Logic Grammars - Syntax

- Rule of a grammar:

```
left -> right1, right2, ..., rightN.
```

- Any symbol of the grammar could have variables that can be used for different purposes
- The input is consumed using the lists syntax:

```
aaaa -> [W],bbbb.
```

- PROLOG code can be embedded enclosing it between brackets:

```
aaaa(W) -> [W],bbbb(W), {number(W)}.
```

- To recognize sentences the main symbol of the grammar is used calling it as a PROLOG predicate, for example:

```
analisis(F,[el,gato,come,pescado],[]).
```

- Only the last two parameters are mandatory, usually a first parameter that returns the result of the analysis is added.

Logic Grammars - Grammar 1

- Parsing of simple sentences (SN+V+CMPL)
- Using top-down decomposition a grammar is obtained
- After determining the structure of each syntactical element, the rules to parse each one are written (rules and terminal symbols).
- Finally a lexicon that covers the sentences we want to recognize are added
- This grammar only recognizes, no representation of the sentence is generated
- Variation: relative clauses with relative pronouns and number and gender agreement

Grammar 1

```
analisis(X,Y):- asercion(X,Y).

asercion --> sn, verb,compl.
compl --> prep,sn.
compl--> sn.
compl --> [].
sn--> npr.
sn--> det,n.

verb--> [W],{verbo(W)}.
npr--> [W],{npropio(W)}.
n--> [W],{nombre(W)}.
det--> [W],{determ(W)}.
prep--> [W],{prepo(W)}.
```

```
npropio(clara).
npropio(maria).
npropio(juan).
npropio(barcelona).
nombre(hombre).
nombre(profesor).
nombre(libro).
determ(un).
determ(el).
verbo(esta).
verbo(rie).
verbo(piensa).
verbo(habla).
verbo(lee).
prepo(en).
prepo(con).
prepo(de).
```

Grammar 1 - relative clauses

```
asercion --> sn, rel, verb,compl.

rel--> prel,verb,compl.

compl --> [].
compl --> prep,sn.
compl--> sn.
sn--> npr.
sn--> det,n.

verb--> [W],{verbo(W)}.
npr--> [W],{npropio(W)}.
n--> [W],{nombre(W)}.
det--> [W],{determ(W)}.
prep--> [W],{prepo(W)}.
prel--> [W],{pronomrel(W)}.
```

```
npropio(clara).
npropio(maria).
npropio(juan).
npropio(barcelona).
nombre(hombre).
nombre(libro).
determ(un).
determ(el).
prepo(en).
prepo(con).
prepo(de).

pronomrel(que).

verbo(esta).
verbo(rie).
verbo(piensa).
verbo(habla).
```

Logic Grammars - Grammar 2

- The grammar is constrained to only accept sentences using the correct complements
- The information in the lexicon is augmented including for each verb the complements that can accept
- The grammar is modified in order to the rules check the correctness of the verb complements

Grammar 2

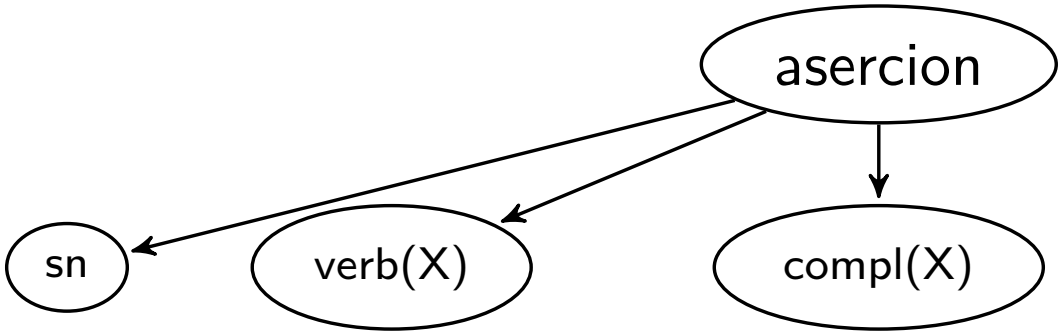
```
analisis(X,Y):- asercion(X,Y).

asercion --> sn, verb(X),compl(X).

compl([])--> [].
compl([])--> sn.
compl([arg(X)|Y])--> prep(X),sn,compl(Y).

sn-->npr.
sn-->det,n.

verb(A)--> [W],{verbo(W,A)}.
npr--> [W],{npropio(W)}.
n--> [W],{nombre(W)}.
det--> [W],{determ(W)}.
prep(W)--> [W],{prepo(W)}.
```

```
npropio(clara).
npropio(maria).
npropio(juan).
npropio(barcelona).

nombre(hombre).
nombre(profesor).
nombre(libro).

determ(un).
determ(el).

prepo(en).
prepo(con).
prepo(de).

verbo(piensa,[arg(en)]).
verbo(esta,[arg(en)]).
verbo(rie,[]).
verbo(habla,[arg(de),arg(con)]).
verbo(lee,[]).
```

Grammar 2 - Example

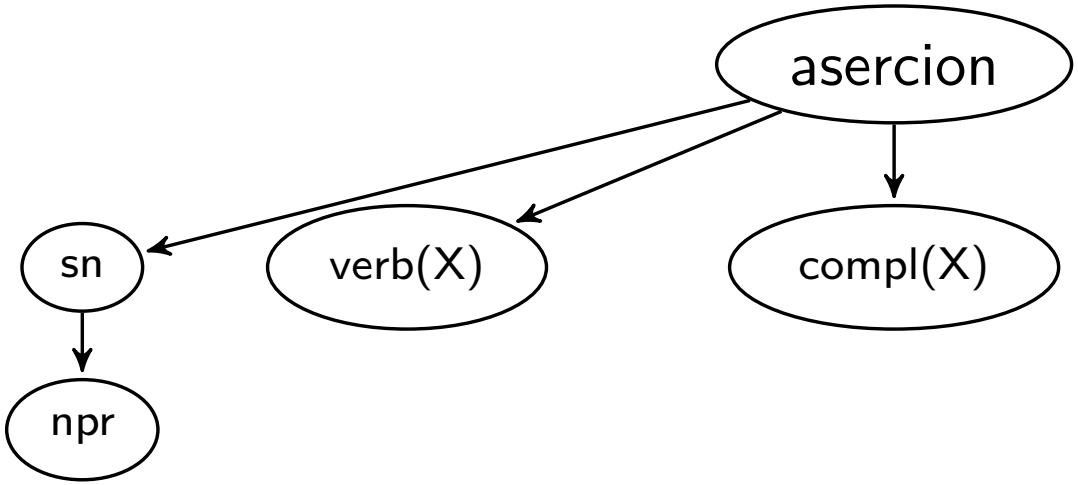
clara habla de un profesor con juan []

Grammar 2 - Example



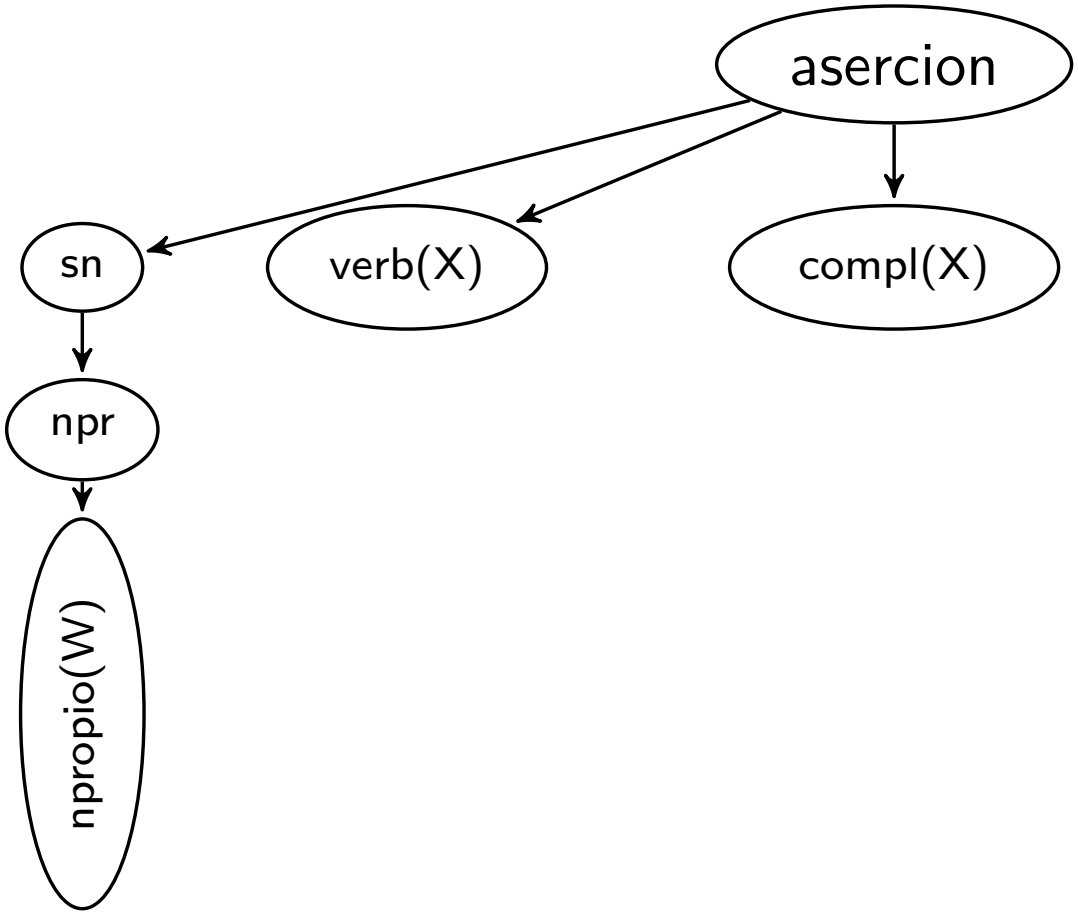
clara habla de un profesor con juan []

Grammar 2 - Example



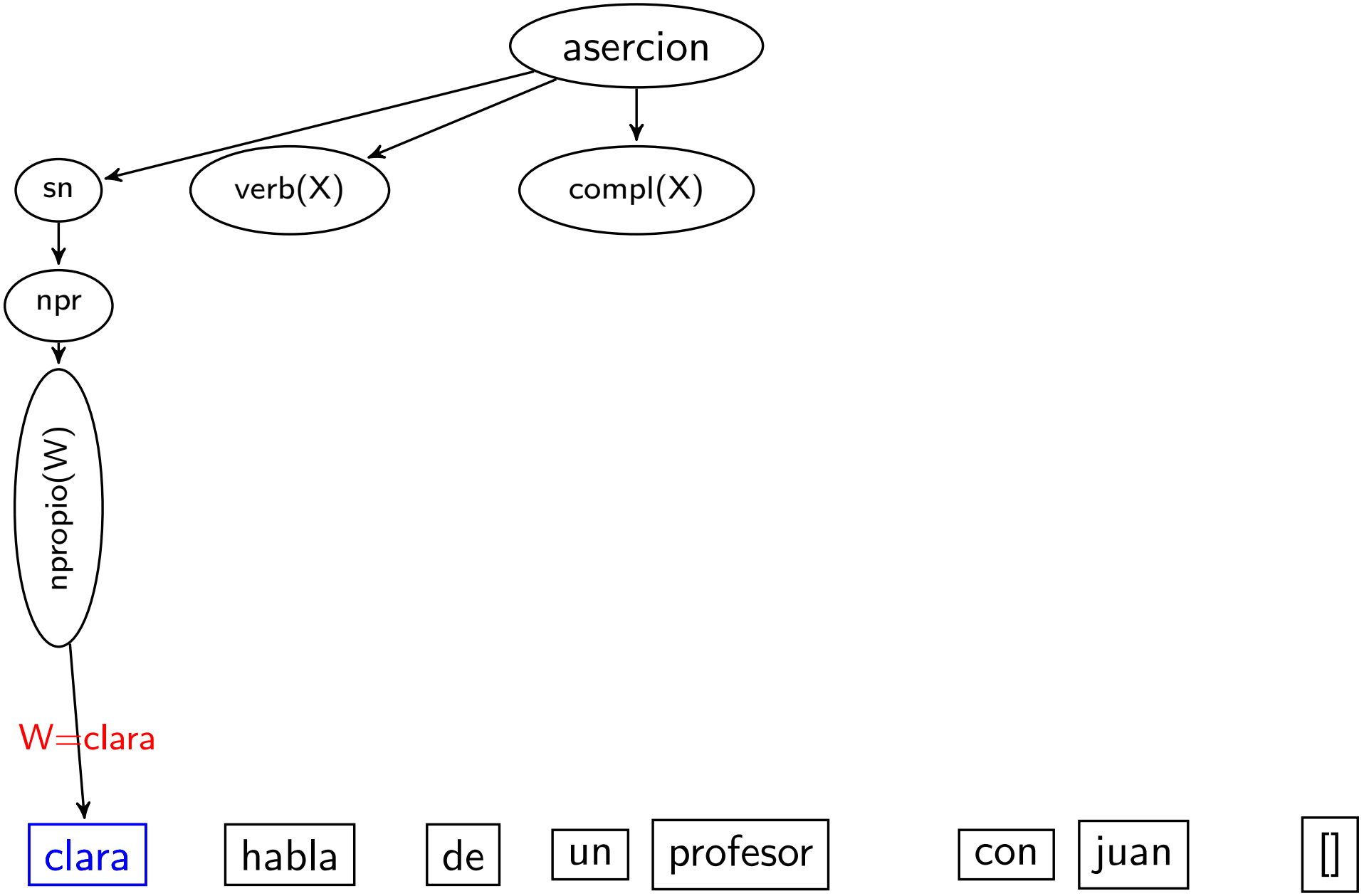
clara habla de un profesor con juan []

