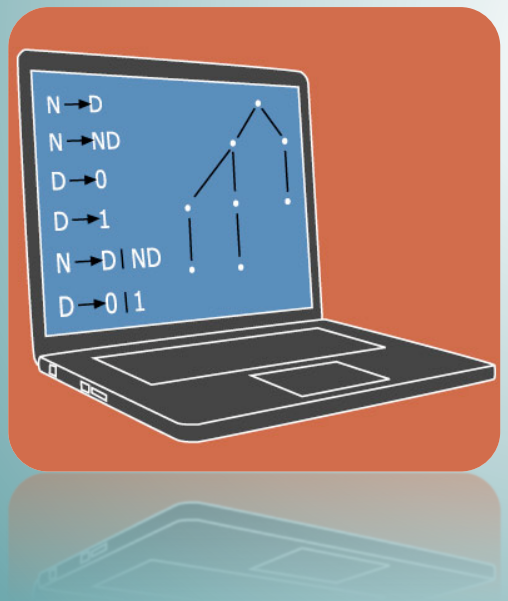
Procesadores de Lenguaje

Analizadores sintácticos descendentes recursivos



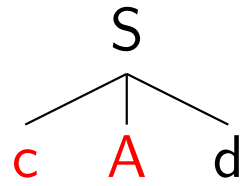
Cristina Tirnauca

DPTO. DE MATEMÁTICAS,

ESTADÍSTICA Y COMPUTACIÓN

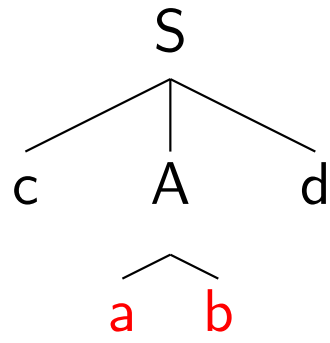
Los analizadores descendentes

El menú para hoy: los analizadores descendetes recursivos

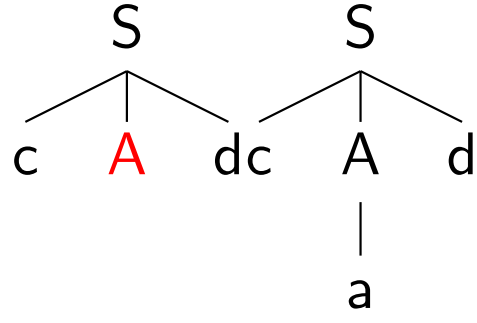


Consideremos la siguiente gramática:

$S \rightarrow c\ A\ d$
$A \rightarrow a\ b$
$A \rightarrow a$



Cadena de entrada: w=cad



Prolog

- Prolog (PROgrammation en LOGique) es un lenguaje de programación lógico.
- Nació de un proyecto que no tenía como objetivo la implementación de un lenguaje de programación, sino el procesamiento de lenguajes naturales.
- Padres: Alain Colmerauer y Philippe Roussel.
- Los programas en Prolog se componen de cláusulas de Horn que constituyen reglas del tipo "modus ponendo ponens" ("Si es verdad el antecedente, entonces es verdad el consecuente").
- Su ejecución se basa en dos conceptos: la unificación y el backtracking.
- Existen dos tipos de cláusulas: hechos y reglas.

Prolog y los analizadores descendentes recursivos (I)

