Теорема Париха. Стековые автоматы

Теория формальных языков

2023 г.

Теорема Париха

Скажем, что множество векторов полулинейно, если оно является конечным объединением множеств векторов вида $(i_1 + \sum p_{t,1} \cdot j_{t,1}, \dots, i_k + \sum p_{t,k} \cdot j_{p,k})$.

КС-язык $\mathscr{L}$ в алфавите Σ можно описать количественно: как множество $V_{\mathscr{L}}$ векторов $\{(k_{j,1}, \dots, k_{j,n}) \mid k_{j,i} = |w_j|_{a_i} \;\&\; w_j \in \mathscr{L}\}$.

. Если $\mathscr{L}$ — КС-язык, тогда $V_{\mathscr{L}}$ полулинейно.

Теорема Парихa

Пусть $V_{\mathscr{L}} = \{(k_{j,1}, \ldots, k_{j,n}) \mid k_{j,i} = |w_j|_{a_i} \;\&\; w_j \in \mathscr{L}\}$. Если $\mathscr{L}$ — КС-язык, тогда $V_{\mathscr{L}}$ полулинейно.

- Если $V_{\mathscr{L}}$ полулинеен, то $V_{h(\mathscr{L})}$ полулинеен (h — гомоморфизм).
- Если w принадлежит языку Шютценберже, то $\forall n(|w|_{(_n} = |w|_{)_n} = |w|_{[_n} = |w|_{]_n})$.
- Рассматриваем префиксные трассы вывода над ГНФ языка Шютценберже и выкидываем из них наиболее короткие отрезки накачки. Их длина ограничена $\Rightarrow$ ограничено их множество.

Следствия теоремы Париха

Множества регулярных и КС-языков над однобуквенным алфавитом совпадают.

Коммутативным образом всякого КС-языка является регулярный язык.

Свойства замкнутости КС-языков

Пусть стартовый нетерминал грамматики G_i — это S_i.

- КС-языки тривиально замкнуты относительно объединения и конкатенации. Объединение: добавим правило $S' \to S_1 \mid S_2$, конкатенация: добавим правило $S' \to S_1 S_2$.
- КС-языки замкнуты относительно реверсирования (достаточно реверсировать левые части правил), а также префиксных и суффиксных замыканий (упражнение после освоения материала по PDA).

Свойства замкнутости КС-языков

- КС-языки не замкнуты относительно пересечения. Универсальный контрпример: пусть \$ – символ-разделитель (отсутствует в словаре). Рассмотрим язык $\{w_1\$(.)^*\$w_3\}$, где слова w_1 и w_2 связаны порождающими правилами (т.е. структура $w_1\$w_3$ не регулярна), и язык $\{w_1\$w_2\$(.)^*\}$, где такими правилами связаны w_1 и w_2. Их пересечение вынудит существование нерегулярной зависимости между w_1 и одновременно w_2 и w_3, что порождает не две, как в КС-языках, а минимум три области накачки.
  Конкретные примеры: пара $L_1 = \{a^n\$a^n\$a^*\}$, $L_2 = \{a^n\$a^*\$a^n\}$; или пара $L_1' = \{w\$w^R\$a^*\}$, $L_2' = \{w\$(a|b)^*\$a^n \mid |w|_a = n\}$.

Свойства замкнутости КС-языков

- КС-языки не замкнуты относительно дополнения. В противном случае пересечение также не нарушало бы свойство контекстной свободы. Контпримеры строятся на основе этой же идеи: берём язык-пересечение КС-языков L_1 и L_2, не являющийся КС. Чаще всего его дополнение — КС-язык.
  Конкретные примеры: КС-язык $L' = \overline{L_1 \cap L_2} = \{a^m\$a^n\$a^k \mid m \neq n \vee m \neq k \vee n \neq k\}$ с не КС-дополнением; КС-язык $L'' = \overline{L_1' \cap L_2'} = \{w\$v\$a^n \mid v \neq w^R \vee n \neq |w|_a\}$ с не КС-дополнением.

