БУ ВО Ханты-Мансийского автономного округа – Югры
«Сургутский государственный университет»

Теория языков программирования и методы трансляции

Часть 2

- Формальные языки и грамматики

Гришмановский Павел Валерьевич
доцент кафедры автоматики и компьютерных систем, к.т.н., доцент

Сургут, 2019

Формальные языки и грамматики

Язык – это заданный набор символов и правил их комбинирования для записи осмысленных текстов

- Набор символов языка – алфавит
- Правила комбинирования – синтаксис
- Смысл конструкций языка – семантика

Язык программирования – формальная знаковая система, предназначенная для записи компьютерных программ, которая определяет набор лексических, синтаксических и семантических правил, задающих внешний вид программы и действия, которые выполнит исполнитель (компьютер) под ее управлением

Задача языка программирования – передача команд и данных от человека к компьютеру

Формальные языки и грамматики

Синтаксис – это

- система языковых категорий, относящихся к соединениям слов и строению предложений
- набор правил, определяющих множество допустимых конструкций языка
- правило записи программ на языке программирования
- система обозначений для обмена информацией между человеком и компьютером

Формальные языки и грамматики

Синтаксические свойства языка

- Самодокументируемость (легкость чтения) – синтаксические конструкции отражают семантические особенности, структуры данных и алгоритмов становятся очевидны при просмотре текста
  - наличие избыточности и умолчаний
  - естественность форматов и идентификаторов
  - наличие свободных комментариев
- Легкость написания – семантика передается минимально необходимыми конструкциями
  - краткость и регулярность конструкций
  - сокращенные формы конструкций и умолчания
  - естественность форматов и идентификаторов

Формальные языки и грамматики

Синтаксические свойства языка

- Верифицируемость – возможность математически строгого доказательства корректности
  - отсутствие избыточности и умолчаний
  - регулярность конструкций
- Легкость трансляции – возможность построения транслятора и приемлемые затраты на трансляцию
  - отсутствие избыточности
  - регулярность конструкций
- Однозначность – отсутствие неопределенности толкований конструкций языка, любая конструкция имеет единственный смысл

Формальные языки и грамматики

Семантика – смысловое значение (интерпретация) абстрактного синтаксиса (множества допустимых видов конструкций), представленное в терминах математически строгой формальной модели

Подходы к заданию семантики

- операционный – в терминах переходов абстрактной машины из одного состояния в другое *(категориальная абстрактная машина, SECD-машина П. Лендина, SK-машина Дэвида Тёрнера)*
- пропозиционный (деривационный) – значение конструкций языка выражается формулами, описывающими состояния объектов программы *(аксиоматический метод Хоара, метод индуктивных утверждений Флойда)*
- денотационный – смысл конструкций языка представлен в терминах абстракции функций, оперирующих состояниями программы *(теория вычислений Д. Скотта)*

Формальные языки и грамматики

Задать язык означает

- задать алфавит – множество допустимых символов
- задать синтаксис – множество правильных программ
- задать семантику – смысл программ

