## Теория автоматов и формальных языков

### 1. Лексикографический номер

Рассмотрим нумерацию называемую лексикографической. Данной нумерацией пустому слову присваивается ∅, а буквам алфавита $\Sigma = \{a_1, a_2, ..., a_n\}$ соответственно. И если слово X имеет номер $l_x$, то слова $x_{ai}$ имеют номер $nl_x + i$. Отсюда следует, что лексикографический номер слова определяется следующим образом: $n_{i1}^{k-1} + n_{i2}^{k-2} + ... + i_k$, те к любому слову можно однозначно вычислить его лексикографический номер.

Обратный случай, где по номеру определяется слово:

Пусть N - лексикографический номер

Если N = 0, то это номер пустого слова

Если N ≠0,то N = $k_1 n + r_0$, где $1 \le n_0 \le n$

Пусть $k_1 \le n$, тогда N будет номером слова $a_{k1} * a_{r0}$

В противном случае (k<n) $k_1 = k_2 n + r_1$, $1 \le r_0 \le n$

$N = (k_2 n + r_1)n + r_0$

$N = k_2 n^2 + r_1 n + r_0$

Если $k_2 > n$, то N номер слова $ak_2 ar_1 ar_0$

Если $k_3 n^3 + r_2 n^2 + r_1 n + r_0$

Продолжим те же действия до тех пор пока $k_3 \le n$

Пример:

Предположим, что нам дан алфавит $\Sigma = \{a, b, c\}$ и слово abccb

Найти лексикографический номер слова

n - количество букв

k - длина слова

i - порядковый номер в алфавите

$3^4 * 1 + 3^3 * 2 + 3^2 * 3 + 3^1 * 3 + 3^0 * 2 = 173$

(a) (b) (c) (c) (b)

$\Sigma^*$ - алфавит со всевозможными комбинациями из букв алфавита Σ

$\Sigma = \{a, b, c\} \Rightarrow \Sigma^* = \{\varepsilon, a, b, c, aa, bb, ac, ba, bb, bc, ca, cb, cc, ...\}$(порядок важен)

$L \subseteq \Sigma^*$

Пример:

Предположим, что нам дан алфавит $\Sigma = \{a, b, c\}$ и номер $N = 321$

Найти слово по лексикографическому номеру

$321 = 107 * 3 + 0 \Rightarrow 106 * 3 + 3 \Rightarrow (35 * 3 + 1) * 3 + 3 \Rightarrow$

$\Rightarrow ((11 * 3 + 2) * 3 + 1) * 3 + 3 \Rightarrow (((3 * 3 + 2) * 3 + 2) * 3 + 1) * 3 + 3$

Раскроем скобки: материал

$((3 * 3^2 + 2 * 3 + 2) * 3 + 1) * 3 + 3 = (3 * 3^3 + 2 * 3^2 + 2 * 3 + 1) * 3 + 3 =$

$= 3 * 3^4 + 2 * 3^3 + 2 * 3^2 + 1 * 3^1 + 3 * 3^0$

k-1 = 4 ⇒k = 5 - длина слова

Первые цифры в произведениях - это номера буквы в алфавите

По итогу получаем: cbbac(32213)

## 2. Способы задания и распознавания формальных языков

Введение

Говоря "формальный язык" мы имеем в виду, что приведенные в этом курсе результаты используется прежде всего при описании искусственных языков. Но нет особой разницы между специально придуманными формальными языками и стихийно возникающими - естественными языками.

Изучая языки следует иметь в виду 3 основных аспекта:

1) Синтаксис языка;

   Язык - это какое-то множество слов, где слово есть определенная конечная последовательность букв(символов алфавита).

   Термин "буква" и "слово" могут пониматься по-разному.

   В том случае, когда словами будут определенные конечные последовательности букв, то они называются лексемами, не каждая последовательность букв будет лексемой данного языка, а только такая, которая отвечает определенным правилам.

   Синтаксис языка и представляет собой систему правил, в соответствии с которыми можно строить правильные последовательности букв. Тогда необходимо с одной стороны разработать механизмы перечисления или порождения слов заданной структурой, а с другой сторона механизмы проверки того, что данное слово принадлежит языку.

   Прежде всего именно эти механизмы изучает классическая теория формальных языков.

2) Семантика языка;

   Семантика предполагает подставление словам языка определенного смысла

3) Прагматика языка;

   Прагматика связана с тем целями, которые ставит перед собой носители языка.

Элементы теории формальных языков

- Алфавит - это конечное множество символов;
- Цепочкой символов в алфавите Σ называется любая конечная последовательность символов этого алфавита.

Цепочка, которая не содержит ни одного символа называется пустой.

Для ее обозначения используют символ $\varepsilon(e)$. Предполагается, что сама буква ε в алфавит Σ не входит.

- Пусть α и β некоторые цепочки, тогда цепочка αβ называется конкатенацией(сцеплением)цепочек α и β. Конкатенацию можно считать двуместной операцией и записывать: $\alpha * \beta = \alpha\beta$ (к концу первой цепочки ставится вторая)

Для любой цепочки α:

$$\alpha * \varepsilon = \varepsilon * \alpha = \alpha$$

Для любых цепочек $\alpha, \beta, \gamma$ справедливо:

$$\alpha * (\beta * \gamma) = (\alpha * \beta) * \gamma$$

- Реверсом или обращением цепочки α называется $\alpha^R$ все элементы исходной цепочки записаны в ней в обратном порядке:

$\alpha = COH$

$a^R = HOC$ $\varepsilon^R = \varepsilon$

- n-ой степенью цепочки α называется конкатенация n цепочек α

  $\alpha^n = \alpha * \alpha *... * \alpha$

- Если $\alpha^0 = \varepsilon,\ \alpha^1 = \alpha,\ \alpha^2 = \alpha^1 * \alpha,\ \alpha^3 = \alpha^2 * \alpha$ и т. д
- Длина цепочки - это число, составляющих ее символов $|\alpha|$

  Пример: $\alpha = abbac$

  $|\alpha| = 5$ $|\varepsilon| = 0$

  Обозначим через $|\alpha|_s$ число символов S в цепочке α:

  $|\alpha|_b = 2$ из α = abbac

- Обозначим через $\Sigma^*$ - множество, содержащее все цепочки в алфавите Σ, включая пустую цепочку.

  Обозначим через $\Sigma^+$ - множество, содержащее все цепочки Σ, исключая пустую цепочку.

  $\Sigma^* = \Sigma^+ \cup \{0\}$

- Языком в алфавите Σ называется подмножество множества всех цепочек в заданном алфавите. Язык обозначим $\alpha \subseteq \Sigma^*$

Известны различные способы определения языков или описание языков.

