تمرین دوم
طراحی زبان‌ها

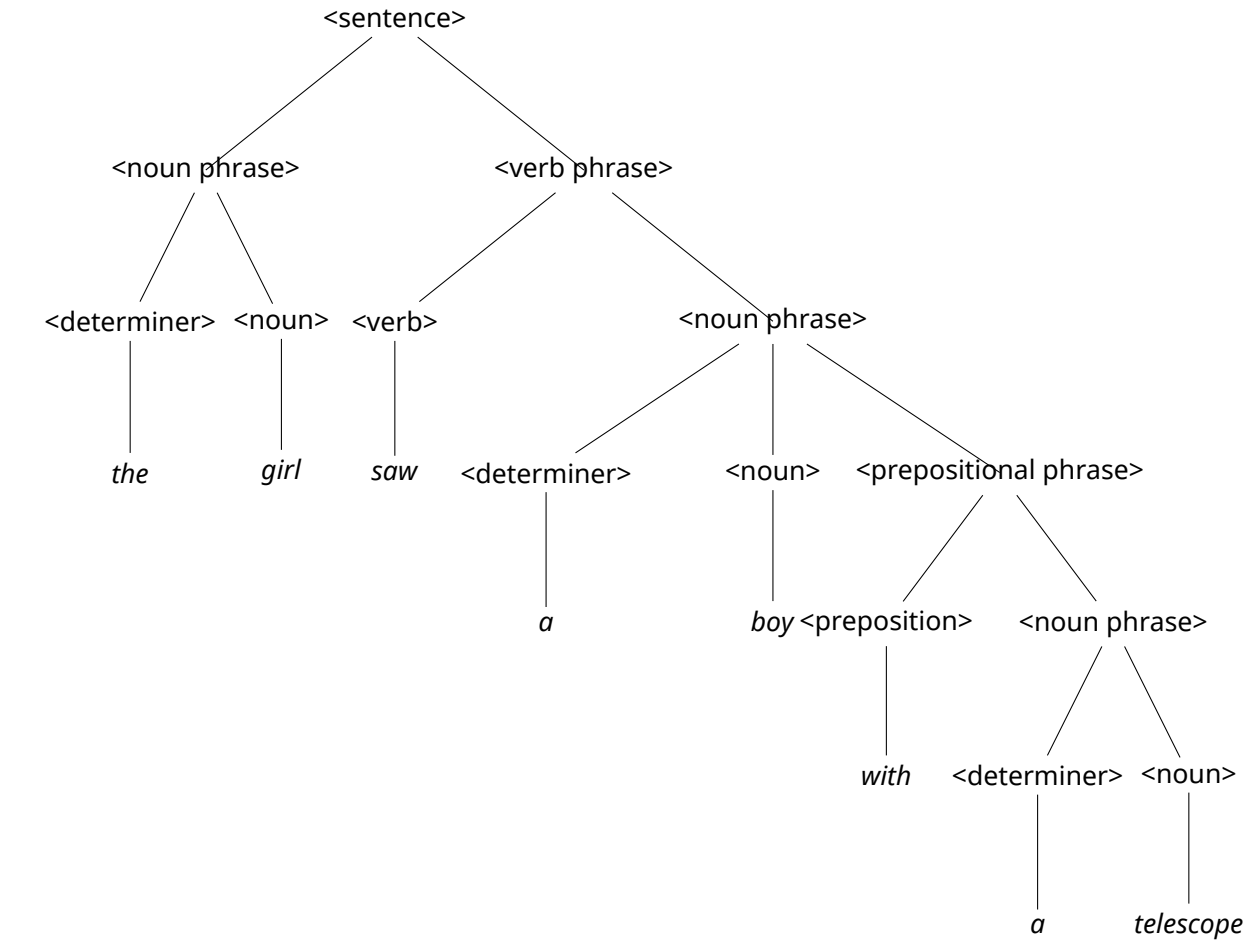
دانیال علی‌حسینی
۹۲۳۱۰۱۴

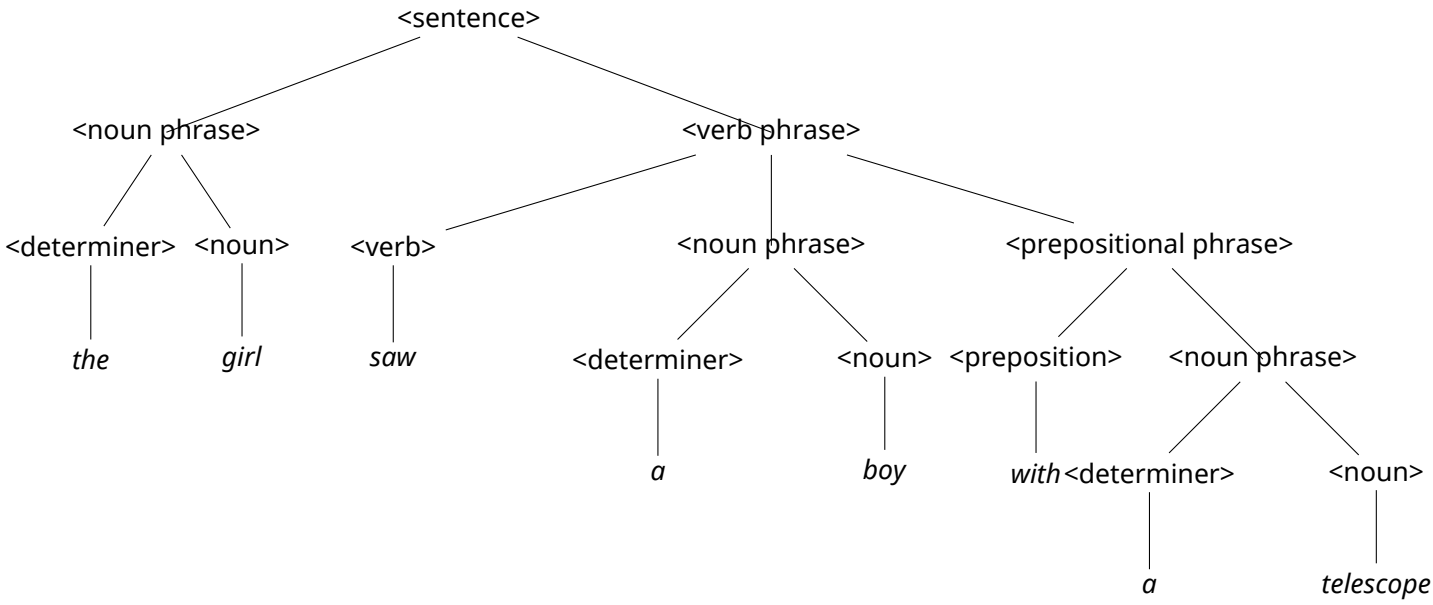
۲۸ مهر ۱۳۹۵

Formal Syntax and Semantics

Page 8

1.