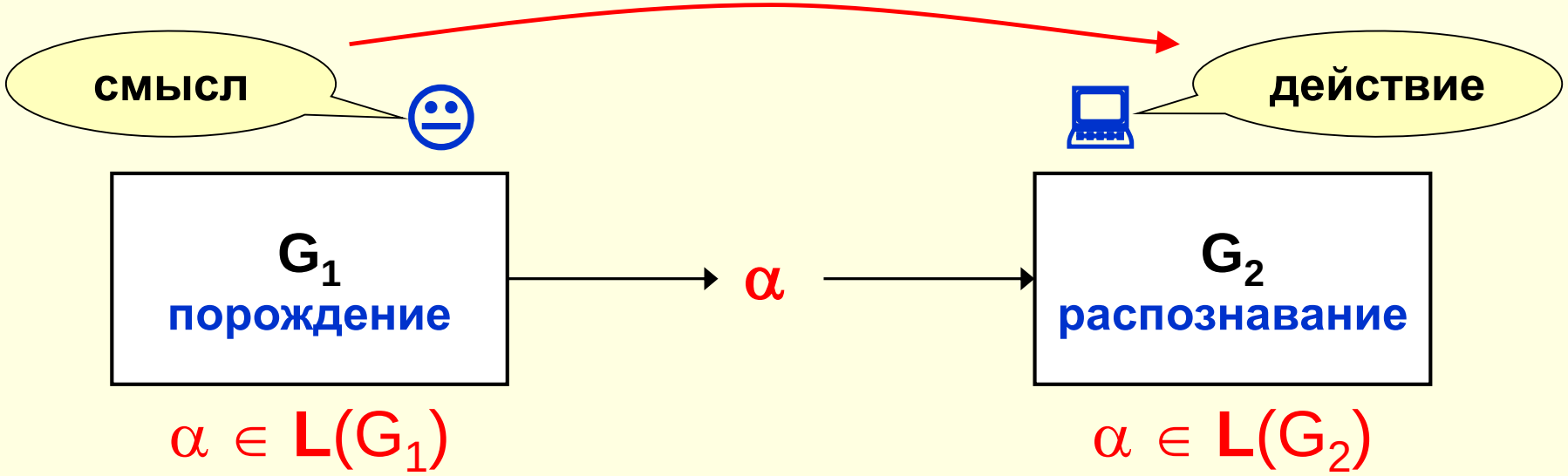
Способы задания языка

- перечисление всех допустимых цепочек языка
- определение способа порождения цепочек языка – генерация
- определение способа распознавания цепочек языка – разбор, трансляция

Формальные языки и грамматики

Грамматика – математическая система, определяющая язык, т.е. формальное описание способа

- порождения цепочек символов – порождающая
- распознавания цепочек символов – распознающая



Формальные языки и грамматики

Язык **L** как символьная система – множество допустимых цепочек α, состоящих из символов алфавита **V**, порождаемое грамматикой G

$$\alpha \in \mathbf{L},\ \mathbf{L} = \mathbf{L}(G(\mathbf{V}))$$

Алфавит **V** – счетное множество допустимых символов языка, может быть бесконечным

V* – множество всех возможных цепочек над **V**, включая λ

$\mathbf{V}^+$ – множество всех возможных цепочек над **V**, ***не*** включая λ

$$\mathbf{V}^* = \mathbf{V}^+ \cup \{\lambda\},\ \text{где } \lambda \text{ – пустая цепочка}$$

Цепочка $\alpha(\mathbf{V})$ – цепочка α над алфавитом **V**, состоящая только из символов множества **V**

$$\forall\alpha(\mathbf{V}) \in \mathbf{V}^*$$

Язык **L**(**V**) – язык **L** над алфавитом **V**, счетное подмножество цепочек конечной длины из множества всех возможных цепочек над алфавитом **V**, может быть бесконечным, а длина любой цепочки может быть сколь угодно большой

$\mathbf{L}(\mathbf{V}) \subseteq \mathbf{V}^*$

$\mathbf{L}_1(\mathbf{V}) \subseteq \mathbf{L}_2(\mathbf{V})$, если $\forall\alpha \in \mathbf{L}_1(\mathbf{V}) :\ \alpha \in \mathbf{L}_2(\mathbf{V})$ – $\mathbf{L}_2$ включает $\mathbf{L}_1$

