



# نظریه‌ی زبان‌ها و ماشین‌ها

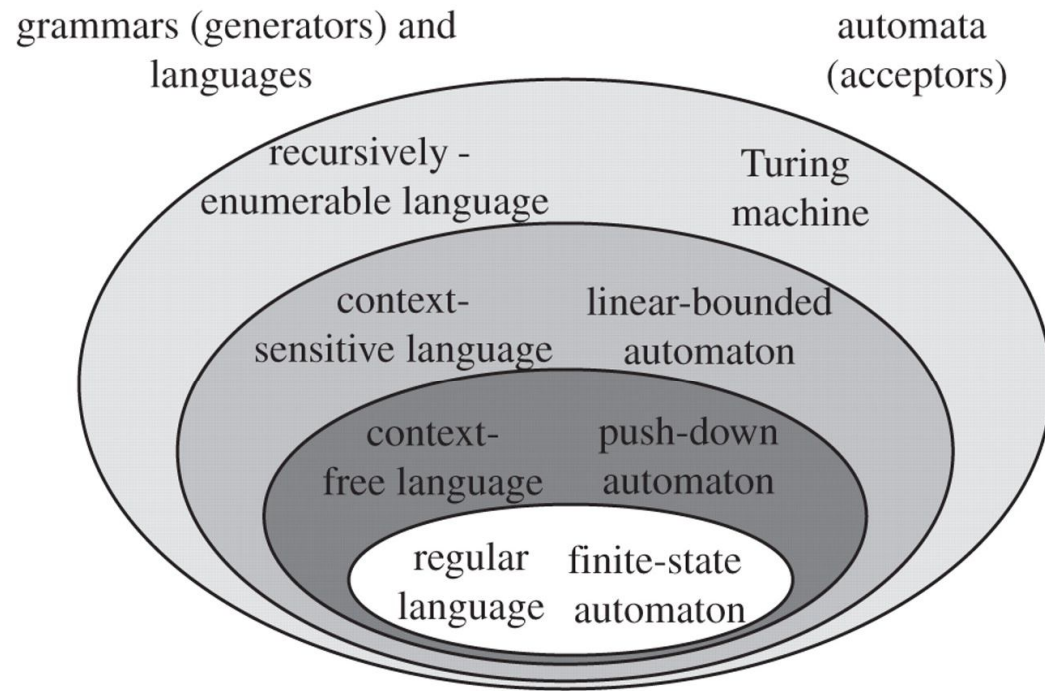
علی شکیبا

دانشگاه ولی‌عصر (عج) رفسنجان

[ali.shakiba@vru.ac.ir](mailto:ali.shakiba@vru.ac.ir)

# فصل ۵: زبان‌های مستقل از متن

# سلسله مراتب زبان‌های چامسکی



the traditional Chomsky hierarchy

# زبان‌های مستقل از متن

• گرامر  $G = (V, T, S, P)$  را **مستقل از متن** گویند هرگاه تمام قوانین آن به صورت  
 $A \rightarrow x$

باشند که  $A \in V$  و  $x \in (V \cup T)^*$ .

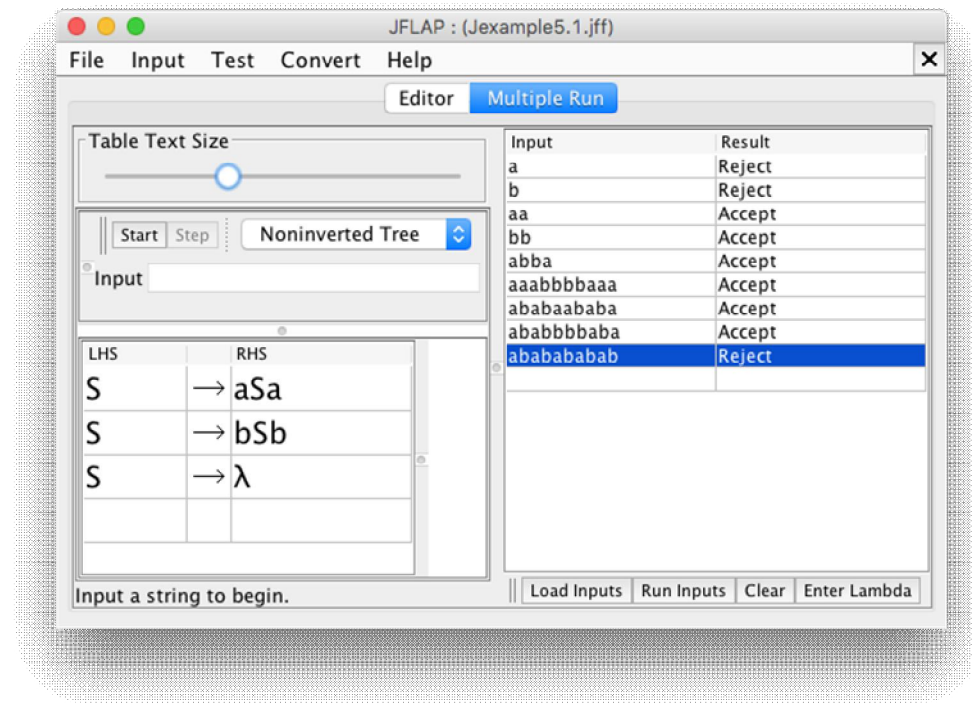
• زبان  $L$  مستقل از متن است هرگاه گرامر مستقل از متنی مانند  $G$  وجود داشته باشد که

