Deterministické zásobníkové automaty

Vrátíme se ještě k problému syntaktické analýzy. Uvedli jsme si jednak dva typy nedeterministické analýzy a poté algoritmus Cocke-Younger-Kasami se složitostí $O(n^3)$, kde n je délka vstupního slova. Zmínili jsme se také o tom, že pro jisté typy gramatik existuje deterministická analýza o lineární složitosti, která funguje na principu konstrukce deterministického zásobníkového automatu a jeho výpočtu. Sami uznáte, že lineární složitost je lákavá věc. Zájemci o hlubší poznání takto fungující analýzy odkazuji na kurz IA006 Vybrané partie z teorie automatů.

V tomto materiálu si podrobně popíšeme deterministický PDA a deterministický bezkontextový jazyk (dále též DCFL), následně probereme uzávěrové vlastnosti DCFL a popíšeme vztah obyčejných CFL a DCFL.

Definice – deterministický PDA

Řekneme, že zásobníkový automat $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ je **deterministický** (DPDA), jestliže jsou splněny podmínky:

1. pro všechna $q \in Q$ a $Z \in \Gamma$ platí: kdykoliv $\delta(q, \varepsilon, Z) \neq \emptyset$, pak $\delta(q, a, Z) = \emptyset$ pro všechna $a \in \Sigma$;
2. pro žádné $q \in Q, Z \in \Gamma$ a $a \in \Sigma \cup \{\varepsilon\}$ neobsahuje $\delta(q, a, Z)$ více než jeden prvek.

Poznámka.

Povšimněte si hlavně první podmínky. Je v ní řečeno, že pokud v nějakém stavu q s vrcholem zásobníku Z může automat pokračovat bez načtení symbolu ze vstupní pásky, pak nesmí mít možnost zvolit v dalším opakování jinou variantu, a to naopak číst symbol a ze vstupu. V případě, že by to bylo možné, jedná se o nedeterministické rozhodování, zda číst ze vstupu či nikoliv.

Definice – deterministický bezkontextový jazyk

Řekneme, že L je deterministický bezkontextový jazyk (DCFL), právě když existuje DPDA M takový, že $L = L(M)$.

Uzávěrové vlastnosti DCFL

V následujícím si na základě rozdílných uzávěrových vlastností třídy DCFL a CFL ukážeme, že tyto třídy jsou rozdílné.

Věta 1 (uzavřenost na doplněk).

Třída deterministických bezkontextových jazyků je uzavřena na operace doplněk do jazyka.

Poznámka.

Důkaz předchozí věty je poněkud komplikovanější, proto si uvedeme pouze jeho ideu.

Idea důkazu.

Základní myšlenkou důkazu je použití stejného principu jako v případě regulárních jazyků. Máme-li deterministický zásobníkový automat $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ a hledáme automat M' rozpoznávající jazyk co-$L(M)$, jednoduše zaměníme koncové stavy za nekoncové a naopak. Je tu však několik komplikací, které musíme ošetřit. Jsou celkem 4 a jádro důkazu je založeno právě na řešení těchto problémů. My si je popíšeme a pouze naznačíme, jak by se daly řešit.

1. **komplikace:** DPDA M nemusí vstupní slovo dočíst do konce, protože vyprázdní zásobník nebo přechod není definován.

   M tedy dané slovo nepřijímá, záměnou koncových a nekoncových stavů však v případě automatu M' dosáhneme stejného efektu. To je však chyba, M' by měl slovo přijmout. Řešením je přidání nového zásobníkového symbolu $D \notin \Gamma$, který určíme jako dno zásobníku M'. Pokud M měl prázdný zásobník, M' má na vrcholu pouze D. V takovém případě se přepneme do nějakého nového koncového stavu $d \notin Q$, ve kterém přečteme celý vstup a akceptujeme.

   Formálně: $\delta'(q, \varepsilon, D) = \{(d, D)\},\ \delta'(d, a, D) = \{(d, D)\}$ pro libovolný stav $q \in Q,\ a \in \Sigma$.

   Ještě bychom měli uvažovat situace, kdy pro nějakou trojici $q \in Q, Z \in \Gamma, a \in \Sigma$ není přechod $\delta(q, a, Z)$ definován. V tom případě doplníme přechodovou funkci δ' automatu M takto: $\delta'(q, a, Z) = (d, \varepsilon)$.

2. **komplikace:** DPDA M nepřečte vstupní slovo do konce, protože přestane číst vstup a provádí ε-kroky, kterými pouze přidává na zásobník další symboly a ten neomezeně roste.

   Pro tento případ si zavedeme následující symboly: $s = |Q|,\ t = |\Gamma|$ a

   $$r = \max\{|\gamma|;\ (q, \gamma) \in \delta(p, \varepsilon, Z),\ p, q \in Q, Z \in \Gamma, \gamma \in (N \cup \Sigma)^*\},$$

   tj. r znamená maximální možnou délku řetězce, kterým můžeme přepsat vrchol zásobníku v situaci, kdy nečteme nic ze vstupní pásky. Řešením této komplikace je zavedení bufferu, který budeme měnit pouze v momentě, kdy automat M provádí ε-kroky, tj. nečte nic ze vstupu. Buffer si bude udržovat informaci o počtu symbolů přidaných na zásobník a jeho kapacita bude přesně $s \cdot t \cdot r$. Vymažeme jej v momentě, když načteme nějaký symbol ze vstupu. Nastane-li situace, že do bufferu již nemůžeme nic zapsat, protože je plný, ukončíme výpočet automatu M' přepnutím do koncového stavu d a přijímáme vstupní slovo. Činíme tak na základě této věty:

   „zásobník roste neomezeně při ε-krocích, právě když během posloupnosti ε-kroků jeho délka vzroste o více než $s \cdot t \cdot r$ symbolů."

