Syntaktická analýza zdola nahor (2. časť)

Peter Kostolányi

16. mája 2017

V prvej časti týchto poznámok sme popísali vo všeobecnosti nedeterministickú schému algoritmu „*posuň a redukuj*“ a špeciálnu triedu tzv. *jednoducho precedenčných gramatík*, pre ktoré je možné schému „posuň a redukuj“ determinizovať; hovoríme potom o *algoritme* „posuň a redukuj“. Rozhodovanie medzi operáciou „posuň“ a operáciou „redukuj“ je pre jednoducho precedenčné gramatiky založené na reláciách $\doteq$, $\lessdot$ a $\gtrdot$ medzi dvojicami po sebe idúcich symbolov.

V nasledujúcom preskúmame ďalšiu triedu bezkontextových gramatík – tzv. LR(0) *gramatiky* – pre ktoré je možné schému „posuň a redukuj“ determinizovať. Pomenovanie „LR(0)“ pochádza zo skutočnosti, že algoritmus „posuň a redukuj“ pre LR(0) gramatiky[1] spracúva vstup *zľava doprava* (angl. ***L**eft-to-Right*) a konštruuje *pravé krajné odvodenie* (angl. ***R**ightmost derivation*). Algoritmus navyše nemá žiadnu informáciu o nasledujúcich neprečítaných symboloch na vstupe, teda má informáciu o **0** nasledujúcich symboloch.

V závere týchto poznámok stručne popíšeme tzv. LR(k) *gramatiky*, ktoré sú zovšeobecnením LR(0) gramatík s informáciou o **k** nasledujúcich symboloch na vstupe. Použitie metód založených na LR gramatikách je momentálne v praxi najrozšírenejším prístupom k syntaktickej analýze.

LR(0) položky a platné LR(0) položky

Prístup cez LR(0) gramatiky je založený na deterministickom rozhodovaní medzi operáciou „posuň“ a operáciou „redukuj“ pomocou tzv. LR(0) *položiek*. LR(0) položky možno definovať pre *ľubovoľnú* bezkontextovú gramatiku G: ide o pravidlá gramatiky G, na ktorých pravej strane je pridaný práve jeden výskyt nového špeciálneho symbolu $\bullet$. Neskôr definujeme LR(0) gramatiky ako bezkontextové gramatiky, ktorých LR(0) položky spĺňajú určité podmienky.

Definícia 1. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika taká, že $\bullet \notin N \cup T$. LR(0) *položka* v gramatike G je dvojica $(\xi, u\bullet v)$, kde $\xi \in N$, $u, v \in (N \cup T)^*$ a $\xi \to uv \in P$. LR(0) položku $(\xi, u\bullet v)$ zvyčajne zapisujeme ako $\xi \to u\bullet v$.

Príklad 1. Uvažujme bezkontextovú gramatiku $G = (N, T, P, \sigma)$ takú, že $N = \{\sigma\}$, $T = \{a, b\}$ a $P = \{\sigma \to a\sigma a \mid b\sigma b \mid \varepsilon\}$. Gramatike G zodpovedajú nasledujúce LR(0) položky (prvý stĺpec zodpovedá pravidlu $\sigma \to a\sigma a$, druhý pravidlu $\sigma \to b\sigma b$ a tretí pravidlu $\sigma \to \varepsilon$).

$$\begin{array}{lll}
\sigma \to \bullet a\sigma a, & \sigma \to \bullet b\sigma b, & \sigma \to \bullet. \\
\sigma \to a\bullet\sigma a, & \sigma \to b\bullet\sigma b, & \\
\sigma \to a\sigma\bullet a, & \sigma \to b\sigma\bullet b, & \\
\sigma \to a\sigma a\bullet, & \sigma \to b\sigma b\bullet, &
\end{array}$$

Treba upozorniť na fakt, že v definícii LR(0) položky môžu byť slová u a v aj prázdne.

Algoritmus „posuň a redukuj“ pre LR(0) gramatiky si počas svojho behu udržiava množinu tzv. *platných* LR(0) *položiek*. Skôr, než uvedieme jeho formálnu definíciu, vysvetlíme intuitívny význam tohto pojmu na príklade. Uvažujme napríklad pravidlo $\xi \to \alpha c\beta$ v nejakej gramatike G. Potom počas vykonávania algoritmu „posuň a redukuj“:

- LR(0) položka $\xi \to \bullet\alpha c\beta$ je platná, ak sa na vstupe ako jedna z eventualít očakáva podslovo odvoditeľné z $\alpha c\beta$, pričom $\alpha c\beta$ v konštruovanom odvodení vznikne z neterminálu ξ.
- LR(0) položka $\xi \to \alpha\bullet c\beta$ je platná, ak bola v minulosti platná položka $\xi \to \bullet\alpha c\beta$, následne bolo zo vstupu prečítané slovo odvoditeľné z α a toto slovo bolo na zásobníku zredukované na α. Na vstupe sa teda ako jedna z eventualít očakáva podslovo odvoditeľné z $c\beta$.

[1] Rovnako ako všetky ostatné algoritmy založené na schéme „posuň a redukuj“ tak, ako bola popísaná v prvej časti týchto poznámok.

- LR(0) položka $\xi \to \alpha c \bullet \beta$ je platná, ak bola platná položka $\xi \to \alpha \bullet c\beta$, zo vstupu bolo prečítané slovo odvoditeľné z c (čiže jedine c samotné) a toto slovo bolo zredukované na c (na nula krokov). Na vstupe sa tak ako jedna z možností očakáva podslovo odvoditeľné z β.
- LR(0) položka $\xi \to \alpha c \beta \bullet$ je platná, ak bola „v dávnejšej minulosti“ platná položka $\xi \to \bullet \alpha c \beta$, zo vstupu bolo prečítané slovo odvoditeľné z $\alpha c \beta$ a toto slovo bolo na zásobníku zredukované na $\alpha c \beta$. Jednou z možností je teda redukovať podslovo $\alpha c \beta$ na ξ.

Symbol $\bullet$ teda udáva „stav rozpracovanosti“ daného pravidla pri spätnom konštruovaní pravého krajného odvodenia. Na začiatku celého procesu sú platné práve všetky LR(0) položky $\sigma \to \bullet u$ pre $\sigma \to u \in P$. Hľadané odvodenie vstupného slova w totiž musí začínať počiatočným neterminálom σ, a preto w musí byť – v prípade jeho príslušnosti do $L(G)$ – odvoditeľné zo σ. Slovo w teda musí byť odvoditeľné z nejakého slova u, pre ktoré existuje pravidlo $\sigma \to u$.