$$L = L(G)$$

باشد.

# مثال ٥-١

$$L(G) = \{ww^R : w \in \{a,b\}^*\}$$



# مثال ۵-۳

$$L(G) = \{a^n b^m : n \neq m\}$$

The screenshot shows the JFLAP software interface for a grammar. The title bar reads "JFLAP : (Jexample5.3rm.jff)". The menu bar includes "File", "Input", "Test", "Convert", and "Help". The "Editor" tab is active, showing a "Table Text Size" slider and a "Start Step" dropdown set to "Noninverted Tree". Below this is an "Input a" field. The main area contains a table of grammar rules:

LHS	RHS
S	→ AT
S	→ TB
T	→ aTb
T	→ λ
A	→ aA
A	→ a
B	→ bB
B	→ b

At the bottom of the editor, there is a text field with the prompt "Input a string to begin." and buttons for "Load Inputs", "Run Inputs", "Clear", and "Enter Lambda". To the right of the editor is a table showing the results of testing various input strings:

Input	Result
a	Accept
b	Accept
ab	Reject
ba	Reject
aab	Accept
baa	Reject
aaaabbb	Accept
aabbbb	Accept
aaabbbba	Reject

The "aaabbbba" row is highlighted in blue. At the bottom right of the window, there is a small number "7".

# اشتقاق سمت راست (چپ) ترین

• گرامر . با قواعد زیر مفروض است:

$$S \rightarrow aAB$$

$$A \rightarrow bBb$$

$$B \rightarrow A \mid \lambda$$

• دو اشتقاق مختلف برای رشته  $abb$  :  
• راست ترین اشتقاق

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abb \quad \bullet$$

• چپ ترین اشتقاق

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abbB \Rightarrow abb \quad \bullet$$

• ترتیب اعمال قوانین



# گرامری برای عبارات محاسباتی

- فرض کنید  $G$  گرامری مستقل از متن برای عبارات ساده‌ی محاسباتی است.
- رشته‌ی  $w$  داده شده است؛ آیا  $w \in L(G)$  یا خیر؟
  - به عبارت دیگر؛ آیا این عبارت محاسباتی از نظر نحوی صحیح است؟
- به منظور یافتن دنباله‌ای از اشتقاق‌ها که از  $S$  به  $w$  ختم شود؛ رشته‌ی  $w$  را **تجزیه** می‌کنیم.
- در صورتی که این تجزیه وجود نداشته باشد؛ آنگاه  $w \notin L(G)$ .

# گرامر $G$

- این گرامر به فرم Backus-Naur نوشته شده است.
- متغیرها در بین  $\langle$  و  $\rangle$  محصور شده‌اند.
- منظور از نماد  $::=$  همان نماد  $\rightarrow$  است.

```
<expr> ::= <digit>
          | <expr> <op> <expr>
          | ( <expr> )
<digit> ::= 0|1|2|3|4|5|6|7|8|9
<op> ::= + | *
```

