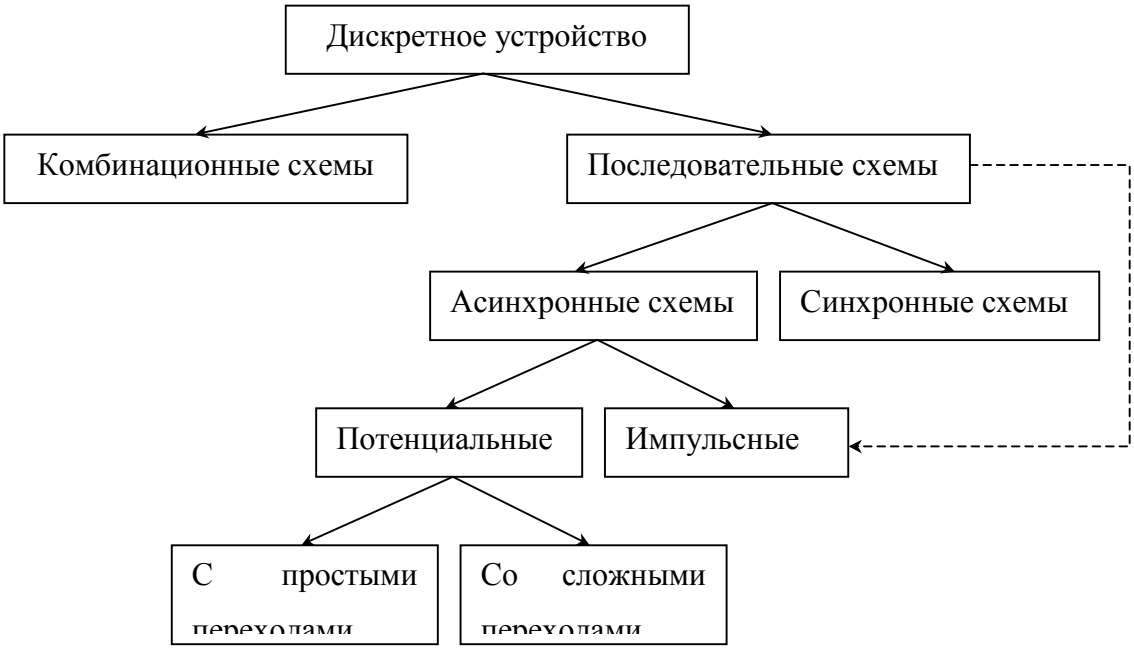
Тема 5.3. Асинхронные схемы

Ранее было показано, что автоматы можно разделить на:

- комбинационные
- асинхронные
- синхронные

Причем, каждый последующий является обобщением предыдущего.

Приведем обобщенную схему классификации дискретных устройств:



Комбинационная схема – это дискретное устройство, функционирование или поведение которого в каждый момент времени определяется только входными сигналами, воздействующими на него.

Последовательная схема – это дискретное устройство, функционирование или поведение которого зависит от предыстории воздействия на него входных знаков.

Эта схема имеет более одного внутреннего состояния. Теоретической моделью для этих схем являются конечные автоматы Милли или Мура.

Синхронный автомат (схема) – это дискретное устройство, такты функционирования которого определяются в моменты импульсного воздействия тактового сигнала, когда остальные входные сигналы удерживаются или остаются неизменными.

Асинхронный автомат (схема) – это дискретное устройство, такты функционирования которого определяются в момент смены входных сигналов.

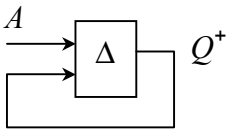
Асинхронный потенциальный автомат (схема) – воспринимает воздействия при активных значениях входных сигналов, которые продолжаются до тех пор, пока сохраняется их активный уровень.

(Асинхронный) импульсный автомат – воспринимает воздействия только при изменении входных сигналов, т.е. входные сигналы оказывают на него импульсное воздействие.