Formálna definícia platných LR(0) položiek

Sformulujme teraz naozajstnú definíciu platných LR(0) položiek. Neprečítaná časť vstupu nemôže mať na takúto definíciu vplyv, pretože algoritmus „posuň a redukuj“ nemá žiadnu informáciu o nasledujúcich symboloch na vstupe – pre každé u „je možné“, že neprečítaná časť vstupu je u. Preto sa platné LR(0) položky definujú pre potenciálne (nie nutne „dosiahnuteľné“) obsahy zásobníka. Presnejšie, platné LR(0) položky sú formálne definované len pre tie obsahy zásobníka, ktoré možno po doplnení vhodným sufixom[2] redukovať na σ, t.j. pre *životaschopné prefixy* (ich definíciu o chvíľu zopakujeme). Rovnako dobre by ale bolo možné definovať platné položky aj pre ľubovoľný obsah zásobníka – platnú položku by aj tak mohol mať iba životaschopný prefix.

Definícia 2. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. Nech w je pravá vetná forma v gramatike G. Prefix x' pravej vetnej formy w je *životaschopný*, ak w obsahuje „handle“ y takú, že $w = xyz$ pre nejaké $x \in (N \cup T)^*$ a $z \in T^*$ a súčasne platí $|x'| \leq |xy|$.

Poznámka 1. V pravej vetnej forme w sú teda životaschopné práve tie prefixy x', pre ktoré vo w existuje aspoň jedna „handle“ y, ktorá „nekončí skôr“ ako x'. To je pre jednoznačné gramatiky práve vtedy, keď *jediná* „handle“ y v danej pravej vetnej forme w „nekončí skôr“ ako x'.

Definícia 3. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. Nech x' je životaschopný prefix v gramatike G a $\xi \to u \bullet v$ je LR(0) položka v G. LR(0) položka $\xi \to u \bullet v$ je *platná* pre životaschopný prefix x', ak v G existuje pravé krajné odvodenie

$$\sigma \Rightarrow_{rm}^* x\xi z \Rightarrow_{rm} xuvz,$$

kde $x \in (N \cup T)^*$, $z \in T^*$ a pre životaschopný prefix x' platí $x' = xu$.

Poznámka 2. Uvedená definícia zrejme zodpovedá intuitívnemu významu platných LR(0) položiek opísanému vyššie. Položka $\xi \to u \bullet v$ je v predchádzajúcej definícii platná pre životaschopný prefix xu. Slovo u pritom zodpovedá „už spracovanej“ časti pravidla. Ak je pri vykonávaní algoritmu „posuň a redukuj“ na zásobníku slovo xu, tak algoritmus „posuň a redukuj“ musel prečítať podslovo odvoditeľné z u a zredukovať ho na u. Navyše je možné, že na vstupe nasleduje podslovo odvoditeľné z v (ak existuje také *terminálne* slovo), pričom predchádzajúca definícia zaručuje, že ak za týmto podslovom nasleduje sufix z, tak je možné zredukovať vstupné slovo postupne až na σ. Položka $\xi \to u \bullet v$ je tak platná aj v intuitívnom zmysle opísanom vyššie.

Úplná LR(0) *položka* je LR(0) položka so symbolom $\bullet$ na konci pravej strany pravidla.

Definícia 4. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. *Úplná* LR(0) *položka* v G je LR(0) položka $\xi \to u \bullet v$ taká, že $v = \varepsilon$.

[2]Nie nutne terminálnym. Pre *redukované* gramatiky ale existuje takýto sufix práve vtedy, keď existuje takýto *terminálny* sufix.

Normálny tvar bezkontextových gramatík

V nasledujúcom budeme uvažovať normálny tvar bezkontextových gramatík, v ktorom sa počiatočný neterminál σ nevyskytuje na pravej strane žiadneho pravidla. Transformácia bezkontextovej gramatiky $G = (N, T, P, \sigma)$ do takéhoto normálneho tvaru je triviálna záležitosť: stačí pridať *nový* neterminál σ' a pravidlo $\sigma' \to \sigma$. Nasledujúce tvrdenie preto uvádzame bez dôkazu.

Tvrdenie 1. *Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. Potom existuje bezkontextová gramatika $G' = (N', T', P', \sigma')$ taká, že $L(G') = L(G)$ a $P' \subseteq N' \times ((N' - \{\sigma'\}) \cup T')^*$.*

Je zrejmé, že ak je gramatika G v uvedenom normálnom tvare, tak v ľubovoľnom akceptačnom behu algoritmu „posuň a redukuj“ sa môže neterminál σ vyskytnúť na zásobníku iba v jeho závere. Výskyt neterminálu σ na vrchu zásobníka je teda signálom na to, že zostáva iba overiť, či je σ jediným symbolom na zásobníku (okrem špeciálneho symbolu $\vdash$ pre „dno zásobníka“; toto overenie možno realizovať napríklad postupným vyprázdnením zásobníka) a či bol prečítaný celý vstup.[3] Ďalším dôsledkom je, že LR(0) položky typu $\sigma \to \bullet u$ môžu byť platné iba na začiatku vykonávania algoritmu „posuň a redukuj“.

Nedeterministický položkový automat

Nech $G = (N, T, P, \sigma)$ je (ľubovoľná) bezkontextová gramatika v normálnom tvare z tvrdenia 1. V nasledujúcom popíšeme konštrukciu nedeterministického konečného automatu, tzv. *nedeterministického položkového automatu*, ktorého stavy (okrem počiatočného) sú LR(0) položky v gramatike G a stav zodpovedajúci položke $\xi \to u\bullet v$ je dosiahnuteľný z počiatočného stavu na slovo x práve vtedy, keď x je životaschopný prefix a $\xi \to u\bullet v$ je jeho platná LR(0) položka. Takýto automat možno zrejme využiť na nájdenie množiny všetkých platných LR(0) položiek pre x – stačí nájsť všetky stavy dosiahnuteľné na x.

V intuitívnej rovine je konštrukcia nedeterministického položkového automatu založená na nasledujúcich úvahách (formálne sa dá zdôvodniť *definíciou* platných LR(0) položiek):

- Na začiatku každého behu algoritmu „posuň a redukuj“ sú platné všetky položky $\sigma \to \bullet u$, kde $\sigma \to u \in P$. Všetky tieto LR(0) položky sú teda platné pre životaschopný prefix ε,[4] keďže na začiatku behu algoritmu „posuň a redukuj“ je zásobník prázdny (až na symbol $\vdash$ reprezentujúci dno zásobníka). Navyše je zrejmé, že zásobník môže byť prázdny *výlučne* na začiatku výpočtu algoritmu „posuň a redukuj“. Z počiatočného stavu q_0 nedeterministického položkového automatu teda vedú prechody na ε práve do všetkých stavov zodpovedajúcich položkám $\sigma \to \bullet u$, kde $\sigma \to u \in P$.