# تجزیه برای $(1+2)^*3$

DERIVATION	PRODUCTION
$\langle \text{expr} \rangle \Rightarrow \langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$	$\langle \text{expr} \rangle ::= \langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$
$\Rightarrow \langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{digit} \rangle$	$\langle \text{expr} \rangle ::= \langle \text{digit} \rangle$
$\Rightarrow \langle \text{expr} \rangle \langle \text{op} \rangle 3$	$\langle \text{digit} \rangle ::= 3$
$\Rightarrow \langle \text{expr} \rangle * 3$	$\langle \text{op} \rangle ::= *$
$\Rightarrow ( \langle \text{expr} \rangle ) * 3$	$\langle \text{expr} \rangle ::= ( \langle \text{expr} \rangle )$
$\Rightarrow ( \langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle ) * 3$	$\langle \text{expr} \rangle ::= \langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$
$\Rightarrow ( \langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{digit} \rangle ) * 3$	$\langle \text{expr} \rangle ::= \langle \text{digit} \rangle$
$\Rightarrow ( \langle \text{exp} \rangle \langle \text{op} \rangle 2 ) * 3$	$\langle \text{digit} \rangle ::= 2$
$\Rightarrow ( \langle \text{expr} \rangle + 2 ) * 3$	$\langle \text{op} \rangle ::= +$
$\Rightarrow ( \langle \text{digit} \rangle + 2 ) * 3$	$\langle \text{expr} \rangle ::= \langle \text{digit} \rangle$
$\Rightarrow ( 1 + 2 ) * 3$	$\langle \text{digit} \rangle ::= 1$

```

<expr> ::= <digit>
         | <expr> <op> <expr>
         | ( <expr> )
<digit> ::= 0|1|2|3|4|5|6|7|8|9
<op> ::= + | *
    
```

# درخت اشتقاق (یا تجزیه)

- ریشه‌ی درخت تجزیه متغیر  $S$  بوده و نام هر یک از گره‌های داخلی
- برگ:  $T \cup \{\lambda\}$
- میانی:  $V$
- فرزندان گره  $A$  اگر از قاعده‌ی  $A \rightarrow a_1 \dots a_n$  استفاده کنیم:
- از چپ به راست دارای نام  $a_1$  تا  $a_n$  هستند.
- گره‌های با برچسب  $\lambda$  فاقد خواهر/برادر هستند.
- **تولید** این درخت؛ الحاق برچسب‌های برگ‌ها از چپ به راست به یکدیگر بدون  $\lambda$  است.

JFLAP : (Jexample5.6.jff)

File Input Test Convert Help

Editor CYK Parse

Table Text Size

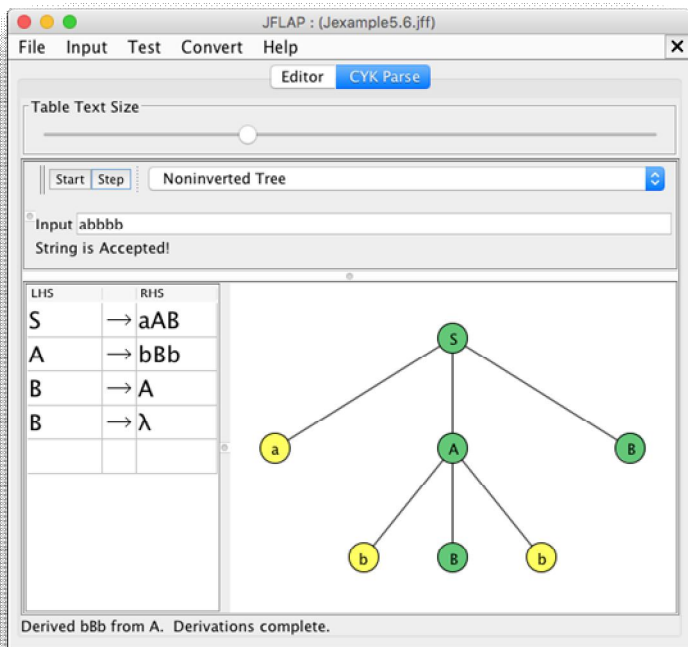
Start Step Noninverted Tree

Input abbbb  
String is Accepted!

LHS	RHS
S	$\rightarrow$ aAB
A	$\rightarrow$ bBb
B	$\rightarrow$ A
B	$\rightarrow$ $\lambda$

Derived  $\lambda$  from B. Derivations complete.

