

Per avere PDA più efficienti / più piccole
e con minor nondeterminismo è necessario

(20)

SEMPLIFICARE LE GRAMMATICHE

- 1) Eliminare le produzioni ϵ (del tipo $A \rightarrow \epsilon$) che sono inadatte al bottom-up parsing
- 2) Eliminare le produzioni unitarie (del tipo $A \rightarrow B$) che possono creare dei archi $A \Rightarrow^* A$
- 3) Eliminare simboli inutili, cioè quei terminali e nonterminali che non sono raggiungibili/generabili a partire dal simbolo iniziale S
- 4) Eliminare la recorsione sinistra (del tipo $A \rightarrow A\alpha$), perché inadatta al top-down parsing
- 5) Fattorizzare la grammatica, per ottenere grammatiche con meno nondeterminismo nel top-down parsing.

Eliminare le produzioni ε

(21)

- Input: G libera con produzioni ϵ (del tipo $A \rightarrow \epsilon$)
- Output: G' libera, senza produzioni ϵ , tale che

$$L(G') = L(G) \setminus \{\epsilon\}$$

Oss: Se $\epsilon \in L(G)$ e vogliamo una G'' tale che
 $L(G) = L(G'')$, basta considerare $G' = (NT, T, S, R')$
e definire $G'' = G' \cup \{S' \rightarrow \epsilon \mid S \in S\}$ output dell'algoritmo

$$G'' = (NT \cup \{S'\}, T, S', R' \cup \{S' \rightarrow \epsilon \mid S \in S\})$$

Simboli annullabili: $A \in NT$ tale che $A \Rightarrow^+ \epsilon$

$N(G) = \{A \in NT \mid A \Rightarrow^+ \epsilon\}$ viene calcolato
induttivamente come segue:

$$N_0(G) = \{A \in NT \mid A \rightarrow \epsilon \in R\}$$

$$N_{i+1}(G) = N_i(G) \cup \{B \in NT \mid B \rightarrow c_1 \dots c_k \in R \\ \text{e } (c_1, \dots, c_k) \in N_i(G)\}$$

Oss: • $N_i(G) \subseteq N_{i+1}(G)$ perché aggiungo qualcosa ad ogni passo

• $\exists i_c$ tale che $N_{i_c}(G) = N_{i_c+1}(G)$ perché NT è finito!

Fatto: L'insieme $N(G) = N_{i_c}(G)$ è esattamente
l'insieme di tutti i simboli annullabili

Una volta calcolato $N(G)$ per $G = (NT, T, S, R)$, (22)
 costruiamo la grammatica $G' = (NT, T, S, R')$ dove
per ogni produzione $A \rightarrow \alpha \in R$, con $\alpha \neq \epsilon$, in cui appaiono
 i simboli annullabili z_1, \dots, z_k , mettiamo in R'
 tutte le produzioni del tipo $A \rightarrow \alpha'$ dove α' si ottiene
 da α cancellando tutti i possibili sottoinsiemi di
 z_1, \dots, z_k (incluso \emptyset), ad eccezione del caso in
cui α' risulta ϵ .

- in G' non mettiamo produzioni $A \rightarrow \epsilon \in R$
- in G' non introduciamo mai produzioni
del tipo $A \rightarrow \epsilon$

Proposizione Data una grammatica libera G , la
 grammatica G' determinata dall'algoritmo sopra
 non ha ϵ -produzioni e $L(G') = L(G) \setminus \{\epsilon\}$

(Come già detto, se vogliamo che la nuova grammatica
 sia del tutto equivalente a G , allora bisogna ammettere
 una produzione ϵ per il nuovo simbolo iniziale;

$$G'' = (NT \cup \{S'\}, T, S'; R' \cup \{S' \rightarrow \epsilon | S\})$$

Esempio:

(23)

$$G = \begin{cases} S \rightarrow AB \\ A \rightarrow aAA \mid \epsilon \\ B \rightarrow bBB \mid \epsilon \end{cases}$$