- Nech x je životaschopný prefix, pre ktorý je platná LR(0) položka $\xi \to u\bullet\eta v$, kde $\eta \in N$. V takom prípade sa na vstupe „očakáva“ výskyt podslova odvoditeľného z neterminálu η. Odvodenie tohto podslova začína použitím nejakého pravidla $\eta \to y$. Pre životaschopný prefix x sú teda platné aj všetky LR(0) položky $\eta \to \bullet y$, kde $\eta \to y \in P$. Zo stavu zodpovedajúceho položke $\xi \to u\bullet\eta v$ teda vedú prechody na ε do všetkých stavov zodpovedajúcich položkám $\eta \to \bullet y$ pre $\eta \to y \in P$.

- Nech x je opäť životaschopný prefix, pre ktorý je platná LR(0) položka $\xi \to u\bullet\eta v$, kde $\eta \in N$. To znamená, že na vstupe sa „očakáva“ výskyt podslova odvoditeľného z neterminálu η. Po „spracovaní“ tohto podslova je obsahom zásobníka životaschopný prefix $x\eta$, pre ktorý je platná LR(0) položka $\xi \to u\eta\bullet v$. Inak povedané, ak je pre x platná položka $\xi \to u\bullet\eta v$, je pre $x\eta$ platná položka $\xi \to u\eta\bullet v$. Zo stavu zodpovedajúceho položke $\xi \to u\bullet\eta v$ preto vedie prechod na η do stavu zodpovedajúceho položke $\xi \to u\eta\bullet v$.

[3] Pri implementácii na zásobníkovom automate je overenie tejto druhej podmienky implicitné, keďže automat akceptuje iba vtedy, keď bol dočítaný celý vstup.

[4] Prázdne slovo ε je životaschopným prefixom práve vtedy, keď existuje aspoň jedno pravidlo typu $\sigma \to u$. Prípad, keď takéto pravidlo neexistuje, je zo zrejmých dôvodov nezaujímavý.

- Nech x je životaschopný prefix, pre ktorý je platná LR(0) položka $\xi \to u\bullet cv$, kde $c \in T$. To znamená, že na vstupe sa „očakáva“ výskyt písmena c. Po spracovaní tohto symbolu je obsahom zásobníka životaschopný prefix xc, pre ktorý je platná LR(0) položka $\xi \to uc\bullet v$. Zo stavu zodpovedajúceho položke $\xi \to u\bullet cv$ teda vedie prechod na c do stavu zodpovedajúceho položke $\xi \to uc\bullet v$.

Formálna definícia nedeterministického položkového automatu zodpovedajúceho bezkontextovej gramatike G je preto nasledovná.

Definícia 5. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika, ktorá je v normálnom tvare z tvrdenia 1. *Nedeterministický položkový automat* pre gramatiku G je nedeterministický konečný automat $\mathscr{N}_G = (K, \Sigma, \delta, q_0, F)$, kde

$$K = \{q_0\} \cup \{\xi \to u\bullet v \mid \xi \in N;\ u, v \in (N \cup T)^*;\ \xi \to uv \in P\},$$

$\Sigma = N \cup T$ a prechodová funkcia δ je daná nasledovne:

$$\begin{aligned}
\delta(q_0, \varepsilon) &= \{\sigma \to \bullet u \mid \sigma \to u \in P\}, \\
\delta(\xi \to u\bullet\eta v, \varepsilon) &= \{\eta \to \bullet y \mid \eta \to y \in P\}, && \forall \xi \to u\bullet\eta v \in K \text{ také, že } \eta \in N, \\
\delta(\xi \to u\bullet\eta v, \eta) &= \{\xi \to u\eta\bullet v\}, && \forall \xi \to u\bullet\eta v \in K \text{ také, že } \eta \in N, \\
\delta(\xi \to u\bullet cv, c) &= \{\xi \to uc\bullet v\}, && \forall \xi \to u\bullet cv \in K \text{ také, že } c \in T.
\end{aligned}$$

Množina akceptačných stavov F nie je podstatná, možno uvažovať napríklad $F = \emptyset$.

Poznámka 3. Priamo z jeho definície je zrejmé, že v automate $\mathscr{N}_G$ vedie z každého stavu najviac jeden prechod na písmeno. Nedeterminizmus automatu $\mathscr{N}_G$ je spôsobený prechodmi na ε, ktorých naopak môže z jedného stavu viesť aj niekoľko.

Poznámka 4. Množina akceptačných stavov nedeterministického položkového automatu nie je dôležitá, pretože tento automat budeme (nepriamo) používať výlučne na zistenie množiny stavov (LR(0) položiek) dosiahnuteľných z počiatočného stavu q_0 na dané slovo x. Dobre odôvodnenou voľbou množiny akceptačných stavov však môže byť napríklad $F = K - \{q_0\}$ – v takom prípade automat $\mathscr{N}_G$ akceptuje práve všetky životaschopné prefixy v gramatike G.

Nasledujúcu vetu uvedieme bez formálneho dôkazu, jej platnosť by ale mala byť čitateľovi zrejmá z predchádzajúcich úvah.

Veta 1. *Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika a $\mathscr{N}_G$ je nedeterministický položkový automat pre G. Potom pre všetky $x \in (N \cup T)^*$ a všetky* LR(0) *položky $\xi \to u\bullet v$ gramatiky G platí $(q_0, x) \vdash^*_{\mathscr{N}_G} (\xi \to u\bullet v, \varepsilon)$ práve vtedy, keď x je životaschopný prefix v gramatike G a* LR(0) *položka $\xi \to u\bullet v$ je platná pre x.*