Асинхронный потенциальный автомат с простыми переходами – это автоматы, у которых все возможные переходы из одного состояния в другое являются простыми.

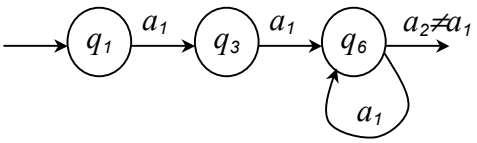
Асинхронный потенциальный автомат со сложными переходами – это автоматы, у которых имеются переходы из одного состояния в другое, которые являются сложными, т.е. это автоматы, у которых возможны неоднократные изменения внутренних состояний при неизменных входных сигналах.

Представим схему асинхронного потенциального автомата со сложными переходами.



Например, при подаче на вход автомата знака a_1, автомат вычисляет состояние q_3, которое попадает на его вход, и автомат вычисляет другое состояние q_6, только после чего автомат успокаивается.

Проиллюстрируем данный пример:



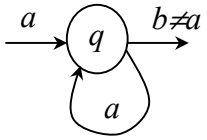
Асинхронные автоматы

Моделью асинхронного автомата, как дискретного устройства, является асинхронный конечный автомат, у которого все состояния устойчивы по всем входным знакам, ведущим в это состояние.

Формально асинхронный конечный автомат можно представить:

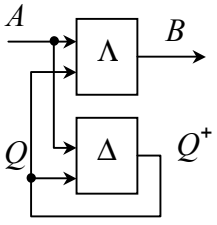
$$K=<A,Q,B,\delta,\lambda> \qquad \forall q\in Q,\ \forall a\in A \quad \delta(q,a)=\delta(q,aa)$$

На графе это выглядит следующим образом: если есть дуга, ведущая в вершину q, помеченная знаком a, то в вершине q есть петля, помеченная знаком a и не существует дуги из вершины q, помеченной знаком a.



Очевидно, только изменение знака на входе может привести к смене состояния автомата. Это значит, что синхронизация работы автомата в автоматной сети, состоящей из асинхронных автоматов, выполняется автоматически, без специальных мер.

Структурная схема асинхронного автомата имеет вид:

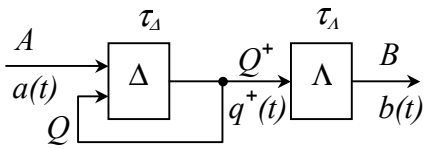


Как видно из схемы выходной знак B и следующее состояние автомата Q^+ определяются через автоматы Λ и Δ соответственно, на входы которых подаются знаки Q и A.

Запишем систему уравнений, которая описывает асинхронный автомат.

$$\begin{cases} b(t+\tau_\Lambda)=\lambda(q(t),a(t)) \\ q^+(t)=\delta(q(t),a(t)) \\ q(t+\tau_\Delta)=q^+(t) \end{cases} \qquad (12)$$

В случае автомата Мура схема примет вид:



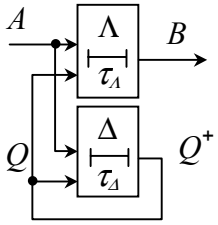
Где τ_Δ и τ_Λ это задержка, вносимая в распространение сигнала комбинационным автоматом Δ и Λ соответственно.

Модели асинхронных автоматов

В основном на практике используются две модели: частная и общая.

Общая модель рассмотрена ранее, где предполагается, что комбинационные автоматы Δ и Λ вносят задержку в распространение сигнала, т.е. это модель с задержками в комбинационных схемах.