Конечный язык можно описать простым перечислением его языков.

Поскольку формальный язык может быть и бесконечным, требуется механизмы, позволяющие конечным образом представить бесконечные языки.

Можно выделить 2 основных подхода для этого представления:

1) механизм распознавания;
2) механизм порождения или генерация;

Механизм распознавания по сути является процедурой специального вида, которая по заданной цепочке определяет - принадлежит она языку или нет.

Язык, определяемый распознавателем - это множество цепочек, которое он допускает.

Приведем примеры распознавателей:

1) Машина Тьюринга;

Язык, который можно задать с помощью МТ- это рекурсивно-перечислимый

2) Линейно-ограниченный автомат;

Представляет собой МТ, в которой лента не бесконечна, а ограничена длиной входного слова. Определяет контекстно-зависимые языки.

3) Автомат с магазинной памятью(МП автоматы);

В отличие от предыдущего автомата головка не может изменять входное слово и не может сдвигаться влево, зато существует дополнительная бесконечная память(магазин или стек), определяет контекстно-свободные языки.

Основной способ реализации механизма порождения - это использование порождающих грамматик, которые иногда называют грамматикой Хомского.

## 3. Языки и операции над языками

Рассмотрим простые примеры языков в некотором алфавите Σ:

1) $\emptyset$ (пустой язык)
2) $\{\varepsilon\}$ (из одного пустого символа)
3) $\{a\}\ a \in \Sigma$ (содержит один символ)

Суммой языков $L_1$ и $L_2$ будем называть язык, полученный в результате объединения множеств $L_1$ и $L_2$.

$$L_1 + L_2 = \{w | w \in L_1 \cup L_2\}$$

Произведением языков $L_1$ и $L_2$ будем называться язык, получаемый в результате конкатенации всех возможных слов первого и второго языков.

$$L_1 * L_2 = \{w_1 * w_2 | w_1 \in L_1, w_2 \in L_2\}$$

Пример:

$L_1 = \{a, bc\}$ $\qquad L_1 * L_2 = \{ab, aac, bcb, bcac\}$

$L_2 = \{b, ac\}$ $\qquad L_2 * L_1 = \{ba, bbc, aca, acbc\}$

Введем следующие обозначения:

$L^0 = \{\varepsilon\}$

$L' = L_1;\ L^2 = L^1 * L;\ L^3 = L^2 * L...$

$L^K = L^{K-1} * L$

Итерацией языка L называется язык, получаемый в результате бесконечного сложения числа языков.

$$\{\varepsilon\} + L^1 + L^2 +... + L^K +... = \sum_{k=0}^{\infty} L^K$$

Пример:

L={a}

$L^* = \{\varepsilon\} + \{a\} + \{aa\} +... = \{a^n | n \geq 0\} = a^*$

L={bc}

$L^* = \{\varepsilon\} + \{bc\} + \{bcbc\} +... = \{(bc)^n | n \geq 0\} = (bc)^*$

Таким образом, с помощью операций сложения, умножения и итерация некоторые языки можно выражать в виде формул через через более простые языки.

## 4. Регулярные языки и регулярные выражения

Очевидно, что для любого языка L справедливо

L+∅ = L

L*∅=∅

Значит $\emptyset^n = \emptyset$

$\emptyset^n = \{\varepsilon\}$

$\varepsilon^* = \{\varepsilon\}$

Пусть имеется некоторый алфавит $A = \{a_1, ... a_s\}$

Одноэлементные языки $a_1, ... a_s$, а также язык, содержащий $\{\varepsilon\}$ будем называть элементарными языками.

Регулярным языком называется такой язык, который можно получить из элементарных языков с помощью конечного числа операций сложения, умножения и итераций. Чтобы доказать регулярность какого-либо языка нужно записать его в виде, так называемого, регулярного выражения, т.е формулы, в которых конечное число раз используются элементарные языки и операции сложения, умножения и итераций.

Давайте рассмотрим пример:

$L = \{a, ab, abc\}$

$a + ab + abc = a(\varepsilon + b + bc) = a(\varepsilon + b(\varepsilon + c))$

(регулярное выр) (регулярное выр)

## 5. Способы задания конечных автоматов распознавателей

Конечный автомат — абстрактный автомат без выходного потока, число возможных состояний которого конечно. Результат работы автомата определяется по его конечному состоянию.

Конченый автомат может быть задан с помощью пяти параметров: $M = (Q, \Sigma, \delta, q_0, F)$

- Q - конечное множество состояний автомата;
- $q_0$- начальное состояние автомата ($q_0 \in Q$);
- F - множество заключительных(или допускающих) состояний, таких, что $F \subseteq Q$;
- Σ - допустимый входной алфавит(конечное множество допустимых входных символов), из которого формируются строки, считываемые автоматом;
- δ - заданное отображение множества $Q \times \Sigma$ во множество P(Q) подмножества Q: δ: $Q \times \Sigma \rightarrow P(Q)$(иногда δ называют функцией переходов автомата).

Способы описания:

1. Диаграмма состояний (или иногда граф переходов)
2. Таблица переходов — табличное представление функции δ.

Конечные автоматы подразделяются на детерминированные и недетерминированные.

Пусть $M = (Q, \Sigma, \delta, q_0, F)$ недетерминированный конечный автомат. Назовем M - детерминированным, если множество $\delta(q, a)$ содержит не более одного состояния для $\forall q \in Q, a \in \Sigma$

Если $\delta(q, a)$ содержит точно одно состояние, то M называем полностью детерминированным.

## 6. Связь между регулярными языками и конечными автоматами

Регулярные языки могут задаваться с помощью 3х основных способов:

- регулярные (праволинейные и леволинейные) грамматики;
- конечные автоматы (КА);

- регулярные множества (равно как и обозначающие их регулярные выражения).

Связь регулярных выражений и конечных автоматов

Регулярные выражения и недетерминированные конечные автоматы связаны между собой следующим образом:

- для любого регулярного языка, заданного регулярным выражением, можно построить конечный автомат, определяющий тот же язык;
- для любого регулярного языка, заданного конечным автоматом, можно получить регулярное выражение, определяющее тот же язык.

## 7. Теорема Клини

Классы детерминированных конечных автоматов и регулярных языков совпадают.
(Бля мне серьезно не понятно что хочет здесь от нас Романенко, потому что доказательство этой штуки существует, но точно не в её лекциях и в принципе на них не похоже)

## 8. Удаление Е-переходов

Пусть $M = (Q, \Sigma, \delta, q_0, F)$ недетерминированный конечный автомат. Назовем М - детерминированным, если множество $\delta(q, a)$ содержит не более одного состояния для $\forall q \in Q, a \in \Sigma$

Если $\delta(q, a)$ содержит точно одно состояние, то М называем полностью детерминированным.