Poznámka 5. Ak je gramatika G v redukovanom normálnom tvare, sú všetky stavy automatu $\mathscr{N}_G$ dosiahnuteľné, a teda každá LR(0) položka je platná pre niektorý životaschopný prefix. Ak je totiž dosiahnuteľný stav zodpovedajúci LR(0) položke $\xi \to \bullet u$, sú zjavne dosiahnuteľné aj všetky ostatné stavy zodpovedajúce LR(0) položkám pravidla $\xi \to u$ (pomocou prechodov na jednotlivé písmená slova u). Keďže je gramatika G v redukovanom normálnom tvare, pre každý neterminál $\xi \in N$ a každé $\xi \to u \in P$ existuje pravé krajné odvodenie

$$\sigma \Rightarrow^*_{rm} x\xi z \Rightarrow_{rm} xuz,$$

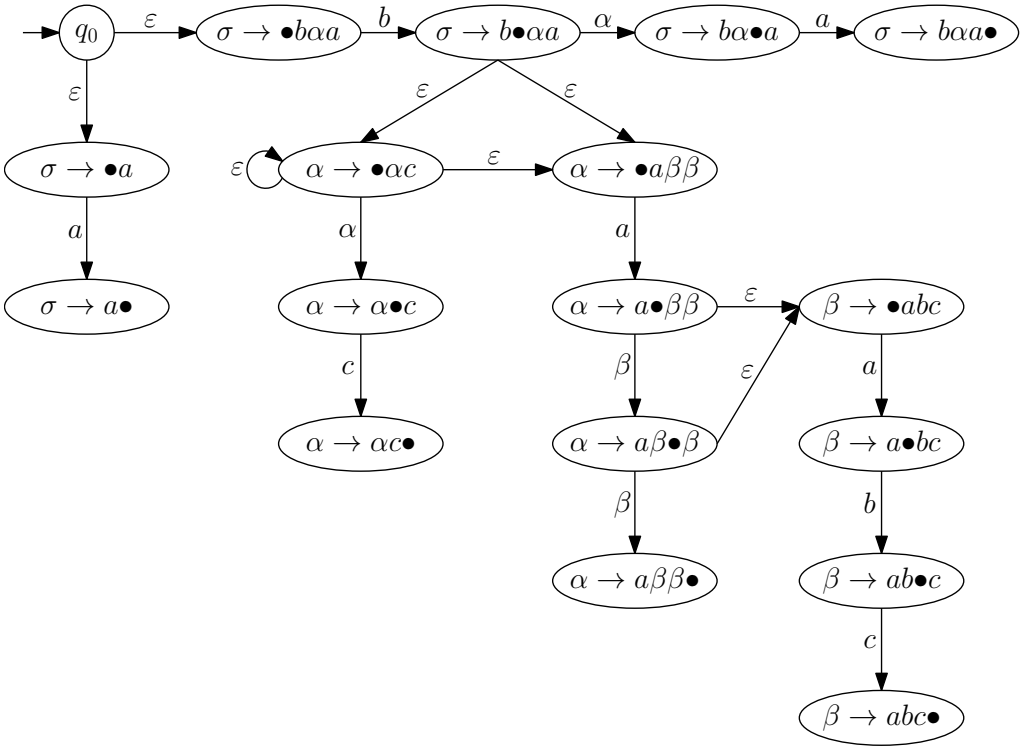
kde $x \in (N \cup T)^*$ a $z \in T^*$. Slovo u je teda „handle“ zodpovedajúca pravidlu $\xi \to u$ a slovo x je životaschopný prefix pravej vetnej formy xuz, pre ktorý je platná LR(0) položka $\xi \to \bullet u$. Zodpovedajúci stav je teda dosiahnuteľný z q_0 na slovo x, čo dokazuje nami vyslovené tvrdenie.

Príklad

Príklad 2. Uvažujme bezkontextovú gramatiku $G = (N, T, P, \sigma)$ s $N = \{\sigma, \alpha, \beta\}$, $T = \{a, b, c\}$ a

$$\begin{aligned} P = \{&\sigma \to b\alpha a \mid a \\ &\alpha \to \alpha c \mid a\beta\beta \\ &\beta \to abc\}. \end{aligned}$$

Nedeterministický položkový automat $\mathscr{N}_G$ pre gramatiku G zostrojený na základe definície 5 – presnejšie jeho „dosiahnuteľná časť“ – je znázornený na obrázku 1.



Obr. 1: Nedeterministický položkový automat $\mathscr{N}_G$ zodpovedajúci gramatike G.

Ľahko možno overiť, že z každého stavu automatu $\mathscr{N}_G$ skutočne vedie najviac jeden prechod na písmeno. Presnejšie, zo stavu q_0 a stavov zodpovedajúcich úplným LR(0) položkám nevedie žiaden prechod na písmeno a z ostatných stavov vedie prechod na písmeno, ktoré je v zodpovedajúcej LR(0) položke za symbolom $\bullet$. Zo stavu q_0 ďalej vedú prechody na ε do stavov zodpovedajúcich LR(0) položkám $\sigma \to \bullet u$ pre nejaké $\sigma \to u \in P$. Z ostatných stavov vedú prechody na ε práve vtedy, keď je v zodpovedajúcej LR(0) položke symbol $\bullet$ pred neterminálom. Ak ide o neterminál ξ, tak tieto prechody vedú do stavov zodpovedajúcich LR(0) položkám $\xi \to \bullet u$ pre $\xi \to u \in P$. Ako vidieť na príklade LR(0) položky $\alpha \to \bullet\alpha c$, „ľavá rekurzia“ má za následok prítomnosť „epsilonových“ slučiek.

Deterministický položkový automat

Účelom, pre ktorý sme zaviedli nedeterministický položkový automat, je nájdenie spôsobu, ako pre každý životaschopný prefix x nájsť zodpovedajúcu množinu platných LR(0) položiek. Ide práve o tie LR(0) položky, pre ktoré sú zodpovedajúce stavy v nedeterministickom položkovom automate dosiahnuteľné na slovo x.

Štandardná konštrukcia determinizácie NKA, prebratá v minulom semestri, spočíva najprv v odstránení prechodov na ε a následne v aplikácii tzv. *podmnožinovej konštrukcie*. Výsledkom je ekvivalentný deterministický konečný automat, ktorého stavy sú množinami stavov pôvodného automatu. Na slovo x je v tomto deterministickom automate dosiahnuteľný stav, ktorý zodpovedá množine stavov dosiahnuteľných na x v pôvodnom nedeterministickom automate. Ak teda štandardným spôsobom determinizujeme nedeterministický položkový automat $\mathscr{N}_G$, dostaneme *deterministický položkový automat* $\mathscr{D}_G$, ktorého výpočet na slove x skončí v stave zodpovedajúcom množine platných LR(0) položiek pre x.[5]

Deterministický položkový automat sa teda javí byť vhodným nástrojom na hľadanie množín platných LR(0) položiek. Jeho horeuvedená konštrukcia je ale značne neefektívna. Už pri odstraňovaní prechodov na ε je potrebné zaviesť pomerne veľké množstvo nových prechodov. Po následnej aplikácii podmnožinovej konštrukcie vznikne vo všeobecnosti *obrovské* množstvo nedosiahnuteľných stavov. A automat $\mathscr{D}_G$ tiež obsahuje „nepodstatné“ prechody do stavu $\emptyset$.