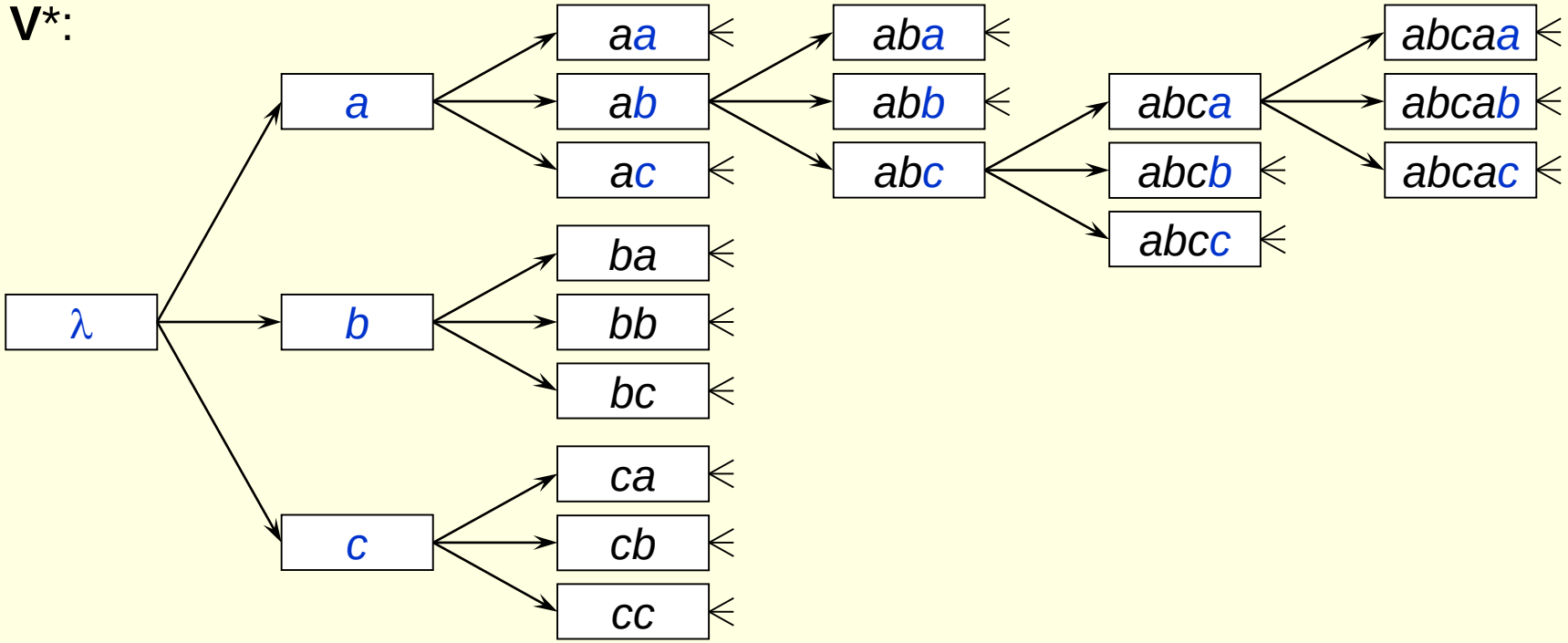
$\mathbf{L}_1(\mathbf{V}) = \mathbf{L}_2(\mathbf{V})$, если $\mathbf{L}_1(\mathbf{V}) \subseteq \mathbf{L}_2(\mathbf{V}) \ \wedge \mathbf{L}_2(\mathbf{V}) \subseteq \mathbf{L}_1(\mathbf{V})$ – $\mathbf{L}_1$ и $\mathbf{L}_2$ эквивалентны

$\mathbf{L}_1(\mathbf{V}) \cong \mathbf{L}_2(\mathbf{V})$, если $\mathbf{L}_1(\mathbf{V}) \cup \{\lambda\} = \mathbf{L}_2(\mathbf{V}) \cup \{\lambda\}$ – $\mathbf{L}_1$ и $\mathbf{L}_2$ почти эквивалентны

Формальные языки и грамматики

V = {*a*, *b*, *c*}

V*:



1 |**V**| |**V**|^2 |**V**|^3 |**V**|^4 |**V**|^5

V* = {λ, *a*, *b*, *c*, *aa*, *ab*, *ac*, *ba*, *bb*, *bc*, *ca*, *cb*, *cc*, *aaa*, *aab*, *aac*, *aba*, …}