Свойства замкнутости КС-языков

- КС-языки тривиально замкнуты относительно объединения и конкатенации.
- КС-языки замкнуты относительно реверсирования (достаточно реверсировать левые части правил), а также префиксных и суффиксных замыканий (упражнение после освоения материала по PDA).
- КС-языки не замкнуты относительно пересечения.
- КС-языки не замкнуты относительно дополнения.
- КС-языки замкнуты относительно пересечения с регулярным языком (см. ниже).

Пересечение КС-грамматики и рег. языка

Утверждение

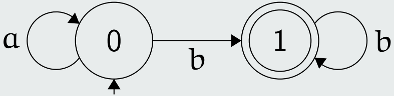
Даны КС-грамматика G и конечный автомат $\mathscr{A}$. Можно построить КС-грамматику G' такую, что $L(G') = L(G) \cap L(\mathscr{A})$.

Предположим, что G — в k-нормальной форме Хомского (т.е. с максимум k нетерминалами в правых частях), q — множество состояний автомата $\mathscr{A}$, q_f — единственное финальное состояние, N — множество нетерминалов грамматики G. Множество нетерминалов G' — множество $\langle q_i, A, q_j \rangle$, $q_i, q_j \in q$, $A \in N$.

- По каждому правилу $A \to A_1 \dots A_n$ из G строим правила $\langle p, A, q \rangle \to \langle p, A_1, q_1 \rangle \langle q_{n-1}, A_n, q \rangle$ для всех возможных p, q, q_i.
- По правилу вида $A \to t$ из G и переходу $p \xrightarrow{t} q$ строим правило $\langle p, A, q \rangle \to t$.
- Нетерминал $\langle q_0, S, q_f \rangle$ объявляем стартовым.

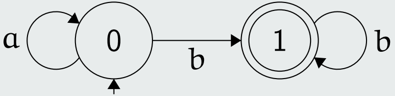
Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$,
$T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



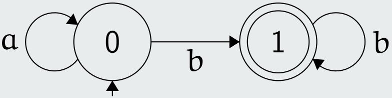
Сначала разберёмся с правилами вида $X \to t$. Если $t = a$, тогда подходящий нетерминал — только G_A, состояния — только 0+0. Если $t = b$, получается четыре комбинации состояний и нетерминалов.

$\langle 0, G_B, 1\rangle \to b \quad \langle 1, G_B, 1\rangle \to b$

$\langle 0, T, 1\rangle \to b \quad \langle 1, T, 1\rangle \to b \quad \langle 0, G_A, 0\rangle \to a$

Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



$\langle 0, G_B, 1\rangle \to b \quad \langle 1, G_B, 1\rangle \to b$

$\langle 0, T, 1\rangle \to b \quad \langle 1, T, 1\rangle \to b \quad \langle 0, G_A, 0\rangle \to a$

Рассмотрим возможные подстановки состояний в правила, порождаемые $S \to G_A T$, $T \to SG_B$. Соответствующие уравнения:

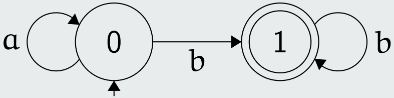
$\langle X1, S, X2\rangle \to \langle X1, G_A, X3\rangle\langle X3, T, X2\rangle$

$\langle Y1, T, Y2\rangle \to \langle Y1, S, Y3\rangle\langle Y3, G_B, Y2\rangle$

Чтобы правила были порождающими, необходимо положить $X1 = X3 = 0$, $Y2 = 1$. Выпишем все такие правила. Заметим, что получившийся в одном из них нетерминал $\langle 0, T, 0\rangle$ — непорождающий, и удалим это правило.

Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



$\langle 0, G_B, 1\rangle \to b$ $\quad\quad$ $\langle 1, G_B, 1\rangle \to b$

$\langle 0, T, 1\rangle \to b$ $\quad\quad$ $\langle 1, T, 1\rangle \to b$ $\quad\quad$ $\langle 0, G_A, 0\rangle \to a$

$\color{blue}{\langle 0, S, 0\rangle \to \langle 0, G_A, 0\rangle\langle 0, T, 0\rangle}$ $\quad\quad$ $\langle 0, S, 1\rangle \to \langle 0, G_A, 0\rangle\langle 0, T, 1\rangle$