V nasledujúcom preto popíšeme efektívnejší (a pomerne intuitívny) spôsob, ako na základe automatu $\mathscr{N}_G$ zostrojiť „užitočnú časť“ zodpovedajúceho deterministického konečného automatu $\mathscr{D}_G$. Táto „užitočná časť“ bude pozostávať iba z dosiahnuteľných stavov automatu $\mathscr{D}_G$. Navyše nebude obsahovať stav zodpovedajúci prázdnej množine, kvôli čomu už nebude mať úplnú prechodovú funkciu. Automat $\mathscr{D}_G$ budeme *stotožňovať* s touto jeho „užitočnou časťou“.

Vstupom nasledujúceho algoritmu je nedeterministický položkový automat pre gramatiku G, teda automat $\mathscr{N}_G = (K, \Sigma, \delta, q_0, F)$. Výstupom je „užitočná časť“ zodpovedajúceho deterministického položkového automatu, ktorú budeme označovať $\mathscr{D}_G = (K', \Sigma, \delta', q_0', F')$. Množina F' je aj v tomto prípade nepodstatná, a preto kladieme $F' = \emptyset$. Počas behu si bude algoritmus udržiavať množinu S „spracovaných“ stavov z K'.

1. Polož $q_0' := I_0 := \{q \in K \mid (q_0, \varepsilon) \vdash^*_{\mathscr{N}_G} (q, \varepsilon)\}$. Počiatočným stavom automatu $\mathscr{D}_G$ teda bude množina stavov automatu $\mathscr{N}_G$ dosiahnuteľných na ε z počiatočného stavu automatu $\mathscr{N}_G$.
2. Inicializuj $K' := \{I_0\}$ a $S := \emptyset$.
3. Opakuj, až kým $K' = S$:
   - 3.1 Vyber (na základe ľubovoľného kritéria) nejaký stav $I \in K' - S$. Stav I je množina stavov z K.
   - 3.2 Pre všetky $c \in \Sigma$ vypočítaj množinu $J_c = \bigcup_{p \in I} \{q \in K \mid (p, c) \vdash^*_{\mathscr{N}_G} (q, \varepsilon)\}$. Množina J_c teda obsahuje tie stavy automatu $\mathscr{N}_G$, ktoré sú dosiahnuteľné zo stavov z I na písmeno c (a ľubovoľný počet krokov na ε). Ak $J_c \notin K'$ a $J_c \neq \emptyset$, pridaj stav J_c do K'. Ak $J_c \neq \emptyset$, polož $\delta'(I, c) = J_c$.
   - 3.3 Pridaj stav I do množiny spracovaných stavov S.

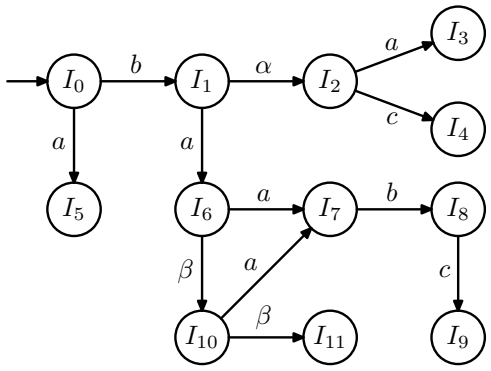
Prvé pokračovanie príkladu

Príklad 2 (pokračovanie). Vyššie sme zostrojili nedeterministický položkový automat $\mathscr{N}_G$ (obrázok 1) zodpovedajúci gramatike $G = (N, T, P, \sigma)$, kde $N = \{\sigma, \alpha, \beta\}$, $T = \{a, b, c\}$ a

$$\begin{aligned} P = \{&\sigma \to b\alpha a \mid a \\ &\alpha \to \alpha c \mid a\beta\beta \\ &\beta \to abc\}. \end{aligned}$$

Deterministický položkový automat $\mathscr{D}_G$ zodpovedajúci automatu $\mathscr{N}_G$, vypočítaný pomocou horeuvedeného algoritmu, je na obrázku 2.

[5] Tu treba upozorniť na dva detaily. Prvým je skutočnosť, že x nemusí byť životaschopný prefix. V takom prípade skončí výpočet na x v stave zodpovedajúcom prázdnej množine (prípadne množine $\{q_0\}$, ale to je možné iba v prípade, že gramatika neobsahuje žiadne pravidlo typu $\sigma \to u$). Druhým je skutočnosť, že táto množina stavov automatu $\mathscr{N}_G$ vo všeobecnosti môže okrem LR(0) položiek obsahovať aj počiatočný stav q_0. Stav automatu $\mathscr{D}_G$ zodpovedajúci takejto množine ale môže byť dosiahnuteľný iba na prázdne slovo ε.



(a) Automat $\mathscr{D}_G$.

I_0: q_0, $\sigma \to \bullet b\alpha a$, $\sigma \to \bullet a$
I_1: $\sigma \to b\bullet\alpha a$, $\alpha \to \bullet\alpha c$, $\alpha \to \bullet a\beta\beta$
I_2: $\sigma \to b\alpha\bullet a$, $\alpha \to \alpha\bullet c$
I_3: $\sigma \to b\alpha a\bullet$
I_4: $\alpha \to \alpha c\bullet$
I_5: $\sigma \to a\bullet$
I_6: $\alpha \to a\bullet\beta\beta$, $\beta \to \bullet abc$
I_7: $\beta \to a\bullet bc$
I_8: $\beta \to ab\bullet c$
I_9: $\beta \to abc\bullet$
I_{10}: $\alpha \to a\beta\bullet\beta$, $\beta \to \bullet abc$
I_{11}: $\alpha \to a\beta\beta\bullet$

(b) Stavy automatu $\mathscr{D}_G$.

Obr. 2: Deterministický položkový automat $\mathscr{D}_G$ pre gramatiku G.

Napríklad výpočet automatu $\mathscr{D}_G$ na slove baa skončí v stave I_7. Jedinou platnou LR(0) položkou pre životaschopný prefix baa je teda $\beta \to a\bullet bc$. Výpočet na slove ε skončí v stave I_0, ktorý okrem LR(0) položiek obsahuje aj stav q_0 (ten ale na platnosť LR(0) položiek pre ε nemá žiaden vplyv). Pre ε sú teda platné LR(0) položky $\sigma \to \bullet b\alpha a$ a $\sigma \to \bullet a$. Napríklad slovo ab automat $\mathscr{D}_G$ nedočíta až do konca.[6] Slovo ab preto nie je životaschopný prefix v gramatike G.