Структурная схема имеет вид:

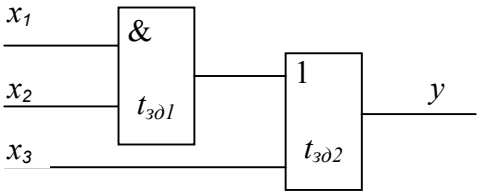


При рассмотрении комбинационной схемы мы видим, что в общем случае задержка в распространение сигнала комбинационной схемой зависит от комбинации входных сигналов.

$$\begin{cases} \tau_\Lambda = \tau_\Lambda(q(t), a(t)) \\ \tau_\Delta = \tau_\Delta(q(t), a(t)) \end{cases}$$

Т.е. задержки τ_Δ и τ_Λ зависят от того, какие сигналы поданы на вход.

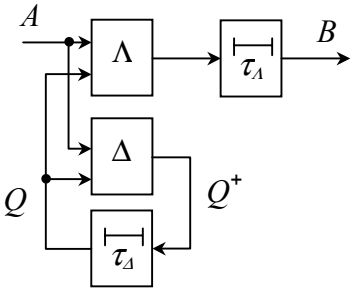
Например, рассмотрим схему:



Очевидно, изменение сигнала x_3 приведет к меньшей задержке в распространение сигнала, чем изменение сигналов x_1 и x_2, т.к. сигнал будет задержан только на $t_{зд2}$.

Вторая модель с элементами задержки в цепи обратной связи.

Структурная схема которой имеет вид:



Здесь предполагается, что комбинационные схемы не вносят задержку в распространение сигнала.

Асинхронные потенциальные автоматы

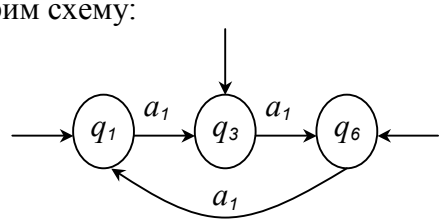
У асинхронных потенциальных автоматов входные сигналы оказывают потенциальное воздействие на схему, т.е. смена сигнала на входе задает такты функционирования автомата, а сами сигналы воздействуют на него все время.

Различают автоматы с простыми и со сложными переходами.

Для синтеза асинхронных потенциальных автоматов, свободных от критических состязаний, т.е. когда автомат функционирует неправильно, необходимо выполнить ряд условий:

Тема 1.5 При переходных процессах не должны возникать автоколебательные процессы, это может произойти, когда используется автомат со сложными переходами, образующими цикл.

Например, рассмотрим схему:



Как видно из рисунка, автомат будет перебирать состояния: $q_1, q_3, q_6, q_1, q_3, q_6, \ldots$

Тема 1.6 Комбинационные схемы должны синтезироваться свободными от критических состязаний, а если критические состязания и присутствуют, то они не должны оказывать воздействия на входы следующего автомата. Это означает, что время, в течении которого на выходе появляются ложные сигналы должно быть намного меньше, чем время реакции комбинационной схемы. На практике это можно организовать т.к. величина времени реакции схемы может быть вычислена.

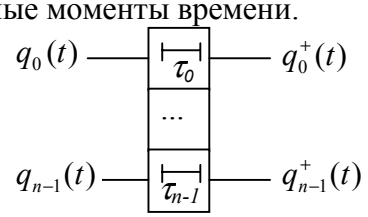
Тема 1.7 Величина задержки сигнала τ_Δ должна быть больше максимальной длительности переходных процессов.

Это требование вытекает из требования № 2. Оно используется при рассмотрении второй модели асинхронного потенциального автомата с элементами задержки в цепи обратной связи.

Тема 1.8 Частота изменения входного сигнала должна быть не больше некоторой частоты f_{max} , при которой в автомате еще успевают завершиться переходные процессы.

$$f_{\max} < \frac{1}{\tau_\Delta}$$

Тема 1.9 Должны отсутствовать критические состязания элементов задержки в цепи обратной связи, т.е. когда разные сигналы обратной связи принимают свои истинные значения в разные моменты времени.



Это условие применяется в том случае, если используется многоразрядное кодирование состояний автомата.

τ_i должны подбираться с точностью, большей чем время реакции схемы. Если же это невозможно обеспечить, то для кодирования знаков состояния необходимо использовать код Грея, когда при смене состояний изменяется только один сигнал.

