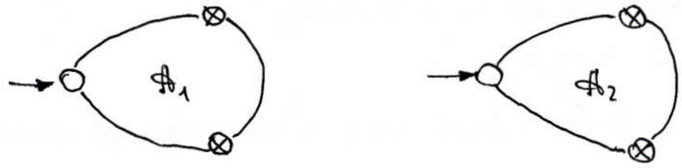
Опр. Язык $L \subseteq \Sigma^*$ наз-ся автоматным, если существует НКА с алфавитом входных символов Σ, задающий язык L.

Теорема. Автоматные языки замкнуты относительно объединения, произведения и операции * (замыкания Клини), т.е. объединение и произведение автоматных языков - это автоматные языки, замыкание Клини автоматного языка - это автоматный язык.

Идея док-ва. Мы не будем формально доказывать это утверждение. Приведем лишь идею.

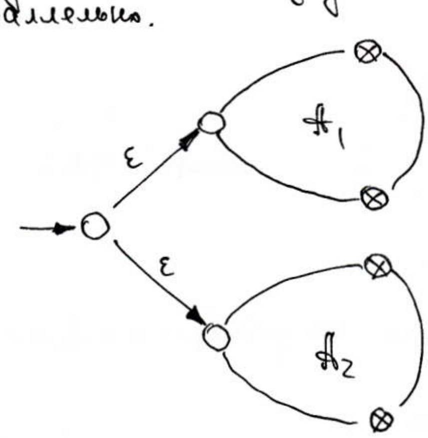
Пусть L_1, L_2 - это автоматные языки. Значит существует НКА $\mathcal{A}_1$, который задаёт язык L_1 и НКА $\mathcal{A}_2$, который задаёт язык L_2.



$L(\mathcal{A}_1) = L_1$ $\qquad$ $L(\mathcal{A}_2) = L_2$.

Комбинируя автоматы $\mathcal{A}_1$ и $\mathcal{A}_2$, построим автоматы, задающие объединение и произведение языков L_1 и L_2.

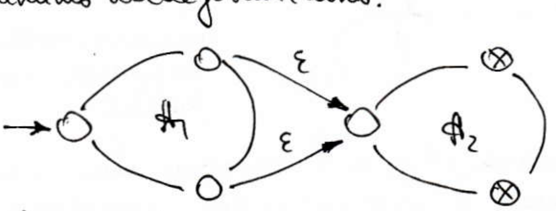
Для того, чтобы получить автомат, задающий объединение языков, необходимо автоматы $\mathcal{A}_1$ и $\mathcal{A}_2$ соединить параллельно.



- автомат, задающий $L_1 \cup L_2$.

Можно по пустому слову пойти в верхний автомат. Тогда мы будем принимать только слова из языка L_1. Можно по пустому слову пойти в нижний автомат. Тогда мы будем принимать только слова из языка L_2.

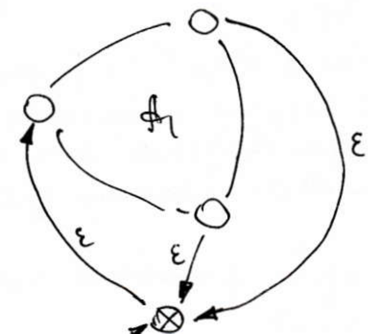
Для того, чтобы получить автомат, задающий произведение языков L_1 и L_2, необходимо автоматы $\mathcal{A}_1$ и $\mathcal{A}_2$ соединить последовательно.



- автомат, задающий $L_1 L_2$

Чтобы прочитать слово из $L_1 L_2$ нужно прочитать первую часть слова из языка L_1 в автомате $\mathcal{A}_1$, а затем перепрыгнуть во второй автомат и прочитать вторую часть слова из языка L_2 в автомате $\mathcal{A}_2$.

Гораздо сложнее устроен автомат, задающий L_1^* (замыкание L_1).



Берем автомат A_1, задающий язык L_1. Добавляем новое состояние. Из нового состояния есть переход в начальное состояние автомата* по пустому слову, из каждого заключительного состояния автомата A_1 есть переход по пустому слову в новое состояние. Сделаем в ~~автомате~~ результирующем автомате одно начальное и одно заключительное состояние. Причем и тем и другим будет новое состояние.