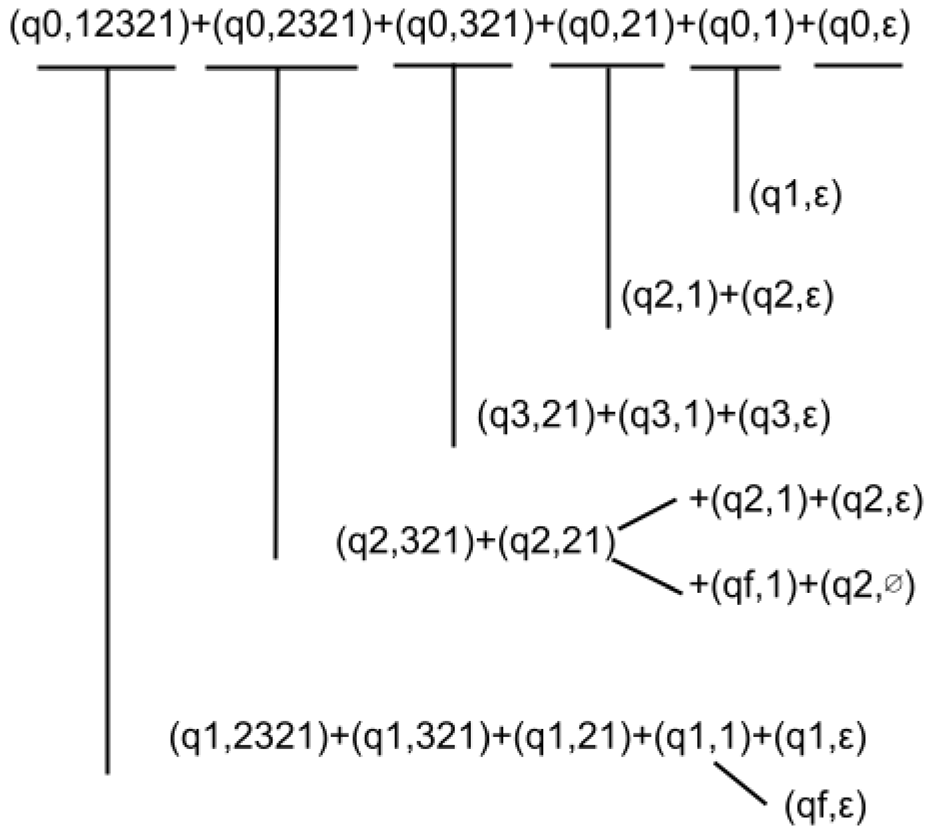
Одним из наиболее важных результатов теории автоматов состоит в том, что класс языков определенных недетерминированным конечным автоматом совпадает с классом языков детерминированных конечных автоматов.

Пример:

Пусть дан автомат

| | 1 | 2 | 3 |
|---|---|---|---|
| q0 | q0,q1 | q0,q2 | q0,q3 |
| q1 | q1,qf | q1 | q1 |
| q2 | q2 | q2,qf | q2 |
| q3 | q3 | q3 | q3,qf |
| qf | Ø | Ø | Ø |

w = 12321




Строим по недетерминированному автомату ⇒ детерминированный
Переделаем начальные состояния:

| | 1 | 2 | 3 |
|---|---|---|---|
| A={q0} | B | C | D |
| B={q0,q1} | E | F | G |
| C={q0,q2} | F | H | I |
| D={q0,q3} | G | I | J |
| E={q0,q1,qf} | E | F | I |
| F={q0,q1,q2} | K | K | L |
| G={q0,q1,q3} | M | L | M |
| H={q0,q2,qf} | F | H | I |
| I={q0,q2,q3} | L | N | N |
| J={q0,q3,qf} | G | I | J |
| K={q0,q1,q2,qf} | K | K | L |
| L={q0,q1,q2,q3} | O | O | O |
| M={q0,q1,q3,qf} | M | L | M |
| N ={q0,q2,q3,qf} | L | N | N |

| | | | |
|---|---|---|---|
| O={q0,q1,q2,q3,qf} | O | O | O |

## 9. Недетерминированные конечные автоматы распознаватели. Теорема о детерминизации

Теорема: Для любого недетерминированного автомата распознавателя существует эквивалентный ему детерминированный конечный автомат.

Доказательство проведем конструктивно.

Сначала приведем недетерминированный конечный автомат с ε-переходами к детерминированному автомату без ε-переходов.

$M = (\Sigma, Q_\varepsilon, \delta_\varepsilon, q_0, F_\varepsilon)$ - с ε-переходами

Приведем его:

$M = (\Sigma, Q, \delta, q_0, F)$ - с без ε-переходами

ε - замыканием состояние $q \in Q$ обозначим через ≊(q) и назовем множество всех состояний, которые достижимы из q без подачи входного сигнала.

Множеству ≊(q) принадлежит состояние q и все состояния, которые достижимы из q по ε-переходам.

Множеством состояний Q является ε-замыкание множества $Q_\varepsilon$.

Множеством начальных состояний автомата M является $S_0 = U\approxeq(q_0)$.

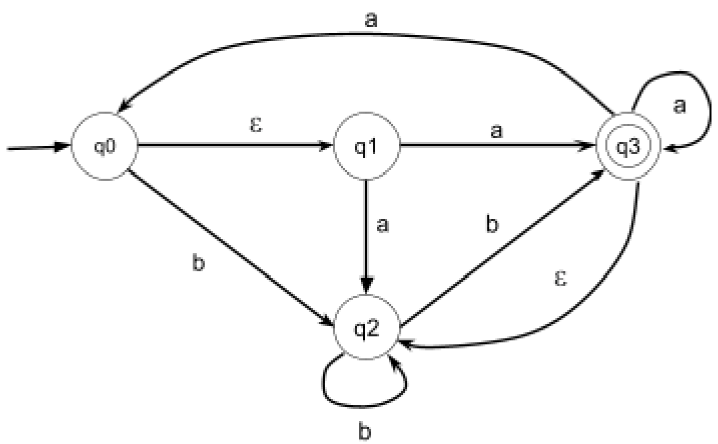
Множеством заключительных состояний F является такие ε-замыкания множества состояний $F_\varepsilon$, и все состояния, которые без подачи сигнала достигают заключительных состояний.

Функция переходов δ автомата M определяется как:

$$\delta(\approxeq(q), a) = \cup S \in \delta_\varepsilon(\approxeq(q), a)$$

Иными словами под воздействием входного сигнала a автомат M переходит из ε -замыканий состояния q в ε-замыкания всех тех состояний S, в которые $M_\varepsilon$ переходит под воздействием a из всех состояний, достижимых q, под воздействием ε.