Ejemplo (CFG):	Una posible solución (ineficiente!!!):
S $\rightarrow$ NP VP	s(Z) :- np(X), vp(Y), append(X,Y,Z).
NP $\rightarrow$ DET N	np(Z) :- det(X), n(Y), append(X,Y,Z).
VP $\rightarrow$ V NP	vp(Z) :- v(X), np(Y), append(X,Y,Z).
VP $\rightarrow$ V	vp(Z) :- v(Z).
DET $\rightarrow$ the	det([the]).
DET $\rightarrow$ a	det([a]).
N $\rightarrow$ woman	n([woman]).
N $\rightarrow$ man	n([man]).
V $\rightarrow$ shoots	v([shoots]).

```
 [trace] 1 ?- s([a,woman,shoots]).
Call: (6) s([a, woman, shoots]) ?
Call: (7) np(_G923) ?
Call: (8) det(_G923) ?
Exit: (8) det([the]) ?
Call: (8) n(_G926) ?
Exit: (8) n([woman]) ?
Call: (8) lists:append([the], [woman], _G934) ?
Exit: (8) lists:append([the], [woman], [the, woman]) ?
Exit: (7) np([the, woman]) ?
Call: (7) vp(_G932) ?
Call: (8) v(_G932) ?
Exit: (8) v([shoots]) ?
Call: (8) np(_G935) ?
Call: (9) det(_G935) ?
Exit: (9) det([the]) ?
Call: (9) n(_G938) ?
Exit: (9) n([woman]) ?
Call: (9) lists:append([the], [woman], _G946) ?
Exit: (9) lists:append([the], [woman], [the, woman]) ?
Exit: (8) np([the, woman]) ?
Call: (8) lists:append([shoots], [the, woman], _G949) ?
Exit: (8) lists:append([shoots], [the, woman], [shoots, the,
woman]) ?
Exit: (7) vp([shoots, the, woman]) ?
Call: (7) lists:append([the, woman], [shoots, the, woman], [a,
woman, shoots]) ?
Fail: (7) lists:append([the, woman], [shoots, the, woman], [a,
woman, shoots]) ?
Redo: (9) n(_G938) ?
Exit: (9) n([man]) ?
Call: (9) lists:append([the], [man], _G946) ?
Exit: (9) lists:append([the], [man], [the, man]) ?
Exit: (8) np([the, man]) ?
Call: (8) lists:append([shoots], [the, man], _G949) ?
Exit: (8) lists:append([shoots], [the, man], [shoots, the, man]) ?
Exit: (7) vp([shoots, the, man]) ?
 Call: (7) lists:append([the, woman], [shoots, the, man], [a,
woman, shoots]) ?
Fail: (7) lists:append([the, woman], [shoots, the, man], [a,
woman, shoots]) ?
Redo: (9) det(_G935) ?
Exit: (9) det([a]) ?
Call: (9) n(_G938) ?
Exit: (9) n([woman]) ?
Call: (9) lists:append([a], [woman], _G946) ?
Exit: (9) lists:append([a], [woman], [a, woman]) ?
Exit: (8) np([a, woman]) ?
Call: (8) lists:append([shoots], [a, woman], _G949) ?
Exit: (8) lists:append([shoots], [a, woman], [shoots, a, woman])
?
Exit: (7) vp([shoots, a, woman]) ?
Call: (7) lists:append([the, woman], [shoots, a, woman], [a,
woman, shoots]) ?
Fail: (7) lists:append([the, woman], [shoots, a, woman], [a,
woman, shoots]) ?
Redo: (9) n(_G938) ?
Exit: (9) n([man]) ?
Call: (9) lists:append([a], [man], _G946) ?
Exit: (9) lists:append([a], [man], [a, man]) ?
Exit: (8) np([a, man]) ?
Call: (8) lists:append([shoots], [a, man], _G949) ?
Exit: (8) lists:append([shoots], [a, man], [shoots, a, man]) ?
Exit: (7) vp([shoots, a, man]) ?
Call: (7) lists:append([the, woman], [shoots, a, man], [a,
woman, shoots]) ?
Fail: (7) lists:append([the, woman], [shoots, a, man], [a,
woman, shoots]) ?
Redo: (7) vp(_G932) ?
Call: (8) v(_G932) ?
Exit: (8) v([shoots]) ?
Exit: (7) vp([shoots]) ?
Call: (7) lists:append([the, woman], [shoots], [a, woman,
shoots]) ?
```