Например:

$Q = \{q_1, q_2, q_3\}$

$q_1 \to 010$

$q_2 \to 100$

$q_3 \to 001$

Первое и последнее условия являются необходимыми, а при невыполнении остальных условий иногда автомат будет работать правильно.

Пример. Синтез RS триггера.

RS триггер имеет два состояния $Q=\{q_0,q_1\}$.

Триггер переходит в состояние q_1 если на входе знак a_2;

Триггер переходит в состояние q_0 если на входе знак a_1;

Триггер не изменяет состояния, если на входе знак a_0.

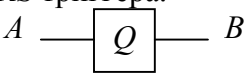
Кратко это можно записать:

$a_2 \to q_1$

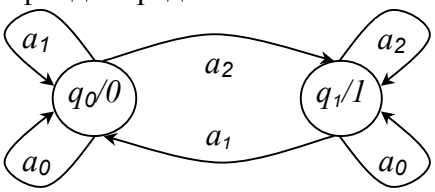
$a_1 \to q_0$

$a_0 \to$ *состояние не меняется.*

Изобразим структурную схему RS триггера.



Нарисуем автоматный граф и доопределим состояния выхода.



Представим автомат в виде таблицы:

$A\backslash Q$	$q_0/0$	$q_1/1$
a_0	q_0	q_1
a_1	q_0	q_0
a_2	q_1	q_1

Перейдем к дискретным функциям:

A	Q	Q^+	B
a_0	q_0	q_0	0
a_0	q_1	q_1	1
a_1	q_0	q_0	0
a_1	q_1	q_0	0
a_2	q_0	q_1	1
a_2	q_1	q_1	1

Закодируем алфавиты:

A	a_0	a_1	a_2
SR	00	01	10

B	0	1
b	0	1

Q	q_0	q_1
q	0	1

Представим поведение автомата в виде булевых функций:

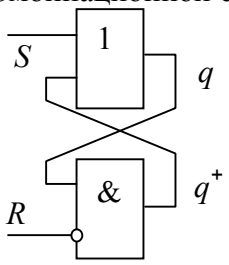
A	Q	Q^+	B

S	R	q	q^+	b
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1

Представим функцию q^+ в виде выражений в автоматном базисе:

$$q^+ = \overline{S}\ \overline{R}q \cup S\overline{R} = \overline{R}(\overline{S}q \cup S) = \overline{R}(S \cup q) \qquad (13)$$

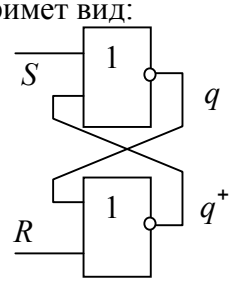
Представим выражение в виде комбинационной схемы:



Преобразуем выражение для q^+:

$$q^+ = \overline{\overline{\overline{R}(S \cup q)}} = \overline{R \cup \overline{(S \cup q)}}$$

Тогда комбинационная схема примет вид:



Замечание 1.

Из выражения (13) $Q^+ = \overline{R}(S \cup Q)$ следует, что комбинационная схема является свободной от критических состязаний.

Замечание 2.

Вход R имеет больший приоритет по отношению ко входу S. По функции переходов триггера можно установить, какие значения входных сигналов изменяют состояние триггера.

А. Пусть состояние триггера изменяется с 1 на 0.

