Programmation Logique

Grammaire non contextuelle

Enseignant: NGUYEN Thi Minh Tuyen

Exemple

S → NP VP

NP → DET N

VP → V NP

VP → V

DET → *the*

DET → *a*

N → *man*

N → *woman*

V → *shoots*

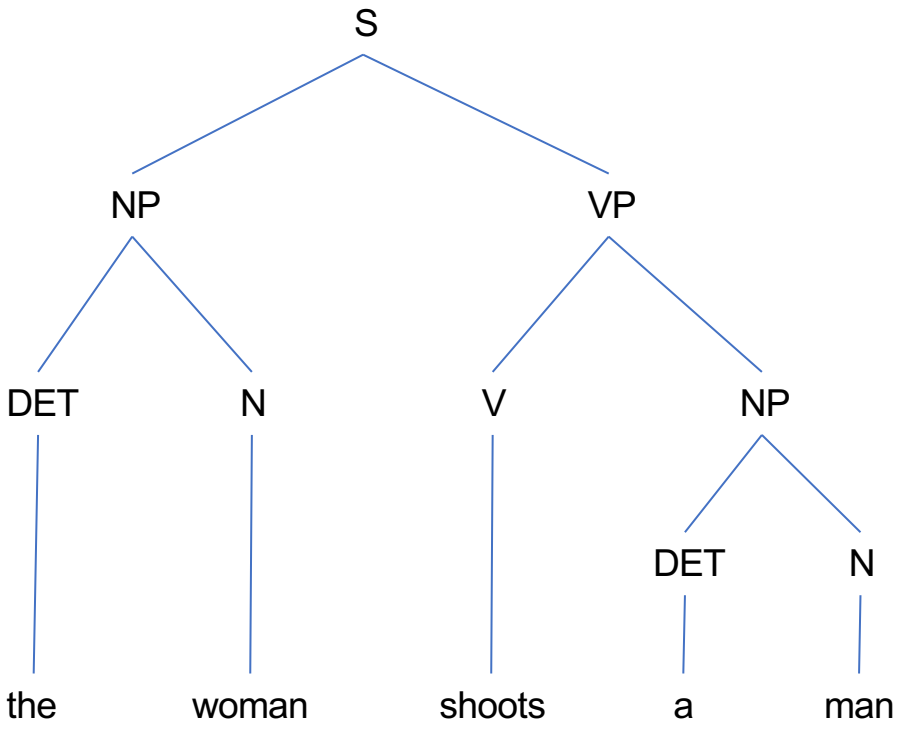
Éléments:

- → : dénote une règle de production
- les symboles en partie droite sont séparés par des virgules
- S, NP, VP, DET, N, V sont appelés des non-terminaux.
- *the, a, man, woman, shoots* sont des terminaux.

- Considère la phrase suivante:

  the woman shoots a man

Est-ce que cette phrase est engendrée par la grammaire définie?



S → NP VP

NP → DET N

VP → V NP

VP → V

DET → *the*

DET → *a*

N → *man*

N → *woman*

V → *shoots*

Grammaire non contextuelle en Prolog

- Nous utilisons des listes pour représenter une séquence des mots
- Par exemple: `[a, woman, shoots, a, man]`
- La règle `S → NP VP` : concaténation de deux listes **np** et **vp** et le résultat est mis dans la liste **s**.

→ le prédicat *append/3* est utilisé.

En utilisant le prédicat append/3 [1]


```
s(C) :- np(A), vp(B), append(A,B,C).
np(C):- det(A), n(B), append(A,B,C).
vp(C):- v(A), np(B), append(A,B,C).
vp(C):- v(C).
det([the]).
det([a]).
n([man]).
n([woman]).
v([shoots]).

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

S → NP VP

NP → DET N

VP → V NP

VP → V

DET → *the*

DET → *a*

N → *man*

N → *woman*

V → *shoots*

En utilisant le prédicat append/3 [2]


```
s(C) :- append(A,B,C),np(A), vp(B).
np(C):- append(A,B,C),det(A), n(B).
vp(C):- append(A,B,C),v(A), np(B).
vp(C):- v(C).
det([the]).
det([a]).
n([man]).
n([woman]).
v([shoots]).
```

Requêtes


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

Problèmes

- Le prédicat `append/3` n'est pas efficace car Prolog essaie chaque combinaison de mots.
- Déplacer le prédicat `append/3` en avance n'est pas intéressant, il y aura beaucoup d'appels à `append/3` avec des variables non-instanciées.

Décomposition par différence

- Listes différentielles (En anglais: difference lists)
- Une liste différentielle est formée de deux listes A et B, où B est une portion finale de A. Par exemple: [1, 2, 3, 4] et [3, 4].
  - On la note: "A - B"
- Par exemple: [1, 2, 3, 4] - [3,4] représente la liste [1, 2]
- Si les deux listes A et B sont complètement instanciées: cette notion n'est pas intéressante.
- Avec des variables logiques et en jouant sur l'unification: nous obtenons des résultats intéressants.
  - Par exemple: le résultat de [1,2|X] - [X] est la liste [1,2]

Exemples

- [a,b,c]-[ ]
  - est la liste [a,b,c]
- [a,b,c,d]-[d]
  - est la liste [a,b,c]
- [a,b,c|T]-T
  - est la liste [a,b,c]
- X-X
  - est la liste vide

Nouvelle version ...