   Důkaz provádět nebudeme, můžeme pouze zmínit, že v takovém případě nastane situace, kdy se výpočet dostane do konfigurace, ve kterém se už jednou během posloupnosti ε-kroků ocitl (proto podmínka $> s \cdot t \cdot r$).

3. **komplikace:** DPDA M nepřečte vstupní slovo do konce, protože přestane číst vstup a provádí ε-kroky, kterými pouze přidává na zásobník další symboly. Ten však narozdíl od předchozí situace roste omezeně a po jisté chvíli se na něj začíná opakovaně zapisovat stejná posloupnost symbolů.

   Zmíněné omezení počtu symbolů na zásobníku je přesně dáno. Začnou-li se provádět ε-kroky, pak přidáním více než $s \cdot t \cdot r$ se dostáváme do situace, kterou jsme odhalili v předchozím bodě. Je tedy jasné, že v průběhu posloupnosti ε-kroků se počet symbolů na zásobníku liší od původního počtu nejvýše o $s \cdot t \cdot r$ symbolů.

Řešením 3. komplikace je zavedení dalšího bufferu, ve kterém budeme udržovat informaci o počtu odlišných konfigurací, ve kterých se automat ocitl v průběhu posloupnosti ε-kroků. Jestliže toto číslo přesáhne hodnotu

$$(*) \quad (s \cdot (t+1))^{(s \cdot t \cdot r)},$$

automat M se dostává do konfigurace, ve kterém už byl. Protože je deterministický, bude se nyní chovat úplně stejně jako předtím a cyklit. Automat M' by se v tu chvíli měl přepnout do akceptujícího stavu d a slovo přijmout.

Otázkou je, proč má mít buffer hodnotu větší než (*). Exponent $s \cdot t \cdot r$ je počet zásobníkových políček, která sledujeme. Hodnota $(t+1)^{s \cdot t \cdot r}$ znamená, že každé políčko může obsahovat t různých symbolů ($t = |\Gamma|$), navíc může být i prázdné (+1). Konečně v s je uložen počet stavů. Z toho vyplývá, že v průběhu ε-kroků můžeme mít nanejvýš (*) různých konfigurací. Přesáhneme-li toto číslo, dostáváme se do konfigurace, která „už tu jednou byla“.

4. **komplikace:** DPDA M přečte celé slovo ze vstupní pásky, pak ale prochází pod ε koncovými i nekoncovými stavy. Zaměníme-li je v automatu M', může nastat situace, kdy jak M, tak i M' přijímá slovo, což je nežádoucí.

   I tento problém se dá elegantně vyřešit, my si však postup uvádět nebudeme.

Věta 2 (neuzavřenost na průnik).

Třída deterministických bezkontextových jazyků není uzavřena na operaci průnik.

Důkaz.

Protipříkladem: mějme dva deterministické CFL $L_1 = \{a^n b^n c^m \mid m, n \geq 1\}$ a $L_2 = \{a^m b^n c^n \mid m, n \geq 1\}$. Jejich průnikem je známý jazyk $L = \{a^n b^n c^n \mid n \geq 1\}$, který není ani bezkontextový, natož pak deterministický bezkontextový.

Věta 3 (uzavřenost na průnik s regulárním jazykem).

Třída deterministických bezkontextových jazyků je uzavřena na operaci průnik s regulárním jazykem.

Důkaz.

Podobně jako v případě třídy bezkontextových jazyků, tj. pomocí synchronní-paralelní kompozice.

Věta 4 (neuzavřenost na operaci sjednocení).

Třída deterministických bezkontextových jazyků není uzavřena na operaci sjednocení.

Důkaz.

Dle DeMorganových pravidel platí pro libovolné (tedy i deterministické bezkontextové) jazyky L_1, L_2 následující rovnost:

$$L_1 \cap L_2 = \text{co-}(\text{co-}L_1 \cup \text{co-}L_2)$$

Sporem předpokládejme, že třída DCFL je uzavřena na operaci sjednocení. Poté díky uzavřenosti na doplněk by z předchozí rovnosti vyplývalo, že jazyk $L_1 \cap L_2$ je deterministický bezkontextový. To je však nesmysl, třída DCFL není uzavřena na průnik. Náš předpoklad byl mylný a platí jeho opak, tedy že třída DCFL není uzavřena na operaci sjednocení.

Věta 5 (vztah CFL a DCFL).

Existují bezkontextové jazyky, které nejsou deterministickými bezkontextovými jazyky.

Poznámka.

Jinými slovy: třída DCFL je vlastní podmnožinou třídy CFL.

Důkaz.

Uvažujme jazyk $L = \{w.w^R \mid w \in \Sigma\}$, který je sice bezkontextový, ale není deterministický. Nedokážeme najít zásobníkový automat, který by deterministicky rozhodnul, kdy už přečetl přesně polovinu slova $w.w^R$.

Poznámka.

Věta 5 platí i proto, že obě třídy CFL i DCFL jsou uzavřeny na jiné operace.