Q^+*:1→0,* тогда dQ^+*=1;*

По формуле (1) оператора переходов $dx = \overline{x}(t)\ \&\ x(t-\Delta t)$ имеем:

$$dQ^+ = \overline{Q}^+(t)\ \&\ Q^+(t-\Delta t) \qquad (14)$$

Для того, чтобы найти $Q^+(t-\Delta t)$ воспользуемся системой (12):

$Q(t+\Delta t) = Q^+(t)$, из него вытекает, что $Q^+(t-\Delta t) = Q(t)$;

Подставив в формулу (14) получаем:

$$dQ^+ = \overline{Q}^+\ \&\ Q = 1;$$

Ранее было показано, выражение (13), что $Q^+ = \overline{R}(S \cup Q)$, тогда

$$dQ^+ = \overline{\overline{R}(S \cup Q)}\ Q = (R \cup \overline{S}\ \overline{Q})Q = RQ$$

Т.о. триггер переходит из состояния 1 в состояние 0 только при сигнале R=1 и состоянии Q=1.

В. Пусть состояние триггера меняется с 0 на 1.

Q^+:*0→1*, тогда $d\overline{Q}^+ = 1$;

Можно показать, что в этом случае $d\overline{Q}^+ = Q^+ \,\&\, \overline{Q} = 1$, тогда

$$d\overline{Q}^+ = \overline{R}(S \cup Q)\overline{Q} = \overline{R}S\overline{Q}$$

Т.о. триггер переходит из состояния 0 в состояние 1 только при состоянии Q=0, сигналах R=0 и S=1.

Вывод: сигнал R более приоритетен, чем сигнал S.

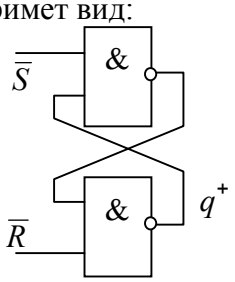
Замечание 3.

Так как функция переходов триггера, заданная в виде таблицы является не полностью определенной и содержит равное число 0 и 1, то в виде формулы она может быть представлена и с использованием СКНФ, тогда получим:

$$Q^+ = S \cup \overline{R}Q = \overline{\overline{S \cup \overline{R}Q}} = \overline{\overline{S}\ \overline{\overline{R}\ Q}}$$

В этом случае можно показать, что приоритетным является вход S.

Тогда комбинационная схема примет вид:

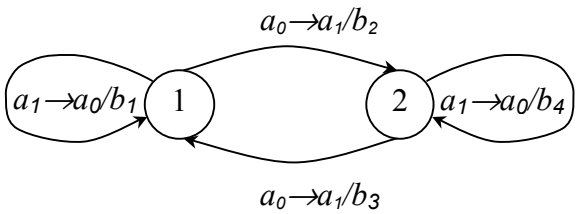


Тема 5.4. Импульсные автоматы

Импульсными автоматами называются такие автоматы, на которые входные сигналы воздействуют импульсно, т.е. кратковременно, в моменты их изменения.

Рассмотрим пример счетного триггера:

$A=\{a_0,a_1\}$;



Этот автомат имеет два типа переходов:

1-й тип: $a_0 \to a_1$

2-й тип: $a_1 \to a_0$

Примем эти переходы за знаки автомата.

Задав выходной алфавит $B=\{b_1,b_2,b_3,b_4\}$, автомат будет сообщать путь, по которому он попал в текущее состояние.

Воспользуемся формулой (1), оператора переходов: $dx = x(t-\Delta t)\,\&\,\overline{x}(t)$

В случае более двух знаков определим оператор переходов так:

Пусть есть алфавит $A=\{a_1,a_2,\ldots,a_m\}$. Оператор переходов должен сопоставлять каждой паре различных знаков новый знак из алфавита, содержащего m(m-1) знак.

Представим такой оператор переходов в виде таблицы.

