Домашнее задание от 26.10.
Теоретическая информатика. 2 курс.
Решения.

Глеб Минаев @ 204 (20.Б04-мкн)

28 октября 2021 г.

Содержание

Задача 2. Пусть даны грамматика $G = (\Sigma, N, R, S)$ и ДКА $A = (\Sigma, Q, q_0, \delta, F)$. Давайте рассмотрим следующий алгоритм.

Будем всё время хранить некоторое семейство $\Omega = \{T_{q,A}\}_{\substack{q \in Q \\ A \in N}}$ подмножеств Q (т.е. $T_{q,A} \subseteq Q$). Мы хотим сделать так, чтобы

$$T_{q,A} = \{p \in Q \mid \exists w \in L_G(A)\colon\ \delta^*(q,w) = p\}.$$

Для этого рассмотрим Ω_0, где

$$T^0_{q,A} := \{\delta^*(q,w) \mid w \in \Sigma^* \wedge (A \to w) \in R\}.$$

Далее по каждому Ω_n будем строить Ω_{n+1} по правилу

$$T^{n+1}_{q,A} := \left\{ p \;\middle|\; \begin{gathered} \exists u_0, \dots, u_m \in \Sigma^*\colon \\ \exists B_1, \dots, B_m \in N\colon \\ \exists q_0, p_0, \dots, q_m, p_m \in Q\colon \\ (A \to u_0 B_1 \dots B_m u_m) \in R \\ \wedge \forall i\ \delta^*(q_i, u_i) = p_i \\ \wedge \forall i\ q_{i+1} \in T^n_{p_i, B_{i+1}} \\ \wedge\ q_0 = q \wedge p_m = p \end{gathered} \right\}.$$

Лемма 1. $T^n_{q,A} \subseteq T^{n+1}_{q,A}$.

Доказательство. Докажем по индукции по n.

Если $n = 0$, то для всякого $p \in T^n_{q,A}$ есть $w \in \Sigma^*$, что $(A \to w) \in R$ и $\delta^*(q,w) = p$. Следовательно, $(A \to u_0) \in R$, где $u_0 = w$, $m = 0$, $q_0 = q$, $p_0 = p$, $\delta^*(q_0, u_0) = p_0$. Таким образом $p \in T^{n+1}_{q,A}$. Следовательно, $T^n_{q,A} \subseteq T^{n+1}_{q,A}$.

Если $n > 0$, то для всякого $p \in T^n_{q,A}$ есть $(A \to u_0 B_1 \dots B_m u_m) \in R$, где $u_0, \dots, u_m \in \Sigma^*$, $B_1, \dots, B_m \in N$, и $q_0, p_0, \dots, q_m, p_m \in Q$, что для вского i верно $\delta^*(q_i, u_i) = p_i$ и $q_{i+1} \in T^{n-1}_{p_i, B_{i+1}}$, а также $q_0 = q$, $p_m = p$. Так как $T^{n-1}_{p_i, B_{i+1}} \subseteq T^n_{p_i, B_{i+1}}$ по предположению индукции, то $q_{i+1} \in T^n_{p_i, B_{i+1}}$, а значит $p \in T^{n+1}_{q,A}$. Следовательно, $T^n_{q,A} \subseteq T^{n+1}_{q,A}$. □

Лемма 2. *Если* $\Omega_n = \Omega_{n+1}$*, то* $\Omega_m = \Omega_{m+1}$ *для всякого* $m \geqslant n$*.*

Доказательство. Докажем утверждение по индукции по m.