# درخت اشتقاق جزئی



- ریشه‌ی درخت تجزیه متغیر  $S$  بوده و نام هر یک از گره‌های داخلی
- برگ:  $VUTU\{\lambda\}$
- میانی:  $V$
- فرزندان گره  $A$  اگر از قاعده‌ی  $A \rightarrow a_1 \dots a_n$  استفاده کنیم:
- از چپ به راست دارای نام  $a_1$  تا  $a_n$  هستند.
- گره‌های با برچسب  $\lambda$  فاقد خواهر/برادر هستند.

قضیه ۵-۱: فرض کنید  $G = (V, T, S, P)$  گرامری مستقل از متن باشد. آنگاه به ازای هر  $w \in L(G)$ ، یک درخت اشتقاق با تولید  $w$  وجود دارد.

همچنین؛ برای هر درخت اشتقاق مبتنی بر قواعد  $G$ ؛ تولید آن در  $L(G)$  است.

علاوه بر آن؛ تولید یک درخت اشتقاق جزئی؛ فرم جمله‌ای از  $G$  است.

درخت‌های اشتقاق هیچ چیزی در مورد ترتیب اشتقاق بیان نمی‌کنند.

درخت‌های اشتقاق صرفاً بیانگر قوانین مورد استفاده در اشتقاق هستند.

# ابهام

• گرامر  $G = \{V, T, E, P\}$  با  $V = \{E, I\}$ ،  $T = \{a, b, c, +, *, (, )\}$  و قواعد زیر مفروض است

$$E \rightarrow I$$

$$E \rightarrow E+E$$

$$E \rightarrow E^*E$$

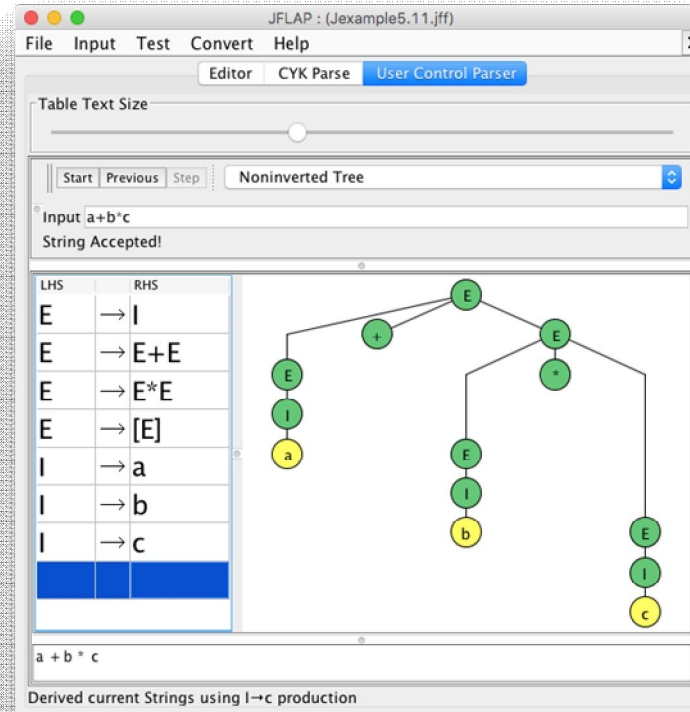
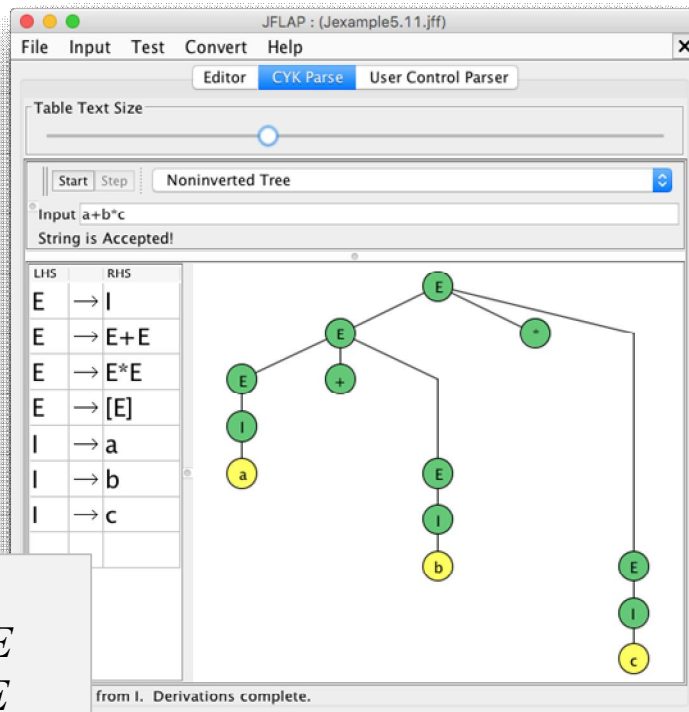
$$E \rightarrow (E)$$

$$I \rightarrow a \mid b \mid c$$

• این گرامر؛ یک گرامر **مبهم** است زیرا برای رشته‌ی  $a+b^*c$  دو درخت تجزیه می‌توان ساخت.