A		dA
a^-	a^+	
a_1	a_1	–
a_1	a_2	a_{12}

…	…	…
a_1	a_m	a_{1m}
a_2	a_1	a_{21}
a_2	a_2	–
…	…	…
a_2	a_m	a_{2m}
…	…	…
…	…	…
…	…	…
…	…	…
a_m	a_1	a_{m1}
a_m	a_2	a_{m2}
…	…	…
a_m	a_m	–

Где a^- - значение входного сигнала до изменения, а a^+ - значение входного сигнала после изменения,

Зададим автоматную таблицу счетного триггера:

A		1	2
a^-	a^+		
a_0	a_1	$2/b_2$	$1/b_3$
a_1	a_0	$1/b_1$	$2/b_4$

Закодируем алфавиты:

A	a_0	a_1
	1	0

С учетом введенного кодирования можно записать:

$$a_1 \to a_0 \Leftrightarrow d\overline{a}$$

$$a_0 \to a_1 \Leftrightarrow da$$

Причем оператор переходов =1, если переход произошел и =0, если не произошел.

Q	1	2
	0	1

B	b_1	b_2	b_3	b_4
	00	01	10	11

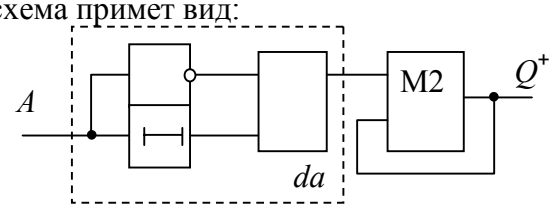
Представим поведение автомата в виде булевых функций:

da	Q	Q^+	B	
1	0	1	0	1
1	1	0	1	0
0	0	0	0	0
0	1	1	1	1

Представим функцию Q^+ в виде выражений в автоматном базисе:

$$Q^+ = Q \oplus da$$

Тогда импульсная схема примет вид:



Теорема 5.2. О реализации импульсных автоматов

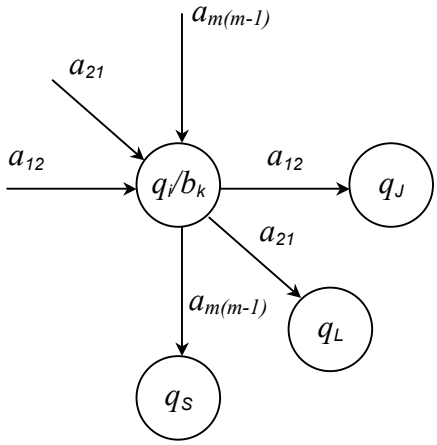
Произвольный импульсный автомат может быть реализован в виде асинхронного потенциального автомата. Верно и обратное.

Доказательство осуществим на примере.

Пусть задан импульсный автомат $K=<dA,B,Q,\delta,\lambda>$.

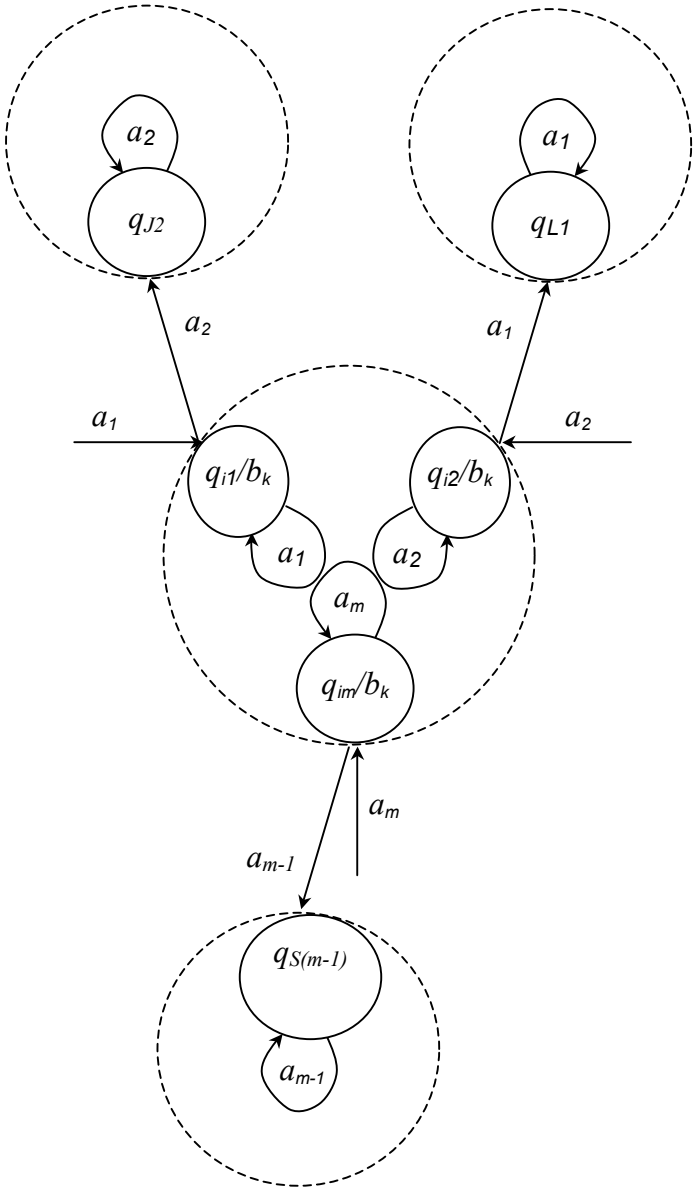
Для построения асинхронного потенциального автомата, неотличимого от заданного преобразуем его в эквивалентный ему автомат Мура $K_M=<dA,B,Q_M,\delta_M,\lambda>$.