Почему этот автомат задаёт L_1^*. Потому что, если слово из L_1^*, т.е. это слово является произведением нескольких слов из L_1, то мы из начального состояния попадаем в автомат A_1, читаем слово из L_1, затем слово читаем следующее слово из L_1 и т.д. ■

Опр. Детерминированный конечный автомат A наз-ся полным, если из любого состояния по любой букве алфавита автомата существует переход

$$\forall q\in Q\ \forall a\in\Sigma\ \exists q'\in Q\quad \delta(q,a)=q'.$$

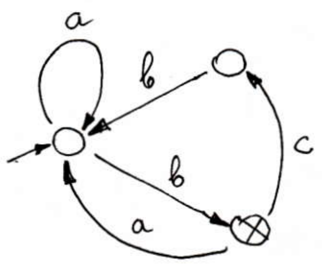
Полный автомат любое непустое слово в алфавите автомата дочитывает слово до конца.

Утв. Любой ДКА эквивалентен полному ДКА.

Без док-ва.

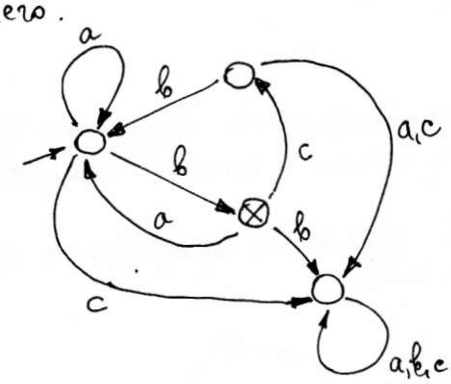
Проиллюстрируем способ преобразования ДКА в полный ДКА.

Пример. Рассмотрим ДКА



неполный ДКА

Автомат не полный. В нем нет всех стрелочек. Мы добавляем новое состояние и все стрелочки, которых нет, отправляем в него.



полный ДКА

для того, чтобы новое состояние не нарушило полноту, его надо зациклить на себя

Все принимающие пути в полном автомате есть и в исходном автомате. Все слова, которые не принимались раньше, они сбрасываются в новое состояние, которое играет роль стокового.

Теорема. Автоматные языки замкнуты относительно дополнения и пересечения

Док-во. Покажем, что дополнение автоматного языка - это автоматный язык.

Пусть $L\subseteq\Sigma^*$ - автоматный язык. Хотим показать, что $\overline{L}=\Sigma^*\setminus L$ - автоматный язык. Пусть язык L задан автоматом $A=(Q,\Sigma,\delta,q_0,F)$. Не теряя общности, мы можем считать, что этот автомат является полным ДКА.

Рассмотрим новый автомат $\overline{A}=(Q,\Sigma,\delta,q_0,Q\setminus F)$. Этот автомат получается из исходного ~~автомата~~ автомата A так. Те состояния, которые в автомате A заключительные делаем не заключительными, а те состояния, которые были не заключительными делаем заключительными. Другими словами, в автомате A мы меняем местами заключительные состояния и не заключительные.

Утверждается, что автомат $\overline{A}$ задаёт дополнение языка L

$$L(\overline{A})=\overline{L}$$

Почему это так? Автомат $\mathcal{A}$ полный. Значит все слова в алфавите Σ автомат дочитывает до конца. Какие-то из них приводят в заключительные состояния, а какие-то в незаключительные. Мы поменяли местами заключительные и незаключительные состояния. Значит те слова, которые раньше принимались, в новом автомате не принимаются, а те слова, которые раньше не принимались, в новом автомате принимаются. Поэтому язык нового автомата – это дополнение языка старого автомата.

Остается разобраться с пересечением.

Пусть $L_1, L_2 \subseteq \Sigma^*$ – это автоматные языки, которые задаются ПДКА $\mathcal{A}_1$ и $\mathcal{A}_2$. Как задать пересечение языков L_1 и L_2 автоматом? По закону де Моргана

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}.$$

пересечение можно выразить через объединение и дополнение. Мы уже знаем как задать объединение языков L_1, L_2 и дополнение языка L_1 и языка L_2 автоматами. Используя известные конструкции для объединения и дополнения, можно построить автомат, задающий пересечение

Автоматные языки – это языки, которые могут быть заданы автоматами. Оказывается, что автоматные языки – это в точности языки, которые могут быть заданы регулярными выражениями.