$N_0(G) = \{A, B\}$
 $N_1(G) = \{A, B, S\}$
 $= N(G)$

Consideriamo $S \rightarrow AB \in R$. Dobbiamo considerare 4 casi:

$$\begin{array}{lll} \emptyset & \Rightarrow & S \rightarrow AB \\ \{B\} & \Rightarrow & S \rightarrow A \\ \{A\} & \Rightarrow & S \rightarrow B \\ \{A, B\} & \Rightarrow & \cancel{S \rightarrow \epsilon} \quad \text{non la mettiamo in } R' \end{array}$$

$S \rightarrow AB \mid A \mid B$

Ora consideriamo $A \rightarrow aAA \in R$. Dobbiamo considerare 4 casi:

$$\begin{array}{lll} \emptyset & \Rightarrow & A \rightarrow aAA \\ \{A\} & \Rightarrow & A \rightarrow aA \quad \left. \begin{array}{l} \\ \end{array} \right\} \text{duplicata} \\ \{A\} & \Rightarrow & A \rightarrow aA \quad \left. \begin{array}{l} \\ \end{array} \right\} \\ \{A, A\} & \Rightarrow & A \rightarrow a \end{array}$$

$\Rightarrow A \rightarrow aAA \mid aA \mid a$

Ora consideriamo $B \rightarrow bBB$, e nello stesso modo di prima, otteniamo $B \rightarrow bBB \mid bB \mid b$

Le due ϵ -produzioni $A \rightarrow \epsilon$ e $B \rightarrow \epsilon$ sono ignorate.

$$\Rightarrow G' = \begin{cases} S \rightarrow AB \mid A \mid B \\ A \rightarrow aAA \mid aA \mid a \\ B \rightarrow bBB \mid bB \mid b \end{cases}$$

N.B. $\epsilon \in L(G)$ $S \Rightarrow AB \Rightarrow B \Rightarrow \epsilon$
 $\epsilon \notin L(G')$

$$G'' = \begin{cases} S' \rightarrow \epsilon \mid S \\ S \text{ come per } G' \end{cases} \quad \text{e tale che } L(G) = L(G'')$$

Eliminare le produzioni unitarie

(24)

- produzioni unitarie: $A \rightarrow B$ con $A, B \in NT$
- coppie unitarie: (A, B) tale che $A \Rightarrow^* B$
usando solo produzioni unitarie

Calcolare le coppie unitarie induttivamente

$$1) U_0(G) = \{(A, A) \mid A \in NT\}$$

$$2) U_{i+1}(G) = U_i(G) \cup \\ \{(A, C) \mid (A, B) \in U_i(G) \\ \text{e } B \rightarrow C \in R\}$$

Oss: $U_i(G) \subseteq U_{i+1}(G)$

• $\exists i_c$ tale che $U_{i_c}(G) = U_{i_c+1}(G)$ perché NT è finito

Per definizione, $U(G) = U_{i_c}(G)$, detto insieme di tutte le coppie unitarie.

Algoritmo: Data $G = (NT, T, R, S)$ libera, si definisce $G' = (NT, T, R', S)$ dove, per ogni $(A, B) \in U(G)$, R' contiene tutte le produzioni $A \rightarrow \alpha$, dove $B \rightarrow \alpha \in R$ e non è unitaria.

Oss: Poiché, per ogni $A \in NT$, la coppia $(A, A) \in U(G)$, R' contiene tutte le produzioni non-unitarie di R e in aggiunta un po' di altre.

Teorema

Sia $G = (NT, T, R, S)$ libera e sia $U(G)$ l'insieme delle sue coppie unitarie. Sia $G' = (NT, T, R', S)$ la grammatica ottenuta dall'algoritmo sopra.
Allora G' non ha produzioni unitarie e $L(G) = L(G')$.

Esempio

$$\begin{array}{l} E \rightarrow E + T \mid T \\ T \rightarrow T * A \mid A \\ A \rightarrow a \mid b \mid (E) \end{array} \quad \left\{ \begin{array}{l} \text{gram. non ambigua} \\ \text{per esp. aritmetiche} \\ G \text{ per\`o ha prod. unitarie} \\ E \rightarrow T \text{ e } T \rightarrow A \end{array} \right.$$

$$U_0(G) = \{(E, E), (T, T), (A, A)\}$$

$$U_1(G) = U_0(G) \cup \{(E, T), (T, A)\}$$

$$U_2(G) = U_1(G) \cup \{(E, A)\} = U_3(G) = U(G)$$

$$G' \quad \left\{ \begin{array}{ll} E \rightarrow E + T & \text{produzioni presenti} \\ T \rightarrow T * A & \text{grasse a } U_0(G) \\ A \rightarrow a \mid b \mid (E) & \end{array} \right.$$