Fail: (7) lists:append([the, woman], [shoots], [a, woman, shoots]) ?
Redo: (8) n(_G926) ?
Exit: (8) n([man]) ?
Call: (8) lists:append([the], [man], _G934) ?
Exit: (8) lists:append([the], [man], [the, man]) ?
Exit: (7) np([the, man]) ?
Call: (7) vp(_G932) ?
Call: (8) v(_G932) ?
Exit: (8) v([shoots]) ?
Call: (8) np(_G935) ?
Call: (9) det(_G935) ?
Exit: (9) det([the]) ?
Call: (9) n(_G938) ?
Exit: (9) n([woman]) ?
Call: (9) lists:append([the], [woman], _G946) ?
Exit: (9) lists:append([the], [woman], [the, woman]) ?
Exit: (8) np([the, woman]) ?
Call: (8) lists:append([shoots], [the, woman], _G949) ?
Exit: (8) lists:append([shoots], [the, woman], [shoots, the, woman]) ?
Exit: (7) vp([shoots, the, woman]) ?
Call: (7) lists:append([the, man], [shoots, the, woman], [a, woman, shoots]) ?
Fail: (7) lists:append([the, man], [shoots, the, woman], [a, woman, shoots]) ?
Redo: (9) n(_G938) ?
Exit: (9) n([man]) ?
Call: (9) lists:append([the], [man], _G946) ?
Exit: (9) lists:append([the], [man], [the, man]) ?
Exit: (8) np([the, man]) ?
Call: (8) lists:append([shoots], [the, man], _G949) ?
Exit: (8) lists:append([shoots], [the, man], [shoots, the, man]) ?
Exit: (7) vp([shoots, the, man]) ?
Call: (7) lists:append([the, man], [shoots, the, man], [a, woman, shoots]) ?
Fail: (7) lists:append([the, man], [shoots, the, man], [a, woman, shoots]) ?
Redo: (9) det(_G935) ?
Exit: (9) det([a]) ?
Call: (9) n(_G938) ?
Exit: (9) n([woman]) ?
Call: (9) lists:append([a], [woman], _G946) ?
Exit: (9) lists:append([a], [woman], [a, woman]) ?
Exit: (8) np([a, woman]) ?
Call: (8) lists:append([shoots], [a, woman], _G949) ?
Exit: (8) lists:append([shoots], [a, woman], [shoots, a, woman]) ?
Exit: (7) vp([shoots, a, woman]) ?
Call: (7) lists:append([the, man], [shoots, a, woman], [a, woman, shoots]) ?
Fail: (7) lists:append([the, man], [shoots, a, woman], [a, woman, shoots]) ?
Redo: (9) n(_G938) ?
Exit: (9) n([man]) ?
Call: (9) lists:append([a], [man], _G946) ?
Exit: (9) lists:append([a], [man], [a, man]) ?
Exit: (8) np([a, man]) ?
Call: (8) lists:append([shoots], [a, man], _G949) ?
Exit: (8) lists:append([shoots], [a, man], [shoots, a, man]) ?
Exit: (7) vp([shoots, a, man]) ?
Call: (7) lists:append([the, man], [shoots, a, man], [a, woman, shoots]) ?
Fail: (7) lists:append([the, man], [shoots, a, man], [a, woman, shoots]) ?
Redo: (7) vp(_G932) ?
Call: (8) v(_G932) ?
Exit: (8) v([shoots]) ?
Exit: (7) vp([shoots]) ?
Call: (7) lists:append([the, man], [shoots], [a, woman, shoots]) ?
Fail: (7) lists:append([the, man], [shoots], [a, woman, shoots]) ?
Redo: (8) det(_G923) ?
Exit: (8) det([a]) ?
Call: (8) n(_G926) ?
Exit: (8) n([woman]) ?
Call: (8) lists:append([a], [woman], _G934) ?