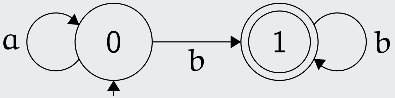
$\langle 0, T, 1\rangle \to \langle 0, S, 0\rangle\langle 0, G_B, 1\rangle$ $\quad\quad$ $\langle 0, T, 1\rangle \to \langle 0, S, 1\rangle\langle 1, G_B, 1\rangle$

$\langle 1, T, 1\rangle \to \langle 1, S, 0\rangle\langle 0, G_B, 1\rangle$ $\quad\quad$ $\langle 1, T, 1\rangle \to \langle 1, S, 1\rangle\langle 1, G_B, 1\rangle$

Осталось разобраться с правилами, порождёнными $S \to SS$. Выпишем их общий вид: $\langle \color{red}{X1}, S, \color{red}{X2}\rangle \to \langle \color{red}{X1}, S, \color{red}{X3}\rangle\langle \color{red}{X3}, S, \color{red}{X2}\rangle$.

Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



$\langle 0, G_B, 1\rangle \to b$ $\quad$ $\langle 1, G_B, 1\rangle \to b$

$\langle 0, T, 1\rangle \to b$ $\quad$ $\langle 1, T, 1\rangle \to b$ $\quad$ $\langle 0, G_A, 0\rangle \to a$

$\langle 0, S, 1\rangle \to \langle 0, G_A, 0\rangle\langle 0, T, 1\rangle$

$\langle 0, T, 1\rangle \to \langle 0, S, 0\rangle\langle 0, G_B, 1\rangle$ $\quad$ $\langle 0, T, 1\rangle \to \langle 0, S, 1\rangle\langle 1, G_B, 1\rangle$

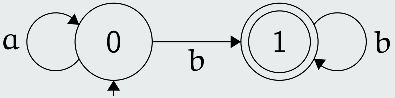
$\langle 1, T, 1\rangle \to \langle 1, S, 0\rangle\langle 0, G_B, 1\rangle$ $\quad$ $\langle 1, T, 1\rangle \to \langle 1, S, 1\rangle\langle 1, G_B, 1\rangle$

Осталось разобраться с правилами, порождёнными $S \to SS$. Выпишем их общий вид: $\langle X1, S, X2\rangle \to \langle X1, S, X3\rangle\langle X3, S, X2\rangle$.

Если положить $X1 = 1$, $X2 = 0$, получим саморекурсивное правило $\langle 1, S, 0\rangle \to \alpha_1\langle 1, S, 0\rangle\alpha_2$. Но в построенной части грамматики нет правил вида $\langle 1, S, \ldots\rangle \to \beta$. Поэтому нетерминал $\langle 1, S, 0\rangle$ — непорождающий. Удалим правила с его вхождением.

Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



$\langle 0, G_B, 1\rangle \to b$

$\langle 0, T, 1\rangle \to b$

$\langle 0, T, 1\rangle \to \langle 0, S, 0\rangle\langle 0, G_B, 1\rangle$

$\langle 1, G_B, 1\rangle \to b$

$\langle 1, T, 1\rangle \to b$ $\quad\langle 0, G_A, 0\rangle \to a$

$\langle 0, S, 1\rangle \to \langle 0, G_A, 0\rangle\langle 0, T, 1\rangle$

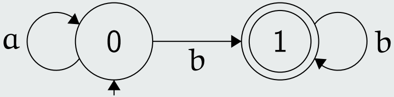
$\langle 0, T, 1\rangle \to \langle 0, S, 1\rangle\langle 1, G_B, 1\rangle$

$\langle 1, T, 1\rangle \to \langle 1, S, 1\rangle\langle 1, G_B, 1\rangle$

Теперь если $X1 = X2 = 1$, то единственный вариант развёртки $S \to SS$ без участия нетерминала $\langle 1, S, 0\rangle$ будет иметь вид $\langle 1, S, 1\rangle \to \langle 1, S, 1\rangle\langle 1, S, 1\rangle$, так что нетерминал $\langle 1, S, 1\rangle$ тоже непорождающий.

Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



$\langle 0, G_B, 1\rangle \to b$

$\langle 0, T, 1\rangle \to b$

$\langle 0, T, 1\rangle \to \langle 0, S, 0\rangle\langle 0, G_B, 1\rangle$

$\langle 1, G_B, 1\rangle \to b$

$\langle 1, T, 1\rangle \to b \qquad \langle 0, G_A, 0\rangle \to a$

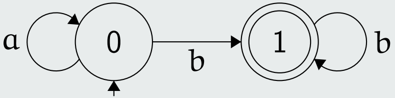
$\langle 0, S, 1\rangle \to \langle 0, G_A, 0\rangle\langle 0, T, 1\rangle$

$\langle 0, T, 1\rangle \to \langle 0, S, 1\rangle\langle 1, G_B, 1\rangle$

Аналогичным образом устанавливаем бесполезность нетерминала $\langle 0, S, 0\rangle$, который обязан ссылаться либо дважды на себя, либо на непорождающий $\langle 1, S, 0\rangle$.

Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



$\langle 0, G_B, 1\rangle \to b$ $\quad$ $\langle 1, G_B, 1\rangle \to b$

$\langle 0, T, 1\rangle \to b$ $\quad$ $\langle 1, T, 1\rangle \to b$ $\quad$ $\langle 0, G_A, 0\rangle \to a$

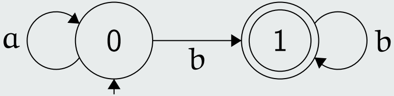
$\langle 0, S, 1\rangle \to \langle 0, G_A, 0\rangle\langle 0, T, 1\rangle$

$\langle 0, T, 1\rangle \to \langle 0, S, 1\rangle\langle 1, G_B, 1\rangle$

Теперь получается, что все варианты раскрытия нетерминала $\langle 0, S, 1\rangle$ по правилу $S \to SS$ включают непорождающие нетерминалы, поэтому никаких других правил в грамматику добавлять не надо. Осталось только удалить правила с недостижимыми нетерминалами $\langle 0, G_B, 1\rangle$, $\langle 1, T, 1\rangle$.

Пример

Построим пересечение языков CFG $S \to G_A T \mid SS$, $T \to b \mid SG_B$, $G_A \to a$, $G_B \to b$, и следующего автомата:



$\langle 0, G_A, 0 \rangle \to a$ $\qquad$ $\langle 1, G_B, 1 \rangle \to b$

$\langle 0, T, 1 \rangle \to b$

$\langle 0, S, 1 \rangle \to \langle 0, G_A, 0 \rangle \langle 0, T, 1 \rangle$ $\qquad$ $\langle 0, T, 1 \rangle \to \langle 0, S, 1 \rangle \langle 1, G_B, 1 \rangle$

Грамматика пересечения языков построена.

Базовый парсинг по КС-грамматике

На идее пересечения с автоматом можно построить примитивный алгоритм разбора, работающий с произвольной КС-грамматикой G в н.ф. Хомского: построить простой ДКА $\mathscr{A}$ с $n+1$ состояниями по слову ω (где $|\omega| = n$) и пересечь его с G, после чего проверить результат на пустоту. Переход по букве ω_i в ДКА $\mathscr{A}$ сопоставим переходу из состояния $i-1$ в i; стартовое состояние назовём 0. Правила пересечения уточнятся так:

- Правила $A \to \gamma$ будут соответствовать переходам $\langle i-1, A, i\rangle \to \gamma$, если $\gamma = \omega_i$.
- Правила $A \to BC$ будут соответствовать переходам $\langle i, A, i+k_1+k_2\rangle \to \langle i, B, i+k_1\rangle\langle i+k_1, C, i+k_1+k_2\rangle$, для всех $k_1+k_2 \leqslant |\omega| - i$, $k_1 > 0$, $k_2 > 0$, $i \geqslant 0$.

Очевидно, правила для пересечений $A \to BC$ более эффективно строить из базовых для $A \to \gamma$ «снизу вверх»: сначала для $k_1 = k_2 = 1$, затем для $k_i = 1$, $k_j = 2$, и т.д.

Алгоритм Кока–Янгера–Касами (CYK)

Задача

Дано слово $\omega_1 \ldots \omega_n \in \Sigma^+$ и грамматика G в н.ф. Хомского. Проверить, выполнено ли $\omega \in \mathscr{L}(G)$.

Идея алгоритма: переход к более простым задачам порождения подстрок $\omega_i \ldots \omega_{i+k}$.