in aggiunta

$$E \rightarrow T * A \quad \text{per\`o } (E, T) \in U_1(G)$$

$$T \rightarrow a \mid b \mid (E) \quad \text{per\`o } (T, A) \in U_1(G)$$

$$E \rightarrow a \mid b \mid (E) \quad \text{per\`o } (E, A) \in U_2(G)$$

Quindi, riassumendo, G' è

$$E \rightarrow E + T \mid T * A \mid a \mid b \mid (E)$$

$$T \rightarrow T * A \mid a \mid b \mid (E)$$

$$A \rightarrow a \mid b \mid (E)$$

non contiene produzioni unitarie ed è equivalente a G

Rimuovere i simboli inutili

(26)

Def: Un simbolo $X \in T \cup NT$ è

- un generatore se $\exists w \in T^*$ con $X \Rightarrow^* w$
- rappresentabile se $S \Rightarrow^* \alpha X \beta$ per qualche $\alpha, \beta \in (T \cup NT)^*$
- utile se è sia generatore, sia rappresentabile
ovvero se $S \Rightarrow^* \alpha X \beta \Rightarrow^* z \in L(G)$, cioè X compare in almeno una derivazione di una stringa $z \in L(G)$.

Esempio

$$\begin{array}{l} S \rightarrow AB | a \\ B \rightarrow b \end{array} G$$

Poiché $\left. \begin{array}{l} a \Rightarrow^* a \\ b \Rightarrow^* b \\ S \Rightarrow^* a \\ B \Rightarrow^* b \end{array} \right\} \Rightarrow \text{generator} = \{S, B, a, b\}$
(manca A)

↓
possiamo eliminare
tutte le produzioni che
includono A

$$G' \left[\begin{array}{l} S \rightarrow a \\ B \rightarrow b \end{array} \right]$$

Ora B non è rappresentabile da S, allora elimino B

$$G'' \left[S \rightarrow a \right]$$

G'' è equivalente a G , ma non contiene simboli inutili

Come calcolare i generatori?

(27)

Ricorda: X è un generatore se $\exists w \in T^*$: $X \Rightarrow^* w$

1) $G_0(G) = T$

se $a \in T$, $a \Rightarrow^* a$!

allora tutti i terminali sono generatori

2) $G_{i+1}(G) = G_i(G) \cup$

$$\{ B \in NT \mid B \rightarrow c_1 \dots c_k \in R \\ \text{e } c_1, \dots, c_k \in G_i(G) \}$$

Oss: Nel punto 2) è contemplato anche il caso $k=0$ (cioè $B \rightarrow \varepsilon \in R$): se $B \rightarrow \varepsilon$, allora B è un generatore.

Oss: • $G_i(G) \subseteq G_{i+1}(G)$

• $\exists i_c$ tale che $G_{i_c}(G) = G_{i_c+1}(G)$ perché TNT è finito

Fatto L'insieme $G(G) = G_{i_c}(G)$ è esattamente l'insieme di tutti i simboli generatori di G .

Come calcolare i raggiungibili?

(28)

Ricordi: X è raggiungibile sse

$$S \xrightarrow{*} \alpha X \beta \text{ per qualche } \alpha, \beta \in (NTUT)^*$$

1) $R_0(G) = \{S\}$

2) $R_{i+1}(G) = R_i(G) \cup$
 $\cup \{x_1, \dots, x_k\}$
 $B \in R_i(G),$
 $B \rightarrow x_1 \dots x_k \in R$

Oss: • $R_i(G) \subseteq R_{i+1}(G)$

• $\exists i_c. R_{i_c}(G) = R_{i_c+1}(G)$ perché TUT è finito

Fatto L'insieme $R(G) = R_{i_c}(G)$ è esattamente
l'insieme di tutti i simboli raggiungibili di G .

Come rimuovere i simboli inutili?