Grammar 2 - Example

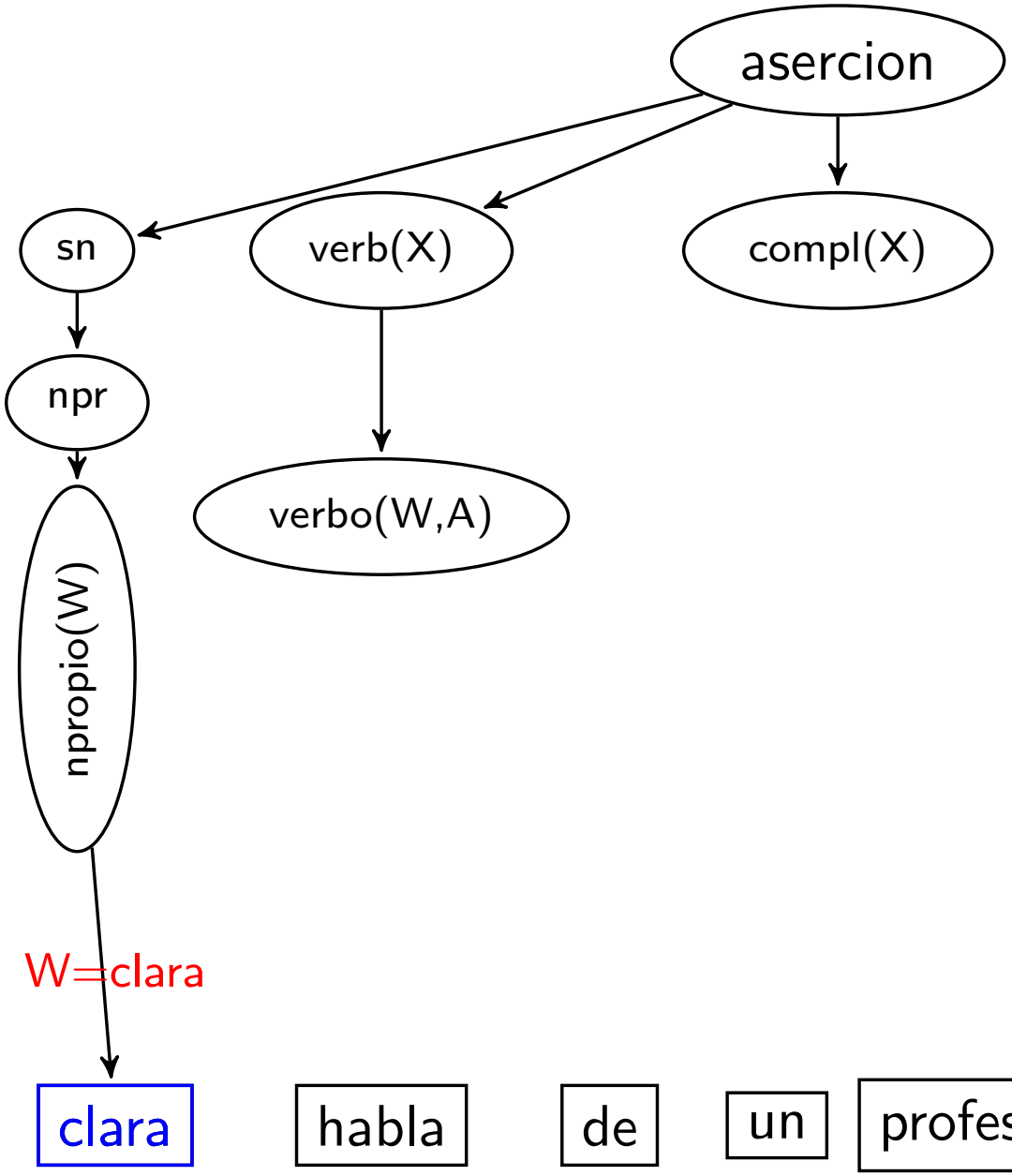


clara habla de un profesor con juan []

