Формальные языки и автоматы.

Андрей Тищенко @AndrewTGk

2024/2025

Лекция 17 января

Преподаватель: Игнатьев Валерий Николаевич.
Немного рекламы ИСП РАН, его направлений. Курс закончится до майских праздников, элементы контроля также пройдут раньше (кроме сессии). На лекциях проведут две контрольные, они будут весить 0.4 от оценки.
Телеграмм курса.

1 Актуальность

- Регулярные выражения (PB, RE)

  текстовые редакторы

  утилиты обработки текстов (sed, grep)

  UNIX shell

  тривиальный разбор текста

- Конечные автоматы — модель для многих HW&SW компонентов

  Лексический анализ компилятора

  Model checking

  Дизайн и верификация цифровых электронных схем

- Контектно-свободные грамматики используются для

  формализации синтаксиса практически всех ЯП

  в распознавании естественного языка (natural language processing)

- Конструирование неоптимизирующего компилятора

2 История

1930-е	• Тьюринг — абстрактная машина • Проблема разрешимости, останова
1940–1950	• теория конечных автоматов, разработка и верификация цифровых схем, лексический анализ, парсеры-краулеры, протоколы связи. • Хомский — оснвоы теории ФЯ
1950	• Формальные грамматики
1950-	• Формальные грамматкии применяются при реализации первых ЯП

3 Основные понятия теории автоматов

Определение

Алфавит (V) — конечное непустое множество символов.

Определение

Слово (строка, предложение) — любая цепочка конечной длины, составленная из символов алфавита.
ε или ϵ, или e — обозначение пустой строки/слова — не содержит ни одного символа.
V^* — множество всех слов, составленных из символов V
$V^+ = V^* \backslash \varepsilon$

- $|x|$ — длина строки $x \in V$
- $|\varepsilon| = 0$
- Пусть $x,\ y \in V^*,\ x = a_1 a_2 \dots a_i,\ y = b_1 b_2 \dots b_j$, тогда xy — конкатенация, $xy = a_1 a_2 \dots a_i b_1 b_2 \dots b_j$

Определение

Язык (L) — любое множество строк в алфавите $L \subseteq V^*$
Примеры записи:

- $L_1 = \varnothing$ — пустой язык.
- $L_2 = \{\varepsilon\}$ — язык, содержащий только пустое слово.
- $L_3 = \{\varepsilon,\ a,\ b,\ aa,\ ab,\ ba,\ bb\}$ — язык, содержащий все цепочки из a и b, длина которых не превосходит 2.
- $L_4 = \{a^{n^2} : n > 0\}$ — язык слов, состоящих из букв a в количестве квадрата натурального числа.

Проблемы

- Как задавать язык?
- Для каждого ли языка существует конечное представление?
- Для каких класов языков существуют конечные представления?
- Является ли данная цепочка элементом определённого языка?

Как задавать язык?

Распознавание	Порождение
• Алгортим или процедура, которые определяют, принадлежит ли заданное слово языку	• Процедура, которая систематически порождает предложения языка последовательно в некотором порядке

Алгоритм: да или нет (принадлежит или нет).
Процедура: да или нет, или не завершается. По алгоритму или процедуре распознавания языка можно построить процедуру, порождающую этот язык.

Построение порождающей процедуры по процедуре распознавания

Пусть V — алфавит из p символов. Занумеруем слова из V^*, зададим способ нумерации (k) пар положительных целых чисел $(i,\ j)$ (вспоминаем доказательство счётности множества $\mathbb{Q}$).

Построение распознающей процедуры по порождающей

Пусть P — порождающая процедура языка L, тогда мы может порождать слова из языка, пока новое слово не совпадёт с нашим (если слова в языке нет, то будет исполняться бесконечно).

Типы языков

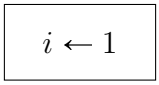
- Рекурсивно перечислимый — существует процедура распознавания или существует порождающая процедура
- Рекурсивный — существует алгоритм распознавания

Теорема

Пусть $L \subseteq V^*$ — некоторый язык, а $\overline{L} = V^* \backslash L$ — его дополнение. Если языки L и $\overline{L}$ рекурсивно перечислимы, то L рекурсивен.

Доказательство

Пусть L распознаётся процедурой P, а $\overline{L}$ распознаётся $\overline{P}$.
Построим алгоритм распознавания L: $(x \in L)$?



Дорисовать схему с лекции

Грамматики

Способ порождения языков.**Нарисовать схему с лекции**.
$\underbrace{(\text{терминалы}) \to (\text{терминалы/нетерминалы})}_{\text{правила}}$
Грамматикой называется чевёрка $G = (V_N,\ V_T,\ P,\ S)$, где
V_N — алфавит нетерминальных символов
V_T – алфавит терминальных символов, причём $V_N \cap V_T = \varnothing \wedge V = V_N \cup V_T$
P — конечное множество правил вида $\alpha \to \beta,\ \alpha \in V^* V_N V^*,\ \beta \in V^*$
$S \in V_N$ — начальный нетерминал (аксиома грамматики, стартовый символ)
Обозначения:
$A,\ B,\ C,\ S \in V_N$ — нетерминалы
$a,\ b,\ c \in V_T$ — терминалы
$x,\ y,\ z \in V_T^*$ — строки терминалов
$\alpha,\ \beta,\ \gamma \in V^*$ — строки из объединённого алфавита V

- Пусть $\alpha \to \beta \in P$ — правило; $\gamma,\ \delta \in V^*$
  Тогда $\gamma\alpha\delta \underset{G}{\Rightarrow} \gamma\beta\delta$ читается как «из $\gamma\alpha\delta$ непосредственно выводится из $\gamma\beta\delta$» в грамматике G.