Если $m = n$, то утверждение вырождается в условие. Тогда $m > n$. Тогда по предположению индукции $T^m_{p,B} = T^{m-1}_{p,B}$ для всех $p \in Q$ и $B \in N$. Тогда для всякого $p \in T^{m+1}_{q,A}$ есть $(A \to u_0 B_1 \dots B_k u_k) \in R$, где $u_0, \dots, u_k \in \Sigma^*$, $B_1, \dots, B_k \in N$, и $q_0, p_0, \dots, q_k, p_k \in Q$, что для вского i верно $\delta^*(q_i, u_i) = p_i$ и $q_{i+1} \in T^m_{p_i, B_{i+1}}$, а также $q_0 = q$, $p_k = p$. Но тогда $q_{i+1} \in T^{m-1}_{p_i, B_{i+1}}$. Следовательно, $p \in T^m_{q,A}$. Т.е. $T^{m+1}_{q,A} \subseteq T^m_{q,A}$ для всех $q \in Q$ и $A \in N$. □

Следствие 2.1. *Если* $\Omega_n = \Omega_{n+1}$*, то* $\Omega_m = \Omega_n$ *для всех* $m \geqslant n$*.*

Лемма 3. $T_{q,A} = \bigcup_{n=0}^{\infty} T^n_{q,A}$.

Доказательство. Сначала покажем по индукции по n, что $T^n_{q,A} \subseteq T_{q,A}$.

Если $n = 0$, то утверждение очевидно. Если $n > 0$, то для всякого $p \in T^n_{q,A}$ есть $(A \to u_0 B_1 \dots B_m u_m) \in R$, где $u_0, \dots, u_m \in \Sigma^*$, $B_1, \dots, B_m \in N$, и $q_0, p_0, \dots, q_m, p_m \in Q$, что для вского i верно $\delta^*(q_i, u_i) = p_i$ и $q_{i+1} \in T^{n-1}_{p_i, B_{i+1}}$, а также $q_0 = q$, $p_m = p$. При этом $q_{i+1} \in T^{n-1}_{p_i, B_{i+1}} \subseteq T_{p_i, B_{i+1}}$. Значит есть $v_i \in L_G(B_{i+1})$, что $\delta^*(p_i, v_i) = q_{i+1}$. Следовательно,

$$p_m = \delta^*(q_0, u_0 v_1 \dots v_m u_m).$$

При этом $u_0 v_1 \dots v_m u_m \in L_G(A)$. Таким образом $p \in T_{q,A}$. Следовательно, $T^n_{q,A} \subseteq T_{q,A}$.

Следовательно, $\bigcup_{n=0}^{\infty} T^n_{q,A} \subseteq T_{q,A}$.

Теперь покажем по индукции по размеру дерева разбора $w \in L_G(A)$, что $\delta(q, w) \in \bigcup_{n=0}^{\infty} T^n_{q,A}$.

Рассмотрим первую подстановку у w: $A \to u_0 B_1 \dots B_m u_m$. Следовательно, $w = u_0 v_1 \dots v_m u_m$, где $v_i \in L_G(B_i)$. Определим $q_i := \delta^*(q, u_0 v_1 \dots v_i)$, $p_i := \delta^*(q, u_0 v_1 \dots v_i u_i)$. Тогда $p_i = \delta^*(q_i, u_i)$, $q_{i+1} = \delta^*(p_i, v_{i+1})$. Так как деревья разборов всех v_i меньше изначального, то по предположению индукции $q_{i+1} \in \bigcup_{n=0}^{\infty} T^n_{p_i, B_{i+1}}$, а значит $q_{i+1} \in T^{n_i}_{p_i, B_{i+1}}$ для некоторого n_i. Пусть $N := \max_i n_i$. Тогда $q_{i+1} \in T^N_{p_i, B_{i+1}}$. Следовательно $\delta^*(q, w) \in T^{N+1}_{q,A}$. А тогда $\delta^*(q, w) \in \bigcup_{n=0}^{\infty} T^n_{q,A}$.