the girl saw a boy with a telescope.

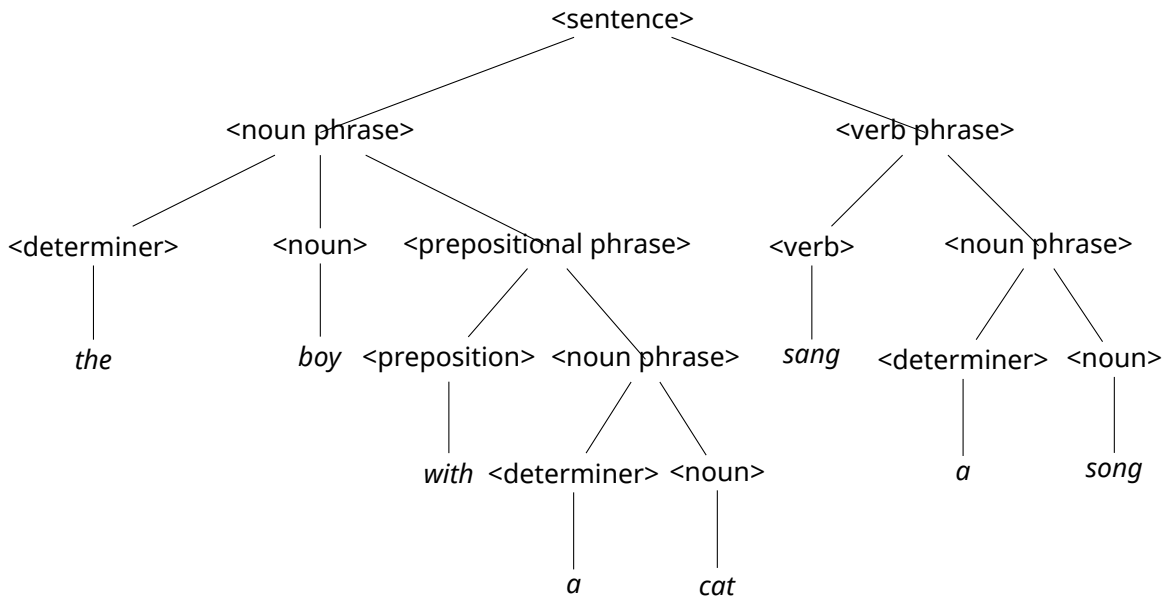




در درخت اشتقاق اول دختری، یک پسر دارای تلسکوپ را میبیند در حالی که در درخت اشتقاق دوم، دختری با استفاده از یک تلسکوپ، پسری را میبیند.

