Листок 7

1. Другое определение нормальной формы Хомского. Заметим, что грамматика в НФХ не может порождать ε, однако мы знаем КС-языки, содержащие ε. В действительности имеет место

Теорема 1. *Пусть G — КС-грамматика, а G' — грамматика в НФХ, полученная из G применением рассмотренного ранее алгоритма. Тогда*

$$L(G') = L(G) \setminus \{\varepsilon\}.$$

Можно сформулировать другое определение НФХ, которое с практической точки зрения не хуже нашего, но позволяет выводить ε.

Определение (другое определение нормальной формы Хомского)**.** Говорят, что КС-грамматика G находится в *нормальной форме Хомского*, если она не содержит бесполезных символов и каждая продукция грамматики имеет один из видов:

(1) $A \to a$,

(2) $A \to BC$,

(3) $S \to \varepsilon$,

где $a \in \Sigma$, $A, B, C, S \in N$, S — стартовый символ, *не встречающийся в правых частях продукций грамматики.*

Чтобы получить НФХ в смысле последнего определения достаточно добавить в алгоритм удаления ε-правил шаг 4:

> Если $S \in \mathrm{Gen}_G(\varepsilon)$, то ввести в грамматику новый стартовый символ S' и две продукции $S' \to S \mid \varepsilon$.

$\otimes$ Скорректировать решения заданий по получению НФХ так, чтобы ответом служила грамматика в НФХ в смысле второго определения.

2. Алгоритмические проблемы контекстно-свободных языков. Тремя основными проблемами теории формальных языков являются:

(1) *проблема пустоты*: для данной грамматики G определить

$$L(G) \stackrel{?}{=} \varnothing;$$

(2) *проблема принадлежности*: для данных грамматики G и слова $w \in \Sigma^*$ определить

$$w \stackrel{?}{\in} L(G);$$

(3) *проблема эквивалентности*: для данных грамматик G_1, G_2 определить

$$L(G_1) \stackrel{?}{=} L(G_2).$$

Проверка пустоты КС-языка сводится к построению $\mathrm{Gen}_G(\Sigma)$ и проверке $S \in \mathrm{Gen}_G(\Sigma)$. Рассмотрим один алгоритм, решающий проблему принадлежности для КС-языков.

Алгоритм (Кок—Янгер—Касами, «CYK-алгоритм»)**.**
Вход: грамматика $G = (\Sigma, N, \mathcal{P}, S \in N)$ в НФХ, слово $w \in \Sigma^*$.
Выход: да, $w \in L(G)$ / нет, $w \notin L(G)$.
Метод: последовательное определение нетерминалов, выводящих всевозможные подстроки w всё большей длины.
Пусть $w = w_1 \dots w_n$. Для всех $1 \leqslant i \leqslant j \leqslant n$ определим множество

$$N_{ij} = \{A \in N \mid A \Rightarrow_G^* w_i \dots w_j\}.$$

Очевидно, что $w \in L(G) \Leftrightarrow S \in N_{1n}$. Приведём алгоритм построения множеств N_{ij}.

for $i \leftarrow 1$ **to** n
  do $N_{ii} \leftarrow \{A \in N \mid A \to w_i \in \mathcal{P}\}$ ▷ Подстроки w длины 1
for $s \leftarrow 2$ **to** n ▷ Цикл по длине подстроки
  do for $i \leftarrow 1$ **to** $n - s + 1$ ▷ Цикл по месту начала подстроки
    $j \leftarrow i + s - 1$ ▷ Позиция конца подстроки с началом в w_i длины s
    $N_{ij} \leftarrow \{A \in N \mid A \to BC \in \mathcal{P};\ \exists k \in [i, j-1]_{\mathbb{Z}} \colon B \in N_{ik},\ C \in N_{k+1j}\}$

Замечание. Алгоритм удобно выполнять, заполняя таблицу с N_{ij} в ячейках.

Используя CYK-алгоритм,

(1) для грамматики G с продукциями:
$S \to AB, \quad A \to BB \mid a, \quad B \to AB \mid b$
определить, принадлежат ли $L(G)$ строки: (а) *aabbb*, (б) *babab*, (в) b^7;

(2) для грамматики G с продукциями:
$S \to AB \mid BC, \quad A \to BA \mid a, \quad B \to CC \mid b, \quad C \to AB \mid a$
определить, принадлежат ли $L(G)$ строки: (а) *ababa*, (б) *baaab*, (в) *aabab*.

Замечание 1 (о применении CYK-алгоритма к решению задачи синтаксического анализа)**.** Несложная модификация CYK-алгоритма позволяет в случае $w \in L(G)$ давать на выходе вывод w в G. С точки зрения теории синтаксического анализа CYK-алгоритм проводит *восходящий (bottom—up) анализ*.

Замечание 2 (о сложности CYK-алгоритма)**.** Нетрудно видеть, что сложность CYK-алгоритма может быть оценена как $O(n^3 \cdot |\mathcal{P}|)$, что ограничивает применение алгоритма на практике. Чаще всего в приложениях рассматривается подкласс КС-грамматик, *детерминированные КС-грамматики* (по-другому, $LL(k)$- и $LR(k)$-грамматики), для которых существуют линейные алгоритмы разбора (сложность $O(n)$).

Утверждение. Проблема эквивалентности КС-грамматик является неразрешимой.