Алгоритм Кока–Янгера–Касами (CYK)

Задача

Дано слово $\omega_1 \dots \omega_n \in \Sigma^+$ и грамматика G в н.ф. Хомского. Проверить, выполнено ли $\omega \in \mathscr{L}(G)$.

Идея алгоритма: переход к более простым задачам порождения подстрок $\omega_i \dots \omega_{i+k}$.

Определим функцию $f(A, i, j)$ (где $i \leqslant j$), возвращающую ответ, можно ли вывести слово $\omega_i \dots \omega_j$ из $A \in N$.

- Если $i = j$, тогда $f(A, i, j) = T \Leftrightarrow A \to \omega_i \in P$, и $f(A, i, j) = F$ иначе.
- Если $i < j$, тогда

$$f(A, i, j) = \bigvee_{(A \to B\,C \in P)} \bigvee_{k=i+1}^{j} \big(f(B, i, k-1)\ \&\ f(C, k, j)\big).$$

Это в точности быстрый алгоритм построения $G \cap \mathscr{A}(\omega)$.

Асимптотика CYK

- По уже существующей грамматике G в н.ф. Хомского и слову ω — $O(|\omega|^3)$.
- Но строить н.ф. Хомского по произвольной G — экспоненциально дорого от числа правил в G.
- Выход — применить алгоритм эффективного пересечения с ДКА к ненормализованной грамматике:
  - Правила $A \to \varepsilon$ снимают ограничение $k_i > 0$ (если $\varepsilon \in \mathscr{L}(A)$, тогда возможно введение нетерминала с индексами $\langle i, A, i \rangle$) (динамическое замыкание по ε).
  - Правила $A \to B$ порождают правила вида $\langle i, A, j \rangle \to \langle i, B, j \rangle$ (динамическое замыкание по цепным правилам).
- Обратно, алгоритм сборки $G \cap \mathscr{A}(\omega)$ снизу вверх можно перенести на пересечение с произвольным автоматом $\mathscr{A}$.

Стековая память

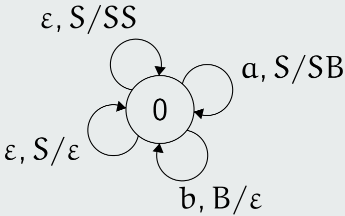
Пусть G — CFG. Неформально представим, что G — это стековый автомат, где состояния стека — нетерминальные сент. формы, порождаемые G. Скажем, что G распознаёт только слова, соответствующие пустому стеку.

Стековая память

Пусть G — CFG. Неформально представим, что G — это стековый автомат, где состояния стека — нетерминальные сент. формы, порождаемые G. Скажем, что G распознаёт только слова, соответствующие пустому стеку.

Грамматика и её стек

$$S \to aSB \mid SS \mid \varepsilon \qquad B \to b$$

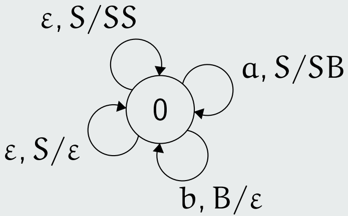


Стековая память

Пусть G — CFG. Неформально представим, что G — это стековый автомат, где состояния стека — нетерминальные сент. формы, порождаемые G. Скажем, что G распознаёт только слова, соответствующие пустому стеку.

Грамматика и её стек

$$S \to aSB \mid SS \mid \varepsilon \qquad B \to b$$



А если в такие автоматы добавить ещё состояния?

Pushdown Automata

Определение

Стековый автомат $\mathscr{A}$ — кортеж $\langle \Pi, \Sigma, Q, \delta, q_0, Z_0 \rangle$, где:

- Π — алфавит стека;
- Σ — алфавит языка;
- Q — множество состояний;
- δ — правила перехода вида $\langle q_i, t, P_i \rangle \to \langle q_j, \alpha \rangle$, где $t \in \Sigma \cup \{\varepsilon\}$, $\alpha \in \Pi^*$;
- q_0 — стартовое состояние, Z_0 – дно стека.

