Московский государственный университет имени М. В. Ломоносова
факультет вычислительной математики и кибернетики
кафедра алгоритмических языков

Дипломная работа

НЕКОТОРЫЕ АЛГОРИТМЫ НА ОСНОВЕ ГРАФОВЫХ ПРЕДСТАВЛЕНИЙ БЕСКОНТЕКСТНЫХ ЯЗЫКОВ

Выполнил: студент 524 группы
Сарафанов Андрей Михайлович

Научный руководитель: ст. преп., к.ф.-м.н.
Вылиток Алексей Александрович

Москва 2015

Содержание

Введение

Формальным языком называется множество цепочек конечной длины, состоящих из элементов некоторого непустого множества Σ. В этом случае говорят, что язык построен над алфавитом Σ. Цепочку нулевой длины обозначают символом ϵ. В теории формальных языков классической считается классификация формальных языков, основанная на т.н. иерархии Хомского, предложенной американским лингвистом Ноамом Хомским. Согласно ей, формальные языки делятся на 4 класса:

0. Рекурсивно перечислимые (неограниченные)
1. Контекстно-зависимые
2. Контекстно-свободные (бесконтекстные)
3. Регулярные

Стоит также отдельно отметить выделяемые в классе 2 подклассы детерминированых и недетерминированных контекстно-свободных языков.
Существует множество способов описания формальных грамматик, из них наиболее распространенными являются формальные грамматики и абстрактные вычислительные устройства, такие как машина Тьюринга, магазинные автоматы, конечные автоматы (скопировал у Касимовой) .
Формальные грамматики применяются для описания формальных языков всех классов и представляют из себя четверки вида (N, Σ, P, S), где

- N - конечное множество нетерминальных символов,
- Σ - конечное множество терминальных символов,
- P - конечное множество правил вывода вида $Left \to Right, Left \in (N \cup \Sigma)^+, Right \in (N \cup \Sigma)^*$,
- $S \in N$ - начальный символ.

Постановка задачи

Обзор имеющихся решений

Понятие L-графов (как лучше назвать?)

4.1 Определение L-графа

Определение 1. Пусть $\Sigma_($ и $\Sigma_)$ — непересекающиеся алфавиты, и существует биективное отображение $\phi : \Sigma_(\to \Sigma_)$. Тогда назовём непустое множество $P \subseteq \Sigma_(\times \Sigma_)$ *D-множеством*.
Определение 2. Пуст P - D-множество. Тогда назовём язык L_P, порождаемый грамматикой $S \to \Lambda | aSbS, (a, b) \in P$, *D-языком* (над D-множеством P).
Определение 3. Пусть $\Delta = \Sigma_(\cup \Sigma_)$. Определим отображение $\mu : \Delta^* \to \Delta^*$. Пусть $\omega \in \Delta^*$. Рассмотрим множество $W' = \{(\omega_1, \omega_2) | \omega_1, \omega_2 \in \Delta^*, (a, b) \in P, \omega = \omega_1 ab\omega_2\}$. Если W' пусто, то $\mu(\omega) = \omega$, иначе среди всех $\mu(\omega) = \mu(\omega_1\omega_2)$, где $|\omega_1|$ минимальна по всем парам $(\omega_1, \omega_2) \in W'$. Назовём μ *стирающим отображением*.
Определение 4. *L-графом* назовём восьмёрку
$G = (V, E, \Sigma, \Sigma_), \Sigma_(, P, S, F)$, где

- V — множество вершин, $S \in V$ - начальная вершина, $F \subseteq V$ — множество заключительных вершин;
- Σ - алфавит входных символов, $\Sigma_($ и $\Sigma_)$ - непересекающиеся алфавиты, P — D-множество над $\Sigma_($ и $\Sigma_)$;
- множество E описывает дуги и их символьные пометки: $E \subseteq \{(v_1, v_2, \alpha, \beta) | v_1, v_2 \in V, \alpha \in \Sigma \cup \{\epsilon\}, \beta \in \Sigma_(\cup \Sigma_)\}$.

4.2 Понятие ядра L-графа

4.3 Понятие памяти L-графа

Определение 5. Основные понятие, такие как *Sentences*, *sCore*, введем как у Касимовой.