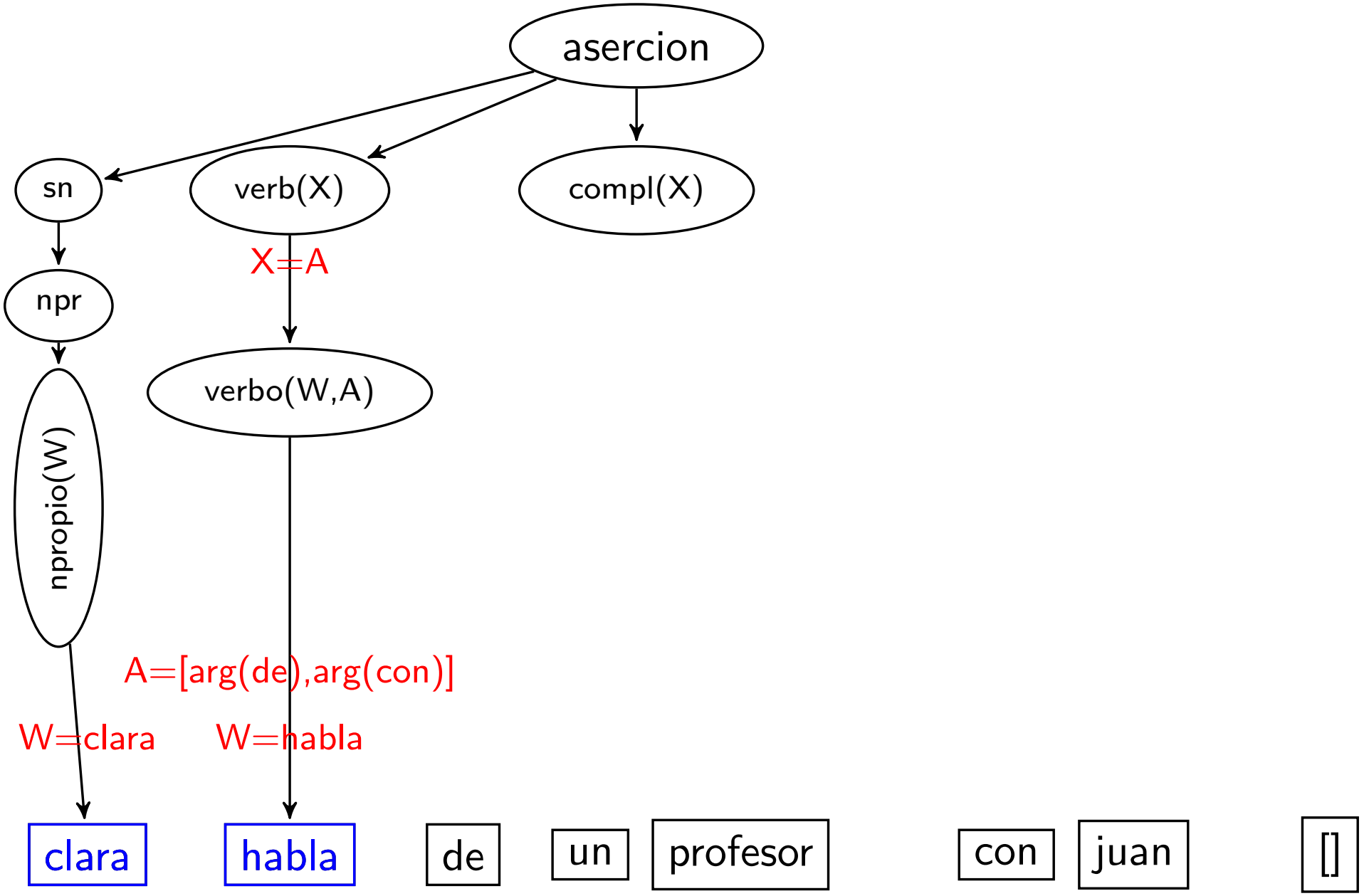
Grammar 2 - Example



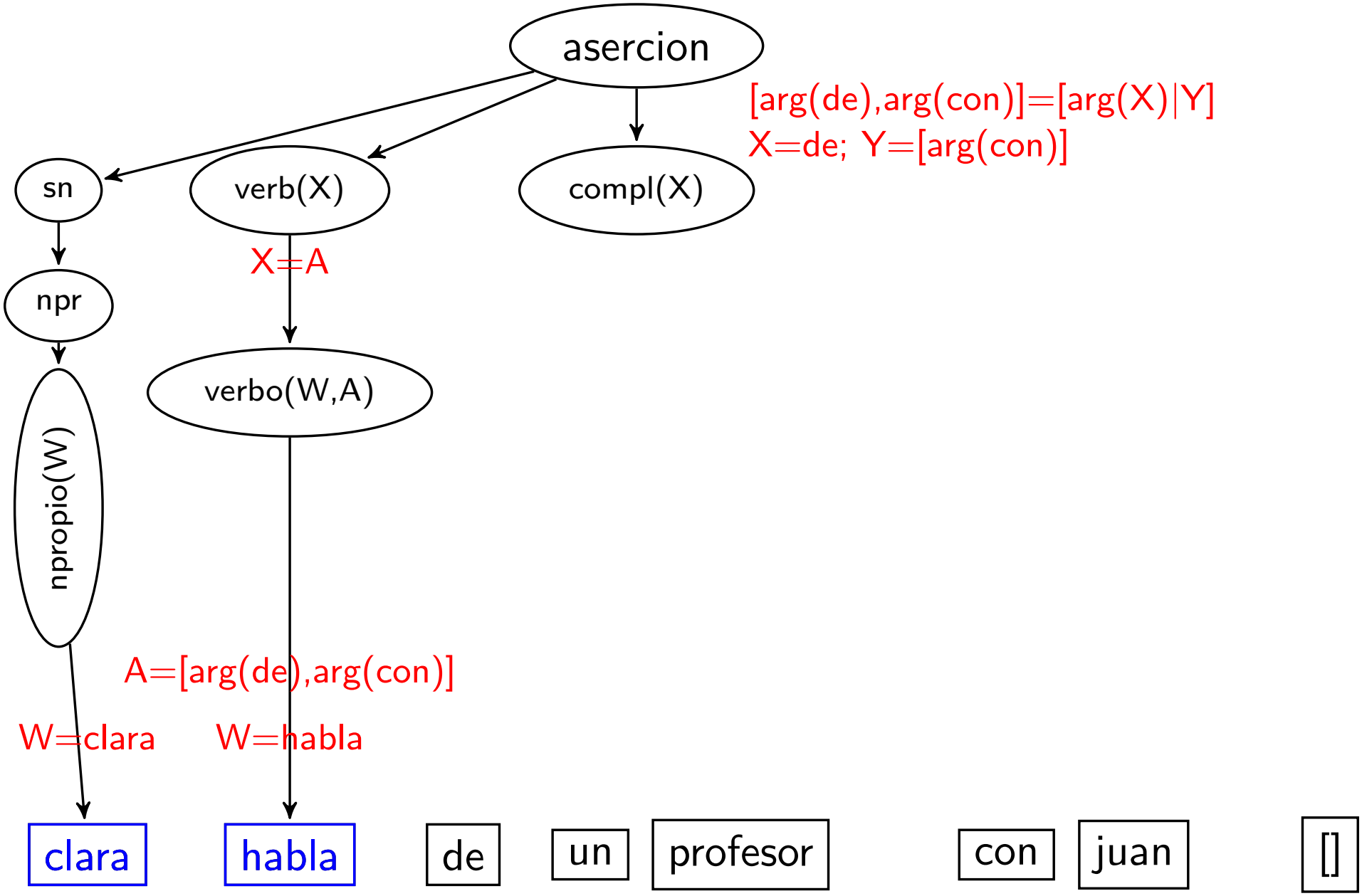
Grammar 2 - Example



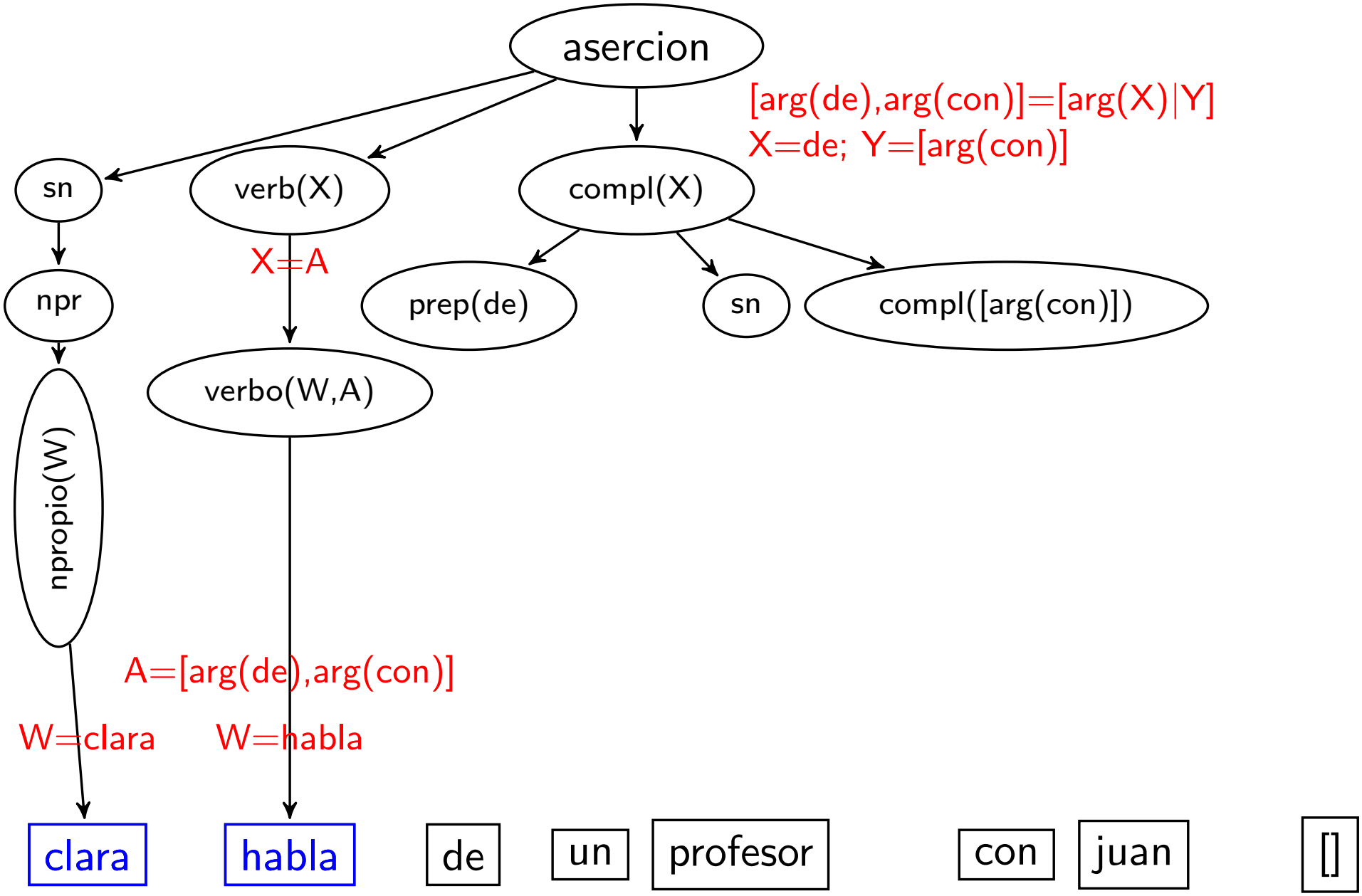
Grammar 2 - Example



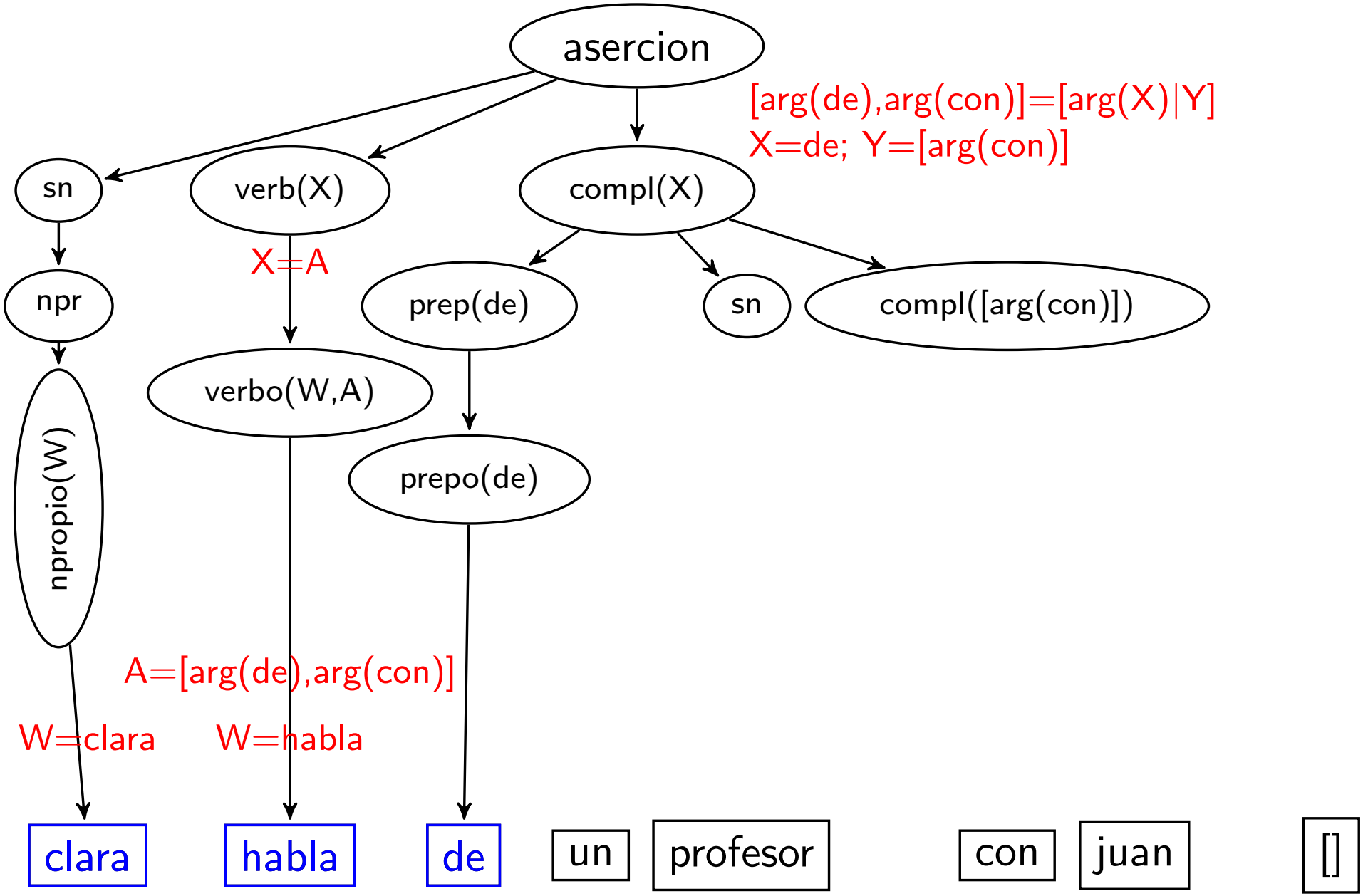
Grammar 2 - Example



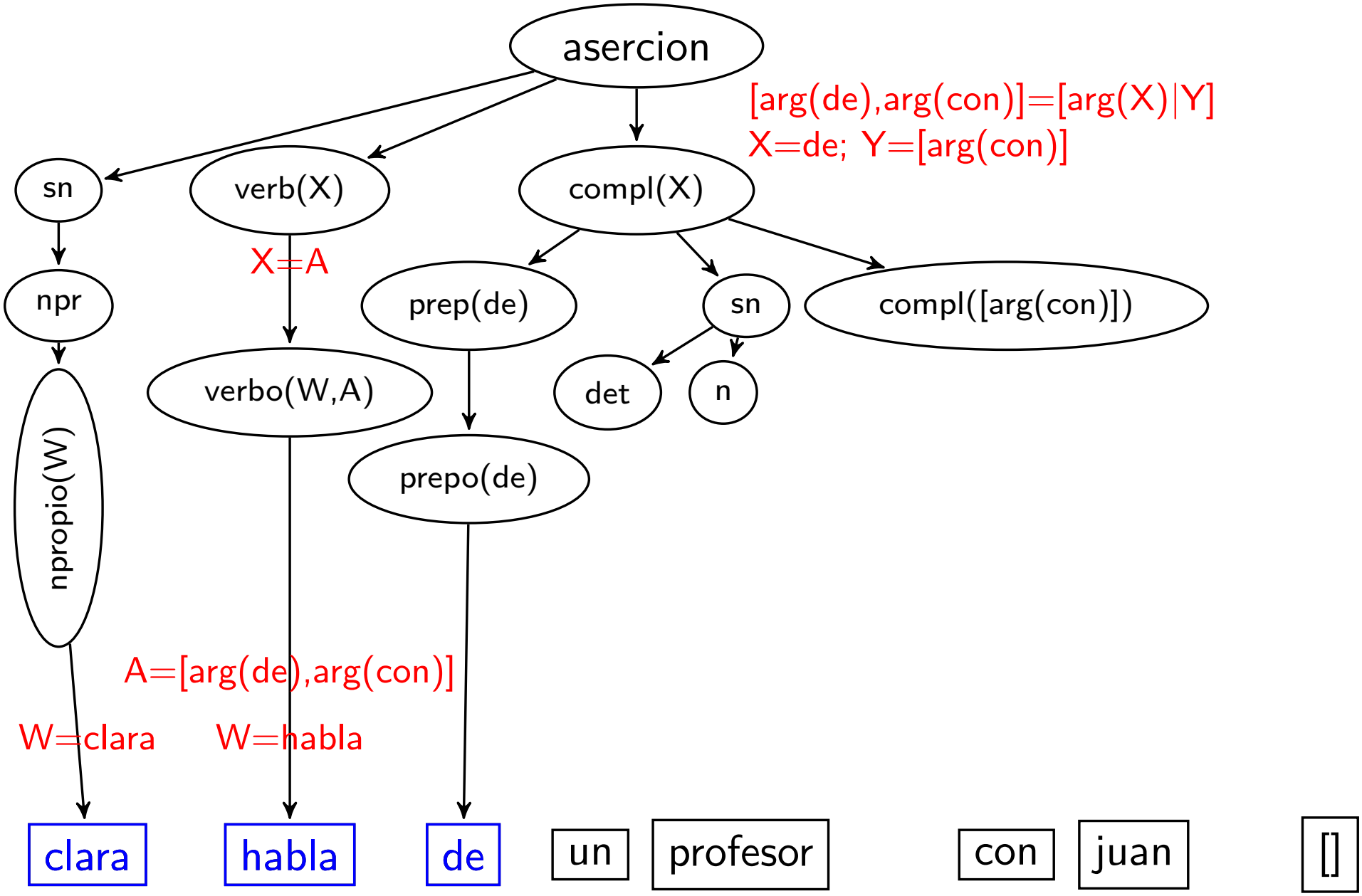
Grammar 2 - Example



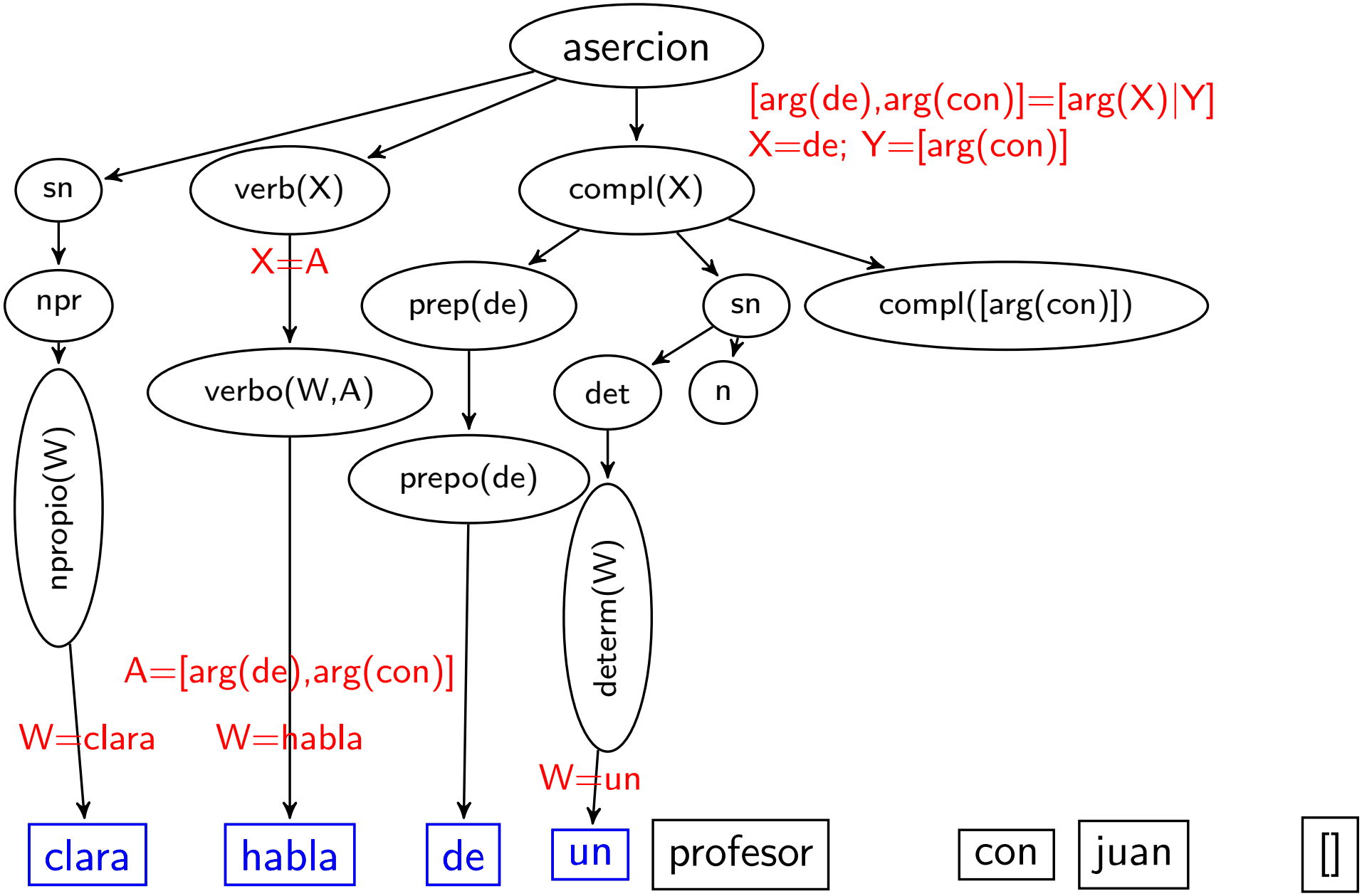
Grammar 2 - Example



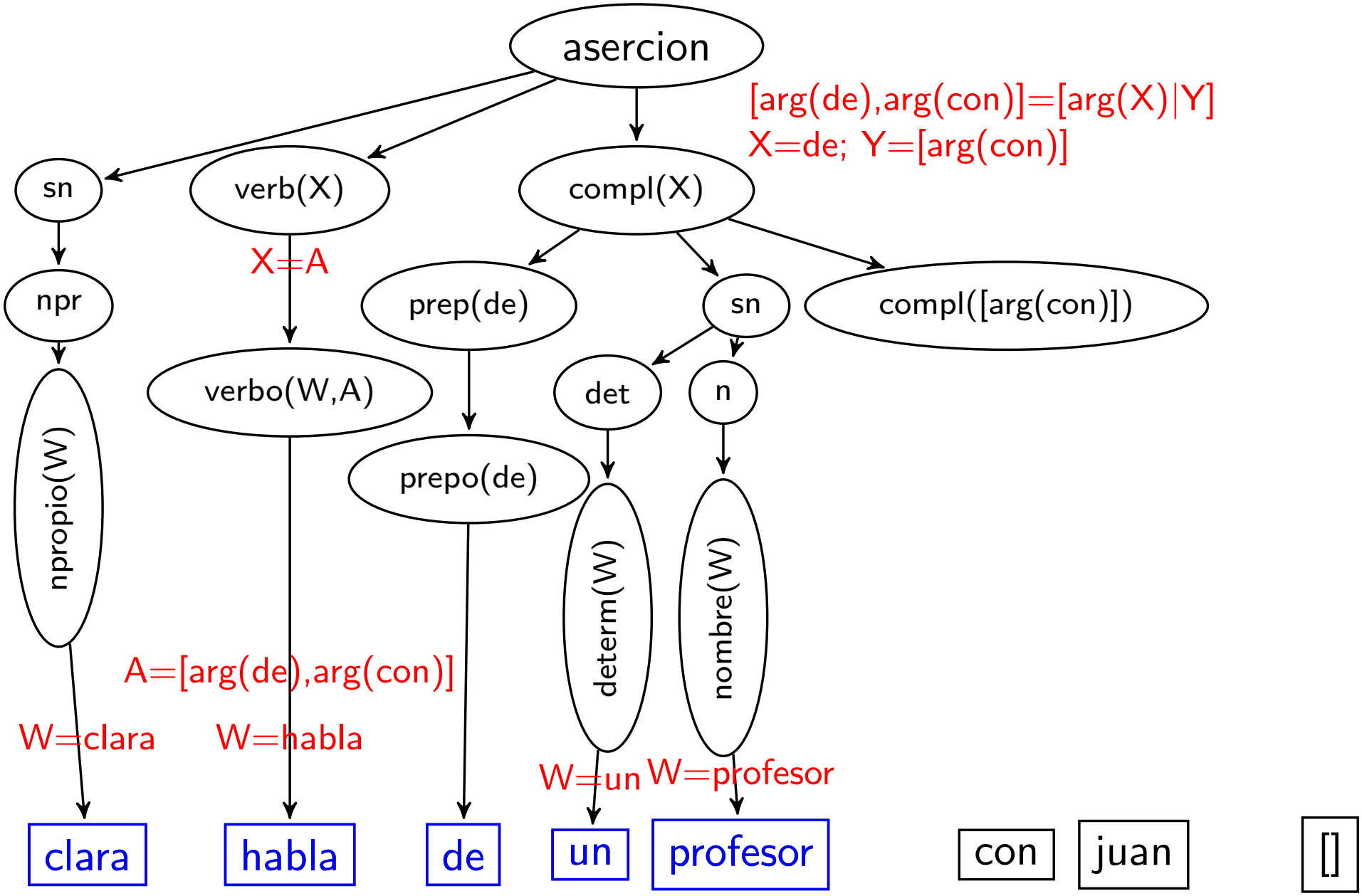
Grammar 2 - Example



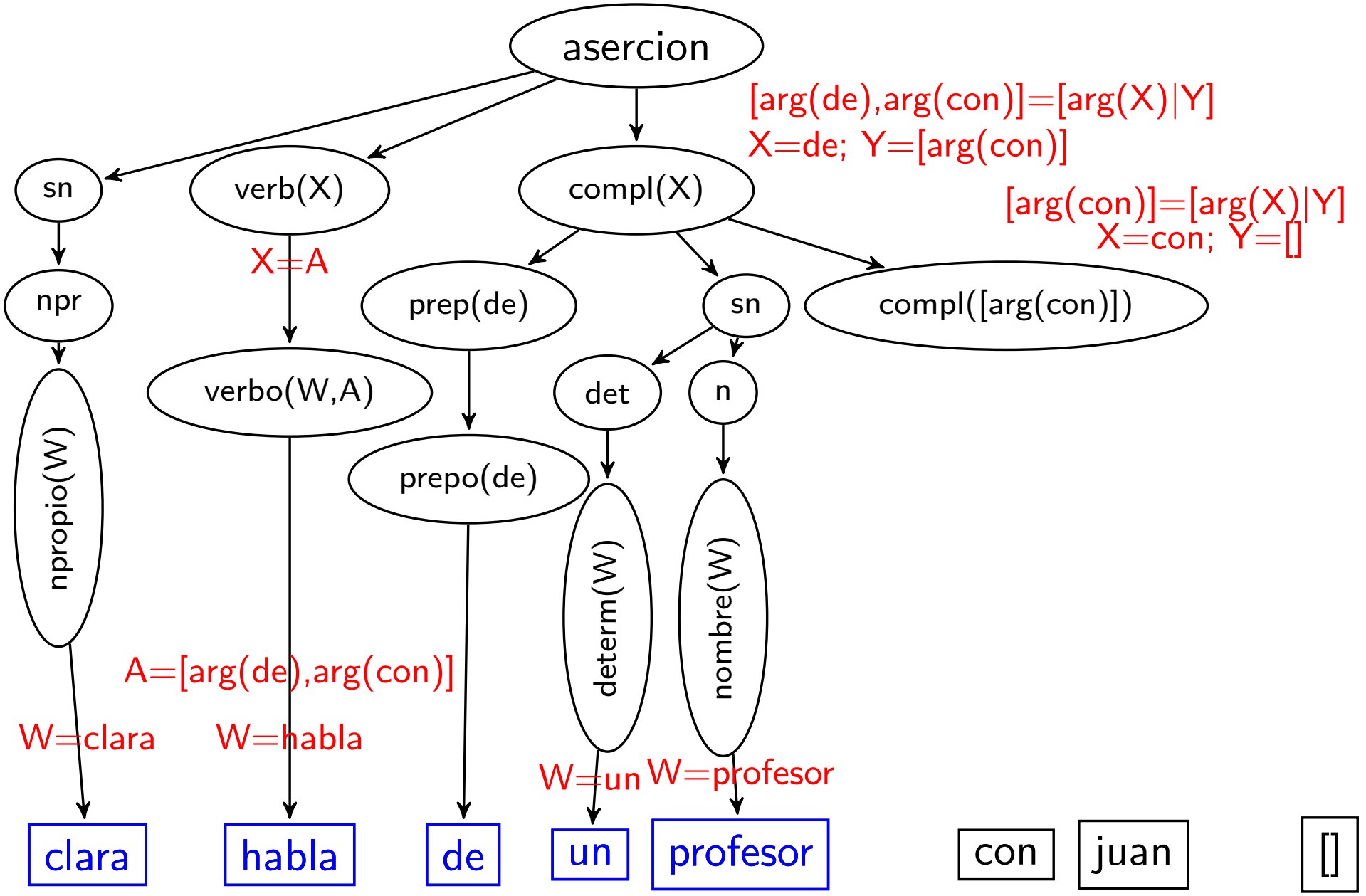
Grammar 2 - Example



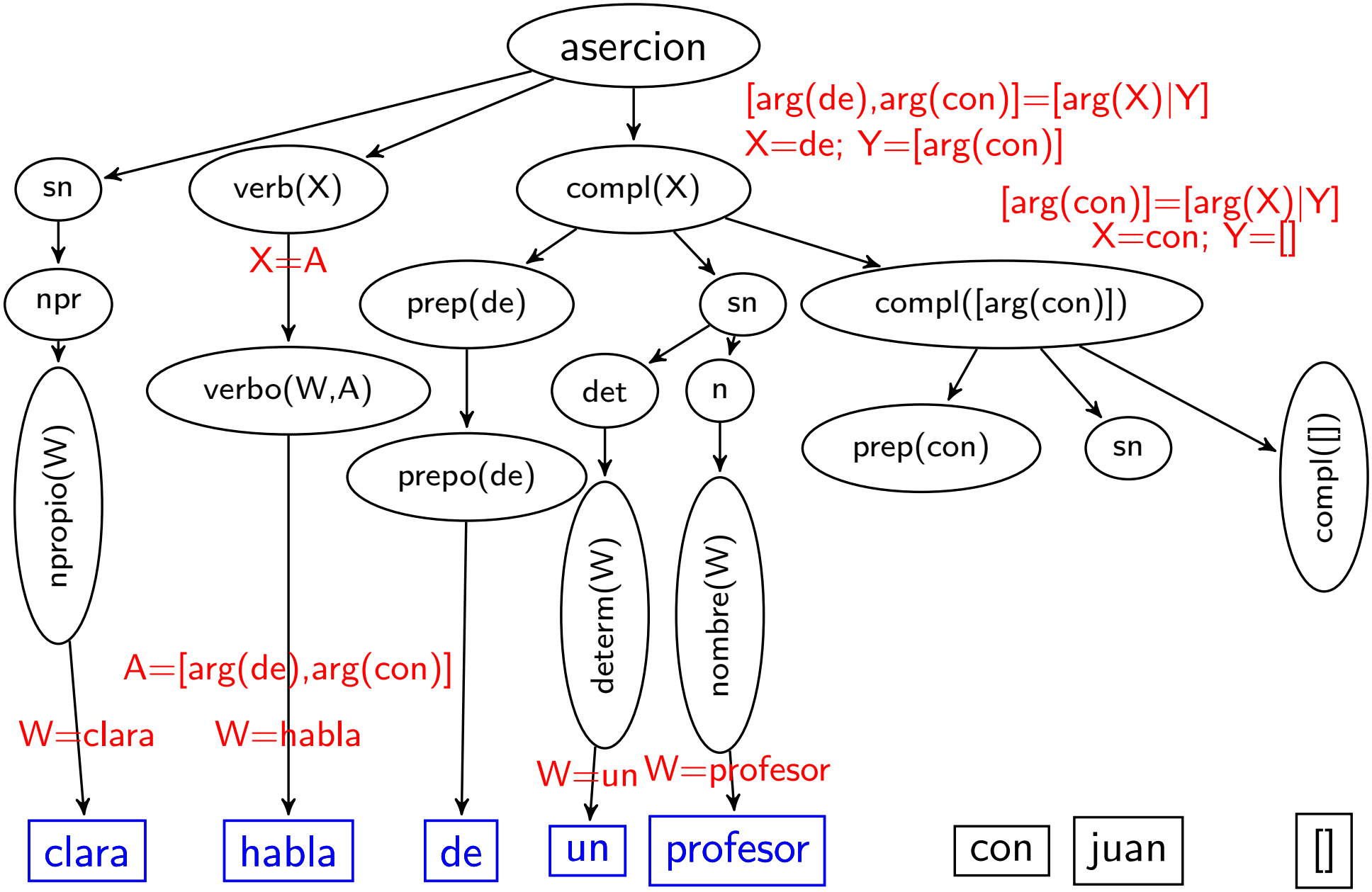
Grammar 2 - Example



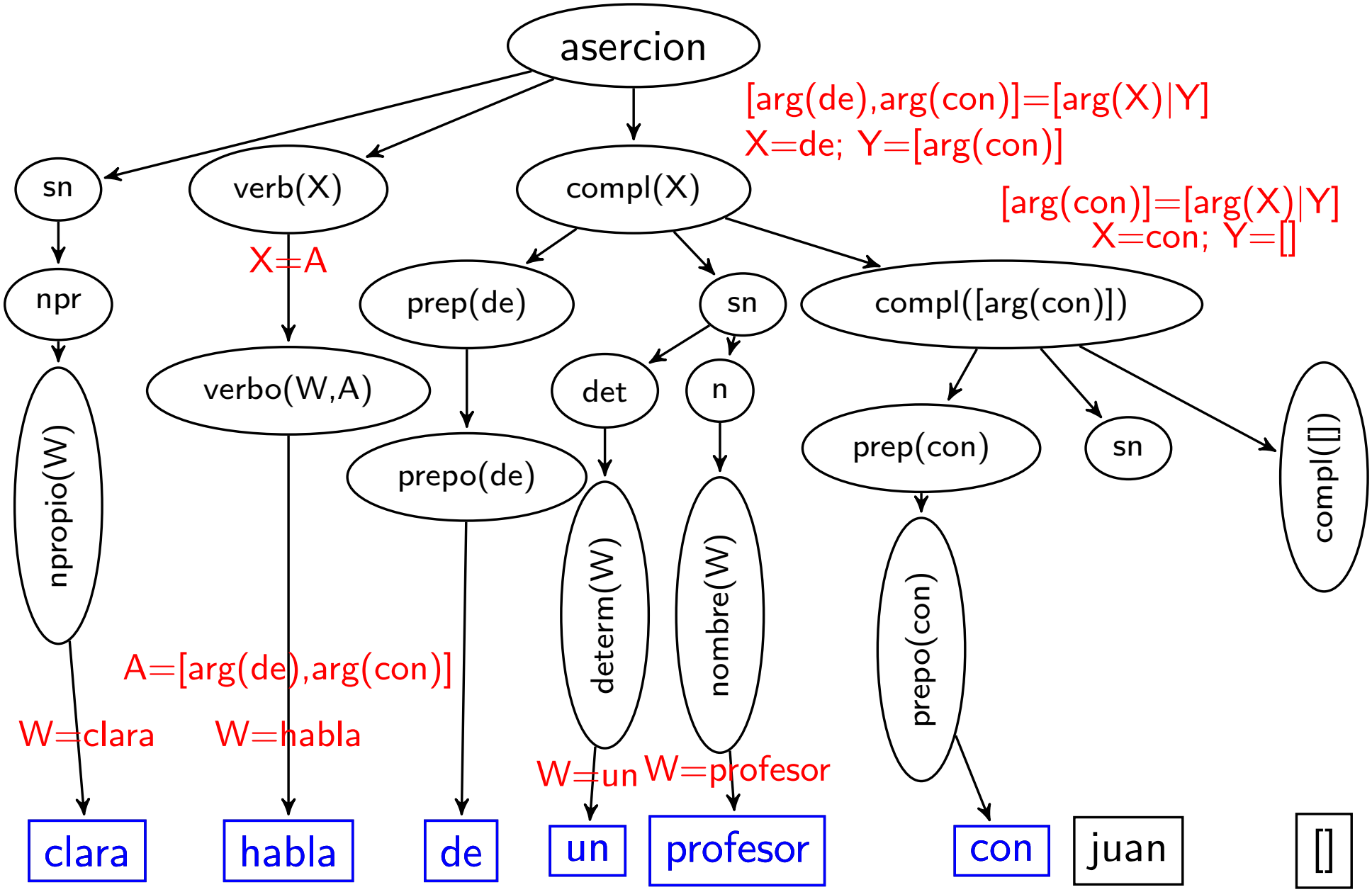
Grammar 2 - Example



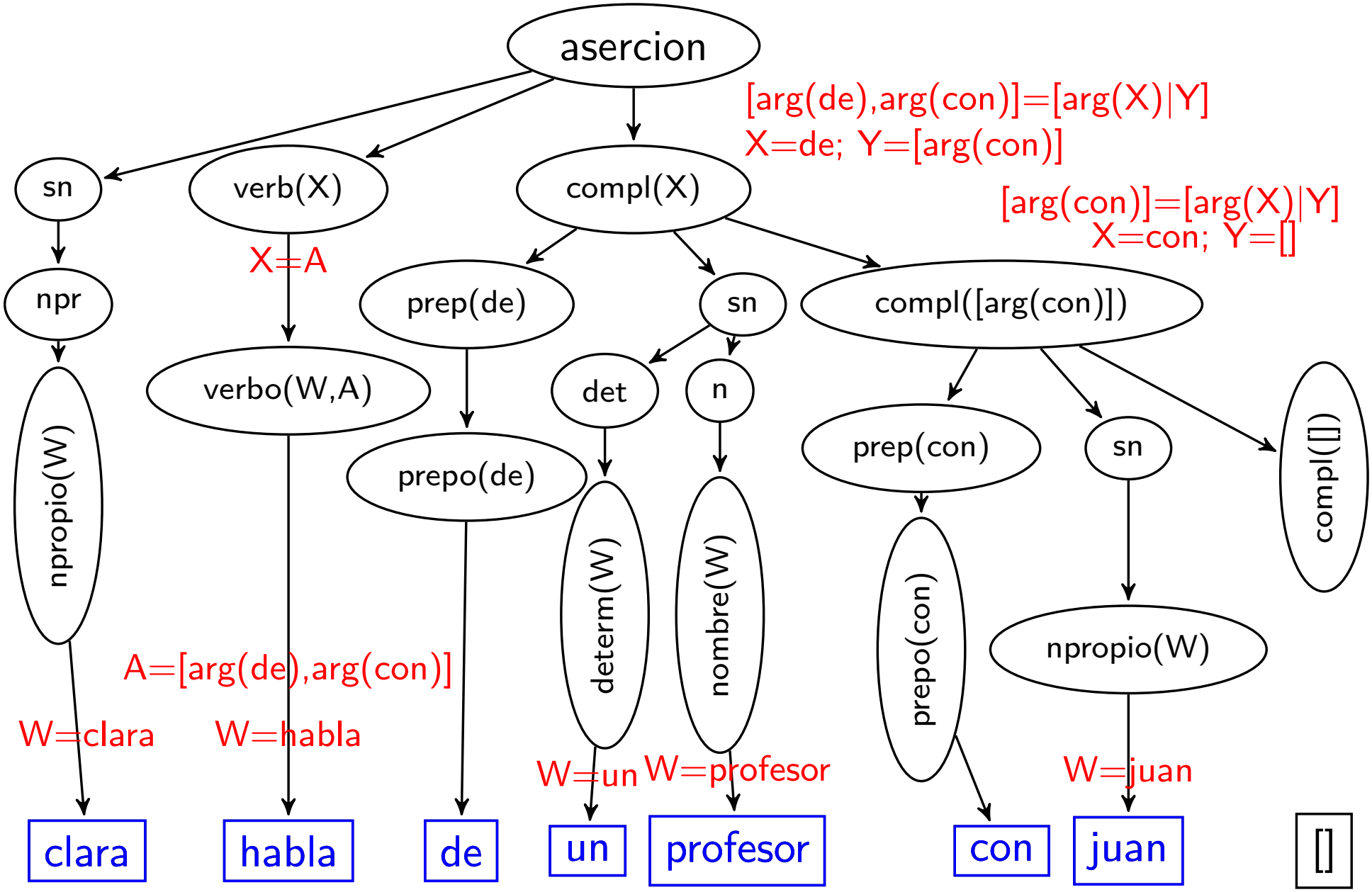
Grammar 2 - Example



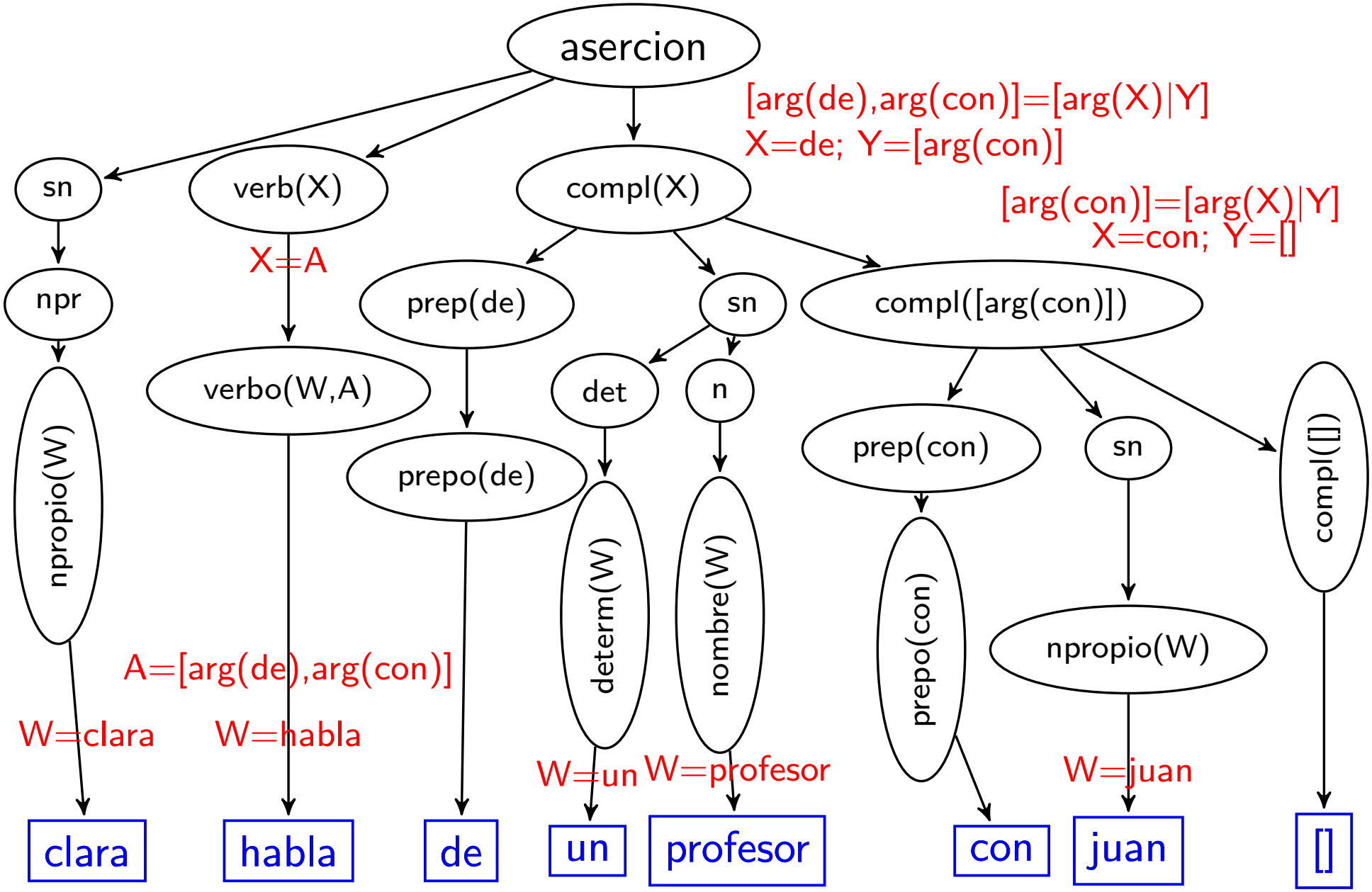
Grammar 2 - Example