- Пусть $\alpha_1,\ \alpha_2, \ldots,\ \alpha_m \in V^*,\ \alpha_1 \underset{G}{\Rightarrow} \alpha_2,\ \alpha_2 \underset{G}{\Rightarrow} \alpha_3,\ \ldots,\ \alpha_{m-1} \underset{G}{\Rightarrow} \alpha_m$
  Тогда $\alpha_1 \underset{G}{\Rightarrow} \alpha_m$ читается как «из α_1 выводится α_m в грамматике G»

- Рефлексивность $\alpha \overset{*}{\underset{G}{\Rightarrow}} \alpha$.
  $\overset{*}{\underset{G}{\Rightarrow}}$ значит в алфавите есть последовательность переходов (какой-то длины), переводящая левый аргумент в правый.

Язык, порождаемый грамматикой G: $L(G) = \{\omega \mid \omega \in V_T^*,\ S \overset{*}{\underset{G}{\Rightarrow}} \omega\}$ — множество всех терминальных строк, выводимых из начального нетерминала грамматики.
Сентенциальной формой называется любая строка α такая, что

$$\alpha \in V^* \wedge S \overset{*}{\underset{G}{\Rightarrow}} \alpha$$

Грамматики $G_1,\ G_2$ эквивалентны, если $L(G_1) = L(G_2)$
Пример:

$$G = (\{S\},\ \{0,\ 1\},\ \{S \to 0S1,\ S \to 01\},\ S)$$

$$S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow 0^3S1^3 \Rightarrow \cdots \Rightarrow 0^{n-1}S1^{n-1} \Rightarrow 0^n1^n$$

$$L(G) = \{0^n1^n \mid n > 0\}$$

Лекция 24 января

Классификация грамматик по Хомскому

Грамматика **типа 0** — без ограничений
Грамматика **типа 1 (неукорачивающая, контекстно-зависимая)**. Каждое правило имеет вид

$$\alpha \to \beta \in P : |\alpha| \leqslant |\beta|$$

Грамматика **типа 2 (контекстно-свободная)**. Каждое правило имеет вид

$$A \to \beta \in P,\ A \in V_n,\ \beta \in V^+$$

Грамматика **типа 3 (регулярная)**. Каждое правило имеет вид

$$A \to \alpha B \text{ или } A \to \alpha,\ \alpha \in V_T,\ A,\ B \in V_N$$

Можно показать, что правила вида $A \to xB$ или $A \to x,\ x \in V_T^+,\ A,\ B \in V_N$ также задают регулярную грамматику.
Язык, порождаемый грамматикой типа i, и не порождаемый грамматикой типа $i+1$, называют языком типа i.
Пусть K_i — класс языков типа i, тогда $K_3 \subset K_2 \subset K_1 \subset K_0$

Проблема с пустым словом

Если какой-то символ может переходить в пустое слово, то грамматика сможет укорачивать слова, поэтому:

$$\neg\exists L \in K_1 \cup K_2 \cup K_3 : \varepsilon \in L$$

Из-за этого единственный возможный способ добавить в грамматику пустое слово:

$$S \to \varepsilon$$

Теорема

Переписать

Примеры

$G = (\{S,\ A,\ B\},\ \{0,\ 1\},\ \{S \to AB,\ A \to 0A,\ A \to A,\ B \to 1B,\ B \to 1\},\ S)$. Контекстно-свободная, но не является регулярной, ведь $S \to AB$ (два терминала справа).
Заметим, что $L(G) = \{0^n 1^m \mid n,\ m > 0\}$
$G_1 = (\{S,\ A\},\ \{0,\ 1\},\ \{S \to 0S,\ S \to 0A,\ A \to 1A,\ A \to 1\},\ S)$. Регулярный

Замечание

Если язык порождается некоторой грамматикой, то это не означает, что не существует грамматики с более сильными ограничениями, порождающей заданный язык.

Лексический анализ

- Первый этап анализа исходного текста программы в компиляторе
- Входная строка разбивается на цепочку лексем (token)
  • Например: <константа, 5>, <ключевое слово, for>, <идентификатор, int>
- Отдельная стадия анализа или «по запросу» синтаксического анализа
- Задаётся конечным автоматом
  • На практике регулярное выражение или регулярная грамматика

Итерация на множествах

Пусть $P,\ Q \subseteq V^*$ — некоторые языки.