<u>Опр.</u> Пусть Σ - алфавит. Регулярные выражения над алфавитом Σ - это выражения, которые получаются с помощью следующих правил:

1. Константы $0, 1$ являются регулярными выражениями;
2. Любой символ $a \in \Sigma$ алфавита является регулярным выражением;
3. Если α, β – это регулярные выражения, то выражения α^*, $(\alpha+\beta)$, $(\alpha\cdot\beta)$ являются регулярными;
4. Регулярные выражения – это те и только те выражения, которые могут быть получены за конечное число применений пунктов 1 – 3.

Договоримся о приоритете операций. Сначала выполняется $*$, затем $\cdot$, а потом $+$.

Каждое регулярное выражение задает язык. Если α – регулярное выражение, то язык, задаваемый регулярным выражением α, обозначается так $L(\alpha)$:

α - регулярное выражение $\Rightarrow$ $L(\alpha)$ - язык, задаваемый регулярным выражением α

Язык, задаваемый регулярным выражением, определяется индуктивно.

1. $L(0) = \varnothing$ (язык, задаваемый константой 0, - пустой язык, т.е. язык без слов);
2. $L(1) = \{\varepsilon\}$ (язык, задаваемый константой 1, состоит из одного слова - пустого слова);
3. $L(a) = \{a\}$ (язык, задаваемый буквой алфавита, - это язык, состоящий из этой буквы);
4. $L(\alpha^*) = (L(\alpha))^*$ (язык, задаваемый звездочкой регулярного выр-ния α, - это замыкание Клини языка, задаваемого регулярным выражением α);
5. $L(\alpha\cdot\beta) = L(\alpha)\cdot L(\beta)$ (язык, задаваемый произведением регулярных выражений α и β, - это произведение языков, задаваемых регулярными выражениями α и β).
6. $L(\alpha+\beta) = L(\alpha) \cup L(\beta)$ (язык, задаваемый суммой регулярных выражений α и β, - это объединение языков, задаваемых регулярными выражениями α и β).

<u>Опр.</u> Язык наз-ся регулярным, если он может быть задан регулярным выражением.

<u>Теорема</u> (Клини, 1956). Классы регулярных и автоматных языков совпадают, т.е. все что можно задать регулярным выражением можно задать автоматом; все что можно задать автоматом можно задать регулярным выражением.

<u>Док-во.</u>

$$\boxed{\text{Регулярные языки} \subseteq \text{Автоматные языки}}$$

Это включение следует из замкнутости автоматных языков относительно объединения, произведения и замыкания Клини. Нам известны конструкции автоматов, задающие объединение, произведение автоматных языков, а также замыкание Клини автоматного языка. Используя эти конструкции мы всегда по регулярному выражению задающему [illegible] задающий тот же самый язык.

Докажем обратное включение.

$\boxed{\text{Автоматные языки} \subseteq \text{Регулярные языки}}$

Определим понятие регулярного автомата.

Регулярный автомат — это автомат, в котором переходы осуществляются по регулярным выражениям над алфавитом автомата. Регулярный автомат — это автомат, в котором на стрелочках написаны не только пустое слово ε и буквы алфавита автомата, но и ещё и регулярные выражения.

Мы расширим класс автоматов (мы разрешим на стрелочках писать не только пустое слово и буквы, но ещё и регулярные выражения). Если все языки, которые можно задать регулярным автоматом, можно также задать регулярным выражением, то все языки, которые можно задать обычным автоматом тем более можно задать регулярным выражением. Потому что обычный автомат — это частный случай регулярного автомата.

Докажем, что любой регулярный автомат, задающий язык, можно преобразовать в регулярное выражение, задающее тот же язык. Конструкция преобразования строится индуктивно по числу состояний.

