Преподаватель: Матросов Александр Васильевич.

Касаемо практики: мы будем делать 2 программных проекта. Первый будет связан с реализацией конечного автомата для регулярного выражения. Второй проект будет связан с конечным автоматом с магазином, который используется для создания контекстно-свободных языков программирования.

Литература: Основная: Джон Хопкрофт, Мотвани, Ульман, «Введение в теорию автоматов и вычислений». Допы: Ахо «Компиляторы: принципы, технологии и инструменты»; Карпов «Теория автоматов», «Основы построения трансляторов».

Теория формальных языков изучает методы задания, распознавания и обработки языков.

Определение 0.1. Словарь — конечное множество объектов.

Определение 0.2. Символ — элемент словаря.

Определение 0.3. Цепочка символов над словарем — конечная последовательность символов словаря.

Пример 0.1. $\alpha = aabb$ — цепочка над словарём $V = \{a, b\}$.

Определение 0.4. V^* — множество всевозможных цепочек, включая пустую.

Определение 0.5. a^n — цепочка из n символов.

Определение 0.6. Операция подстановки выполняет замену подцепочки заданной цепочки на другую цепочку.

Определение 0.7. Языком L над словарем V называется произвольное множество цепочек над этим словарем.

Пример 0.2. $V = \{a, b\},\ L = \{a^n b^n | n \geq 0\}$.

Проблема задания бесконечного множества цепочек, составляющих нас язык. Необходим конечный механизм, который может описать бесконечное множество цепочек.

Определение 0.8. Любой конечный механизм задания языка называется грамматикой.

Существует два типа грамматик:

(1) Порождающая: за конечное число шагов построить правильные цепочки языка. Удобна и естественна для задания спецификации языка, по существу задает правила построения правильных предложений.
(2) Распознающая: за конечное число шагов определяет, принадлежит ли данная цепочка данному языку. По существу — алгоритм распознавания правильных цепочек языка для дальнейшего анализа и трансляции цепочек языка в некоторый код.

Определение 0.9. Порождающая грамматика Хомского называется четверка объектов $G = (T, N, S, R)$, где

T — термальный словарь, состоящий из терминальных символов (строчные латинские);

N — нетерминальный словарь, состоящий из нетерминальных символов (прописные латинские);

S из N называется начальным символом.

R — конечное непустое множество правил вывода (продукций) вида $\alpha \to \beta$; α, β из $(T \cup N)^*$.

Классификация грамматик Хомского:

- Тип 0: не накладывается никаких ограничений. Для распознавания необходимо абстрактное устройтво «машина Тьюринга».

- Тип 1: ограничения: $\gamma A\delta \to \gamma\sigma\delta$; A — нетерминал; грамматики типа 1 называются контекстно-зависимыми — для распознавания необходим «линейно ограниченный автомат».
- Тип 2: правила ограничения: $A \to \beta$, где A — терминал, а β — цепочка контекстно-зависимыми грамматика. Для распознавания необходим «конечный автомат с магазином».
- Тип 3: $A \to tN|t$; t терминал; N нетерминал; автоматная (регулярная) грамматика. Для распозанавания необходим «конечный автомат».
- Вложения: $T^3 \subset T^2 \subset T^1 \subset T^0$ и $L^3 \subset L^2 \subset L^1 \subset L^0$.

Преобразователи информации.

В информатике одна из изучаемых проблем — конечное преобразование информации, где один набор символов преобразуется в другой.

Определение 0.10. Конечный автомат — не функциональный преобразователь. Абстрактная машина, в которой есть конечное число состояний и сигналов, в зависимости от комбинации которых автомат меняет свое состояние.

Определение 0.11. (формальное определение детерминированного конечного автомата)

Детерминированный конечный автомат A представляет собой пятерку компонентов $A = (Q, \Sigma, \delta, q_0, F)$, где:

- Q — конечное множество состояний.
- Σ — конечное множество входных символов.
- σ — функция перехода $\sigma(q, a) = p$, где $p, q \in Q,\ a \in \Sigma$.
- q_0 — начальное состояние из Q.
- F — множество финальных (завершающих) состояний $F \in Q$.

Пример 0.3. ДКА, допускающий битовые цепочки, содержащие подцепочку 01.

Состояния:

- 01 не прочитана и либо автомат на предыдущем шаге ничего не читал, либо прочитан символ 1 (состояние q_0).
- 01 еще не прочитана, но автомат на предыдущем шаге прочитал 0 (состояние q_1).
- 01 прочитана.

Функции перехода:

- $\delta(q_0, 1) = q_0$; $\delta(q_0, 0) = q_1$;
- $\delta(q_1, 1) = q_2$; $\delta(q_1, 0) = q_1$;
- $\delta(q_2, 1) = q_2$; $\delta(q_2, 0) = q_2$;

$Q = \{q_0, q_1, q_2\},\ \Sigma = \{0, 1\},\ F =$ (что-то там).

Мы можем задать автомат с помощью диаграммы переходов.

- Всякому состоянию соответствует вершина графа.
- Для любого перехода ДКА $\delta(q, a) = p$, определяемого его функцией перехода, в графе существует дуга из вершины состояния q в вершину состояния p размеченная символом a.
- Существует стрелка в начальное состояни, не выходящая ни из одного состояния.
- Вершина из множества допускаемых состояний обозначается двойным кружком.

Или мы можем задать ДКА с помощью таблицы переходов.

- Таблица переходов представляет собой табличное представление функции перехода $\delta(q, a)$ (в левом столбце — состояния, в первой строке — символы алфавита).

Определение 0.12. Расширенная функция переходов $\widehat{\delta}(q, w)$ ставит в соответствие состоянию q и цепочке w состояние p, в которое попадает автомат из состояния q, обработав цепочку w.

Определение 0.13. Языком ДКА $A = (Q, \Sigma, \delta, q_0, F)$ называются цепочки... ну блин. В жопу такие конспекты

Определение 0.14. Недетерминированный конечный автомат — $A(Q, \Sigma, \delta, q_0, F)$, где все, кроме δ имеют те же значения, а δ —функция, аргументами которой являются состояние из Q и входной символ из Σ, а значением — подмножество множества Q. Различие между ДКА и НКА состоит в типе функции δ.

Может в каждый момент времени может находиться в нескольких состояниях.