Pushdown Automata

Определение

Стековый автомат $\mathscr{A}$ — кортеж $\langle \Pi, \Sigma, Q, \delta, q_0, Z_0 \rangle$, где:

- Π — алфавит стека;
- Σ — алфавит языка;
- Q — множество состояний;
- δ — правила перехода вида $\langle q_i, t, P_i \rangle \to \langle q_j, \alpha \rangle$, где $t \in \Sigma \cup \{\varepsilon\}$, $\alpha \in \Pi^*$;
- q_0 — стартовое состояние, Z_0 – дно стека.

Два варианта допуска слова:

- если слово полностью прочитано, и стек пуст;
- если слово полностью прочитано, и состояние финальное.

Виды допуска

Утверждение

PDA с допуском по конечному состоянию распознают те же языки, что и PDA с допуском по пустому стеку.

Виды допуска

Утверждение

PDA с допуском по конечному состоянию распознают те же языки, что и PDA с допуском по пустому стеку.

- Пусть PDA допускает пустой стек. Добавим новый символ дна Z_1 и добавим по нему ε-переходы из всех состояний в новое финальное состояние.

Виды допуска

Утверждение

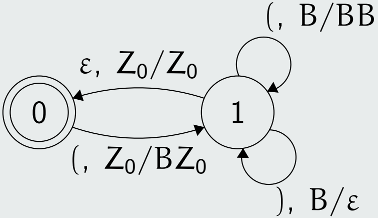
PDA с допуском по конечному состоянию распознают те же языки, что и PDA с допуском по пустому стеку.

- Пусть PDA допускает пустой стек. Добавим новый символ дна Z_1 и добавим по нему ε-переходы из всех состояний в новое финальное состояние.
- Пусть PDA допускает финальные состояния. Добавим из них ε-переходы в состояние, опустошающее стек, а также новый символ стека Z_1 и новое стартовое состояние q_0' с переходом $\langle q_0', \varepsilon, Z_0 \rangle \to \langle q_0, Z_0 Z_1 \rangle$.

Пример оформления PDA

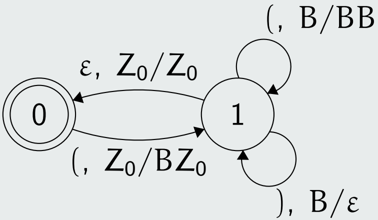
Обычно PDA изображается в виде автомата, в котором стрелки помечены сигнатурой $\alpha,\ \mathsf{T}/\Phi$, где α — это символ терминального алфавита (или пустое слово), T — символ на вершине стека, Φ — последовательность стековых символов, помещаемая на вершину стека вместо T.

Следующий PDA распознаёт правильные скобочные последовательности (включая пустое слово).



Пример оформления PDA

Следующий PDA распознаёт правильные скобочные последовательности (включая пустое слово).



Заметим, что перехода из состояния 0 по символу) нет. Так же как и в случае конечных автоматов, можно добавить для такого перехода состояние-ловушку, потому что он порождает слово, в префиксе которого количество закрывающих скобок превышает количество открывающих, а такие слова не являются ПСП.

От CFG к PDA

Утверждение

По всякой CFG G можно построить PDA $\mathscr{A}$ такой, что $L(G) = L(\mathscr{A})$.

От CFG к PDA

Утверждение

По всякой CFG G можно построить PDA $\mathscr{A}$ такой, что $L(G) = L(\mathscr{A})$.

Переведём G в GNF и построим по ней PDA с единственным состоянием 0 и допуском по пустому стеку, такой что $Z_0 = S$, правилу $A \to a$ соответствует переход $(0, a, A) \to (0, \varepsilon)$; правилу $A \to aB_1 \dots B_n$ — переход $(0, a, A) \to (0, B_1 \dots B_n)$.

От PDA к CFG

Утверждение

По всякому PDA $\mathscr{A}$ можно построить CFG G такую, что $\mathsf{L}(\mathsf{G}) = \mathsf{L}(\mathscr{A})$.

От PDA к CFG

Утверждение

По всякому PDA $\mathscr{A}$ можно построить CFG G такую, что $L(G) = L(\mathscr{A})$.

Пусть $\mathscr{A}$ допускает слова по пустому стеку.