Формальные языки и грамматики

α, β	Цепочки символов	Любое количество символов, записанных друг за другом	$\alpha = a_1a_2a_3\ldots a_n$ $\beta = b_1b_2b_3\ldots b_m$
$\alpha = \beta$	Совпадение, эквивалентность, равенство	Одинаковый состав, количество и порядок символов	$a_i = b_j$: $i = j$, $n = m$, $i \in [1, n]$, $j \in [1, m]$
$\lvert\alpha\rvert$	Длина цепочки	Количество символов в цепочке	$\lvert\alpha\rvert = n$, $\lvert\beta\rvert = m$ $\lvert\alpha\rvert = \lvert\beta\rvert$: $\alpha = \beta$
$\alpha\beta$	Конкатенация, объединение	Дописывание символов второй цепочки к первой в том же порядке	$\alpha\beta = a_1a_2\ldots a_nb_1b_2\ldots b_m$ $\lvert\alpha\beta\rvert = \lvert\alpha\rvert + \lvert\beta\rvert$ $\exists\alpha, \beta$: $\alpha\beta \neq \beta\alpha$
α^R	Обращение	Запись символов цепочки строго в обратном порядке	$\alpha^R = a_na_{n-1}a_{n-2}\ldots a_2a_1$ $\lvert\alpha^R\rvert = \lvert\alpha\rvert$ $(\alpha\beta)^R = \beta^R\alpha^R$
α^k	Итерация, повторение	Конкатенация цепочки с собой k раз, $k \geq 0$	$\alpha^k = (a_1\ldots a_n)_1\ldots (a_1\ldots a_n)_k$ $\lvert\alpha^k\rvert = k\cdot\lvert\alpha\rvert$

Формальные языки и грамматики

Пустая цепочка λ (ε, E, e) – это цепочка, которая не содержит ни одного символа

- $|\lambda| = 0$
- $\lambda^R = \lambda$
- $\lambda^k = \lambda, \forall k \geq 0$
- $\lambda\alpha = \alpha\lambda = \alpha, \forall \alpha \in \mathbf{V}^*$
- $\alpha^0 = \lambda, \forall \alpha \in \mathbf{V}^*$

Формальные языки и грамматики

Способы задания грамматик

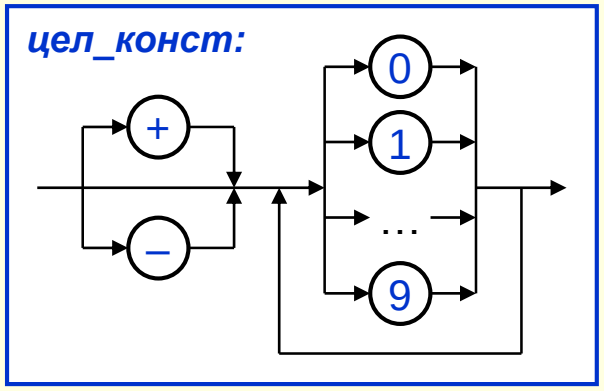
- На естественном языке – формально точное описание правил построения лексем, конструкций языка и их комбинирования
- Графически (диаграммы Вирта) – показывают порядок «сборки» конструкций языка
- Математическое определение множеств – использование формального аппарата теории множеств, алгебры логики и т.п.
- Нормальная форма Бэкуса (НФБ), форма Бэкуса-Наура (ФБН, БНФ) – формальная запись правил построения конструкций языка
- Расширенная форма Бэкуса-Наура с использованием метасимволов – аналогично, но предоставляет большую компактность, наглядность и гибкость описания
- Лямбда-исчисление – функциональная модель вычислителя на цепочках символов, эквивалент НФБ
- Конечные автоматы – простейшая формальная машина, применяемая в практике, но имеющая существенные ограничения описательной мощности
- Формальные машины – автоматы с памятью (магазинной, с произвольным доступом), в качестве управляющего устройства которых выступает конечный автомат
- Регулярные выражения – операции с цепочками символов, эквивалент конечных автоматов