2.

the boy with a cat sang a song.



معنی اول: پسری با استفاده از یک گربه آواز خواند.
معنی دوم: پسری همراه با یک گربه آواز خواندند.

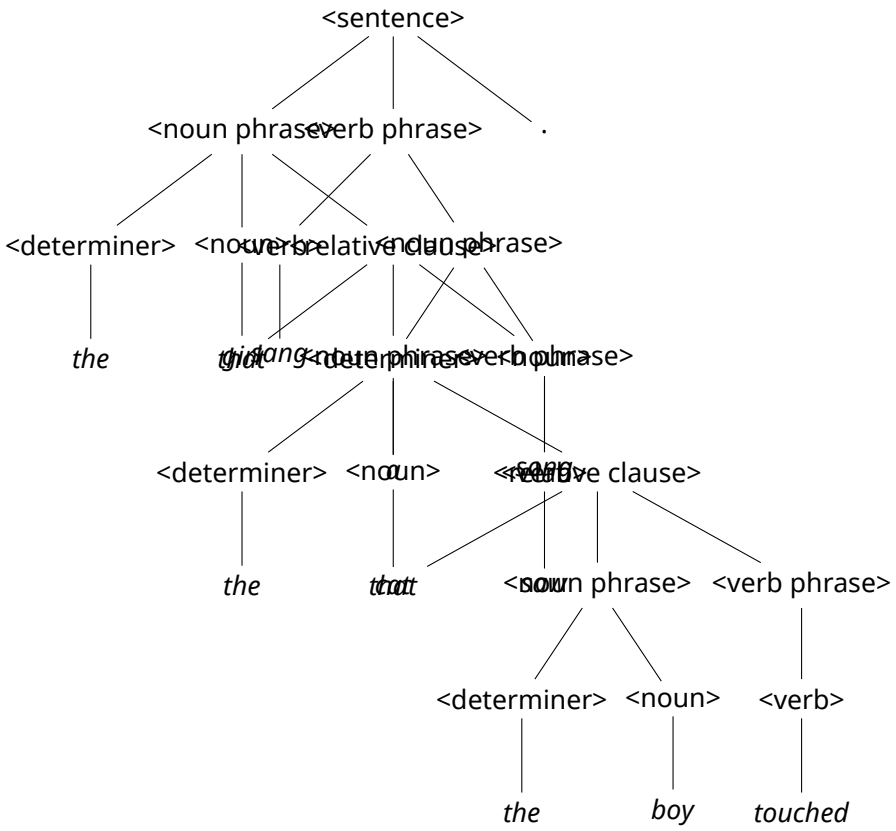
7.

a) $L = \{a^n b^n : n \geq 1\}$

b) $L = \{wy : w, y \in \{a, b\}^*, w = y^{-1}\}$

c) $L = \{w : w \in \{a, b\}^*, w_a = w_b\}$

8.



9.

در ابتدا به اثبات این موضوع میپردازیم که concat دو زبان منظم نیز یک زبان منظم را نتیجه میدهد.
اثبات:
DFA هر دو زبان را میکشیم. حالت پایانی زبان منظم اول را به حالت ابتدایی زبان منظم دوم را تحت انتقال λ. DFA حاصل، DFA مربوط به concat دو زبان خواهد بود. پس concat دو زبان منظم نیز منظم خواهد بود.