Следовательно, $T_{q,A} \subseteq \bigcup_{n=0}^{\infty} T^n_{q,A}$. □

Заметим, что Ω_0 строится алгоритмически, а Ω_{n+1} строится по Ω_n алгоритмически. Заметим, что последовательность $(\Omega_n)_{n=0}^{\infty}$ стабилизируется, так как при переходе от Ω_n к Ω_{n+1} каждое $T^n_{q,A}$ сохраняет все старые элементы и, возможно, подбирает новые, значит бесконечно "расти" данная последовательность не может. Поэтому чтобы построить Ω достаточно начать строить последовательность $(\Omega_n)_{n=0}^{\infty}$ и строить, пока последние два члена не совпадут. Тогда получится последовательность $(\Omega_n)_{n=0}^{k}$, что с Ω_{k-1} бесконечная последовательность стабилизируется. Тогда понятно, что

$$T_{q,A} = \bigcup_{n=0}^{k-1} T^n_{q,A}.$$

Таким образом можно алгоритмически построить Ω.

Как только мы построили Ω, вся задача заключается в проверке того, что $T_{q_0,S} \subseteq F$. Эта задача, очевидно, алгоритмически разрешима, а значит и вся задача алгоритмически разрешима.

Задача 4.

(a) Предъявим алгоритм, который распознаёт свойство $L(G_1) \neq L(G_2)$.

Действительно, давайте просто будем перебирать все слова подряд (это, понятно, алгоритмически разрешимо) и для каждого слова запустим алгоритм алгоритм Кокка—Касами—Янгера сначала на G_1, а потом на G_2. Если на каком-то слове ответы алгоритма Кокка—Касами—Янгера не совпали, то скажем "да". Иначе будем идти дальше.

Если $L(G_1) \neq L(G_2)$, то есть слово $w \in L(G_1) \triangle L(G_2)$. Тогда алгоритм рано или поздно дойдёт до него, получит разные ответы алгоритма Кокка—Касами—Янгера на нём и каждой из грамматик G_1 и G_2 и скажет "да". Если же $L(G_1) = L(G_2)$, то для всех слов ответы будут совпадать, а значит алгоритм просто не закончит свою работу.

Следовательно, свойство $L(G_1) \neq L(G_2)$ распознаётся. Следовательно, свойство $L(G_1) = L(G_2)$ не распознаётся, так как иначе бы $L(G_1) = L(G_2)$ было бы разрешимым. Но мы доказали на лекции обратное.

(b) Предъявим алгоритм, который распознаёт свойство $L(G_1) \cap L(G_2) \neq \varnothing$.

Действительно, давайте просто будем перебирать все слова подряд (это, понятно, алгоритмически разрешимо) и для каждого слова запустим алгоритм алгоритм Кокка—Касами—Янгера сначала на G_1, а потом на G_2. Если на каком-то слове ответы алгоритма Кокка—Касами—Янгера оба будут "да то скажем "да". Иначе будем идти дальше.

Если $L(G_1) \cap L(G_2) \neq \varnothing$, то есть слово $w \in L(G_1) \cap L(G_2)$. Тогда алгоритм рано или поздно дойдёт до него, получит оба ответа "да"и скажет "да". Если же $L(G_1) \cap L(G_2) = \varnothing$, то алгоритм не будет находить такого слова и не закончит свою работу.

(c) Предъявим алгоритм, который распознаёт свойство неоднозначности G_1.

Действительно, давайте просто будем перебирать все слова подряд (это, понятно, алгоритмически разрешимо) и для каждого слова запустим алгоритм алгоритм Кокка—Касами—Янгера на G_1. Далее по полученной таблице каждого слова будем восстанавливать все возможные деревья. Если на каком-то слове получилось два различных дерева разбора, то скажем "да". Иначе будем идти дальше.

Если G_1 неоднозначно, то есть слово w, которое неоднозначно задаётся грамматикой G_1. Тогда алгоритм рано или поздно дойдёт до него, получит два разных дерева по модификации алгоритма Кокка—Касами—Янгера на нём и грамматике G_1 и скажет "да". Если же G_1 однозначна, то для всех слов двух деревьев не найдётся, а значит алгоритм просто не закончит свою работу.