Из недетерминированного автомата с ε-переходами ⇒ недетерминированный автомат без ε-переходов.



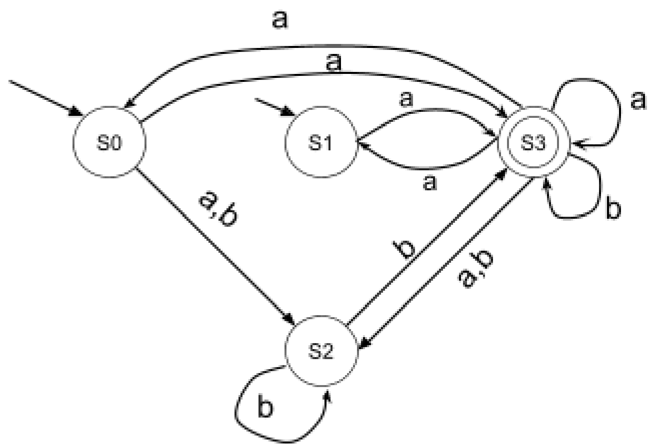

$S_0\approxeq(q_0) = \{q_0, q_1\}$

$S_1\approxeq(q_1) = \{q_1\}$

$S_2\approxeq(q_2) = \{q_2\}$

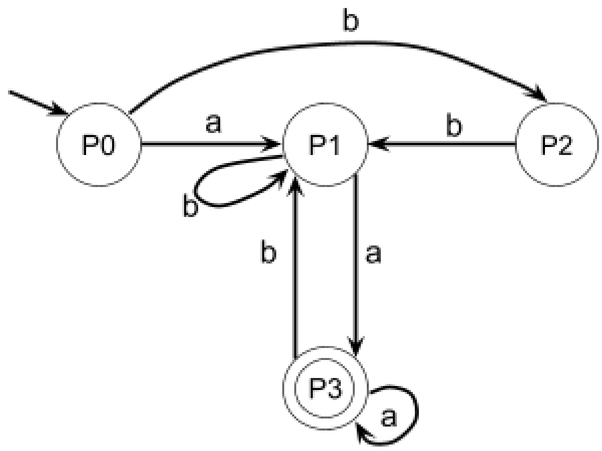
$S_3\approxeq(q_3) = \{q_3, q_2\}$

| | a | b |
|---|---|---|
| $\rightarrow S_0 = \{q_0, q_1\}$ | $S_3, S_2$ | $S_2$ |
| $\rightarrow S_1 = \{q_1\}$ | $S_3, S_2$ | $\emptyset$ |
| $S_2 = \{q_2\}$ | $\emptyset$ | $S_2, S_3$ |
| $S_3 = \{q_3, q_2\}$ | $S_1, S_2, S_3, S_0$ | $S_2, S_3$ |




Начальным детерминированным автоматом будут все начальные состояние недетерминированного автомата без ε-переходов.

| | a | b |
|---|---|---|
| $\rightarrow P_0 = \{S_0, S_1\}$ | $P_1$ | $P_2$ |
| $P_1 = \{S_2, S_3\}$ | $P_3$ | $P_1$ |
| $P_2 = \{S_2\}$ | $\emptyset$ | $P_1$ |
| $P_3 = \{S_0, S_1, S_2, S_3\}$ | $P_3$ | $P_1$ |




## 10. Минимизация конечных автоматов распознавателей

По данному конечному автомату M можно найти наименьший эквивалентный ему автомат, исключив все недостижимые состояния и затем склеив лишние состояния. Лишние состояния определяются с помощью разбиения множества всех достижимых состояний на классы эквивалентных состояний так, что каждый класс содержит не различные состояния.

Потом из каждого класса берется один представитель в качестве сокращенного состояния.

В силу Теоремы о детерминизации можно считать, что исходный автомат детерминирован. Будем также предполагать, что в исходном конечном автомате нет состояний недостижимых из начальной вершины. На множестве состояний M зададим семейство отношений эквивалентности следующим образом:

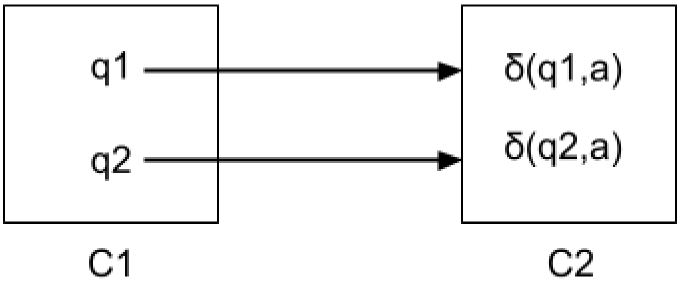
- О-эквивалентность

Два состояния $q_1$и $q_2$ состоят в О-эквивалентности ($q_1 \overset{0}{\equiv} q_2$) тогда и только тогда, когда они оба являются заключительными состояниями или оба не являются заключительными состояниями.

- K-эквивалентность

При $k \geq 1$ $q_1 \overset{k}{\equiv} q_2$ тогда и только тогда, когда два состояния $q_1 \overset{k-1}{\equiv} q_2$.

Чтобы понять смысл К-эквивалентности нарисуем схему:




будем предполагать, что $q_1 \equiv^{k-1} q_2$ и принадлежат одном и тому же классу $C_1$. Эти два состояния будут К-эквивалентны, если состояния $\delta(q_1, a)$ и $\delta(q_2, a)$ являются также К-1-эквивалентны содержащимся в другом классе $C_2$, причем классы $C_1$ и $C_2$ могут совпадать.