Пусть задан импульсный автомат Мура.



Где a_{ij} это изменение входного знака $a_i \rightarrow a_j$.

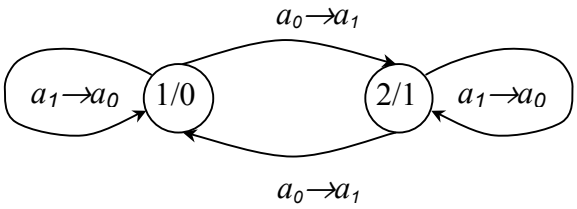
Тогда, каждое состояние $q_i \in Q_M$ расщепляется на $m(m-1)$ состояний следующим образом:



ч.т.д.

Пример:

Рассмотрим импульсный автоматный граф:

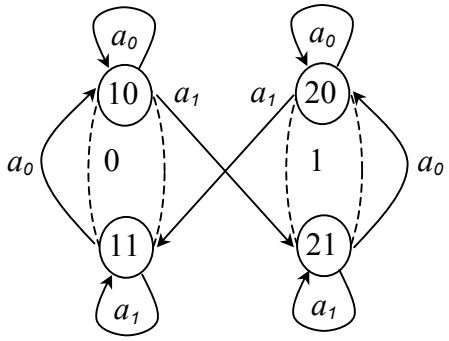


Преобразуем данный автомат в асинхронный автоматный граф.

Как видно из рисунка, каждое состояние (1 и 2) расщепляется на два состояния (10, 11 и 20,21). Причем, состояния 10 и 11 имеют один и тот же выходной знак 0, а состояния 20 и 21 – знак 1.

Из состояния 1 в состояние 2 имеется дуга $a_0 \to a_1$, следовательно, из состояния 10 имеется дуга в состояние 21, помеченная знаком a_1. И к состоянию 21 добавляем петлю, помеченную знаком a_1 .

Аналогично строятся остальные дуги.

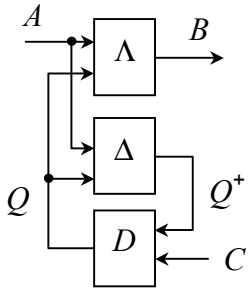


Тема 5.5. Синхронные схемы

Синтез синхронных автоматов, свободных от критических состязаний.

Введение тактового сигнала позволяет исключить из рассмотрения переходные процессы.

Структурная схема синхронного автомата имеет вид:



Тактовый сигнал *C* выполняет функцию временного селектирования (импульсного воздействия) сигналов элемента памяти *D* в дискретные моменты времени, поэтому необходимо, чтобы значения сигнала Q^+ были истинными только в моменты импульсного воздействия тактового сигнала *C*.