- 1) Prima elimino tutti i non-generatori
(e tutte le produzioni che usano almeno uno di questi)
- 2) Poi, dalla nuova grammatica, elimino tutti i
non-raggiungibili (e tutte le prod. che lo usano)

\Rightarrow La grammatica risultante è equivalente
all'originale e non contiene simboli inutili

Eliminare i simboli inutili

(29)

Teorema Sia $G = (NT, T, R, S)$ una grammatica libera tale che $L(G) \neq \emptyset$.

- 1) Sia G_1 la grammatica che si ottiene da G eliminando tutti i simboli che non appartengono a $G(G)$, e tutte le produzioni che fanno uso di almeno uno di tali simboli. \times
- 2) Sia G_2 la grammatica che si ottiene da G_1 eliminando tutti i simboli che non appartengono a $R(G_1)$, e tutte le produzioni che fanno uso di almeno uno di tali simboli.

Allora G_2 non ha simboli inutili e $L(G_2) = L(G)$.

Dimostrazione

- $L(G_2) \subseteq L(G)$ è ovvio, perché G_2 contiene meno produzioni di G .
- $L(G) \subseteq L(G_2)$: dobbiamo dimostrare che se $S \xrightarrow{G}^* w$ allora $S \xrightarrow{G_2}^* w$. Ma ogni simbolo usato nella derivazione $S \xrightarrow{G}^* w$ è ovviamente sia raggiungibile, sia generatore!! Allora quella derivazione è anche una derivazione per G_2 .

Poiché $L(G) \neq \emptyset$, allora S è un generatore e così rimane in G_1 , e poi in G_2

L'ordine dei 2 pass è importante!

(30)

- 1) prima elimino i non-generatori
- 2) poi i non-rappresentabili.

Ma se inverso l'ordine, allora puoi captare
che non elimini tutti i simboli inutili !!

Esempio

$$S \rightarrow AB | a$$

$$B \rightarrow b$$

1) Elimino i non-generatori

$$S \rightarrow a$$

$$B \rightarrow b$$

2) Elimino i non-rappresentabili

$$S \rightarrow a$$

2) Elimino i non-rappresentabili

$$S \rightarrow AB | a$$

$$B \rightarrow b$$

Tutti
rapp.

1) Elimino i non-generatori

$$S \rightarrow a$$

$$B \rightarrow b$$

CORRETTO!

tutti i simboli inutili
sono stati rimossi

è rimasto un
simbolo inutile: B
e la pd. $B \rightarrow b$

Esempio

(31)

$$\left. \begin{array}{l} S \rightarrow AB \mid ac \\ A \rightarrow a \\ B \rightarrow bB \\ C \rightarrow b \mid AC \\ D \rightarrow a \mid as \end{array} \right\} G$$

$$G(G) = \{a, b, A, C, D, S\}$$

solo B non è generatore
 ↓

$$R(G_1) = \{S, a, c, b, A\}$$

$$G_1 \left[\begin{array}{l} S \rightarrow ac \\ A \rightarrow a \\ C \rightarrow b \mid AC \\ D \rightarrow a \mid as \end{array} \right]$$

solo D non è raggiungibile



$$\left. \begin{array}{l} S \rightarrow ac \\ A \rightarrow a \\ C \rightarrow b \mid AC \end{array} \right\} G_2$$

è la grammatica
 semplificata, equivalente
 a G, senza simboli inutile.

$$L(G_2) = \{ab, aab, aaab, \dots\} = a^+b$$

È possibile trovare una grammatica più semplice di G₂
 per il lang. a⁺b?

$$S \rightarrow aS \mid ab$$

Mettere le cose insieme...

(32)

Se, nel semplificare la grammatica G , seguiamo questo ordine:

- 1) eliminare le ϵ -produzioni
- 2) eliminare le produzioni unitarie (e quindi i "cicli")
- 3) eliminare i simboli inutile

Allora la grammatica risultante è garantita non avere né ϵ -produzioni, né produzioni unitarie, né simboli inutili, ed è equivalente a quella di partenza!

N.B. L'ordine è importante poiché alcune delle costituzioni possono interagire tra loro!

Ad es: durante la fase di eliminazione delle ϵ -produzioni, potremmo introdurre prod. unitarie!