Exit: (8) lists:append([a], [woman], [a, woman]) ?
Exit: (7) np([a, woman]) ?
Call: (7) vp(_G932) ?
Call: (8) v(_G932) ?
Exit: (8) v([shoots]) ?
Call: (8) np(_G935) ?
Call: (9) det(_G935) ?
Exit: (9) det([the]) ?
Call: (9) n(_G938) ?
Exit: (9) n([woman]) ?
Call: (9) lists:append([the], [woman], _G946) ?
Exit: (9) lists:append([the], [woman], [the, woman]) ?
Exit: (8) np([the, woman]) ?
Call: (8) lists:append([shoots], [the, woman], _G949) ?
Exit: (8) lists:append([shoots], [the, woman], [shoots, the, woman]) ?
Exit: (7) vp([shoots, the, woman]) ?
Call: (7) lists:append([a, woman], [shoots, the, woman], [a, woman, shoots]) ?
Fail: (7) lists:append([a, woman], [shoots, the, woman], [a, woman, shoots]) ?
Redo: (9) n(_G938) ?
Exit: (9) n([man]) ?
Call: (9) lists:append([the], [man], _G946) ?
Exit: (9) lists:append([the], [man], [the, man]) ?
Exit: (8) np([the, man]) ?
Call: (8) lists:append([shoots], [the, man], _G949) ?
Exit: (8) lists:append([shoots], [the, man], [shoots, the, man]) ?
Exit: (7) vp([shoots, the, man]) ?
Call: (7) lists:append([a, woman], [shoots, the, man], [a, woman, shoots]) ?
Fail: (7) lists:append([a, woman], [shoots, the, man], [a, woman, shoots]) ?
Redo: (9) det(_G935) ?
Exit: (9) det([a]) ?
Call: (9) n(_G938) ?
Exit: (9) n([woman]) ?
Call: (9) lists:append([a], [woman], _G946) ?
Exit: (9) lists:append([a], [woman], [a, woman]) ?
Exit: (8) np([a, woman]) ?
Call: (8) lists:append([shoots], [a, woman], _G949) ?
Exit: (8) lists:append([shoots], [a, woman], [shoots, a, woman]) ?
Exit: (7) vp([shoots, a, woman]) ?
Call: (7) lists:append([a, woman], [shoots, a, woman], [a, woman, shoots]) ?
Fail: (7) lists:append([a, woman], [shoots, a, woman], [a, woman, shoots]) ?
Redo: (9) n(_G938) ?
Exit: (9) n([man]) ?
Call: (9) lists:append([a], [man], _G946) ?
Exit: (9) lists:append([a], [man], [a, man]) ?
Exit: (8) np([a, man]) ?
Call: (8) lists:append([shoots], [a, man], _G949) ?
Exit: (8) lists:append([shoots], [a, man], [shoots, a, man]) ?
Exit: (7) vp([shoots, a, man]) ?
Call: (7) lists:append([a, woman], [shoots, a, man], [a, woman, shoots]) ?
Fail: (7) lists:append([a, woman], [shoots, a, man], [a, woman, shoots]) ?
Redo: (7) vp(_G932) ?
Call: (8) v(_G932) ?
Exit: (8) v([shoots]) ?
Exit: (7) vp([shoots]) ?
Call: (7) lists:append([a, woman], [shoots], [a, woman, shoots]) ?
Exit: (7) lists:append([a, woman], [shoots], [a, woman, shoots]) ?
Exit: (6) s([a, woman, shoots]) ?
true .