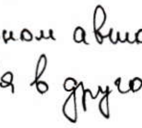
Можно считать, что в регулярном автомате

1) переходы из одного состояния в другое осуществляются по одному регулярному выражению

если $\bigcirc \xrightarrow{\alpha,\beta} \bigcirc$, то преобразуем это в $\bigcirc \xrightarrow{\alpha+\beta} \bigcirc$

2) ровно одно заключительное состояние

если $\otimes$ $\otimes$, то

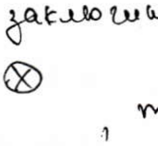


(если заключительных состояний несколько, то добавим новое состояние, а также переходы по 1 из заключительных состояний в новое; наконец, старые заключительные состояния превратим в незаключит., а новое сделаем заключ.)

База индукции

Пусть регулярный автомат состоит из одного состояния. Тогда возможны следующие три случая

Случай 1.



завершающих состояний

= регулярное выр-ние 0.

этот автомат задаёт пустой язык — тот же язык, что и регулярное выр-ние 0

Случай 2.

переходов нет, завершающее состояние есть

= регулярное выр-ние 1.

этот автомат задает язык, состоящий из пустого слова, т.е. тот же язык, что и регулярное выр-ние 1

Случай 3

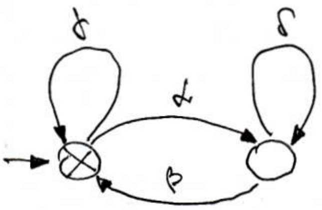
есть переходы и есть заключительное состояние

= регулярное выр-ние α^*

этот автомат задаёт тот же язык, что и регулярное выр-ние α^*

Пусть теперь регулярный автомат состоит из двух состояний. Здесь возможны два случая (начальное состояние совпадает с заключительным, и начальное состояние отличается от заключительного).

Случай 1.

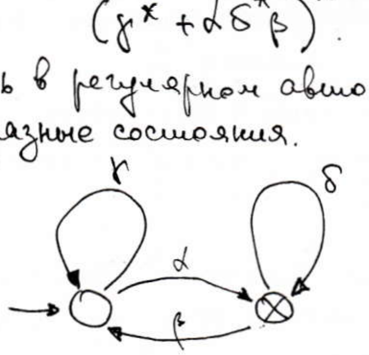


Начальн. сост. = заключит. сост.

Смотрим на картинку и пишем регулярное выражение. Любой принимающий путь начинается и заканчивается в начальном состоянии, т.е. представляет собой какое-то количество циклов из начального состояния в начальном. При этом циклы бывают двух видов. Цикл первого типа состоит из нескольких итераций по маленькой петле. Соответствующее регулярное выражение – γ^*. Цикл второго типа – пойти по α, покрутиться по δ и вернуться по β. Соответствующее регулярное выр-ние $\alpha\cdot\delta^*\beta$. Каждый из этих циклов может повторяться сколько угодно раз. Значит итоговое регулярное выражение имеет вид:

$$(\gamma^*+\alpha\delta^*\beta)^*$$

Случай 2. Пусть теперь в регулярном автомате начальное состояние и заключительное состояние – это разные состояния.



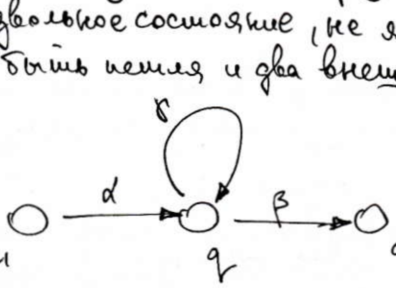
Любой принимающий путь, в общем случае, состоит из трёх частей: 0 или больше итераций по циклам из начального состояния в начальное состояние; затем по дуге α идём в заключительное состояние; наконец, крутимся 0 или больше раз по петле δ, оставаясь в заключительном состоянии

$$(\gamma^*+\alpha\delta^*\beta)^*\cdot\alpha\cdot\delta^*$$

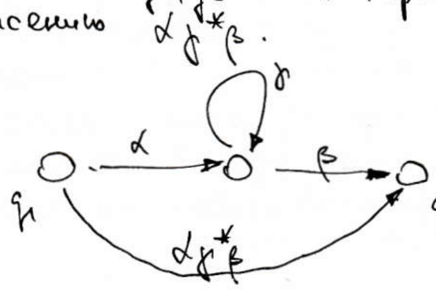
Пусть теперь в регулярном автомате три или больше состояний. Тогда в таком ~~автом~~ автомате есть состояние, которое не начальное и не заключительное. Зафиксируем одно из таких состояний и обозначим его через q.

q – это произвольное состояние, не являющееся начальн. и заключит.