Grammar 2 - Example



Grammar 2 - Example



Logic Grammars - Grammar 3

- Now we will modify the grammar to output a representation of the sentences that recognizes (semantics)
- The representation language will be predicates logic, we identify the elements of the sentences with the language of PC_1
  - Verbs $\longrightarrow$ predicate with the appropriate arity
  - Names $\longrightarrow$ constants
  - Nouns $\longrightarrow$ unary predicates
- The lexicon is updated including the representation of each element
- Variables are added to the adequate symbols in the grammar rules in order to obtain the representation from the lexicon and to build the representation of the sentence

Grammar 3 - I

```
analisis(F,X,Y):- asercion(F,X,Y).

asercion(F) --> sn(S), verb(S,X,F),compl(X).

compl([])--> [].
compl([arg(X,O)|Y])--> prep(X),sn(O),compl(Y).
compl([arg(nulo,O)|Y])--> sn(O),compl(Y).

sn(S)--> npr(S).
sn(S)--> det,n(S).

verb(S,A,F)--> [W],{verbo(W,S,A,F)}.
npr(W)--> [W],{npropio(W)}.
n(F)--> [W],{nombre(W,_,F)}.
det--> [W],{determ(W)}.
prep(W)--> [W],{prepo(W)}.
```

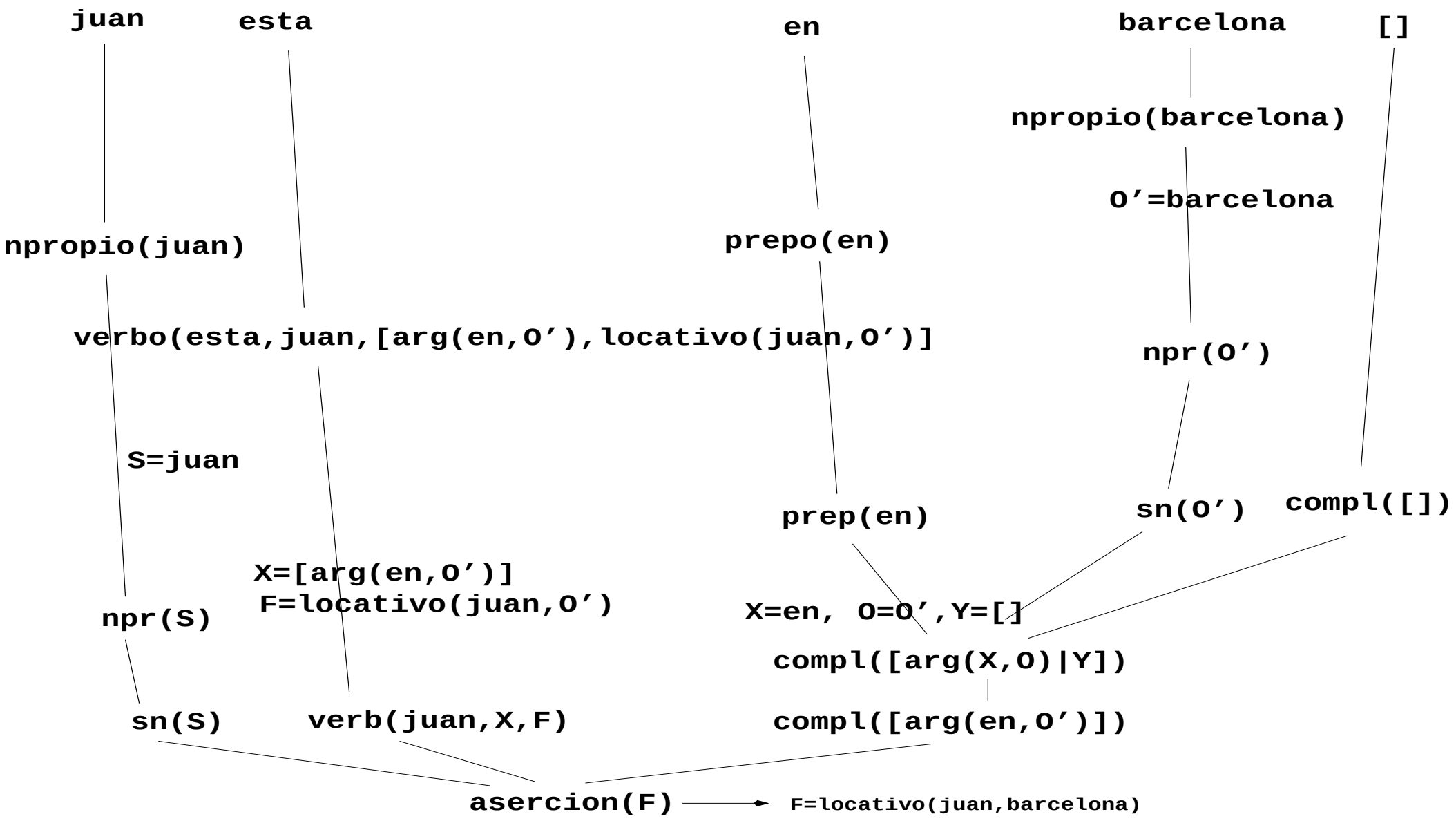
Grammar 3 - II

```
npropio(clara).
npropio(maria).
npropio(juan).
npropio(barcelona).

nombre(libro,K,libro(K)).
nombre(hombre,K,hombre(K)).
nombre(profesor,K,profesor(K)).

determ(un).
determ(el).

prepo(en).
prepo(con).
prepo(de).

verbo(rie,S,[],reir(S)).
verbo(piensa,S,[arg(en,O)],pensar_en(S,O)).
verbo(habla,S,[arg(de,O),arg(con,O1)],comunica(S,O,O1)).
verbo(habla,S,[arg(con,O),arg(de,O1)],comunica(S,O1,O)).