\Rightarrow le ϵ -produzioni vanno eliminate prima di procedere all'eliminazione delle produzioni unitarie

Esempio

(33)

$$\begin{array}{l} S \rightarrow aAa \\ A \rightarrow C \\ C \rightarrow S \mid \epsilon \end{array} \quad]_G$$

1) Togliere ε-produzioni

$$N(G) = \{C, A\}$$



$$G' \left[\begin{array}{l} S \rightarrow aAa \mid aa \\ A \rightarrow C \\ C \rightarrow S \end{array} \right]$$

- tolgo $C \rightarrow \epsilon$
- non metto $A \rightarrow \epsilon$

2) Togliere le prod. unitarie

$$U(G') = \{(A, A), (C, C), (S, S), (A, C), (C, S), (A, S)\}$$

$$G'' \left[\begin{array}{ll} S \rightarrow aAa \mid aa & \text{perché } (S, S) \in U(G') \\ C \rightarrow aAa \mid aa & \text{perché } (C, S) \in U(G') \\ A \rightarrow aAa \mid aa & \text{perché } (A, S) \in U(G') \end{array} \right]$$

3) Rimuovere i simboli inutili

$$G(G'') = \{S, a, A, C\} \text{ tutti generatori}$$

$$R(G'') = \{S, a, A\} \text{ ma non } C!$$



$$\begin{array}{l} S \rightarrow aAa \mid aa \\ A \rightarrow aAa \mid aa \end{array} \quad]_{G'''}$$

È possibile trovare una grammatica ancora più semplice per $L(G''') = (aa)^+$?

$$\begin{array}{l} S \rightarrow aSa \mid aa \quad \text{o anche} \\ S \rightarrow aaS \mid aa \end{array}$$

Forme Normali

- di Chomsky
- di Greibach

garantiscono proprietà particolarie alle grammatiche

Forma normale di Chomsky

Tutte le sue produzioni sono della forma

$$\begin{array}{ll} A \rightarrow BC & (\text{con } \varepsilon \text{ trattato a parte:} \\ A \rightarrow a & S \rightarrow \varepsilon \mid BC \dots) \end{array}$$

Oss: se G libera è in forma normale di Chomsky, allora

- non ha ε -produzioni
- non ha produzioni unitarie

(S non compare mai a dx in una produzione)

Oss: ogni grammatica libera G può essere trasformata in una equivalente G' in forma normale di Chomsky.

Forma Normale di Greibach

Tutte le produzioni sono della forma

$$\begin{array}{ll} A \rightarrow aBC & (\text{con } \varepsilon \text{ trattato a} \\ A \rightarrow aB & \text{parte, come sopra}) \\ A \rightarrow a \end{array}$$

Oss: - no ε -produzioni / no prod. unitarie

- non è mai ricorsiva sx!

- ogni prod., applicata in una derivazione, allunga il prefisso di terminale \Rightarrow parser costituito a partire da f.n. d. Greibach sono meno nondet.

Oss: Ogni G libera può essere trasformato in una equiv. f.n. Greibach

Eliminare la ricorsione sinistra

(35)

(problema per parser
top-down)

- produzione ricorsiva sx: $A \rightarrow A\alpha \in R$

- G è ricorsiva sx: $A \xrightarrow{*} A\alpha$ per qualche AGNT e $\alpha \in (TUNT)^*$

Come rimuove la ricorsione sx immediata?

$$A \rightarrow A\alpha_1 | \dots | A\alpha_n | \beta_1 | \dots | \beta_m$$

(le stringhe β_i non cominciano per A)

Queste produzioni possono essere rimpiazzate da

$$A \rightarrow \beta_i A' | \dots | \beta_m A'$$

$$A' \rightarrow \alpha_1 A' | \dots | \alpha_n A' | \varepsilon$$

Se nella gr. originale abbiamo la derivazione

$$A \Rightarrow A\alpha_{i_1} \Rightarrow A\alpha_{i_2}\alpha_{i_1} \Rightarrow \dots \Rightarrow A\alpha_{i_k}\dots\alpha_{i_2}\alpha_{i_1} \Rightarrow \beta_i\alpha_{i_k}\dots\alpha_{i_2}\alpha_{i_1}$$