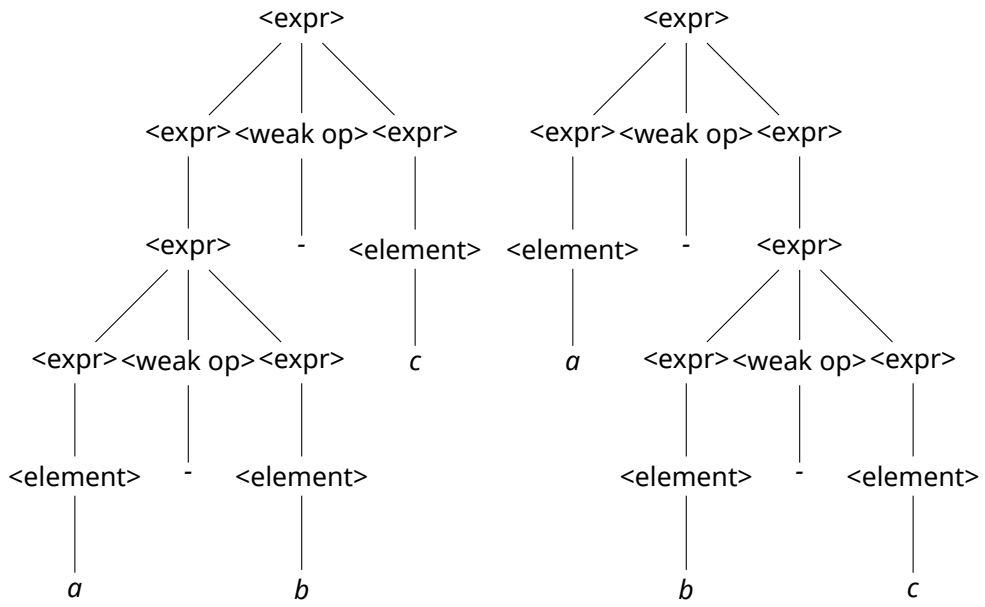
<noun phase> and <prepositional phrase> are regular (based on the lower grammar) so the concatenation of them are regular; and **saw** itself is a regular language so the concatenation of **saw** and <noun phrase> <preposition phrase> is still a regular grammar. In conclusion <verb phrase> ::= saw <noun phrase> <preposition phrase> is a correct grammar based on regular grammars.
With the same inference provided, we can prove that <sentence> ::= <noun phrase><verb phrase> is regular too.

<sentence> ::= <noun phrase><verb phrase> <noun phrase> ::= the boy | the boy <preposition phrase> <verb phrase> ::= saw | saw <noun phrase> | saw <noun phrase> <preposition phrase> <prepositional phrase> ::= by <noun phrase>

Page 16

2.

a-b-c



زبان Wren سعی کرده است تا با استفاده از لایه لایه کردن قواعد، به حل این مشکل بپردازد. در این زبان قواعد به صورتی هستند که تنها عبارات میتوانند از سمت چپ باز شوند؛ پس این زبان از چپ به راست محاسبه میکند؛ که این کار را با استفاده از دو nonterminal <integer expr> و <term> انجام داده است.

3.

```
program errors was
        var a,b : integer ;
        var p,b ; boolean ;
begin
        a := 34;
        if b ≠ 0 then p := true else p := (a+1);
        write p; write q
end
```

Context free errors

```
was –> is
a := 34
if b <> 0 then p := true else p := (a+1) end if
```

Context sensitive or Static semantic errors

1. b has multiple declarations
2. integer can not be assigned to a boolean variable in p := (a+1)
3. q is not declared
4. p must be in integer type in **write p**

5.

a) From high priority to low: - + *

b) * and - : Right to left
   +: Left to right

c)

در واقع طراح زبان برای اینکه بتواند عملکرد پرانتز را پیاده سازی کند، سعی کرده است تا بااستفاده از ایجاد یک لایه بیشتر یا به عبارت دیگر، عمیق‌تر کردن گره مربوط به عبارت داخل پرانتز، اولویت پرانتز را بیشتر از بقیه عملگرها کند. محدودیت این پرانتز این است که تنها به دور عبارت‌هایی میتواند بیاید که در آن تنها از عملگر - استفاده شده است.

6.

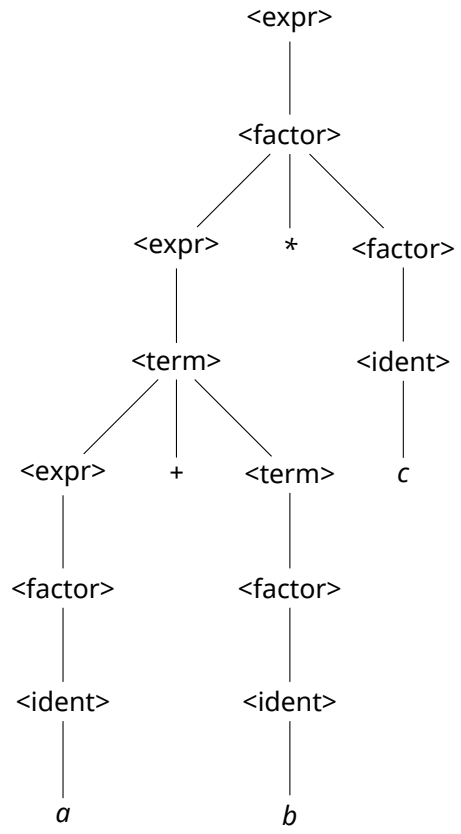
- identifier:
  L := La | Lb | ... | Lz
  L := a | b | ... | z
  گرامر فوق چپگرد است که قابل تبدیل به گرامر راستگرد است. پس میتوان با گرامر سطح ۳ identifier را نوشت
- numeral:
  N := 0N | 1N | ... | 9N
  N := 0 | 1 | ... | 9

9.