```
s(A-C):- np(A-B), vp(B-C).
np(A-C):- det(A-B), n(B-C).
vp(A-C):- v(A-B), np(B-C).
vp(A-C):- v(A-C).
det([the|W]-W).
det([a|W]-W).
n([man|W]-W).
n([woman|W]-W).
v([shoots|W]-W).
```

S → NP VP

NP → DET N

VP → V NP

VP → V

DET → *the*

DET → *a*

N → *man*

N → *woman*

V → *shoots*

Requêtes


```
?- s([the,man,shoots,a,man]-[]).
true

?- s(X-[]).
X = [the,man,shoots,the,man];
X = [the,man,shoots,a,man];
....
```

Résumé

- Utilisation des listes différentielles sont plus éfficaces.
- Par contre
  - pas facile à comprendre
  - mal à suivre toutes les variables des listes différentielles.

➔ utilise DCG

Notion DCG

- DCG
  - Direct Clause Grammar
  - Une notion simple pour écrire la grammaire en cachant les listes différentielles.
- Une notation DCG :

s(L1, L2) :- sn(L1, L3), sv(L3, L2).

peut s'écrire sous la forme :

s --> sn, sv.

La notion DCG de Prolog

- Les symboles de prédicats et de fonctions, les variables et les constantes obéissent à la syntaxe habituelle de Prolog.
- Les symboles adjacents dans une partie droite de règle sont séparés par une virgule.
- La flèche est le symbole "–-->"
- Les terminaux sont écrits entre " [" et "] "
- La chaîne vide ε est représentée par " [ ] ".

Exemple

S → NP VP	`s-->np,vp.`
NP → DET N	`np-->det,n.`
VP → V NP	`vp-->v,np.`
VP → V	`vp-->v.`
DET → *the*	`det-->[the].`
DET → *a*	`det-->[a].`
N → *man*	`n-->[man].`
N → *woman*	`n-->[woman].`
V → *shoots*	`v-->[shoots].`

Exercices

- Définir le DCG qui engendre des langages suivants:
  1. $a^n b^n$.
  2. $a^n b^n$ - {Λ}.
  3. $a^n b^{2n}$.
  4. palindrome pair, $\sum$ = {a,b}.
  5. palindrome impair, $\sum$ = {a,b}.

Exemple


```
s --> sn, sv.
sn --> np.
sv --> vt, sn.
sv --> vi.
np --> [paul].
np --> [ils].
vi --> [dort].
vi --> [dorment].
vt --> [ecoutent].
```

Prendre en compte des éléments contextuels


```
s --> sn(Nombre), sv(Nombre).
sn(Nombre) --> np(Nombre).
sv(Nombre) --> vt(Nombre), sn(_).
sv(Nombre) --> vi(Nombre).
np(sing) --> [paul].
np(plur) --> [ils].
vi(sing) --> [dort].
vi(plur) --> [dorment].
vt(plur) --> [ecoutent].
```

Grammaire DCG où le lexique est donné sous la forme d'une base de faits


```
s --> sn(Nombre), sv(Nombre).
sn(Nombre) --> np(Nombre).
sv(Nombre) --> vt(Nombre), sn(_).
sv(Nombre) --> vi(Nombre).
np(Nombre) --> [Mot], {lexique(Mot, np, Nombre)}.
vi(Nombre) --> [Mot], {lexique(Mot, vi, Nombre)}.
vt(Nombre) --> [Mot], {lexique(Mot, vt, Nombre)}.
lexique(paul, np, sing).
lexique(ils, np, plur).
lexique(dort, vi, sing).
lexique(dorment, vi, plur).
lexique(ecoutent, vt, plur).
```

Exemple: $a^n b^n c^n$


```
s-->ablock(Count),bblock(Count),cblock(Count).
ablock(0)  -->  [].
ablock(NewCount)  -->  [a], ablock(Count),
                            {NewCount  is  Count  +  1}.
bblock(0)  -->  [].
bblock(NewCount)  -->  [b], bblock(Count),
                            {NewCount  is  Count  +  1}.
cblock(0)  -->  [].

cblock(NewCount)  -->  [c], cblock(Count),
                            {NewCount  is  Count  +  1}.
```

DCG n'est pas magique!

- DCG est une belle notion mais nous ne pouvons pas écrire n'importe quelle grammaire non contextuelle et éxécuter sans problème.
- DCGs sont des règles ordinaires déguisées.
- Faire attention à la récursion gauche

Exercice

- Nous avons un ensemble
  - des verbes: sat, saw, hears, took, sees, will_see.
  - des adjs: large, small, brown, orange, green, blue.
  - des noms: cat, mat, man, boy, dog.
  - des déterminants: the, a, an.
- Donner une grammaire non contextuelle en utilisant DCG (avec "extra goal") pour créer des phases englaises valides.

Question?