Минимизация конечного автомата состоит в последующем измельчении Q, разбиением Q на классы эквивалентности до тех пор, пока не получится разбиение, которое уже нельзя измельчить ($\equiv^{k-1} = \equiv^{k}$). Тогда минимальный конечный автомат: $M' = (\Sigma, Q', \delta', q_0', F')$

- $Q'$- это новое множество состояний которое соответствует последнему классу эквивалентности;
- $q_0' = [q_0]$ - класс эквивалентности, который содержит состояние $q_0$;
- $F' = \{[q_0] | q_0 \in F\}$
- $\delta'([q], a) = \delta(q, a)$

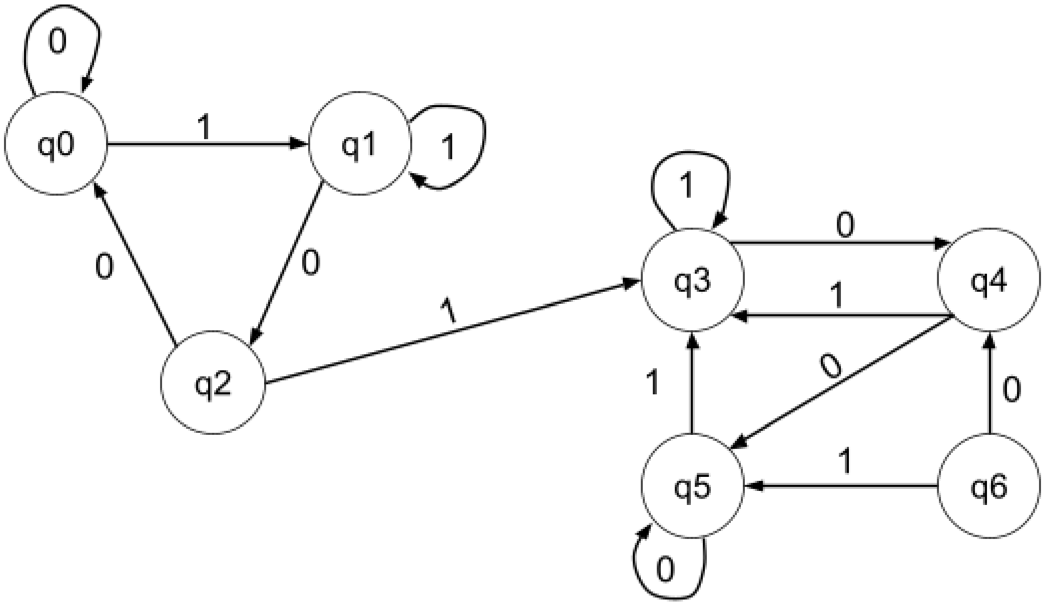
Теорема:

Для произвольного конечного автомата может быть построен эквивалентный ему минимальный автомат.

Из этой теоремы вытекает алгоритм построения минимального автомата:

1) Если автомат недетерминированный ⇒ детерминируем;
2) Если автомат содержит недостижимые из начального состояния состояния, то удаляем их вместе с инцидентными дугами;
3) К полученному автомату применяем алгоритм разбиения на классы эквивалентных состояний;
4) По данным класса строим минимальный автомат.

Пример:




$q_6$ − не достижима

1) Автомат детерминированный
2) $q_6$ - не достижима ⇒ удаляем
3) $\equiv^o\{q_3, q_4, q_5\}\{q_0, q_1, q_2\}$

$(C_1) \qquad (C_2)$

$\equiv^1\{q_3, q_4, q_5\}\{q_0, q_1\}, \{q_2\}$

$(q_3, 0) \rightarrow q_4 \qquad (q_3, 1) \rightarrow q_3$

$(q_4, 0) \rightarrow q_5 \qquad (q_4, 1) \rightarrow q_3$

$(q_5, 0) \rightarrow q_5 \qquad (q_5, 1) \rightarrow q_3$ , где $(q_4, q_5, q_3) \in C_1$

⇒класс $C_1$ не разбивается, тк все состояния класса относятся к одному и тому же классу.

$(q_0, 0) \rightarrow q_0 \qquad (q_0, 1) \rightarrow q_1$

$(q_1, 0) \rightarrow q_2 \qquad (q_1, 1) \rightarrow q_1$

$(q_2, 0) \rightarrow q_0 \qquad (q_2, 1) \rightarrow q_3$ , где $(q_3) \in C_1$, $(q_0, q_1, q_2) \in C_2$

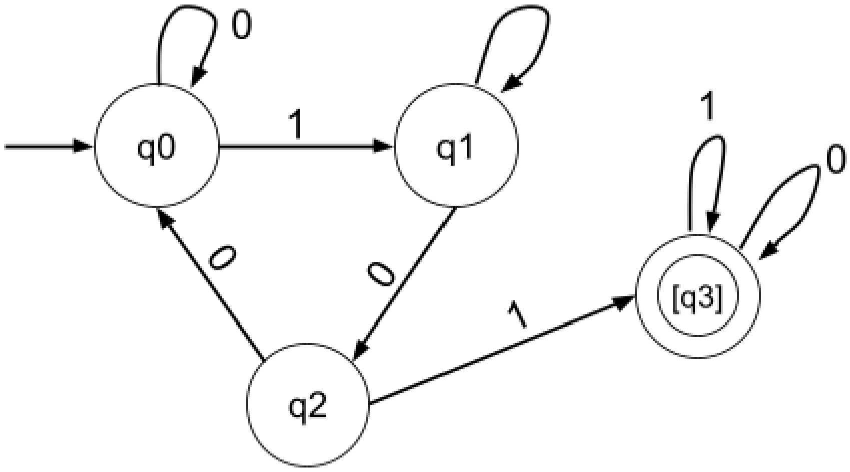
⇒класс $C_2$ разбивается, тк $q_2$ переходит в два разных класса

$\equiv^2\{q_3, q_4, q_5\}\{q_0\}\{q_1\}\{q_2\}$

$\equiv^3\{q_3, q_4, q_5\}\{q_0\}\{q_1\}\{q_2\}$

$\equiv^2 = \equiv^3$ совпадает следовательно переходим к шагу 4)

4) $\equiv^3 = \equiv^2 \quad [q_3] = \{q_3, q_4, q_5\}$ - (взяли любое, обычно первое значение из класса)




Крайние случаи:

1) Если разбито на $\{q_i\}$, то он уже минимальный
2) Если начальная и конечная вершина одна и та же, то это один класс и он не разбивается, а циклится сам на себе.

## 11. Автоматы преобразователи

Конечным автоматом преобразователем называется автомат $M = (\varepsilon, Q, q_0, \delta, \lambda, F)$

λ - функция выходов

δ - функция переходов

F - множество выходных символов

Мы объединим 2 таблицы: функцию переходов и функцию выходов($\delta \times \lambda$).

| $\delta \times \lambda$ | a | b |
|---|---|---|
| $q_0$ | $q_1\|x$ | $q_2\|y$ |
| $q_1$ | $q_2\|y$ | $q_0\|x$ |
| $q_2$ | $q_1\|x$ | $q_1\|y$ |
| $q_3$ | $q_3\|x$ | $q_2\|y$ |

Пусть W = abba

$(q_0, abba) + (xq_1, bba) + (xxq_0 ba) + (xxyq_2 a) + (xxyxq_1 \varepsilon)$

- Два конечных автомата:

$M_1 = (\Sigma_1, Q_1, q_{01}, \delta_1, \lambda_1, F_1)$

$M_2 = (\Sigma_2, Q_2, q_{02}, \delta_2, \lambda_2, F_2)$

называются эквивалентными, если выполняются два условия:

1) Их входные алфавиты совпадают;
2) Реализуемые ими отображения совпадают.