Построение ДКА-кандидата

5.1 Построение всех (w,d)-остовных памятей L-графа

Определение 6. Будем говорить, что в маршруте T *фигурирует* память $m = (v, \gamma)$, если T можно представить в виде $T = T_1T_2$, где $end(T_1) = v, \mu(T) = \gamma$. Обозначим это соотношение как $HasMemory(T, m)$. Если в этом определении $T_2 = \epsilon$, то будем говорить, что *память* T *равна* m ($Mem(T) = m$). Или Mem два раза лучше не использовать?

Определение 7. *Множеством (w,d)-остовных памятей* L-графа G будем называть множество $Mem(G, w, d) = \{(v, \gamma) | \exists T \in sCore(G, w, d) : HasMemory(T, (v, \gamma))\}$.

а

Тут должен быть алгоритм построения этого множества, но простой обход в глубину/ширину, естественно, не подходит. Наличие хотя бы одного цикла приведет к зацикливанию.

5.2 Построение графа (w,d)-остовных памятей L-графа

Определение 8. *Графом* (w, d)*-остовных памятей* L-графа $G(V, E', \Sigma, \Sigma_(, \Sigma_), P, S, F)$ назовём пятёрку $MemGraph(G, w, d) = (M, F, S, E, \Sigma)$, где

- $M = Mem(G, w, d)$ — множество вершин графа, $S \in M$ - начальная вершина, $F \subseteq V$ — множество заключительных вершин;
- Σ - алфавит входных символов;
- множество E описывает дуги и их символьные пометки: $E \subseteq \{(m_1, m_2, \alpha) | m_1, m_2 \in M, \alpha \in \Sigma \cup \{\epsilon\}\}$.

В роли вершин графа выступает множество (w, d)-остовных памятей L-графа G. Опишем алгоритм создания множества E. Для каждой вершины $m_1 = (v_1, \gamma_1) \in M$ рассмотрим поочередно все дуги из E' вида $(v_1, v_2, \alpha, \beta)$. Для каждой вершины $m_2 = (v_2, \gamma_2) \in M$, такой что $\mu(\gamma_1\beta) = \gamma_2$ добавим в E дугу (m_1, m_2, α). Получилось криво, к тому же не ясно ещё, так ли стоит этот граф определять. Альтернативный вариант - добавлять дугу (m_1, m_2, α), если $\exists T \in sCore(G, w, d), T = T_1T_2T_3, mem(T_1) = m_1, mem(T_1T_2) = m_2, \omega(T_1T_2) = \omega(T_1)\beta$

5.3 Построение графа-кандидата

На основе $M_1 = MemGraph(G, 1, 1)$ и $M_2 = MemGraph(G, 2, 2)$ построим граф недетерминированного конечного автомата $Cand(G)$. Обозначим $ToRemove(G) = \{m | m \in M_2, m \notin M_1\}$. Определим отображение $Reduction : ToRemove(G) \to M_1$.

Проверка эквивалентности исходного L-графа и построенного ДКА

6.1 Алгоритм проверки эквивалентности ДКА и L-графа

Заключение

В рамках данной дипломной работы исследовалась проблема регулярности бесконтекстных языков, представленных в виде L-графов.
Предложено условие регулярности детерминированных L-графов: предложены алгоритм построения по детерминированному L-графу (?детерминированного?) конечного автомата, который будет эквивалентен исходному L-графу, только если тот регулярен, и алгоритм проверки эквивалентности детерминированного L-графа и (?детерминированного?) конечного автомата.
Также выделен подкласс детерминированных L-графов, на котором указанное условие регулярности является критерием.

Литература

1. Ахо А. Ульман Дж. Теория синтаксического анализа, перевода и компиляции. Синтаксический анализ. М.: Мир, 1986. Т. 1.

2. E Stearns Richard. A regularity test for pushdown machines // Information and control. 1967. Т. 11, № 3. С. 323–340.

3. Shankar Priti Adiga B. S. A Graph-Based Regularity Test for Deterministic Context-free Languages // Theor. Comput. Sci. 1991. Т. 88, № 1. С. 117–125.

4. Л.И. Станевичене. К теории бесконтекстных языков. М.: МГУ им. М.В. Ломоносова, 2000.

5. Vylitok A. Gomozov A. Stanevichene L. The power of printing ink // V-я международная конференция. Информатика. Образование. Экология и здоровье человека. Издательство Астраханского государственного педагогическиого университета Астрахань, 2000. С. 270–270.

6. G Valiant Leslie. Regularity and related problems for deterministic pushdown automata // Journal of the ACM. 1975. Т. 22.