LR(0) **gramatiky**

V nasledujúcom definujeme LR(0) gramatiky ako bezkontextové gramatiky, pre ktoré možno využiť množiny platných LR(0) položiek na determinizáciu algoritmu „posuň a redukuj“. Na základe platných LR(0) položiek teda treba vedieť rozhodnúť, či vykonať operáciu „posuň“ alebo operáciu „redukuj“. Z definície platných LR(0) položiek zjavne vyplýva:

- Ak je pre životaschopný prefix na zásobníku platná *úplná* LR(0) položka $\xi \to u\bullet$, tak jednou z možností je redukovať podľa pravidla $\xi \to u$.
- Ak pre životaschopný prefix na zásobníku nie je platná žiadna úplná LR(0) položka, tak nemožno redukovať podľa žiadneho pravidla.
- Ak je pre životaschopný prefix na zásobníku platná LR(0) položka $\xi \to u\bullet cv$, kde $c \in T$ je *terminálny symbol*, tak je jednou z možností posunúť symbol zo vstupu (je totiž možné, že nasledujúci neprečítaný symbol na vstupe je c).
- Ak pre životaschopný prefix na zásobníku nie je platná žiadna LR(0) položka s „bodkou“ pred terminálom, tak nie je možné posunúť na zásobník symbol zo vstupu. Z deterministického položkového automatu je totiž zrejmé, že posunutím by vzniklo slovo, ktoré nie je životaschopným prefixom.

Ak teda pre daný životaschopný prefix nie je platná žiadna úplná LR(0) položka, jedinou možnosťou je posunúť symbol zo vstupu – toto rozhodnutie možno urobiť deterministicky. Problém nastane v situácii, keď pre daný životaschopný prefix *je platná* úplná LR(0) položka $\xi \to u\bullet$. Ak je totiž v takom prípade platná aj nejaká iná úplná LR(0) položka $\eta \to y\bullet$, nie je jasné, podľa ktorého z pravidiel redukovať. Podobne v prípade, keď je okrem položky $\xi \to u\bullet$ platná položka so symbolom $\bullet$ pred terminálom, nie je jasné, či redukovať podľa $\xi \to u$, alebo posunúť symbol zo vstupu.

[6] Ak by sme sa neobmedzovali iba na „užitočnú časť“ $\mathscr{D}_G$, skončil by výpočet na ab v stave $\emptyset$. V „kompletnom“ automate $\mathscr{D}_G$ sú životaschopnými prefixmi práve tie slová, ktoré sa dočítajú v stave rôznom od $\emptyset$ a $\{q_0\}$.

LR(0) gramatiky definujeme ako bezkontextové gramatiky, pre ktoré takéto konflikty nikdy nenastanú. Navyše požadujeme, aby gramatika bola v normálnom tvare z tvrdenia 1.

Definícia 6. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. Gramatika G je LR(0) *gramatika*, ak sú splnené nasledujúce dve podmienky:

(*i*) Počiatočný neterminál σ sa nevyskytuje na pravej strane žiadneho pravidla, a teda platí $P \subseteq N \times ((N - \{\sigma\}) \cup T)^*$.

(*ii*) Pre všetky životaschopné prefixy x v G platí, že ak je pre x platná úplná LR(0) položka $\xi \to u\bullet$, tak pre x nie je platná žiadna iná úplná LR(0) položka, ani žiadna LR(0) položka so symbolom $\bullet$ pred terminálom.

Poznámka 6. Overenie, či je daná bezkontextová gramatika LR(0), je teda možné realizovať jednoduchou inšpekciou zodpovedajúceho deterministického položkového automatu – pre každý jeho stav musí platiť, že ak zodpovedajúca množina LR(0) položiek obsahuje úplnú položku $\xi \to u\bullet$, tak neobsahuje žiadnu inú úplnú položku, ani žiadnu položku so symbolom $\bullet$ pred terminálom. Príklad takéhoto overenia uvedieme nižšie.

Poznámka 7. Pomerne jednoducho sa dá dokázať, že ak je gramatika súčasne v redukovanom normálnom tvare a v „bezepsilonovom“ normálnom tvare (čo je zväčša rozumný predpoklad), tak podmienku (*ii*) definície 6 možno zapísať o poznanie jednoduchšie – stačí požadovať, aby platnosť úplnej LR(0) položky implikovala, že nie je platná *žiadna iná položka*. Dôkaz tejto skutočnosti prenechávame ako jednu z úloh na nasledujúce cvičenie.

Druhé pokračovanie príkladu

Príklad 2 (pokračovanie). Vyššie sme zostrojili deterministický položkový automat $\mathscr{D}_G$ (obrázok 2) zodpovedajúci gramatike $G = (N, T, P, \sigma)$, kde $N = \{\sigma, \alpha, \beta\}$, $T = \{a, b, c\}$ a

$$\begin{aligned} P = \{&\sigma \to b\alpha a \mid a \\ &\alpha \to \alpha c \mid a\beta\beta \\ &\beta \to abc\}. \end{aligned}$$

Gramatika G očividne spĺňa podmienku (*i*) definície LR(0) gramatiky, keďže σ nie je na pravej strane žiadneho pravidla. Z obrázku 2 zisťujeme, že úplné LR(0) položky sa nachádzajú v množinách zodpovedajúcich stavom I_3, I_4, I_5, I_9 a I_{11}. V každej z týchto množín je práve jedna úplná LR(0) položka a žiadna LR(0) položka so symbolom $\bullet$ pred terminálom. Gramatika G teda spĺňa aj podmienku (*ii*) definície LR(0) gramatiky a možno tak uzavrieť, že G *je* LR(0) *gramatika*.

Algoritmus „posuň a redukuj“ pre LR(0) gramatiky

Princíp algoritmu „posuň a redukuj“ pre LR(0) gramatiky sme v zásade už vysvetlili pri odôvodňovaní definície LR(0) gramatík. Ide o determinizáciu všeobecnej schémy „posuň a redukuj“, ktorá v každom kroku pracuje nasledovne:

a) Ak je na vrchu zásobníka σ, zisti, či ide o jediný symbol na zásobníku (okrem $\vdash$). Ak áno, prejdi do akceptačného stavu (čiže akceptuj, ak na vstupe nezostal žiaden neprečítaný symbol). Ak nie, zamietni.

b) V opačnom prípade zisti množinu platných LR(0) položiek pre slovo na zásobníku.

c) Ak táto množina obsahuje úplnú položku $\xi \to u\bullet$, *redukuj* podľa $\xi \to u$.

d) Ak obsahuje (aspoň jednu) položku $\xi \to u\bullet cv$, kde c je terminál, *posuň* symbol zo vstupu.

e) Ak nemožno posunúť ani redukovať, zamietni vstup.