- Прямым произведением автоматов $M_1$ и $M_2$ с одинаковым входным алфавитом называется автомат:

$M_1 \times M_2 = (\Sigma, Q_{1\times 2}, (q_{01} \times q_{02}), \delta_{1\times 2}, \lambda_{1\times 2}, F_{1\times 2})$, где для $\forall q_1$ и $q_2$ выполняется $\delta_{1\times 2}((q_1, q_2), a) = (\delta_1, q_1) \times (\delta_2, q_2) = (\delta_1(q_1, a), \delta_2(q_2, a))$

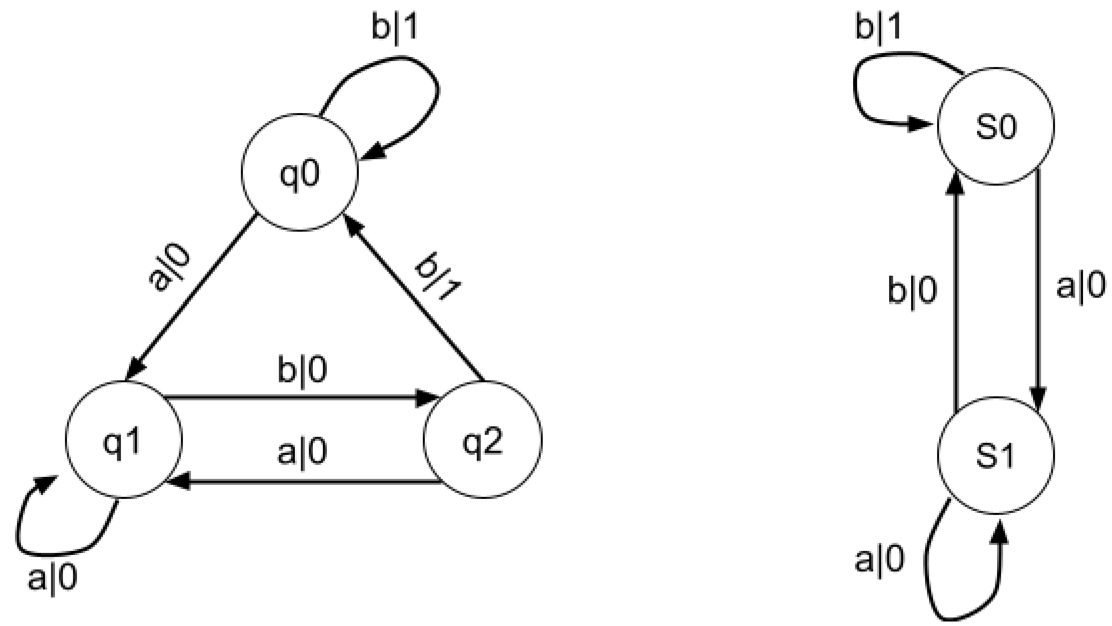
- Произведение автоматов - два рядом стоящих невзаимодействующих, но синхронно работающих автомата.

## 12. Эквивалентность автоматов преобразователей. Теорема Мура

Два конечных автомата $M_1$ и $M_2$ с одинаковыми выходными алфавитами являются эквивалентными тогда и только тогда, когда для любого достижимого состояния $(q_1, q_2)$ в их прямом произведении $M_1 \times M_2$ справедливо равенство:

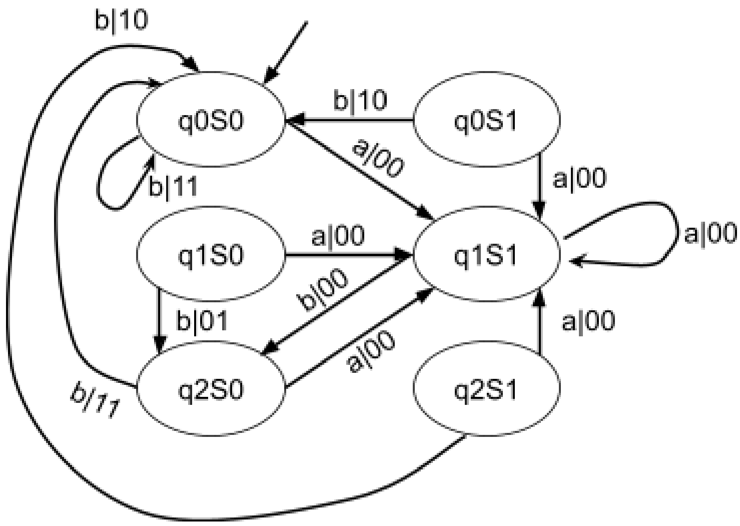
$\lambda_1(q_1, a) = \lambda_2(q_2, a)$ для $\forall a$

Пример:




Являются ли автоматы эквивалентными?

Решение:




1)
2) Удаляем $q_1S_0$, $q_2S_1$, $q_0S_1$,тк они недостижимы;
3) Выходы 1 и 2 автоматов должны совпадать. В нашем случае совпадают следовательно автоматы эквивалентны (либо 00, либо 11)

## 13. Минимизация конечного автомата преобразователя

Пусть имеется автомат преобразователь M. Рассмотрим алгоритм минимизации Мили. На каждом шаге алгоритма будем разбивать множество Q на классы, причем разбиение на каждом наше алгоритмы будет получатся расщеплением некоторых классов предыдущих разбиений.

- 0-эквивалентность

Два состояния $q_1$ и $q_2$ отнесем в один класс $C_1$, если для каждого входного символа совпадают его выходные символы:

$\lambda(q_1, a) = \lambda(q_2, a)$

- K-эквивалентность

Два состояния $q_1$ и $q_2$ из одного класса $C_{Kj}$ отнесем в один класс $C_{K+1j}$, если для каждого входного символа осуществляется переход из состояний $q_1$ и $q_2$ в состояния принадлежащие одному и тому же классу $C_{KS}$, те $\delta(q_1, a)$ и $\delta(q_2, a)$ $\in$ одному и тому же классу.

Если K+1 не изменяет разбиение, то мы получим искомое разбиение на классы эквивалентных состояний.