# ابهام رشته‌ی $a+b^*c$



$E \rightarrow I$   
 $E \rightarrow E+E$   
 $E \rightarrow E^*E$   
 $E \rightarrow (E)$   
 $I \rightarrow a | b | c$

# زبان ذاتاً مبهم

- زبان L را ذاتاً مبهم گویند هرگاه هر گرامر آن، مبهم باشد.
- زبان  $L = \{a^n b^n c^m\} \cup \{a^n b^m c^m\}$  ذاتاً مبهم است.

# تجزیه و عضویت

- تجزیه‌ی جستجوی کامل
- مدلی از تجزیه بالا به پایین

$$S \rightarrow SS \mid aSb \mid bSa \mid \lambda$$

• برای رشته‌ی  $w = aabb$  داریم:

$$\begin{aligned} S &\Rightarrow SS \\ S &\Rightarrow aSb \\ S &\Rightarrow bSa \\ S &\Rightarrow \lambda \end{aligned}$$

$$\begin{aligned} S &\Rightarrow SS \Rightarrow SSS \\ S &\Rightarrow SS \Rightarrow aSbS \\ S &\Rightarrow SS \Rightarrow bSaS \\ S &\Rightarrow SS \Rightarrow S \end{aligned}$$

$$\begin{aligned} S &\Rightarrow aSb \Rightarrow aSSb \\ S &\Rightarrow aSb \Rightarrow aaSbb \\ S &\Rightarrow aSb \Rightarrow abSab \\ S &\Rightarrow aSb \Rightarrow ab \end{aligned}$$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$$

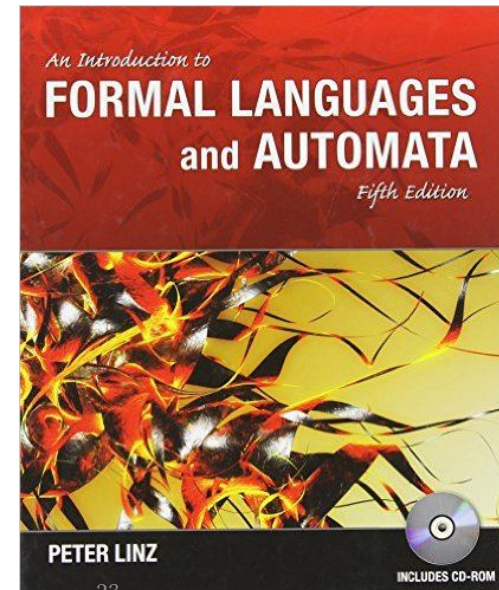
قضیه ۵-۲: اگر گرامر مستقل از متن  $G$  فاقد قوانین مانند  $A \rightarrow B$  باشد؛ انگاه الگوریتم تجزیه‌ی جستجوی کامل برای هر رشته‌ی  $w \in \Sigma^*$ ؛ یا تجزیه‌ای از  $w$  را تولید می‌کند یا اعلام می‌کند که هیچ تجزیه‌ای برای  $w$  ممکن نیست.

## گرامر ساده (یا s-گرامر)

- گرامر مستقل از متن  $G$  را **ساده** یا **s-گرامر** گویند هرگاه تمام قوانین آن به صورت  $A \rightarrow ax$  باشند که  $A \in V$ ،  $a \in T$ ،  $x \in V^*$  و هر زوج  $(A, a)$  حداکثر یک بار در مجموعه قوانین وجود داشته باشند.
- اگر  $G$  یک s-گرامر باشد؛ آنگاه هر رشته‌ی عضو  $L(G)$  را می‌توان با مجموعه عملیات‌های متناسب با  $|W|$  تجزیه نمود.
  - چگونه؟

# در این جلسه آموختیم ...

• فصل ۵



# در جلسه‌ی آینده خواهیم آموخت ...

• فصل ۶