У состояния q может быть петля и два внешних состояния q_1, q_2



Для каждой пары состояний q_1, q_2 как на картинке, мы добавляем переход между q_1 и q_2 по регулярному выражению $\alpha\gamma^*\beta$.



Удаляем состояние q. В первоначальном автомате состояние q - не начальное и не заключительное. Значит в любом принимающем пути оно может быть только промежуточным. В новом автомате все такие пути можно просимулировать без участия состояния q по новым дугам. В новом автомате ничего не приобрели не потеряли. В новом автомате состояний меньше на одно. По индуктивному предположению, новый автомат можно преобразовать в регулярное выражение ■

Лемма (лемма о разрастании для автоматных языков). Пусть L - автоматный язык. Тогда существует константа $n=n(L)$ (зависящая от языка L), для которой каждое слово $w \in L$ длины не меньше n можно разбить на три части $w=xyz$ так, что

1) $|y|>0$
2) $|xy| \leq n$
3) $xy^kz \in L$ для любого целого неотрицательного $k \geq 0$.

(это значит, что всегда можно найти такое слово y недалеко от начала слова w, которое можно удалить или повторить сколь угодно раз, при этом результирующее слово по-прежнему будет принадлежать языку L).

Док-во: Пусть L - автоматный язык. Пусть L задаётся ПДКА A. Для языка L выполняется лемма о разрастании. Выберем значение константы n из леммы. Положим n равным числу состояний в автомате A.

n = число состояний в автомате A.

Рассмотрим произвольное слово w из языка L длины не меньше n

$w \in L$, $|w|=n$.

Автомат прочитает слово w полностью. При этом автомат пройдёт хотя бы $(n+1)$ состояний: начальное состояние + n состояний после прочтения каждой буквы слова w. Автомат пройдёт $(n+1)$ состояний. Всего состояний в автомате n. Значит найдётся состояние, через которое автомат пройдёт два или более раз.

Автомат начинает читать слово w в начальном состоянии. Он читает несколько первых букв слова w и переходит в состояние, которое повторится через несколько шагов. Затем читает ещё несколько букв слова w и автомат возвращается в то же самое состояние. После чего автомат дочитывает слово w до конца и останавливается в заключительном состоянии.

$q_{повт}$ - это первое повторяющееся состояние.

Ту часть слова w, которая была прочитана до входа в повторяющееся состояние обозначим через x. Ту часть слова w, которая читается на цикле обозначим через y. Оставшуюся часть слова w обозначим через z.

1) Почему $|y|>0$? Для того, чтобы перейти из повторяющегося состояния в него же необходимо прочитать хотя бы одну букву слова w. Значит y непустое слово.
2) Почему $|xy| \leq n$? Повторяющееся состояние - это первое из возможных. Первый повтор произойдёт не позднее прочтения первых n символов слова w.
3) Почему слово вида xy^kz принадлежит языку L? Автомат отработает первую часть слова - часть x - и окажется в повторяющемся состоянии. Затем он прочитает часть y k раз, т.е. он пройдёт петлю k раз. После этого автомат окажется в том же состоянии. Затем прочитав слово z, он окажется в заключительном состоянии. Значит слово $xy^kz \in L$.

С помощью леммы о разрастании докажем, что язык $L = \{a^k b^k : k \in \mathbb{N} \cup \{0\}\}$ не автоматный.

От противного. Допустим, что язык L автоматный. Значит для языка L справедлива лемма о разрастании. Возьмем константу n для языка L из леммы о разрастании. Рассмотрим ~~слово~~ в языке L слово

$$w = a^n b^n \in L.$$

Длина этого слова больше, чем n. Значит это слово можно разбить на три части $w = xyz$ как в лемме о разрастании.

$$w = \underbrace{aa\ldots a}_{n}\underbrace{bb\ldots b}_{n}$$

Т.к $|y| > 0$, $|xy| \le n$, то слово ~~xy~~ y состоит из нескольких букв a.