Пример:

Пусть дан автомат:

| q | 0 | 1 |
|---|---|---|
| 1 | 3\|a | 2\|b |
| 2 | 1\|b | 6\|a |
| 3 | 5\|a | 2\|b |
| 4 | 5\|b | 6\|a |
| 5 | 5\|a | 4\|b |
| 6 | 2\|a | 2\|b |

Решение:

$\equiv^0 \{q_1, q_3, q_5, q_6\}\{q_2, q_4\}$

$(C_{11}) \qquad (C_{12})$

$\equiv^1 \{q_1, q_3, q_5\}\{q_6\}\{q_2, q_4\}$

$\equiv^2 \{q_1, q_3, q_5\}\{q_6\}\{q_2, q_4\}$

$[q_1] \qquad [q_2]$

$\equiv^1 = \equiv^2$

| | 0 | 1 |
|---|---|---|
| $[q_1]$ | $[q_1]\|a$ | $[q \ \|b]$ |
| $[q_2]$ | $[q_1]\|$b | $[q_6\|a]$ |
| $q_6$ | $[q_2]\|a$ | $[q_1]\|b$ |

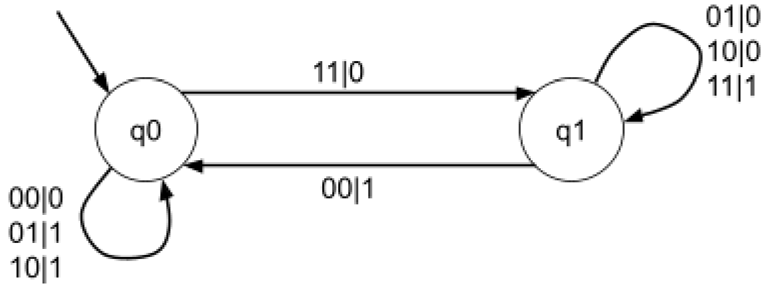
Пример:

Построить автомат, который осуществляет сложение двух двоичных чисел. Двоичные числа обрабатываются начиная с младших разрядов. При этом подразумевается, что вслед за старшим разрядом идет необходимое количество нулей.

$$\begin{array}{r} 1011 \\ +\underline{0110} \\ 10001 \end{array}$$

Входной алфавит $\Sigma = \{00, 01, 10, 11\}$

Выходной алфавит $F = \{0, 1\}$

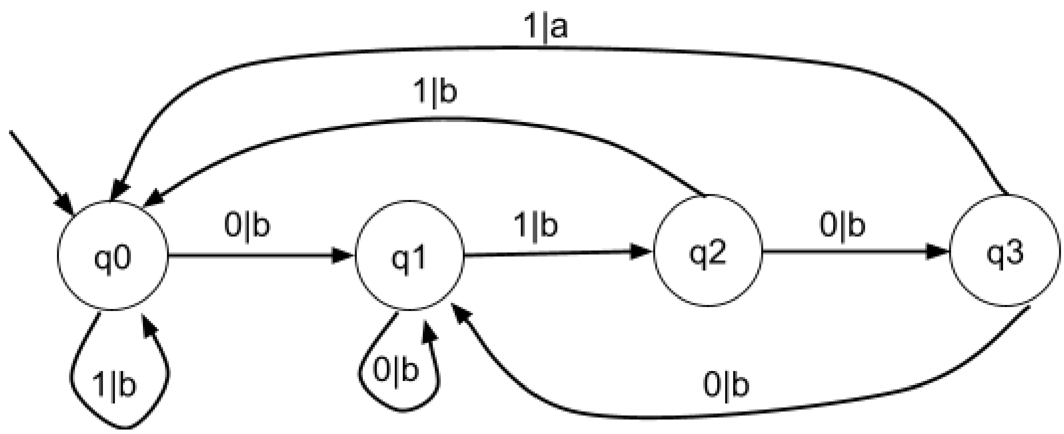



Пример:

Построить автомат с входным алфавитом Σ и выходным алфавитом $F$, порождающих на выход ‘a’ тогда и только тогда, когда последние 4 входных символа равны 0101, в остальных случаях ‘b’.

Входной алфавит $\Sigma = \{0, 1\}$

Выходной алфавит $F = \{a, b\}$




## 14. Грамматики

Порождающей грамматикой G называется четверка, где

$G =< T, N, P, S >$

- N - алфавит нетерминальных символов(не терминалов)
- T - алфавит терминальных символов (терминалы)
- P - правило вывода(запись:$\alpha \rightarrow \beta$)
- S - начальный символ грамматики($S \in N$)

Для записи правил вида с одинаковыми левыми частями($\alpha \to \beta_1,..., \alpha \to \beta_n$) используется запись:

$\alpha \to \beta_1|\beta_2|...|\beta_n$

Пример:

$G = < \{0, 1\}, \{A, S\}, P, S >$

P: $S \to 0A1$

$0A \to 00A1$

$A \to \varepsilon$

$S \to 0A1 \to 0\varepsilon 1 \to 01$

$S \to 0A1 \to 00A11 \to 00\varepsilon 11 \to 0011$

$S \to 0A1 \to 00A11 \to 000A111 \to 000111$

Цепочка $\beta \in (T \cup N)^*$ непосредственно выводима из цепочки $\alpha \in (T \cup N)^+$ в грамматике G, если $\alpha = \xi_1\gamma\xi_2$

$$\beta = \xi_1\delta\xi_2, \text{ где } \xi_1, \xi_2, \gamma \in (T \cup N)^*, \ \delta(T \cup N)^+$$

и правило вывода $\gamma \to \delta$ содержится в P.

Цепочка $\beta$ выводима из цепочки $\alpha$, если существуют такие $\gamma_1, \gamma_2,..., \gamma_n$, что $\alpha = \gamma_0 \to \gamma_1 \to \gamma_2 \to ... \to \gamma_n = \beta$

Языком, порождающей грамматикой G, называется $L(G) = \{\beta^* | S \Rightarrow \beta\}$

Пример: $L(G) = \{0^n, 1^n | n \geq 1\}$

## 15. Классификация грамматик по Хомскому

Определим с помощью ограничений на вид грамматики(на вид правил вывода) 4 типа грамматик:

- 0-тип

Любая порождающая грамматика является типом 0, те никаких ограничений на вид правил вывода не накладывается.

Класс языков типа 0 совпадает с классом рекурсивно-перечислимых языков;

- 1-тип

- Грамматика G называется неукорачивающей, если правая часть каждого правила вывода не короче левой части:

  $\alpha \to \beta, |\alpha| \leq |\beta|$

  В виде исключения в неукорачивающей грамматике допускается правило $S \to \varepsilon$

- Грамматика G называется контекстно-зависимой(КЗ), если каждое правила P имеет вид:

  $\alpha \to \beta$

  $\alpha = \xi_1 A \xi_2$

$\beta = \xi_1 \delta \xi_2$

$\xi_1, \xi_2 \in (T \cup N)^*$

$A \in N$

$\delta \in (T \cup N)^+$

В виде исключения в конце грамматики допускается наличие правила $S \to \varepsilon$ при условии, что S не встречается в правых частях правил.