allora nella nuova grammatica avremo

$$\begin{aligned}
 A \Rightarrow \beta_i A' &\Rightarrow \beta_i\alpha_{i_k} A' \Rightarrow \dots \Rightarrow \beta_i\alpha_{i_k}\dots\alpha_{i_2} A' \Rightarrow \beta_i\alpha_{i_k}\dots\alpha_{i_2}\alpha_{i_1} A' \\
 &\Rightarrow \beta_i\alpha_{i_k}\dots\alpha_{i_2}\alpha_{i_1}
 \end{aligned}$$

e viceversa

Esempio

$$A \rightarrow Aa \mid b$$



$$A \rightarrow bA'$$

$$A' \rightarrow aA' \mid \epsilon$$

Esempio

$$A \rightarrow Ab \mid Ac \mid d$$



$$A \rightarrow dA'$$

$$A' \rightarrow bA' \mid cA' \mid \epsilon$$

OSS: Se $G = [A \rightarrow Aa, \dots]$, non si può applicare l'algoritmo, perché mancano le produzioni di base da cui partire ($A \rightarrow \beta_1 \dots \mid \beta_m$)

Infatti, $L(G) = \emptyset$ e la grammatica corrispondente non ha produzioni.

Ricorsione SX non-immediata

(37)

Consideriamo

$$\begin{array}{l} S \rightarrow Ba \mid b \\ B \rightarrow Bc \mid Sc \mid d \end{array} \quad]^G$$

In G c'è ricorsione SX immediata ($B \rightarrow Bc$),
ma anche non-immediata ($S \Rightarrow Ba \Rightarrow Sc$).

Algoritmo: Input: G libera senza ϵ -produzioni, senza
prod. unitarie, ma con ricorsione SX
non-immediata

Output: G' libera, senza ricorsione SX, ma
può avere ϵ -produzioni

Sia $NT = \{A_1, A_2, \dots, A_n\}$ in un ordine fissato

For i from 1 to n {

1) For j from 1 to $i-1$ {

sostituisci ogni produzione della forma

$A_i \rightarrow A_j \alpha$ con le produzioni

$A_i \rightarrow \beta_1 \alpha \mid \dots \mid \beta_k \alpha$ dove $A_j \rightarrow \beta_1 \mid \dots \mid \beta_k$

sono le produzioni correnti per A_j

}

2) Elimina la ricorsione immediata su A_i :

}

$$S \rightarrow Ba \mid b$$

$$B \rightarrow Bc \mid Sc \mid d$$

$$NT = \begin{cases} S, B \\ 1 \quad 2 \end{cases}$$

- Per $i=1$ (cioè per S), il ciclo interno (J da 1 a ϕ) non viene eseguito; siccome non c'è ricorsione immediata per S , non faccio niente!
 - Per $i=2$ (cioè $A_1 = B$), il ciclo interno (J da 1 a 1) si esegue solo per $A_J = A_1 = S$. Allora la produzione $B \rightarrow Sc$ viene rimpiazzata con

$$B \rightarrow Bac \mid bc$$
- Ora le produzioni complessive per B sono
- $$B \rightarrow Bc \mid Bac \mid bc \mid d$$
- dalle quali dobbiamo eliminare la ricorsione immediata!

Il risultato è

$$B \rightarrow bcB' \mid dB'$$

$$B' \rightarrow cB' \mid acB' \mid \epsilon$$

ovvero la grammatica risultante è

$$S \rightarrow Ba \mid b$$

$$B \rightarrow bcB' \mid dB'$$

$$B' \rightarrow cB' \mid acB' \mid \epsilon$$

Perché si richiede che la G in input (38 bis)
non abbia produzioni unitarie e produzioni epsilon?

$$\begin{array}{l} S \rightarrow AB | a \\ A \rightarrow \epsilon \\ B \rightarrow S | b \end{array} \quad]$$

$S = 1 \quad A = 2 \quad B = 3$

OSS: $S \Rightarrow AB \Rightarrow B \Rightarrow S$

$S \xrightarrow{*} S$

(quindi c'è un ciclo
ric. sx. non immediato)

per $i=1$, $J=1$ a 0 non viene fatto niente

per $i=2$, $J=1$ \Rightarrow ancora niente

$$S \rightarrow AB | a$$

$$\begin{array}{l} A \rightarrow \epsilon \\ B \rightarrow S | b \end{array}$$

ma per $i=3$ e $J=1$

$$S \rightarrow AB | a$$

$$A \rightarrow \epsilon$$

$$B \rightarrow AB | a | b$$

e per $i=3$ e $J=2$

$$S \rightarrow AB | a$$

$$A \rightarrow \epsilon$$

$$B \rightarrow B | a | b$$

e compare una prod.