Рассмотрим слово xy^2z. С одной стороны, $xy^2z = a^m b^n$, $m > n$, и $xy^2z \notin L$. С другой стороны, по лемме о разрастании $xy^2z \in L$, что противоречит предыдущему утверждению. Значит наше предположение о том, что язык L автоматный не верно. Язык L автоматный.

Есть не автоматные языки. Необходима модель вычислений, которая позволяет распознавать более сложные языки.

§ Порождающие грамматики.

Порождающие грамматики порождают языки из начального символа с помощью правил вывода. Правила вывода показывают как из слов, принадлежащих языку, можно получить новые слова, принадлежащие языку.

Примеры: (1) Рассмотрим язык правильных скобочных последовательностей. Каждое слово этого языка состоит из символов открывающей скобочки и закрывающейся скобочки

$$\Sigma = \{ l = \text{'('}, r = \text{')'} \}.$$

По определению, пустое слово является ПСП. Пишем правило вывода:

$$S \to \varepsilon.$$

Пусть S - это произвольная ПСП. Как из нее получить новые ПСП?

Во-первых, можно в конец последовательности S приписать ту же самую последовательность S (т.е. продублировать S):

$$S \to SS$$

в результате получим новую ПСП.

Во-вторых, можно последовательность S заключить в скобки (слева приписать открывающуюся скобку, а справа - закрывающуюся):

$$S \to (S).$$

в результате получим снова новую ПСП.

Итак, получили еще два правила вывода. Оказывается, что используя эти три правила вывода из символа S можно получить любую ПСП

(2) Рассмотрим язык $L = \{a^n b c^n : n \in \mathbb{N} \cup \{0\}\}$.

Этот язык состоит из слов b, abc, a^2bc^2, a^3bc^3, ... Однобуквенное слово b принадлежит языку. Пишем правило вывода

$$S \to b.$$

Пусть $S \in L$, т.е. $S = a^k b c^k$. Как из этого слова получить новое слово из языка L? Нужно к нему слева приписать букву a, а справа - букву c, т.е. применить правило вывода:

$$S \to aSc.$$

Язык L можно описать набором таких правил

Опр. Порождающей грамматикой наз-ся четверка $G=(N,\Sigma,P,S)$, где

N - это алфавит вспомогательных символов;

Σ - это алфавит основных символов (считаем, что $N\cap\Sigma=\varnothing$ и вспомогательные символы будем обозначать прописными буквами, а основные символы - строчными буквами).

P - это множество правил вывода, где правило вывода - это слово вида

$$\underset{(N\cup\Sigma)^+}{\alpha\ni}\to\beta\in(N\cup\Sigma)^*$$

$S\in N$ - начальный символ (в алфавите вспомогательных есть специальный символ - символ S, - который наз-ся начальным символом).

Опр. Пусть $\alpha\to\beta\in P$ - некоторое правило вывода в грамматике G. Рассмотрим слово $w_1=\alpha x\in(N\cup\Sigma)^+$ (слово w_1 есть произведение слов α и x). Применим к слову w_1 правило вывода, т.е. в слове w_1 заменим подслово α на слово β. В результате получим новое слово $w_2=\beta x$. В этом случае мы говорим, что слово w_1 порождает слово w_2 в грамматике G и пишем

$$w_1\underset{G}{\to}w_2.$$

Если существует последовательность слов

$$w_1\underset{G}{\to}w_2\underset{G}{\to}w_3-\ldots\underset{G}{\to}w_n\underset{G}{\to}w_{n+1},$$

в которой каждое следующее слово порождается предыдущим, то мы говорим, что слово w_{n+1} выводимо в грамматике G из слова w_1. и пишем

$$w_1\vdash_G w_{n+1}.$$

Опр. Язык, порождаемый грамматикой G, состоит из всех слов в алфавите основных символов, выводимых из начального символа.

$$L(G)=\{w\in\Sigma^*:S\vdash_G w\}.$$

Любое слово языка получается так: мы начинаем с начального символа S, затем последовательно применяем какие-то правила вывода сколько-то раз так, что в конце получается слово без вспомогательных символов - это слово из языка $L(G)$, порождаемого грамматикой G.