Замечание:

Из определений следует, что если язык, порождаемый грамматикой КЗ или неукорачивающей грамматики G, содержит пустую цепочку, то это цепочка выводится за один шаг с помощью правил вида $S \to \varepsilon$. Других выводов для ε не существует.

Язык порождаемый КЗ-грамматикой называется КЗ-языком.

Утверждение:

Пусть L - формальный язык

Следующее утверждение эквивалентны:

1) Существует КЗ-грамматика такая, что $L = L(G_1)$
2) Существует неукорачивающая грамматика такая, что $L = L(G_2)$

- 2 - тип

  Грамматика G называется контекстно-свободной(КС), если каждое правило имеет вид:

  $A \to \beta$, где $A \in N,\ \beta \in (T \cup N)^*$

  Язык порождаемый КС-грамматикой называется КС-языком.

- 3 - тип
  - Грамматика G называется праволинейной(пл), если каждое правило имеет вид

    $A \to wB$

    $A \to w$, где $A, B \in N,\ w \in T^*$
  - Грамматика G называется леволинейной(лл), если каждое правило имеет вид

    $A \to Bw$

    $A \to w$, где $A, B \in N,\ w \in T^*$

Утверждение:

Пусть L-формальный язык

Языки пл-грамматик и лл-грамматик совпадают. Тогда пл и лл грамматики определяют регулярные языки.

- Автоматной грамматикой называется грамматика вида:

  $A \to aB$ $\qquad\qquad$ $A \to Ba$

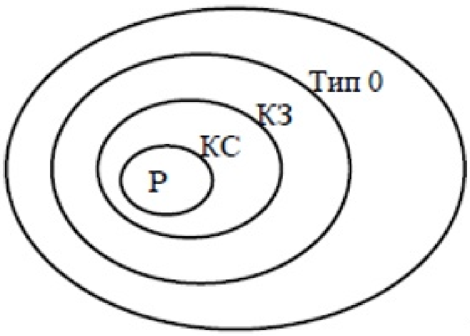
  $A \to a$ $\qquad$ или $\qquad$ $A \to a$

  где $A, B \in N,\ a \in T$

Утверждения:

Следующие соотношения справедливы:

1) Любая регулярная грамматика является контекстно-свободной грамматикой;
2) Любая неукорачиваемая контекстно-свободная грамматика является контекстно-зависимой;
3) Любая неукормачиваемая грамматика является грамматикой типа 0.




Р – регулярная грамматика;
КС – контекстно-свободная грамматика;
КЗ – контекстно-зависимая грамматика;
Тип 0 – грамматика типа 0.

## 16. КС - грамматики. Примеры

2 - тип

Грамматика G называется контекстно-свободной(КС), если каждое правило имеет вид:

$A \rightarrow \beta$, где $A \in N,\ \beta \in (T \cup N)^*$

Пример:

$G = < \{0, 1\}, \{A, S\}, P, S >$

P: $S \rightarrow 0A1$ ⇐КС-грамматика

$A \rightarrow 0A0$

$A \rightarrow \varepsilon$

$S \rightarrow 0A1 \rightarrow 01$

$S \rightarrow 0A1 \rightarrow 00A01 \rightarrow 0001$

$S \rightarrow 0A1 \rightarrow 00A01 \rightarrow 000A001 \rightarrow 000001$

$L = \{0^{2n+1}1 | n \geq 0\}$

Пример:

$G = < \{a, b\}, \{S\}, P, S >$

P: $S \rightarrow aSb|\varepsilon$ ⇐КС-грамматика

$S \rightarrow \varepsilon$

$S \rightarrow aSb \rightarrow ab$

$S \rightarrow aSb \rightarrow aaSbb \rightarrow aabb$

$L = \{a^n, b^n | n \geq 0\}$

## 17. Деревья вывода. Левые и правые выводы

## 18. Построение конечного автомата по регулярной грамматике

На вход предполагаем, что подается грамматика G = (T,N,P,S), а на выход $M = (\Sigma, Q, \delta, q_0, F)$

Шаг 1: Пополним грамматику правилом, где $A \to aN'$: $A = N, a \in T, N' \in N$ для каждого правила, если детерминал следует терминалу.

Если в грамматике нет соответствующего ему правила:$A \to a;\ A \to aB;\ A, B \in N$

Шаг 2: Начальный символ S примем за начальное состояние автомата M

$S \to q_0$

Из нетерминалов образуем множество состояний Q.

$S = q_0,\ Q = N \cup \{N'\}$

А из терминалов алфавит входных символов: $\Sigma = T$

Шаг 3:

Каждое правило $A \to aB$ преобразовать в функцию переходов $\delta(A, a) = B$, где A и B - нетерминальналы, a - терминал

Шаг 4:

Во множество заключительных состояний включить все вершины, помеченные символами $B \in N$ из правила вида $A \to aB$, для которых имеются соответствующие правила $A \to a$.

Шаг 5:

Если в грамматике имеется правило $S \to \varepsilon$, то помещаем S во множество финальных состояний.

Шаг 6:

Если мы получим недетерминальный конечный автомат, то мы его детерминируем.

Пример:

$S \to aA|bB|\varepsilon$ $\qquad T = \{a, b, c, \perp\}$

$B \to bB|cC$ $\qquad N = \{A, B, C, S\}$ +N’

$A \to aA|cC$

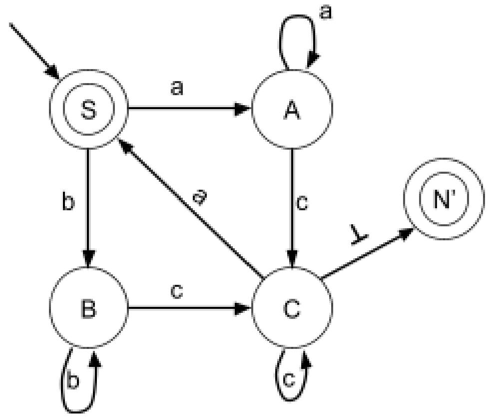
$C \to cC|aS|\perp$

$C \to \perp N'$ - добавленное

Строим таблицу:

| | a | b | c | ⊥ |
|---|---|---|---|---|
| S | A | B | Ø | Ø |
| A | A | Ø | C | Ø |
| B | Ø | B | C | Ø |
| C | S | Ø | C | N’ |
| N’ | Ø | Ø | Ø | Ø |

Финальное состояние: S, N’.




19. **Построение МП-автомата по КС - грамматике**
20. **Построение  расширенного МП - автомата по КС - грамматике**