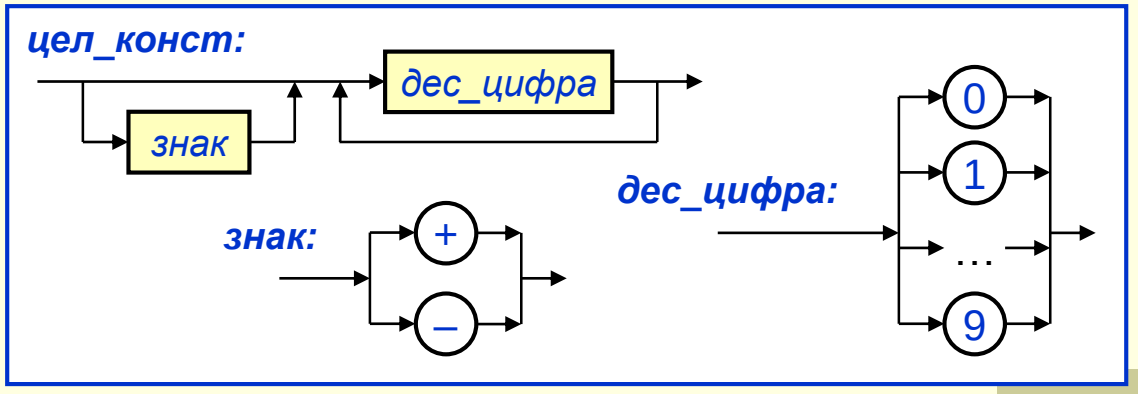
Формальные языки и грамматики

Описание на естественном языке – формально точное описание строения термина языка, лингвистическая формула

- *Идентификатор – это любая последовательность латинских больших и маленьких букв, символа «_» и цифр от 0 до 9, начинающаяся не с цифры.*
- *Целочисленная константа в десятичной системе счисления представляет собой любую последовательность десятичных цифр от 0 до 9, начинающуюся не с нуля, если она не представляет значение 0, которой может предшествовать один из символов знака числа «+» или «-», и вслед за последней цифрой может быть в любой комбинации записано по одному из модификаторов «U» или «u» для обозначения беззнакового типа, и «L» или «l» для обозначения длинного целого числа.*

Диаграммы Вирта – ориентированный граф с вершинами двух типов: терминальными символами (символами алфавита языка) и нетерминальными символами (металингвистическими переменными, т.е. терминами языка, требующими раскрытия в цепочки терминальных символов)





Формальные языки и грамматики

Математическое определение множеств – формулы

- $L_1 = \{ a^n b^n : a,b \in \mathbf{V}, n > 0 \}$
- $L_2 = \{ a^n b^m c^n d^m : a,b,c,d \in \mathbf{V}, n,m \geq 0 \}$
- $L_3 = \{ a^n b^{f(n)} : a,b \in \mathbf{V}, n \geq 0, f(n) \geq 0$ – *некоторая (любая) вычислимая функция* $\}$
- $L_B = \{ \forall \alpha : \alpha \in \{0, 1\}^*, |\alpha| = 8 \}$
- $L_C = \{ L_{prep}(\mathbf{V}) \cup L_{type}(\mathbf{V}) \cup L_{var}(\mathbf{V}) \cup L_{func}(\mathbf{V}) \}^*$

Формальные языки и грамматики

Регулярные выражения

1. Отдельные символы являются регулярными выражениями
2. Если *a* и *b* – регулярные выражения, то регулярными выражениями также являются:
   - *a*∨*b* – выбор, альтернация
   - *ab* – конкатенация
   - (*a*) – группировка
   - *a** – повторение, замыкание Клини (0 или более раз)
3. Ничто другое не является регулярным выражением

- **(** (0∨1∨2∨3∨4∨5∨6∨7∨8∨9) ∨ (+∨–)(0∨1∨2∨3∨4∨5∨6∨7∨8∨9) **)** (0∨1∨2∨3∨4∨5∨6∨7∨8∨9)*
- (a∨b∨c∨d∨e∨f∨g∨h∨i∨j∨k∨l∨m∨n∨o∨p∨q∨r∨s∨t∨u∨v∨w∨x∨y∨z∨_) (a∨b∨c∨d∨e∨f∨g∨h∨i∨j∨k∨l∨m∨n∨o∨p∨q∨r∨s∨t∨u∨v∨w∨x∨y∨z∨_∨ ∨0∨1∨2∨3∨4∨5∨6∨7∨8∨9)*
- (0∨1)(0∨1)(0∨1)(0∨1)(0∨1)(0∨1)(0∨1)(0∨1)