- Построим по стеку $\mathscr{A}$ вспомогательную G':
  - введём новые стековые символы и заменим правила $\langle q_i, t, A \rangle \to \langle q_j, A_1 \dots A_n \rangle$ $(n \geqslant 1)$ на пары $\langle q_i, \varepsilon, A \rangle \to \langle q_i, A_0 \dots A_n \rangle$, $\langle q_i, t, A_0 \rangle \to \langle q_j, \varepsilon \rangle$.
  - переход $\langle q_i, \varepsilon, A \rangle \to \langle q_j, A_0 A_1 \dots A_n \rangle$ поставим в соответствие правилу $A \to A_0 A_1 \dots A_n$; переход $\langle q_i, t, A \rangle \to \langle q_j, \varepsilon \rangle$ поставим в соответствие правилу $A \to t_{i,j}$. Z_0 объявим стартовым символом. Пустой символ введём явно и так же пометим.

От PDA к CFG

Пусть $\mathscr{A}$ допускает слова по пустому стеку.

- Построим по стеку $\mathscr{A}$ вспомогательную G':
  - введём новые стековые символы и заменим правила $(q_i, t, A) \to (q_j, A_1 \ldots A_n)$ $(n \geqslant 1)$ на пары $(q_i, \varepsilon, A) \to (q_i, A_0 \ldots A_n)$, $(q_i, t, A_0) \to (q_j, \varepsilon)$.
  - переход $(q_i, \varepsilon, A) \to (q_j, A_0 A_1 \ldots A_n)$ поставим в соответствие правилу $A \to A_0 A_1 \ldots A_n$; переход $(q_i, t, A) \to (q_j, \varepsilon)$ поставим в соответствие правилу $A \to t_{i.j}$. Z_0 объявим стартовым символом. Пустой символ введём явно и так же пометим.
- Построим $\mathscr{A}'$ — FA с правилами вида $(q_i, t_{i,j}) \to q_j$, если для каких-нибудь A, α $(q_i, t, A) \to (q_j, \alpha)$ — переход $\mathscr{A}$. Все состояния объявим финальными.

От PDA к CFG

Пусть $\mathscr{A}$ допускает слова по пустому стеку.

- Построим по стеку $\mathscr{A}$ вспомогательную G':
    - введём новые стековые символы и заменим правила $(q_i, t, A) \to (q_j, A_1 \dots A_n)$ $(n \geqslant 1)$ на пары $(q_i, \varepsilon, A) \to (q_i, A_0 \dots A_n)$, $(q_i, t, A_0) \to (q_j, \varepsilon)$.
    - переход $(q_i, \varepsilon, A) \to (q_j, A_0 A_1 \dots A_n)$ поставим в соответствие правилу $A \to A_0 A_1 \dots A_n$; переход $(q_i, t, A) \to (q_j, \varepsilon)$ поставим в соответствие правилу $A \to t_{i.j}$. Z_0 объявим стартовым символом. Пустой символ введём явно и так же пометим.
- Построим $\mathscr{A}'$ — FA с правилами вида $(q_i, t_{i,j}) \to q_j$, если для каких-нибудь A, α $(q_i, t, A) \to (q_j, \alpha)$ — переход $\mathscr{A}$. Все состояния объявим финальными.
- Теперь построим CFG — пересечение G' и $\mathscr{A}'$ и сотрем все $\varepsilon_{i,j}$ и разметку терминалов. Грамматика G готова!

PDA в CFG формально

- Нетерминалы — тройки $[p, A, q]$, где $p, q \in Q$, $A \in \Pi$.
- По каждому переходу вида $(q, t, A) \to (p, A_1 \ldots A_n)$ добавим правила для всех возможных q_i вида $[q, A, q_n] \to t[p, A_1, q_1] \ldots [q_{n-1}, A_n, q_n]$.
- По каждому переходу вида $(q, t, A) \to (p, \varepsilon)$ добавим правило $[q, A, p] \to t$.
- Разрешим стартовому состоянию переписываться в любое из $[q_0, Z_0, q]$.

DPDA

Определение

PDA $\mathscr{A}$ детерминированный, если:

- если есть переход $\langle q, \varepsilon, Z\rangle \to \ldots$, то больше никаких переходов по Z из состояния q нет;
- каждой тройке $\langle q, a, Z\rangle$, $a \in \Sigma$, соответствует не больше одной правой части.