ric. sx. $B \rightarrow B$

del tutto insensata

perché non serve a
niente

Esercizio

$$\left. \begin{array}{l} S \rightarrow AB \mid D \\ A \rightarrow \epsilon \mid Ab \mid aB \\ B \rightarrow b \mid bB \\ C \rightarrow c \\ D \rightarrow aD \end{array} \right\} G$$

- 1) Togliere simboli inutile
- 2) Togliere ricorsione sx
- 3) Togliere ϵ -produzione
- 4) Togliere prod. unitarie

1) D non generatore
 $C \rightarrow c$ non raggiungibile \Rightarrow

$$\left. \begin{array}{l} S \rightarrow AB \\ A \rightarrow \epsilon \mid Ab \mid aB \\ B \rightarrow b \mid bB \end{array} \right\} G'$$

2) Ricorsione sx

$$(A \rightarrow Ab) \Rightarrow$$

$$\left. \begin{array}{l} S \rightarrow AB \\ A \rightarrow A' \mid aBA' \\ A' \rightarrow bA' \mid \epsilon \\ B \rightarrow b \mid bB \end{array} \right\} G''$$

3) Togliere le ϵ -prod.

$$N(G'') = \{A, A'\} \Rightarrow$$

$$\left. \begin{array}{l} S \rightarrow AB \mid B \\ A \rightarrow A' \mid aBA' \mid aB \\ A' \rightarrow bA' \mid b \\ B \rightarrow b \mid bB \end{array} \right\} G'''$$

4) Togliere le prod. unitarie

$$U(G''') = \text{Id} \cup \{(S, B), (A, A')\}$$

$$\Rightarrow \left. \begin{array}{l} S \rightarrow AB \mid \underline{b} \mid bB \\ A \rightarrow \underline{bA'} \mid b \mid aBA' \mid aB \\ A' \rightarrow bA' \mid b \\ B \rightarrow b \mid bB \end{array} \right.$$

Fattoriizzazione a Sinistra

(importante per top-down parsing)

$$A \rightarrow aBbc \mid aBd$$

Se, in top-down parsing, sulla pila ha A e legge l'input a, non sono in grado di determinare quale produzione scegliere (nondeterminismus)

\Rightarrow raccolgo la parte comune ("aB") alle 2 produzioni e introduco un nuovo nonterminale per rappresentare il "resto"

$$A \rightarrow aBA'$$

$$A' \rightarrow bC \mid d$$

Algorithm generale

- Inizializza $N = NT$
- Ripeti: il ciclo presente finché nessuna modifica è più possibile a N o all'insieme delle produzioni
 - for each $A \in N$ {
 - Sia α il prefisso più lungo comune alle parti destre di alcune produzioni di A;
 - if $\alpha \neq \epsilon$ {
 - Sia A' un nuovo nonterminale; $N := N \cup \{A'\}$
 - Rimpiatta tutte le produzioni per A

$$A \rightarrow \alpha\beta_1 \dots \mid \alpha\beta_k \gamma_1 \dots \mid \gamma_n$$

con le produzioni:

$$A \rightarrow \alpha A' \mid \gamma_1 \dots \mid \gamma_n$$

$$A' \rightarrow \beta_1 \mid \dots \mid \beta_k$$

} }

Esempio

(41)

$$\begin{array}{l} E \rightarrow T \mid T+E \mid T-E \\ T \rightarrow A \mid A*T \\ A \rightarrow a \mid b \mid (E) \end{array} \quad \left\{ \begin{array}{l} \text{se puo'} \\ \text{fattorizzare!} \end{array} \right.$$

$$\begin{array}{l} E \rightarrow TE' \\ E' \rightarrow \epsilon \mid +E \mid -E \\ T \rightarrow AT' \\ T' \rightarrow \epsilon \mid *T \\ A \rightarrow a \mid b \mid (E) \end{array} \quad \left\{ \begin{array}{l} \text{Versione} \\ \text{equivalente} \\ \text{fattorizzata} \end{array} \right.$$