Следовательно, свойство неоднозначности G_1 распознаётся. Следовательно, свойство однозначности G_1 не распознаётся, так как иначе бы однозначаность G_1 была бы разрешима. Но мы доказали на лекции обратное.

Задача 5. Пусть дана грамматика $G = (\Sigma, N, R, S)$. Рассмотрим грамматику $G' = (\Sigma, N', R', S')$, где

- $N' := N \cup \{S_A\}_{A \in N} \cup \{S'\}$, где S_A — копия S для каждого $A \in N$,
- R' состоит из правил
  - все правила из R,
  - $S_A \to qS_Bp$ для всякого $(B \to pAq) \in R$ и всяких $A, B \in N$,
  - $S' \to qS_Ap$ для всякого $(A \to pq) \in R$ и всякого $A \in N$,

– $S_S \to \varepsilon$.

Покажем по индукции по размеру дерева w, что если $w \in L_{G'}(S_A)$, то есть некоторое разбиение $w = vu$, что uAv порождается S в G.

Рассмотрим первую подстановку. Если это подстановка $S_S \to \varepsilon$, то действительно, S порождается символом S в G. Тогда это $S_A \to qS_Bp$, где $(B \to pAq) \in R$. Далее по дереву p реализует некоторое слово u, а q — v. Причём поддерево S_B имеет меньший размер, чем размер дерева S_A, значит по предположению индукции S_B реализует какое-то слово $v'u'$, что $u'Bv'$ реализуется символом S в G. Тогда $w = vv'u'u$, а применяя правило $B \to pAq$, получаем что S в G также реализует $u'pAqv'$. Прикрепляя к p и q поддеревья, порождающие u и v, получаем, что S в G порождает ещё и $u'uAvv'$. Следовательно, $w = (vv')(u'u)$ — искомое разбиение, а $(u'u)A(vv')$ действительно порождается.

Теперь покажем по индукции по размеру дерева uAv, что для всякого выражения uAv $(u, v \in \Sigma^*)$, порождаемого из S в G, $vu \in L_{G'}(S_A)$.

Если $uAv = S$, то, действительно, $\varepsilon \in L_{G'}(S_S)$, так как есть правило $S_S \to \varepsilon$. Иначе дерево uAv нетривиально (состоит из > 1 вершин, а значит имеет хотя бы второй уровень). Тогда рассмотрим подстановку, которой получается A в uAv: $(B \to pAq) \in R$. Тогда p порождает некоторое слово u, q — v, а строка uAv разбивается как $u'uAvv'$. Тогда S в G порождает $u'Bv'$. Следовательно, $v'u' \in L_{G'}(S_B)$. Также в G' из S_A можно получить qS_Bp. Следовательно, $qv'u'p$ получается из S_A. Вставляя поддеревья для p и q, получаем, что $vv'u'u \in L_{G'}(S_A)$.

Теперь заметим, что если $w \in L_{G'}(S')$, то w — циклический сдвиг слова из $L_G(S)$. Действительно, первая подстановка — $S' \to qS_Ap$, где $(A \to pq) \in R$. Значит p порождает какое-то слово u, а q — v, т.е. uv порождается A в G. При этом S_A порождает какое-то слово $v'u'$, где $u'Av'$ порождается S в G. Значит $u'uvv' \in L(S)$. При этом $w = vv'u'u$.

А если $uv \in L_G(S)$, то $vu \in L_{G'}(S')$. Действительно, рассммотрим первый (нижний) нетерминал в дереве uv, который разрывается границей между u и v, — A. Тогда A порождает слева от границы u', а справа v' и тогда $u = u''u'$, $v = v'v''$. Причём пусть на место данной A была подставлена строка pq, где p реализует u', а q — v'. Тогда S' реализует qS_Ap, а значит и $v'S_Au'$, а значит и $v'v''u''u' = vu$, так как S_A раелизует $v''u''$ (так как S в G реализует $u''Av''$).

Следовательно, $L(G') = L_{G'}(S')$ — циклический сдвиг $L_G(S) = L(G)$.