verbo(esta,S,[arg(en,O)],locativo(S,O)).
verbo(lee,S,[arg(nulo,O)],leer(S,O)).
```

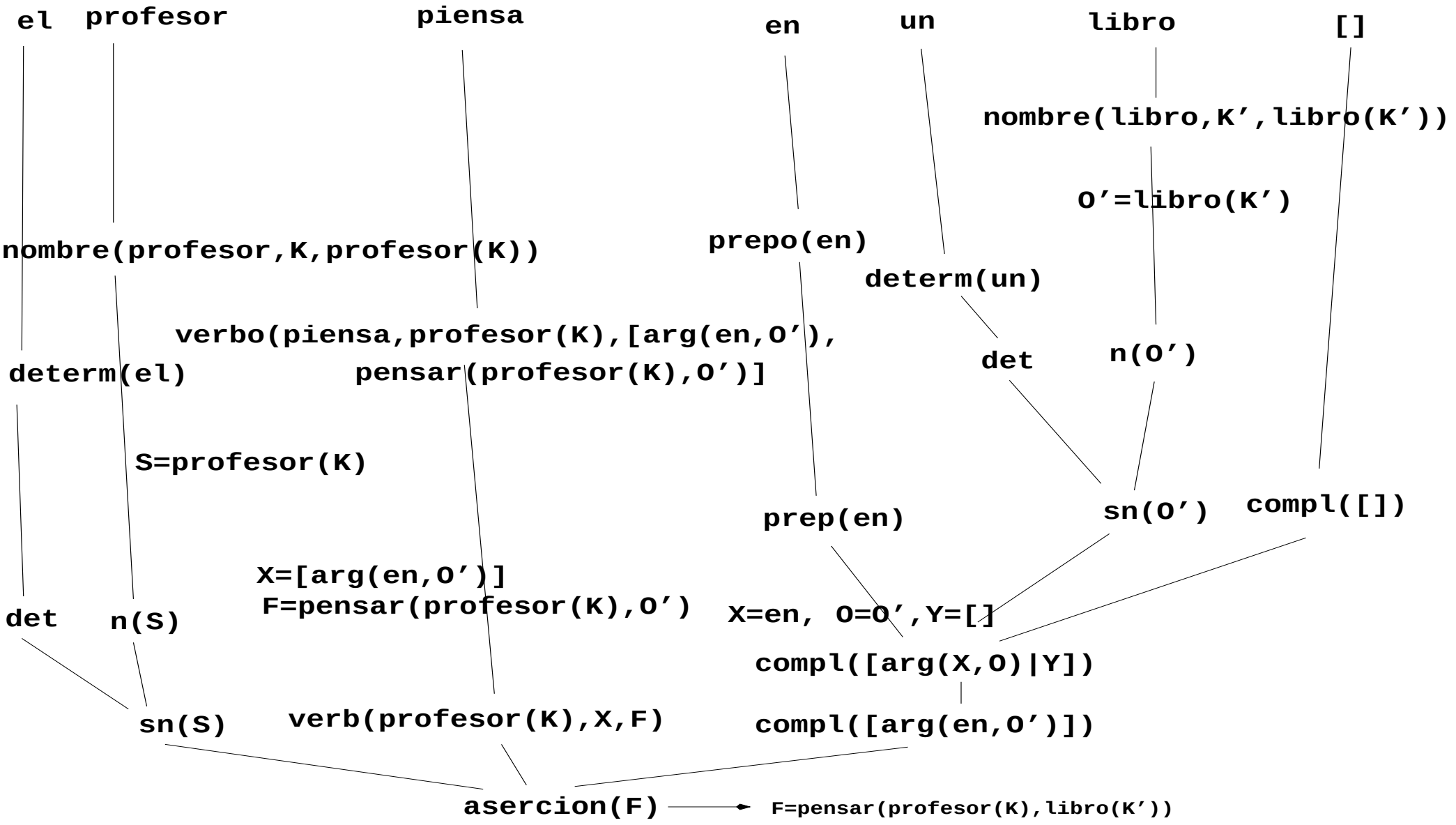
Grammar 3 - Sentences

```
| ?-  analisis(F,[juan,esta,en,barcelona],[]).

F = locativo(juan,barcelona) ?
yes
| ?- analisis(F,[juan,piensa,en,maria],[]).

F = pensar_en(juan,maria) ?
yes
| ?- analisis(F,[el,libro,esta,en,barcelona],[]).

F = locativo(libro(_A),barcelona) ?
yes
| ?- analisis(F,[juan,lee,un,libro],[]).

F = leer(juan,libro(_A)) ?
yes
| ?- analisis(F,[el,hombre,habla,de,juan,con,maria],[]).

F = comunica(hombre(_A),juan,maria) ?
yes
| ?- analisis(F,[el,hombre,rie],[]).

F = reir(hombre(_A)) ?
yes
| ?- analisis(F,[el,profesor,piensa,en,un,libro],[]).

F = pensar_en(profesor(_B),libro(_A)) ?
yes
```