Формальные языки и грамматики

Метасимволы – используются самостоятельно или для сокращения записи в расширенной форме Бэкуса–Наура

- <…> – нетерминальный символ (металингвистическая переменная)
- "..." – запись терминальных символов, совпадающих с метасимволами
- […] – необязательная часть
- {…} – повторение (0 или более раз)
- (…) – группировка
- …|… – альтернативы

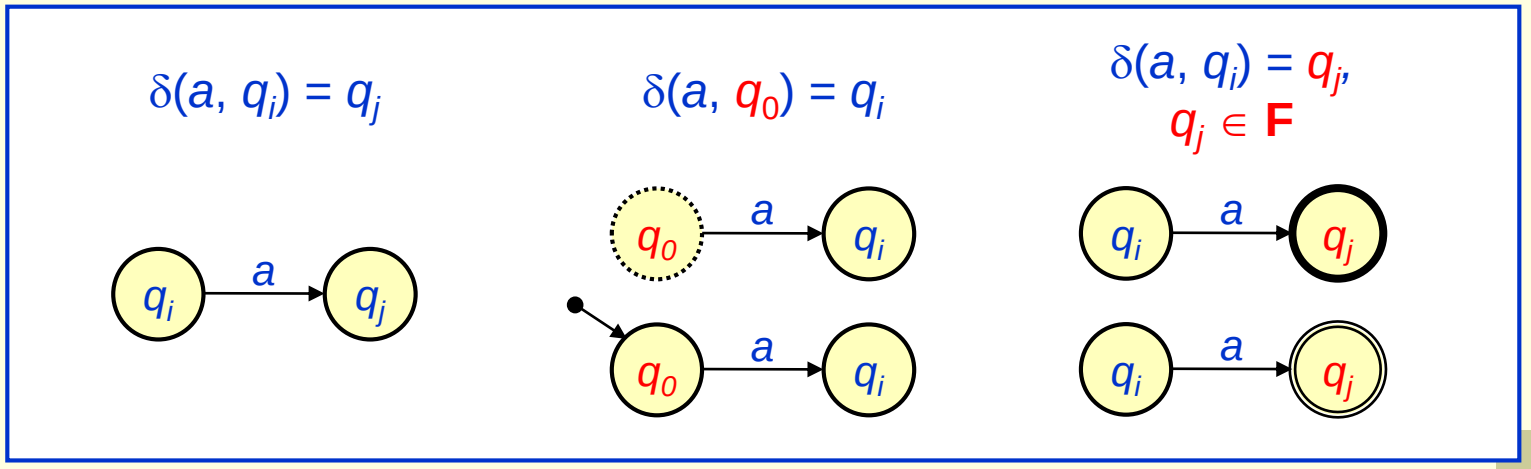
- [+|–](0|1|2|3|4|5|6|7|8|9){0|1|2|3|4|5|6|7|8|9}
- <*цел_конст*> ::= [<*знак*>]<*дес_цифра*>{<*дес_цифра*>}
  <*знак*> ::= +|–
  <*дес_цифра*> ::= 0|1|2|3|4|5|6|7|8|9
- <*идентификатор*> ::= <*буква*>{<*буква*>|<*цифра*>}
  <*буква*> ::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|_
  <*цифра*> ::= 0|1|2|3|4|5|6|7|8|9
- [<класс_памяти>] <тип_возвр_знач> [<согл_вызова>]
  <имя>"("<список_форм_парам>")" (";"|"{"<тело_функции>"}")