Vo všeobecnosti je možné zistenie množiny platných LR(0) položiek (v bode b) realizovať ľubovoľne, ale nie každé riešenie je implementovateľné na zásobníkovom automate. Napríklad triviálne riešenie, v ktorom sa zakaždým odsimuluje výpočet deterministického položkového automatu na celom obsahu zásobníka nie je realizovateľné na zásobníkovom automate, pretože je pri ňom nutné pristupovať dovnútra zásobníka. Na zásobníkovom automate naopak *je realizovateľné* riešenie, v ktorom sa na každej pozícii v zásobníku (pomocou špeciálnych symbolov) zaznamená stav, v ktorom je deterministický položkový automat po prečítaní zodpovedajúceho prefixu. Pri realizácii operácií „posuň" resp. „redukuj" potom stačí odsimulovať jeden krok deterministického položkového automatu na symbole vkladanom na zásobník, pričom simulácia začne v stave uloženom na vrchu zásobníka pred pridaním tohto symbolu. Príklad uvádzame nižšie.

Tretie pokračovanie príkladu

Príklad 2 (pokračovanie). Vyššie sme dokázali, že gramatika $G = (N, T, P, \sigma)$ s $N = \{\sigma, \alpha, \beta\}$, $T = \{a, b, c\}$ a

$$\begin{aligned} P = \{&\sigma \to b\alpha a \mid a \\ &\alpha \to \alpha c \mid a\beta\beta \\ &\beta \to abc\} \end{aligned}$$

je LR(0) gramatika. Deterministický položkový automat tejto gramatiky je na obrázku 2. V nasledujúcom odsimulujeme algoritmus „posuň a redukuj" pre gramatiku G na vstupe *baabcabccca*. Na zásobníku si budeme na každej pozícii pamätať stav, v ktorom je automat $\mathscr{D}_G$ po prečítaní zodpovedajúceho prefixu.

Zásobník	**Zvyšok vstupu**	**Stav automatu $\mathscr{D}_G$**	**Operácia**
$\vdash I_0$	*baabcabccca*	I_0: q_0, $\sigma \to \bullet b\alpha a$, $\sigma \to \bullet a$	Posuň
$\vdash I_0 b I_1$	*aabcabccca*	I_1: $\sigma \to b\bullet\alpha a$, $\alpha \to \bullet\alpha c$, $\alpha \to \bullet a\beta\beta$	Posuň
$\vdash I_0 b I_1 a I_6$	*abcabccca*	I_6: $\alpha \to a\bullet\beta\beta$, $\beta \to \bullet abc$	Posuň
$\vdash I_0 b I_1 a I_6 a I_7$	*bcabccca*	I_7: $\beta \to a\bullet bc$	Posuň
$\vdash I_0 b I_1 a I_6 a I_7 b I_8$	*cabccca*	I_8: $\beta \to ab\bullet c$	Posuň
$\vdash I_0 b I_1 a I_6 \underline{a I_7 b I_8 c I_9}$	*abccca*	I_9: $\beta \to abc\bullet$	Redukuj
$\vdash I_0 b I_1 a I_6 \beta I_{10}$	*abccca*	I_{10}: $\alpha \to a\beta\bullet\beta$, $\beta \to \bullet abc$	Posuň
$\vdash I_0 b I_1 a I_6 \beta I_{10} a I_7$	*bccca*	I_7: $\beta \to a\bullet bc$	Posuň
$\vdash I_0 b I_1 a I_6 \beta I_{10} a I_7 b I_8$	*ccca*	I_8: $\beta \to ab\bullet c$	Posuň
$\vdash I_0 b I_1 a I_6 \beta I_{10} \underline{a I_7 b I_8 c I_9}$	*cca*	I_9: $\beta \to abc\bullet$	Redukuj
$\vdash I_0 b I_1 \underline{a I_6 \beta I_{10} \beta I_{11}}$	*cca*	I_{11}: $\alpha \to a\beta\beta\bullet$	Redukuj
$\vdash I_0 b I_1 \alpha I_2$	*cca*	I_2: $\sigma \to b\alpha\bullet a$, $\alpha \to \alpha\bullet c$	Posuň
$\vdash I_0 b I_1 \underline{\alpha I_2 c I_4}$	*ca*	I_4: $\alpha \to \alpha c\bullet$	Redukuj
$\vdash I_0 b I_1 \alpha I_2$	*ca*	I_2: $\sigma \to b\alpha\bullet a$, $\alpha \to \alpha\bullet c$	Posuň
$\vdash I_0 b I_1 \underline{\alpha I_2 c I_4}$	*a*	I_4: $\alpha \to \alpha c\bullet$	Redukuj
$\vdash I_0 b I_1 \alpha I_2$	*a*	I_2: $\sigma \to b\alpha\bullet a$, $\alpha \to \alpha\bullet c$	Posuň
$\vdash I_0 \underline{b I_1 \alpha I_2 a I_3}$	ε	I_3: $\sigma \to b\alpha a\bullet$	Redukuj
$\vdash I_0 \sigma$	ε	(Slovo σ na zásobníku, ε na vstupe)	Akceptuj

LR(k) **gramatiky**

Prirodzeným rozšírením LR(0) gramatík sú LR(k) gramatiky, určené na spracovanie algoritmom „posuň a redukuj", ktorý má navyše informáciu o k nasledujúcich neprečítaných symboloch na vstupe (a schopnosť detegovať koniec vstupu). V nasledujúcom popíšeme LR(1) gramatiky – zovšeobecnenie na ľubovoľné k je priamočiare.

Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. LR(1) položka v gramatike G je LR(0) položka v G rozšírená o nejakú podmnožinu S abecedy $T \cup \{\dashv\}$, kde $\dashv$ je špeciálny symbol reprezentujúci koniec vstupu. Ak napríklad $\xi \to u\bullet v$ je LR(0) položka v gramatike G a $T = \{a, b\}$, tak napríklad „$\xi \to u\bullet v, \{a\}$" a „$\xi \to u\bullet v, \{a, \dashv\}$" sú LR(1) položky v G.

Neformálne povedané, LR(1) položka $\xi \to u\bullet v, S$ je počas vykonávania algoritmu „posuň a redukuj“ platná vtedy, keď algoritmus v predchádzajúcom prečítal slovo odvoditeľné z u, toto slovo na zásobníku zredukoval na u a na vstupe sa ako jedna z eventualít očakáva podslovo odvoditeľné zo slova v, *za ktorým nasleduje symbol z množiny* S,[7] pričom podslovo uv v konštruovanom pravom krajnom odvodení vznikne použitím pravidla $\xi \to uv$. V úvode výpočtu algoritmu „posuň a redukuj“ sú platné práve všetky LR(1) položky $\sigma \to \bullet u, \{\dashv\}$.