Запишем систему уравнений, которая описывает синхронный автомат:

$$\begin{cases} b(t+\tau_\Lambda) = \lambda(q(t), a(t)) \\ q^+(t+\tau_\Delta) = \delta(q(t), a(t)) \\ q(t+\tau_D) = q^+(t)dc \cup q(t)\overline{dc} \end{cases}$$

Основным условием синтеза синхронных автоматов, свободных от критических состязаний является то, что минимальное значение периода тактового сигнала *C* должно быть не меньше максимального времени переходного процесса. При этом комбинационные схемы можно синтезировать и с критическими состязаниями и использовать произвольное кодирование внутренних состояний автомата.

Синхронный элемент памяти.

В качестве синхронного элемента памяти могут быть использованы импульсные или асинхронные потенциальные автоматы. Пример синтеза D триггера был рассмотрен ранее.

Поведение синхронного элемента памяти можно описать уравнением (*3*):

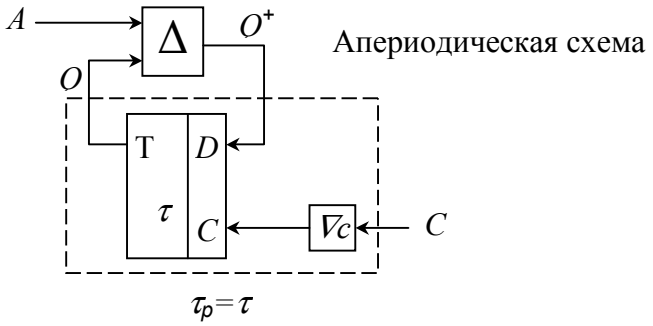
$$Q = Q^{+} dc \cup Q\overline{dc}$$

Если воспользоваться выражением (2) $\nabla x = dx \cup d\bar{x}$;

То можно записать:

$$Q = Q^{+} \nabla c \cup Q\overline{\nabla c} \qquad (15)$$

Рассмотрим синхронный элемент памяти, реализующий выражение (15), в цепи обратной связи



Очевидно, что значение выходного сигнала *Q* определяется значением сигнала, поступающего на вход *D,* в момент изменения тактового сигнала *C* с 1 на 0 (∇c=1) и с 0 на 1 (∇c=1).

СПИСОК ЛИТЕРАТУРЫ

1. Рейуорд-Смит В.Дж. Теория формальных языков. Вводный курс: Пер. с англ. – М.: Радио и связь, 1988.

2. Кузнецов О.П., Адельсон-Вельский Г.М. Дискретная математика для инженера. – 2-е изд., перераб. и доп. – М.: Энергоатомиз-дат, 1988. – 480 с.: ил.

3. Пухальский Г.И., Новосельцева Т.Я. Цифровые устройства: Учебное пособие для втузов. – СПб.: Политехника, 1996. – 885 с.

4. Кудрявцев В.Б., Алешин С.В., Подколзин А.С. Введение в теорию автоматов. М.: Наука, 1985.

5. Горбатов В.А. Основы дискретной математики: Учебное пособие для студентов вузов. – М.: Высш.шк, 1986.

6. Брой М. Информматика. Теоретическая информатика, алгоритмы и структуры данных, логическое программирование, объектная ориентация: В 4-х ч. Ч. 4. – М.: Диалог-МИФИ, 1998. – 224 с.

7. Братчиков И.Л. Синтаксис языков программирования. – М.: Наука, 1975. – 232 с.

Учебное издание

Выхованец Валерий Святославович

Теория автоматов

Учебное пособие для вузов

Редактор О. Д. Выхованец

Компьютерный набор и верстка Д. П. Епифанов, А.В. Булда, С.П. Алешков

Компьютерная графика Л. А. Стоянова