Формальные языки и грамматики

Конечные автоматы

Конечный автомат M = (**Q**, **V**, δ, q_0, **F**), где

- **Q** – конечное множество состояний
- **V** – конечное множество символов (алфавит языка)
- δ – функция переходов, $\delta : \mathbf{V} \times \mathbf{Q} \to \mathbf{Q}$
  $\delta(a, q_i) = \mathbf{Q}_i$, $a \in \mathbf{V}$, $q_1 \in \mathbf{Q}$, $\mathbf{Q}_i \subseteq \mathbf{Q}$
  - $\forall i : |\mathbf{Q}_i| \le 1$ – **детерминированный** конечный автомат
  - $\exists i : |\mathbf{Q}_i| > 1$ – **недетерминированный** конечный автомат
- q_0 – начальное состояние, $q_0 \in \mathbf{Q}$
- **F** – множество конечных состояний, $\mathbf{F} \subseteq \mathbf{Q}$



Формальные языки и грамматики

Форма Бэкуса–Наура или **грамматики Хомского**

- Грамматика G = ($\mathbf{V}^T$, $\mathbf{V}^N$, **P**, S), где
  - $\mathbf{V}^T$ – множество терминальных символов, соответствующее алфавиту порождаемого языка
  - $\mathbf{V}^N$ – множество нетерминальных символов (металингвистических переменных), используемых в записи грамматики

    $\mathbf{V}^T \cap \mathbf{V}^N = \varnothing$ – множества не пересекаются

    $\mathbf{V}^T \cup \mathbf{V}^N = \mathbf{V}$ – объединение множеств образует алфавит **грамматики**
  - **P** – множество правил грамматики

    $\mathbf{P} : \alpha \to \beta,\ \alpha \in \mathbf{V}^+,\ \beta \in \mathbf{V}^*$
  - S – начальный (целевой) символ грамматики, $S \in \mathbf{V}^N$

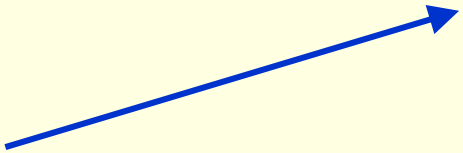
- В левой части правила должен присутствовать хотя бы один нетерминальный символ – порождение цепочек языка не может осуществляться на основе других цепочек того же языка
- Каждый нетерминальный символ должен хотя бы раз встретиться в левой части правила – в противном случае из него нельзя вывести цепочку терминальных символов
- Правила могут (должны!) быть рекурсивными – возможен вывод бесконечного числа цепочек языка при конечном множестве правил
- Грамматика должна содержать хотя бы одно нерекурсивное правило – иначе невозможно закончить вывод цепочки языка

Формальные языки и грамматики

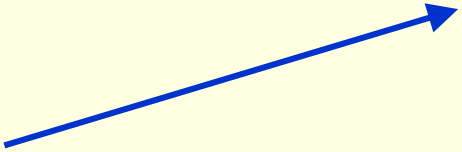
$G_1 = (\mathbf{V}^T, \mathbf{V}^N, \mathbf{P}, S)$,
$\mathbf{V}^T = \{ 0, 1 \}$
$\mathbf{V}^N = \{ A, S \}$
$\mathbf{P} = \{ S \rightarrow 0A1$
$0A \rightarrow 00A1$
$A \rightarrow \lambda \}$

01
0011
000111
00001111
…

$G_2 = (\mathbf{V}^T, \mathbf{V}^N, P, S)$,
$\mathbf{V}^T = \{ 0, 1 \}$
$\mathbf{V}^N = \{ S \}$
$\mathbf{P} = \{ S \rightarrow 0S1 \mid 01 \}$



$G_3 = (\mathbf{V}^T, \mathbf{V}^N, \mathbf{P}, S)$,
$\mathbf{V}^T = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, - \}$
$\mathbf{V}^N = \{ S, T, F \}$
$\mathbf{P} = \{ S \rightarrow T \mid +T \mid -T$
$T \rightarrow F \mid TF$
$F \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \}$



0
12
32453
+1
+83745658749372 5633
-9876
-0
007