DPDA слабее, чем NPDA. Например, язык $\{a^n b^m \mid n = m \vee m = 2 * n\}$ не распознается DPDA. DPDA с допуском по пустому стеку ещё слабее — язык $\{a^n\}$ не может быть распознан DPDA с таким допуском.

DPDA

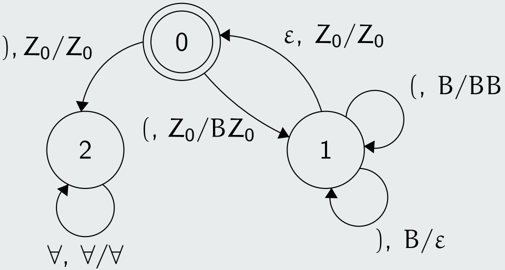
DPDA слабее, чем NPDA. Например, язык $\{a^n b^m \mid n = m \vee m = 2 * n\}$ не распознается DPDA. DPDA с допуском по пустому стеку ещё слабее — язык $\{a^n\}$ не может быть распознан DPDA с таким допуском.

Предположим, что существует DPDA, распознающий язык $\{a^n b^m \mid n = m \vee m = 2 * n\}$. Тогда после чтения префикса $a^n b^n$ слова $a^n b^{2n}$ он должен находиться в финальном состоянии. Далее он должен распознать ровно n букв b. Заменим часть автомата, распознающую этот фрагмент слова, на изоморфную ей, но читающую только буквы c. Получим PDA, распознающий язык $\{a^n b^n\} \cup \{a^n b^n c^n\}$, не являющийся КС.

Предположим, что существует DPDA с допуском по пустому стеку, распознающий язык $\{a^n\}$. Тогда на слове a стек этого автомата должен быть уже точно пуст $\Rightarrow$ в этом состоянии вообще невозможно сделать дальнейшие переходы.

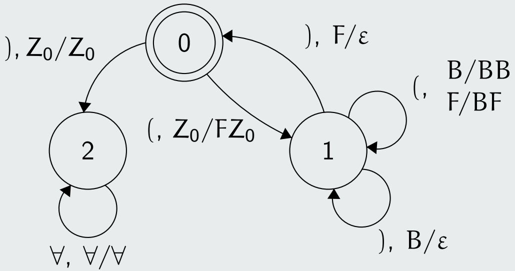
Пример DPDA

PDA для ПСП, приведённый выше, является DPDA, в чём нетрудно убедиться, проверив, что ε-переход совершается лишь в том случае, когда никакие другие совершить невозможно. Добавим в него состояние-ловушку.



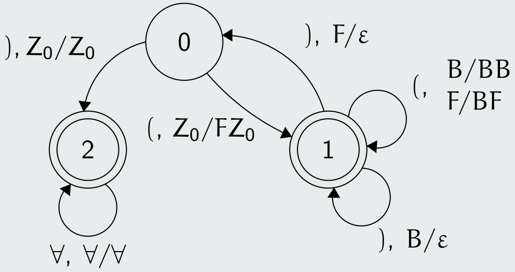
Пример DPDA

Чтобы не описывать многочисленные переходы из состояния-ловушки в себя по всем парам «символ ленты — символ стека», мы воспользовались сокращённым обозначением $\forall,\ \forall/\forall$, подразумевая следующее: «по любой паре $\langle$ терминал, символ стека $\rangle$ в состоянии 2 переходим в себя, сохраняя символ стека на вершине». Также избавимся от ε-перехода, введя символ стека F, т.е. «самая первая скобка».



Пример DPDA

Поскольку автомат $\mathscr{A}$ — детерминированный, в нём существуют переходы по всем комбинациям ⟨ терминал, символ стека ⟩, и нет ε-переходов, связывающих нефинальное и финальное состояния, то автомат, в котором все конечные состояния $\mathscr{A}$ заменены на нефинальные и наоборот, распознаёт дополнение языка, распознаваемого PDA $\mathscr{A}$. Значит, мы показали, что дополнение языка ПСП контекстно-свободно, и предъявили PDA, который распознаёт его.



Двухсторонние PDA

Утверждение

Двухсторонние PDA распознают больше языков, чем односторонние.

Доказательство: язык $\{a^n b^n c^n\}$ распознаваем двухсторонним PDA.