Formálne sa platné LR(1) položky opäť definujú pre životaschopné prefixy.

Definícia 7. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. Nech x' je životaschopný prefix v gramatike G a $\xi \to u\bullet v, S$ je LR(1) položka v G. LR(1) položka $\xi \to u\bullet v, S$ je *platná* pre životaschopný prefix x', ak v G existuje pravé krajné odvodenie

$$\sigma \Rightarrow_{rm}^{*} x\xi z \Rightarrow_{rm} xuvz,$$

kde $x \in (N \cup T)^*$, $z \in T^*$, pre životaschopný prefix x' platí $x' = xu$ a je splnená jedna z nasledujúcich podmienok:

(i) Slovo z je neprázdne a jeho prvý symbol je v S.

(ii) Slovo z je prázdne a symbol $\dashv$ je v S.

Konštrukcia *nedeterministického položkového automatu* zodpovedajúceho danej bezkontextovej gramatike $G = (N, T, P, \sigma)$ je pre LR(1) položky založená na rovnakom princípe ako pre LR(0) položky:

- Zo stavu q_0 vedú prechody na ε do práve všetkých stavov zodpovedajúcich LR(1) položkám $\sigma \to \bullet u, \{\dashv\}$ pre $\sigma \to u \in P$.
- Zo stavu zodpovedajúceho LR(1) položke $\xi \to u\bullet\eta v, S$ (pre $\eta \in N$) vedú prechody na ε do všetkých stavov zodpovedajúcich položkám $\eta \to \bullet y, S'$ pre $\eta \to y \in P$, kde S' je množina symbolov taká, že $a \in S'$ práve vtedy, keď zo slova v možno odvodiť slovo začínajúce symbolom a, alebo z v možno odvodiť ε a $a \in S$.
- Zo stavu zodpovedajúceho LR(1) položke $\xi \to u\bullet\eta v, S$ (pre $\eta \in N$) vedie prechod na η do stavu zodpovedajúceho LR(1) položke $\xi \to u\eta\bullet v, S$.
- Zo stavu zodpovedajúceho LR(1) položke $\xi \to u\bullet c v, S$ (pre $c \in T$) vedie prechod na c do stavu zodpovedajúceho LR(1) položke $\xi \to uc\bullet v, S$.

Deterministický položkový automat zodpovedajúci gramatike G potom možno skonštruovať štandardným algoritmom, rovnako ako pre LR(0) položky.

Zhruba povedané, algoritmus „posuň a redukuj“ pre LR(1) gramatiky[8] pracuje ako algoritmus pre LR(0) gramatiky s tým rozdielom, že operácia „posuň“ sa vykonáva iba v prípade, že je platná nejaká LR(1) položka so symbolom pred terminálom, *ktorý je nasledujúcim neprečítaným symbolom na vstupe* a operácia „redukuj“ sa vykonáva iba v prípade, že je platná úplná LR(1) položka *s množinou symbolov S obsahujúcou prvý neprečítaný symbol na vstupe* (alebo $\dashv$ v prípade, že na vstupe nezostáva žiaden neprečítaný symbol).

Keďže sa operácie „posuň“ a „redukuj“ vykonávajú „viac s rozumom“ ako pre LR(0) gramatiky, je aj „menšia šanca“, že nastane konflikt. Definícia LR(1) gramatík, zaručujúca nemožnosť konfliktov, preto kladie na gramatiku slabšie podmienky, než definícia LR(0) gramatík.

Definícia 8. Nech $G = (N, T, P, \sigma)$ je bezkontextová gramatika. Gramatika G je LR(1) *gramatika*, ak sú splnené nasledujúce dve podmienky:

(i) Počiatočný neterminál σ sa nevyskytuje na pravej strane žiadneho pravidla, a teda platí $P \subseteq N \times ((N - \{\sigma\}) \cup T)^*$.

[7] Alebo koniec slova, ak $\dashv \in S$.

[8] Definíciu LR(1) gramatík uvedieme o chvíľu.

(*ii*) Pre všetky životaschopné prefixy x v G platí, že ak je pre x platná úplná LR(1) položka $\xi \to u\bullet, S$, tak pre x nie je platná žiadna iná úplná LR(1) položka $\eta \to y\bullet, S'$ taká, že $S \cap S' \neq \emptyset$, ani žiadna LR(1) položka so symbolom $\bullet$ pred terminálom patriacim do S.

Generatívna sila LR gramatík

V závere týchto poznámok už len bez dôkazu uvedieme niekoľko klasických tvrdení o generatívnej sile LR gramatík a o vzťahu tried nimi generovaných jazykov k triedam jazykov akceptovaných deterministickými zásobníkovými automatmi.

Každá LR(k) gramatika je očividne aj LR($k+1$) gramatika. Naopak, pre každé $k \in \mathbb{N}$ existuje LR($k+1$) *gramatika*, ktorá nie je LR(k). Na úrovni jazykov sa však táto hierarchia zrúti už na úrovni LR(1).

Veta 2. *Existuje bezkontextový jazyk L, pre ktorý existuje* LR(1) *gramatika G taká, že $L(G) = L$, ale neexistuje žiadna* LR(0) *gramatika generujúca L.*

Veta 3. *Nech $k \geq 1$ a L je bezkontextový jazyk generovaný nejakou* LR(k) *gramatikou. Potom existuje* LR(1) *gramatika G taká, že $L(G) = L$.*

Pamäťová náročnosť syntaktickej analýzy založenej na LR(1) gramatikách môže byť pomerne veľká. V praxi sa preto často používajú metódy, ktoré sú akýmsi „kompromisom“ medzi LR(1) a LR(0) – používajú síce informáciu o nasledujúcom symbole na vstupe, avšak objem nimi využívanej pamäte je na úrovni algoritmu „posuň a redukuj“ pre LR(0) gramatiky. Príkladmi takýchto metód sú napríklad SLR (angl. ***S**imple **LR***) a najmä LALR (angl. ***L**ook**A**head **LR***).

Dá sa tiež dokázať, že trieda jazykov generovaných LR(0) gramatikami je rovná triede jazykov akceptovaných deterministickými zásobníkovými automatmi *prázdnym zásobníkom*. Dôsledkom napríklad je, že jazyk generovaný LR(0) gramatikou je nutne bezprefixový. Trieda jazykov generovaných LR(1) gramatikami (alebo LR(k) gramatikami pre $k \geq 1$) je zas rovná triede jazykov akceptovaných deterministickými zásobníkovými automatmi *stavom*, t.j. triede $\mathscr{L}_{detCF}$ všetkých deterministických bezkontextových jazykov.