Grammar 3 - Example 1



Grammar 3 - Example 2



Logic Grammars - Grammar 4

- Quantification is included to obtain a better representation of the sentences
- The representation of the determinant is changed in order to obtain the type of quantification
- Appropriate variables are added to the grammar rules in order to obtain the proper unification of variables during the execution of the grammar

Grammar 4 - I

```
analisis(F,X,Y):- asercion(F,X,Y).

asercion(F) --> sn(K,S2,F), verb(K,X,S1), compl(X,S1,S2).

compl([],S,S)--> [].
compl([arg(X,K)|Y],S1,S)--> prep(X), sn(K,S2,S), compl(Y,S1,S2).
compl([arg(nulo,K)|Y],S1,S)--> sn(K,S2,S), compl(Y,S1,S2).

sn(K,F,F)--> npr(K).
sn(K,S2,F)--> det(K,S1,S2,F),n(K,S1).

verb(S,A,F)--> [W],{verbo(W,S,A,F)}.
npr(W)--> [W],{npropio(W)}.
n(K,F)--> [W],{nombre(W,K,F)}.
det(K,S1,S2,F)--> [W],{determ(W,K,S1,S2,F)}.
prep(W)--> [W],{prepo(W)}.
```

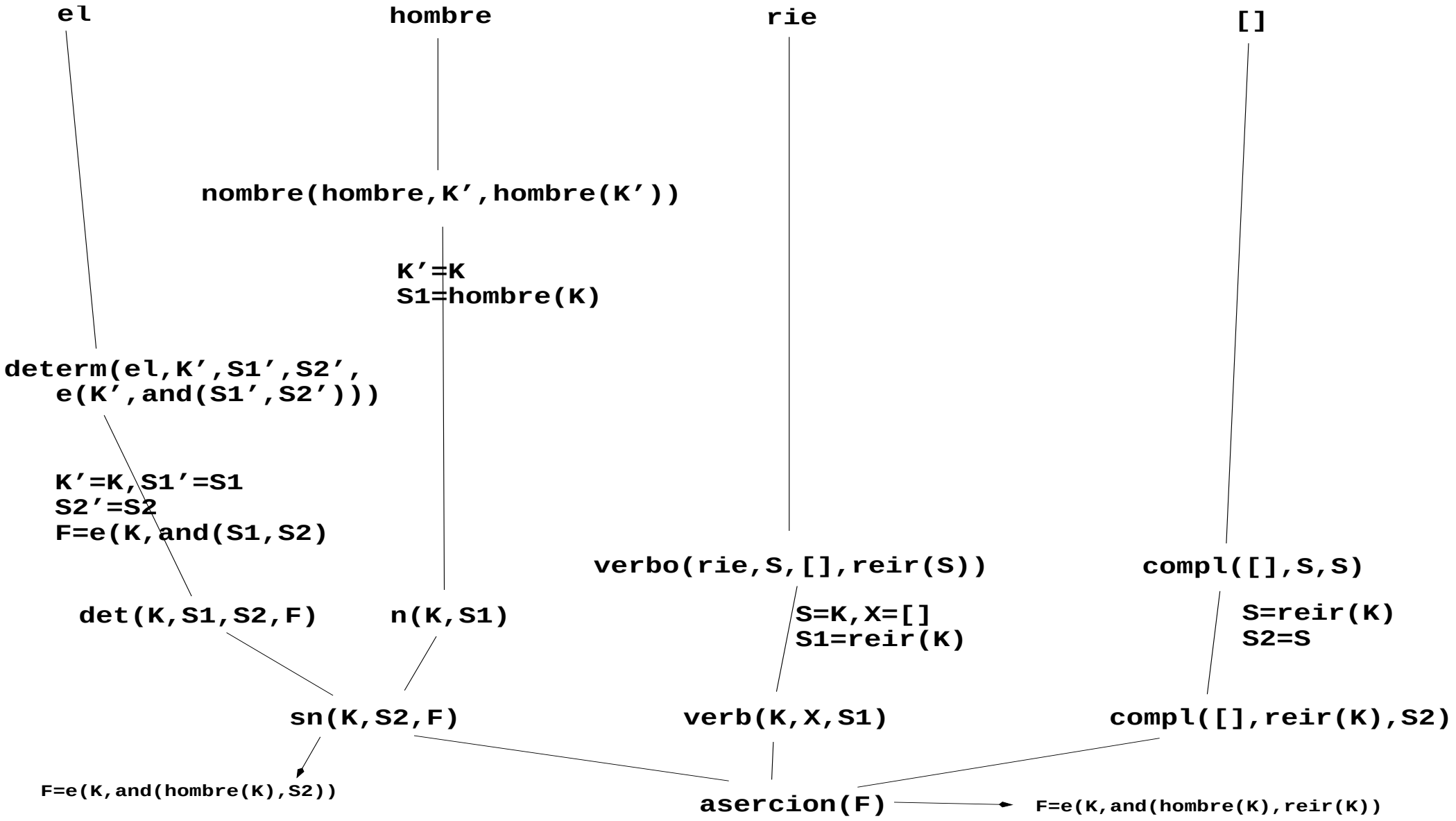
Grammar 4 - II

```
npropio(clara).
npropio(maria).
npropio(juan).
npropio(barcelona).

nombre(libro,K,libro(K)).
nombre(hombre,K,hombre(K)).
nombre(profesor,K,profesor(K)).

determ(el,K,S1,S2,e(K,and(S1,S2))).
determ(un,K,S1,S2,e(K,and(S1,S2))).
determ(los,K,S1,S2,a(K,implies(S1,S2))).
determ(todo,K,S1,S2,a(K,implies(S1,S2))).

prepo(en).
prepo(con).
prepo(de).

verbo(rie,S,[],reir(S)).
verbo(piensa,S,[arg(en,O)],pensar_en(S,O)).
verbo(habla,S,[arg(de,O),arg(con,O1)],comunica(S,O,O1)).
verbo(habla,S,[arg(con,O),arg(de,O1)],comunica(S,O1,O)).
verbo(esta,S,[arg(en,O)],locativo(S,O)).
verbo(lee,S,[arg(nulo,O)],leer(S,O)).
```

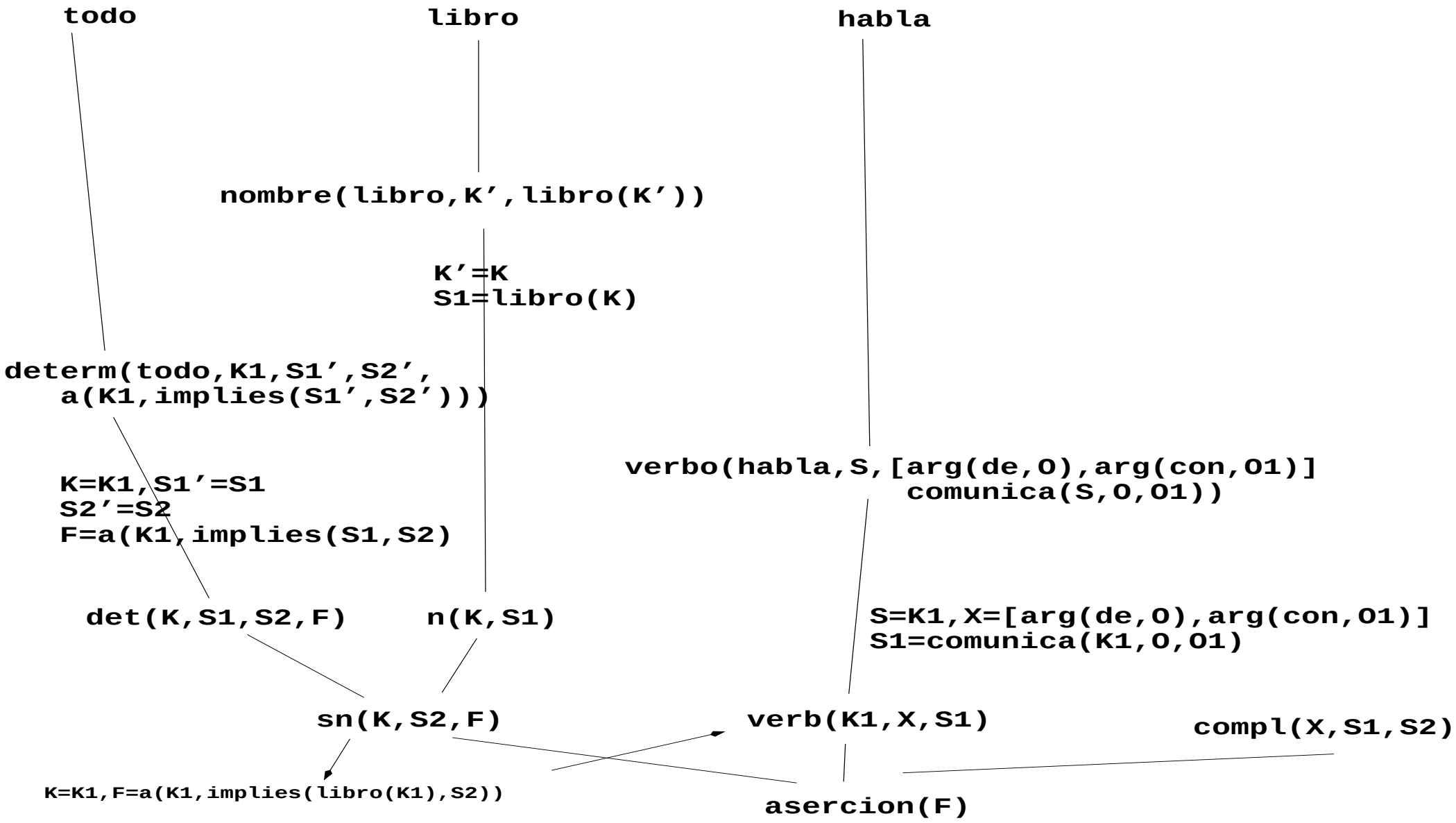
Grammar 4 - Sentences

```
| ?- analisis(F,[el,hombre,rie],[]).

F = e(_A,and(hombre(_A),reir(_A))) ?
yes
| ?- analisis(F,[el,profesor,piensa,en,un,libro],[]).

F = e(_B,and(profesor(_B),e(_A,and(libro(_A),pensar_en(_B,_A))))) ?
yes
| ?- analisis(F,[el,hombre,habla,de,juan,con,maria],[]).

F = e(_A,and(hombre(_A),comunica(_A,juan,maria))) ?
yes
| ?- analisis(F,[todo,hombre,piensa,en  ,un,libro],[]).

F = a(_B,implies(hombre(_B),e(_A,and(libro(_A),pensar_en(_B,_A))))) ?
yes
| ?- analisis(F,[todo,libro,esta,en,barcelona],[]).

F = a(_A,implies(libro(_A),locativo(_A,barcelona))) ?
yes
| ?- analisis(F,[todo,libro,habla,de,un,hombre],[]).

no
| ?- analisis(F,[todo,libro,habla,de,un,hombre,con,un,profesor],[]).

F = a(_C,implies(libro(_C),e(_B,and(hombre(_B),
     e(_A,and(profesor(_A),comunica(_C,_B,_A)))))))) ?
yes
```