<expr> := <sexpr> | <expr> **@** <sexpr> | **(** <expr> **)**
<sexpr> := <pexpr> | <pexpr> **+** <sexpr> | <pexpr> **-** <sexpr> | **(** <sexpr> **)**
<pexpr> := <list> | <list> ***** <pexpr> | **(** <pexpr> **)**
<list> := **[ ]** | **[** <list content> **]**
<list content> := <integer> | **,** <integer>

10.

a+b*c



Page 29

2.

Page 71

1.
value: A synthesized attribute associated with <numeral> indicating the numeral value of <numeral>
length: A synthesized attribute associated with <string> indicating the length of <string>

<string literal> ::= <numeral> H <string>
$< numeral > .value \Leftarrow calculate\_value(< numeral >)$
$< string > .length \Leftarrow calculate\_length(< string >)$
Predicate: $< numeral > .value ==< string > .length$

2.
value: A synthesized attribute associated with <numeral> indicating the numeral value of <numeral>
actual_length: A synthesized attribute associated with <string> indicating the actual length of <string>
expected_length: A inherited attribute associated with <string> indicating the expected length of <string>

<string literal> ::= <numeral> H <string>
$< numeral > .value \Leftarrow calculate\_value(< numeral >)$
$< string > .actual\_length \Leftarrow calculate\_length(< string >)$
$< string > .expected\_length \Leftarrow < numeral > .value$
Predicate: $< string > .actual\_length ==< string > .expected\_length$

4.
n: A synthesized attribute associated with <low hundreds>, <low tens>, <low units> indicating the actual number of C's, X's and I's in them. It is initialized with 0.
value: A synthesized attribute associated with <low hundreds>, <hundreds>, <low tens>, <tens>, <low units>, <units> indicating the actual value of them. It is initialized with 0.

<low hundreds> $[1]$ ::= ε | <low hundreds>$[2]$ C
$< lowhundreds > [1].n \leftarrow < lowhundreds > [2].n + 1$
Predicate: $< lowhundreds > .n \leq 3$

<low tens> $[1]$ ::= ε | <low tens>$[2]$ X
$< lowtens > [1].n \leftarrow < lowtens > [2].n + 1$
Predicate: $< lowtens > .n \leq 3$

<low units> $[1]$ ::= ε | <low units>$[2]$ I
$< lowunits > [1].n \leftarrow < lowunits > [2].n + 1$
Predicate: $< lowunits > .n \leq 3$

<hundreds> ::= <low hundreds>
$< hundreds > .value \leftarrow < lowhundreds > .value$
<hundreds> ::= CD
$< hundreds > .value \leftarrow 4$
<hundreds> ::= D <low hundreds>
$< hundreds > .value \leftarrow 5 + < lowhundreds > .value$
<hundreds> ::= CM
$< hundreds > .value \leftarrow 9$
<low hundreds> ::= ε
$< lowhundreds > .value \leftarrow 0$
<low hundreds>$[1]$ ::= <low hundreds>$[2]$ C
$< lowhundreds > [1].value \leftarrow < lowhundreds > [2].value + 1$

<tens> ::= <low tens>
$< tens > .value \leftarrow < lowtens > .value$
<tens> ::= XL
$< tens > .value \leftarrow 4$

<tens> ::= L <low tens>
$< tens > .value \leftarrow 5 + < lowtens > .value$
<tens> ::= XC
$< tens > .value \leftarrow 9$
<low tens> ::= ε
$< lowtens > .value \leftarrow 0$
<low tens>[1] ::= <low tens>[2] X
$< lowtens > [1].value \leftarrow < lowtens > [2].value + 1$

<units> ::= <low units>
$< units > .value \leftarrow < lowunits > .value$
<units> ::= IV
$< units > .value \leftarrow 4$
<units> ::= V <low units>
$< units > .value \leftarrow 5 + < lowunits > .value$
<units> ::= IX
$< units > .value \leftarrow 9$
<low units> ::= ε
$< lowunits > .value \leftarrow 0$
<low units>[1] ::= <low units>[2] I
$< lowunits > [1].value \leftarrow < lowunits > [2].value + 1$

<roman> ::= <hundreds> <tens> <units>
$< roman > .value \leftarrow < hundreds > .value * 100 + < tens > .value * 10 + < units > .value$