Примеры. 1) Грамматика $G=(\{S\},\{a,b,c\},\{S\xrightarrow{\text{ПВ1}}b,\ S\xrightarrow{\text{ПВ2}}aSc\},S)$ порождает язык

$$L(G)=\{a^nbc^n:n\in\mathbb{N}\cup\{0\}\}.$$

Как слово выводится в грамматике G? Начинаем с начального символа S:

$$S\underset{G}{\xrightarrow{\text{ПВ2}}}aSc\underset{G}{\xrightarrow{\text{ПВ2}}}a^2Sc^2\underset{G}{\xrightarrow{\text{ПВ1}}}\underbrace{a^2bc^2}\to\text{слово, принадлежащее языку } L(G).$$

Определим основные классы порождающих грамматик.

Опр. Порождающая грамматика G наз-ся праволинейной, если любое правило вывода этой грамматики имеет вид

$$A\to wB \text{ или } A\to w,$$

где $A,B\in N$ и $w\in\Sigma^*$.

Порождающая грамматика G наз-ся контекстно-свободной, если любое правило вывода в этой грамматике имеет вид

$$A\to\alpha,$$

где $A\in N$ и $\alpha\in(N\cup\Sigma)^*$.

Порождающая грамматика G наз-ся контекстно-зависимой, если любое правило вывода этой грамматики имеет вид

$$\eta A\gamma\to\eta\alpha\gamma,$$

где $A\in N$, $\alpha\in(N\cup\Sigma)^+$, $\eta,\gamma\in(N\cup\Sigma)^*$

Опр. Язык праволинейной грамматики наз-ся праволинейным.

Язык контекстно-свободной грамматики наз-ся контекстно-свободным.

Язык контекстно-зависимой грамматики наз-ся контекстно-зависимым.

Заметим, что правила вывода праволинейной грамматики являются частным случаем правил вывода контекстно-свободной грамматики. Значит любая праволинейная грамматика является контекстно-свободной. Стало быть, праволинейные языки являются контекстно-свободными.

$\boxed{\text{Праволинейные языки} \subseteq \text{Контекстно-свободные языки}}$

При этом есть контекстно-свободные языки, которые не являются праволинейными, т.е. обратного включения нет.

Контекстно-свободные языки не вкладываются в контекстно-зависимые, т.к. в контекстно-зависимых языках нет пустых слов, а в контекстно-свободных пустые слова могут быть.

$\boxed{\text{Контекст.-своб. языки} \not\subseteq \text{Контекстно-зависимые языки}}$

Оказывается, что праволинейные языки – это в точности автоматные.

Теорема. L – праволинейный язык $\Leftrightarrow$ L – автоматный язык.

Без док-ва.

Лемма (о разрастании для контекстно-свободных языков). Пусть L – контекстно-свободный язык. Существует константа $n = n(L)$ (зависящая от языка L), для которой каждое слово $w \in L$ длины не меньше n можно разбить на 5 частей $w = xuyvz$ так, что

(1) $|uv| > 0$;

(2) $|uyv| \leq n$;

(3) $xu^k y v^k z \in L$ для любого неотрицательного целого числа $k \geq 0$.

Без док-ва

Покажем, что язык $L = \{a^k b^k c^k : k \in \mathbb{N} \cup \{0\}\}$ не контекстно-свободный язык.

От противного. Допустим, что L – контекстно-свободный язык. Значит для него выполняется лемма о разрастании. Возьмём n из леммы о разрастании для языка L. Рассмотрим слово $w = a^n b^n c^n$ из языка L. Его длина больше n. Значит это слово можно разбить на 5 частей

$$a^n b^n c^n = xuyvz.$$

Посмотрим на середину uyv. Длина этого слова $\leq n$. Значит слово uyv не может содержать одновременно и букву a, и букву b, и букву c. Пусть слово uyv не содержит букву a. При этом хотя бы одно слов u, v не пусто. Значит в слове

$$xu^2yv^2z$$

буква a ровно n, а букв b или c больше n. (т.к. мы продублировали части u, v в слове) Стало быть, $xu^2yv^2z \notin L$. Но по лемме о разрастании $xu^2yv^2z \in L$. Противоречие