Prolog y los analizadores recursivos descendentes (II)

Ejemplo (CFG):

$S \rightarrow NP\ VP$

$NP \rightarrow DET\ N$

$VP \rightarrow V\ NP$
$VP \rightarrow V$

$DET \rightarrow the$
$DET \rightarrow a$
$N \rightarrow woman$
$N \rightarrow man$
$V \rightarrow shoots$

Otra solución: difference lists
Definite Clause Grammars (DCG):

```
s --> np,vp.

np --> det,n.

vp --> v,np.
vp --> v.

det --> [the].
det --> [a].
n --> [woman].
n --> [man].
v --> [shoots].
```

```
?- s([the,woman,shoots,a,man],[]).
?- s(X,[]).
?- np([a,woman],[]).
```

Reglas recursivas (I)

CFG:	DCG:
CONJ $\rightarrow$ and	conj $--> [and]$
CONJ $\rightarrow$ or	conj $--> [or]$
CONJ $\rightarrow$ but	conj $--> [but]$
S $\rightarrow$ S CONJ S	s $-->$ s,conj,s
S $\rightarrow$ NP VP	s $-->$ np,vp.
NP $\rightarrow$ DET N	np $-->$ det,n.
VP $\rightarrow$ V NP	vp $-->$ v,np.
VP $\rightarrow$ V	vp $-->$ v.
DET $\rightarrow$ the	det $--> [the]$.
DET $\rightarrow$ a	det $--> [a]$.
N $\rightarrow$ woman	n $--> [woman]$.
N $\rightarrow$ man	n $--> [man]$.
V $\rightarrow$ shoots	v $--> [shoots]$.

Reglas recursivas (II)

DCG:

conj --> [and].
conj --> [or].
conj --> [but].

s --> s,conj,s.
s --> np,vp. s --> np,vp.
s --> s,conj,s.

np --> det,n.

vp --> v,np.
vp --> v.

det --> [the].
det --> [a].

n --> [woman].
n --> [man].

v --> [shoots].

?- s([a,woman,shoots],[]).
ERROR: Out of local stack
true.
Idea: cambiar orden
?- s([a,woman],[]).
ERROR: Out of local stack

Reglas recursivas (III)

DCG:

conj --> [and].
conj --> [or].
conj --> [but].

s --> ss.
s --> ss,conj,s.
ss --> np,vp.

np --> det,n.

vp --> v,np.
vp --> v.

det --> [the].
det --> [a].

n --> [woman].
n --> [man].

v --> [shoots].

?- s([a,woman,shoots],[]).
true
?-s([a,woman],[]).
false.

CFGs con características (I)

DCG:

s --> np,vp.

np --> det,n.

np --> pro.

vp --> v,np.

vp --> v.

det --> [the].

det --> [a].

n --> [woman].

n --> [man].

v --> [shoots].

pro --> [he].

pro --> [she].

pro --> [him].

pro --> [her].

?- s([he,shoots,her],[]).

true.

?- s([he,shoots,she],[]).

true.

CFGs con características (II)

DCG:

s --> np,vp.

np --> det,n.

np --> pro.
vp --> v,np.
vp --> v.
det --> [the].
det --> [a].
n --> [woman].
n --> [man].
v --> [shoots].
pro --> [he].
pro --> [she].
pro --> [him].
pro --> [her].

Nuevo DCG:

s --> npsubject,vp.
npsubject --> det,n.
npobject --> det,n.
npsubject --> prosubject.
npobject --> proobject.
vp --> v,npobject.
vp --> v.

det --> [the].
det --> [a].

n --> [woman].
n --> [man].

v --> [shoots].

prosubject --> [he].
prosubject --> [she].
proobject --> [him].
proobject --> [her].

CFGs con características (III)

El DCG con características:

s --> np(subject),vp.

np(_) --> det,n.

np(X) --> pro(X).

vp --> v,np(object).

vp --> v.

det --> [the].

det --> [a].

n --> [woman].

n --> [man].

v --> [shoots].

pro(subject) --> [he].

pro(subject) --> [she].

pro(object) --> [him].

pro(object) --> [her].

Nuevo DCG:

s --> npsubject,vp.

npsubject --> det,n.

npobject --> det,n.

npsubject --> prosubject.

npobject --> proobject.

vp --> v,npobject.

vp --> v.

det --> [the].

det --> [a].

n --> [woman].

n --> [man].

v --> [shoots].

prosubject --> [he].

prosubject --> [she].

proobject --> [him].

proobject --> [her].