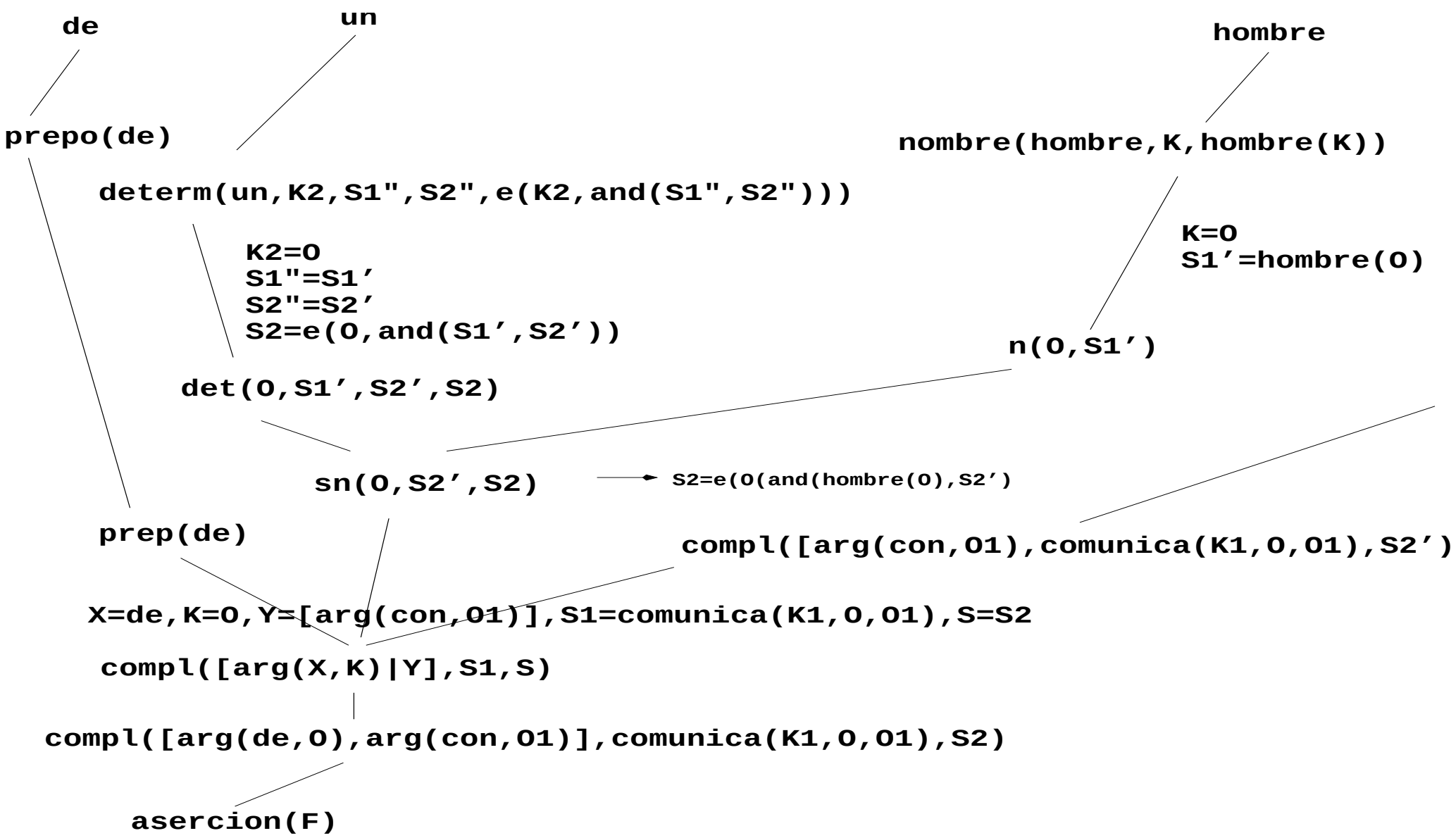
Grammar 4 - Example 1



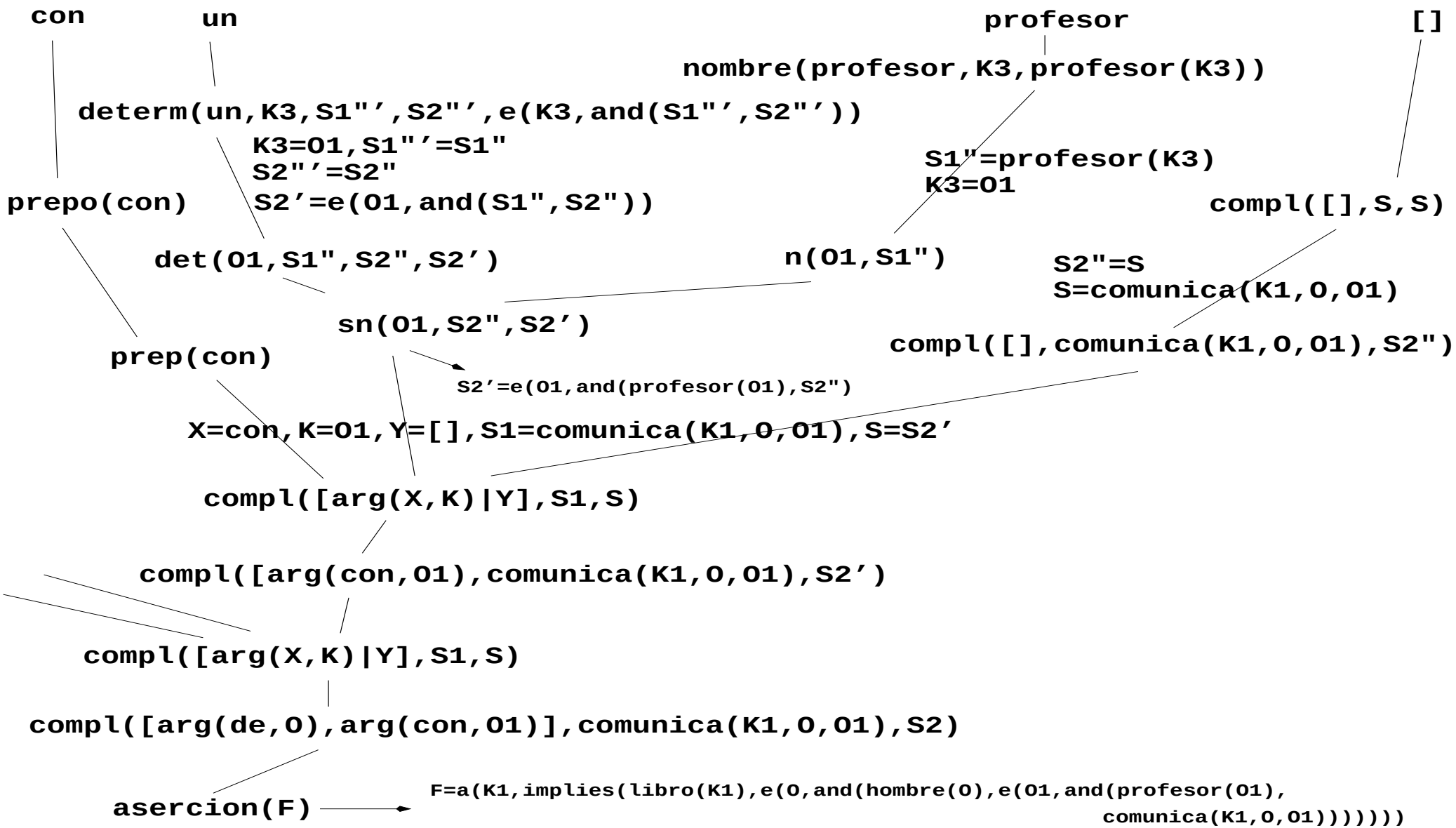
Grammar 4 - Example 2 (I)



Grammar 4 - Example 2 (II)



Grammar 4 - Example 2 (III)



Logic Grammars - Grammar 5

- Finally semantic information is added to the lexicon
- Semantic constraints are added to rules of the verb complements
- Semantic information for the nouns (theme) and verbs (compatible themes) are included in the lexicon

Grammar 5 - I

```
analisis(F,X,Y):- asercion(F,X,Y).

asercion(F) --> sn(K,S2,F,Sem1), verb(K,X,S1,asem(Sem1,Sem2)),
                compl(X,S1,S2,Sem2).

compl([],S,S,[])--> [].
compl([arg(X,K)|Y],S1,S,[Sem1|Sem2])--> prep(X), sn(K,S2,S,Sem1),
                                        compl(Y,S1,S2,Sem2).
compl([arg(nulo,K)|Y],S1,S,[Sem1|Sem2])--> sn(K,S2,S,Sem1),
                                           compl(Y,S1,S2,Sem2).

sn(K,F,F,Sem)--> npr(K,Sem).
sn(K,S2,F,Sem)--> det(K,S1,S2,F),n(K,S1,Sem).

verb(S,A,F,AS)--> [W],{verbo(W,S,A,F,AS)}.
npr(W,S)--> [W],{npropio(W,S)}.
n(K,F,S)--> [W],{nombre(W,K,F,S)}.
det(K,S1,S2,F)--> [W],{determ(W,K,S1,S2,F)}.
prep(W)--> [W],{prepo(W)}.
```

Grammar 5 - II

```
npropio(clara,humano).
npropio(maria,humano).
npropio(juan,humano).
npropio(barcelona,locativo).

nombre(libro,K,libro(K),inanimado).
nombre(hombre,K,hombre(K),humano).
nombre(profesor,K,profesor(K),humano).
nombre(perro,K,profesor(K),animado).
nombre(gato,K,gato(K),animado).
nombre(camino,K,camino(K),locativo).
nombre(pescado,K,pescado(K),inanimado).

determ(el,K,S1,S2,e(K,and(S1,S2))).
determ(un,K,S1,S2,e(K,and(S1,S2))).
determ(los,K,S1,S2,a(K,implies(S1,S2))).
determ(todo,K,S1,S2,a(K,implies(S1,S2))).

prepo(en).
prepo(con).
prepo(de).

verbo(rie,S,[],reir(S),asem(humano,[])).
verbo(piensa,S,[arg(en,O)],pensar_en(S,O), asem(humano,[_])).
verbo(habla,S,[arg(de,O),arg(con,O1)], comunica(S,O,O1),asem(humano,[_,humano])).
verbo(habla,S,[arg(con,O),arg(de,O1)], comunica(S,O1,O),asem(humano,[humano,_])).
verbo(esta,S,[arg(en,O)],locativo(S,O), asem(_,[locativo])).
verbo(lee,S,[arg(nulo,O)],leer(S,O), asem(humano,[inanimado])).
verbo(corre,S,[arg(en,O)],correr(S,O), asem(animado,[locativo])).
verbo(come,S,[arg(en,O)],comer(S,O), asem(animado,[locativo])).
verbo(come,S,[arg(nulo,O)],comer(S,O), asem(animado,[inanimado])).
```