Определение

Произведением PQ языков P и Q называется множество, состоящее из всех слов в алфавите A, которые составлены конкатенацией некоторого слова из P и некоторого слова из Q.

$$PQ = \{pq \mid p \in P,\ q \in Q\}$$

Определение

Итерация (замыкание Клини) языка L обозначается L^* и представляет собой множество всех слов, которые можно образовать путем конкатенации любого количества слов из L.
При этом допускаются повторения, то есть одна и та же цепочка из L может быть выбрана для конкатенации более одного раза

$$P^* = \bigcup_{n=0}^{\infty} P^n,\ P^0 = \{\varepsilon\},\ P^1 = P$$

Регулярные множества/языки

Определение

Регулярное множество в алфавите V_T определяется рекурсивно:

1. $\varnothing$ — регулярное множество в алфавите V_T
2. $\{\varepsilon\}$. **Продолжить**

Форма записи регулярных множеств

Регулярное выражение в алфавите V_T и обозначаемое им регулярное множество определяется следующим образом:

1. $\varnothing$ — регулярное выражение, обозначающее регулярное множество $\varnothing$
2. ε — регулярное выражение, обозначающее регулярное множество $\{\varepsilon\}$
3. a — регулярное выражение, обозначающее регулярное множество $\{a\}$
4. Если p и q — регулярные выражения, обозначающие регулярные множества P и Q соответственно, то:
   (a) $(p|q)$ — обозначает $P \cup Q$
   (b) (pq) — обозначает PQ,
   (c) p^* — обозначает P^*
5. **Продолжить**

Примеры

- $a(\varepsilon|a)|b$ обозначает $\{a,\ b,\ aa\}$
- $a(a|b)^*$ — обозначает множество всевозможных цепочек, состоящих из a и b, начинающихся с a
- $(a|b)^*(a|b)(a|b)^*$ — любая непустая последовательность из $a,\ b$
- $\big((1|0)(1|0)(1|0)\big)^*$ — множество всех цепочек, состоящих из нулей и единиц, длины которых делятся на 3.

Напишем регклярное выражение, задающее множество всех цепочек в алфавите $\{0,\ 1\}$, в которых символ 1 является третьим с конца:

$$(0|1) * 1(0|1)(0|1)$$

Свойства регулярных выражений

Пусть $p,\ q,\ r$ — регулярные выражения, тогда:

1. $p|q = q|p$
2. $\varnothing^* = \varepsilon$
3. $\varepsilon^* = \varepsilon$
4. $p|(q|r) = (p|q)|r$
5. $p(qr) = (pq)r$
6. $p(q|r) = pq|pr$
7. $(p|q)r = pr|qr$
8. $p\varepsilon = \varepsilon p = p$
9. $\varnothing p = p\varnothing = \varnothing$
10. $p^* = p|p^*$
11. $(p^*)^* = p^*$
12. $p|p = p$
13. $p|\varnothing =$ p

Именование регулярных выражений

Пример:
$Letter = a|b|c|\ldots|x|y|z$,
$Digit = 0|1|\ldots|9$
$Identifier = Letter(Letter|Digit)^*$

Конечные автоматы

Регулярные ыражения — для задания регулярных множеств
Конечные автоматы — для распознавания регулярных множеств
Недетерминированный конечный автомат (НКА) — $M = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$

- Q — конечное непустое множество состояний
- Σ — конечный входной алфавит
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to Q$ — функция переходов
- $q_0 \in Q$ — начальное состояние
- $F \subseteq Q$ — множество заключительных (допускающих) состояний.

Конфигурация автомата $(q,\ w) \in Q \times \Sigma^*$. q — текущее состояние, w — цепочка символов под головкой и справа на ленте.

- $(q_0,\ w)$ — начальная конфигурация.
- $(q,\ \varepsilon),\ q \in F$ — заключительная (допускающая) конфигурация.

Такт автомата $M(\vdash)$ — бинарное отношение, заданное на конфигурациях: если $p \in \delta(q,\ a),\ a \in \Sigma \sup\{\varepsilon\}$, то $(q,\ aw) \vdash (p,\ w)$ для всех $w \in \Sigma^*$
$\vdash^*,\ \vdash^+$ — рефлексивно-транзитивное (транзитивное) замыкание $\vdash$
Автомат M допускает цепочку w, если $(q_0,\ w) \vdash^* (q,\ \varepsilon),\ q \in F$
Язык L, распознаваемый (допускаемый, определяемый) M:

$$L = L(M) = \{w \mid w \in \Sigma^* \wedge (q_0,\ w) \vdash^* (q,\ \varepsilon),\ q \in F\}$$

Детерминированный конечный автомат (ДКА) — НКА $M = (Q,\ \Sigma,\ \delta,\ p_0,\ F)$, если

- $\forall q \in Q : \delta(p,\ \varepsilon) = \varnothing$
- $\forall (q,\ a) \in Q \times \Sigma : |\delta(q,\ a)| \leqslant 1$

Ура, рисуночки.
Все состояния обозначаются окружностями.
Начальное состояние — окружность, в которую проведена стрелка из пустоты.
Допускающее состояние — две концентрические окружности.
Обычное состояние — окружность.