4 – Features and unification

4.1 – Introduction

- To represent some linguistic properties (agreement, subcategorization …) with CFG, it is necessary to multiply the number of non terminals and rules in a non reasonable way with a loss of generality.

- **Features structures** are used to capture these kind of properties.

- Feature structures interact in the parsing process through **unification**.

4.2 – Feature structures

- A feature structure (FS) is a mapping from a finite domain of **features** to a range of **values**. The value of feature F for the FS S is denoted by S.F

- A FS is usually presented in the shape of an **Attribute-Value Matrix (AVM**).

- The most general FS (empty mapping) is denoted by [ ].

- A FS is **recursive** if some of its values themselves are feature structures.

4.2 – Feature structures

- A recursive FS can be defined more precisely as a **directed acyclic graph** (DAG) with a unique root:
  - edges are labelled with features; two edges starting from the same vertex have two different labels;
  - terminal vertices are labelled with atoms, unless the FS reduces to a single vertex; in this case, if it has a label, the FS is an atomic FS; if not, it is the most general FS.
- **Features** are identified with edges and the **value** of a feature is the subgraph rooted at the vertex targeted by the feature.
- A **feature path** is a path in the graph of a FS starting from its root. If the FS is denoted by S, and if the path follows the features F_1 , F_2 , … F_n in this order, it leads to a value that is denoted by $S. F_1 .F_2 \ldots F_n$.

4.2 – Feature structures

- A recursive FS is **reentrant** if its graph has a non terminal value with two incoming features. In other terms, the graph has two different feature paths leading to the same value.

4.3 – Unification of feature structures

- A FS S_1 **subsumes** a FS S_2 ($S_1 \sqsubseteq S_2$) if S_1 and S_2 are equal atomic FS or if S_1 is not atomic and each feature F from S_1 is present in S_2 with the following property:
  - If the value of F is atomic in S_1 , it is atomic in S_2 and the two values are equal.
  - If not, S_1.F subsumes S_2.F

  Moreover, if two feature paths P_1 and P_2 lead to the same value in S_1, there are two paths P_1 and P_2 in S_2 leading to the same value.

  - In other terms, a FS S_1 subsumes a FS S_2 if there exists a **morphism** from the graph of S_1 to the graph of S_2 that preserves the root and the labels.

- The relation of subsumption is a **preorder** with [ ] as its minimum element. A top FS, denoted by $\top$, is added as a maximum element for the preorder.

4.3 – Unification of feature structures

- The **unification** of a FS S_1 with a FS S_2 is the FS $S_1 \sqcup S_2$, if it exists, constituted of the following features and values:
  - The features with their value that are present in only one of the two FS.
  - The features F that are present in both FS with an atomic value in S_1. They must be present in S_2 with an equal value, which is also the value of F in $S_1 \sqcup S_2$.
  - The features F that are present in both FS with a non atomic value in S_1. They must be present in S_2 with a non atomic value and the value in $S_1 \sqcup S_2$ is S1.F $\sqcup$ S_2.F.

Moreover, if two feature paths P_1 and P_2 lead to the same value in S_1 or S_2, there are two paths P_1 and P_2 in $S_1 \sqcup S_2$ leading to the same value.

4.3 – Unification of feature structures

- In other terms, $S_1 \sqcup S_2$ is the most general FS that is subsumed by both S_1 and S_2 . If this FS is $\top$, it means that S_1 and S_2 are not unifiable.

- The set of FS on a given signature with the relation of subsumption is a **semilattice.**

4.3 – Unification of feature structures

1. Find all subsumption relations among the following FS:

a)
cat: S
mood: ind
tense: pres
dg1: cat: NP
  agr: num: sg
    pers: 1
dg2: cat: VP
  mood: ind
  tense: pres
  agr: num: sg
    pers:1

b)
cat: S
mood: <1>
tense: <2>
dg1: cat: NP
  agr: num: sg
    pers: 1
dg2: cat: VP
  mood:<1> ind
  tense:<2> pres
  agr: num: sg
    pers:1

c)
cat: S
mood: ind
tense: pres
dg1: cat: NP
  agr: <3> num: sg
    pers: 1
dg2: cat: VP
  mood: ind
  tense: pres
  agr: <3>

d)
cat: S
mood: <1> ind
tense: <2> pres
dg1: cat: NP
  agr: <3>
dg2: cat: VP
  mood: <1>
  tense: <2>
  agr: <3> num: sg
    pers:1

e)
cat: S
mood: <1>
tense: <2>
dg1: cat: NP
  agr: <3>
dg2: cat: VP
  mood: <1>
  tense: <2>
  agr: <3>

f)
cat: S
mood: <1> ind
tense: <2> pres
dg1: cat: NP
  agr: num: pl
    pers: 1
dg2: cat: VP
  mood: <1>
  tense: <2>
  agr: num: sg
    pers:1

4.3 – Unification of feature structures

2. Compute the following FS if they exist: A ⊔ D B ⊔ C E ⊔ F

A.
- cat: S
- mood: ind
- tense: pres
- dg1:
  - cat: NP
  - agr:
    - num: sg
- dg2:
  - cat: VP
  - mood: ind
  - tense: pres
  - agr:
    - pers:1

B.
- cat: S
- mood: <1>
- tense: <2>
- dg1:
  - cat: NP
  - agr:
    - num: sg
    - pers: 1
- dg2:
  - cat: VP
  - mood:<1> ind
  - tense:<2> pres
  - agr:
    - num: sg
    - pers:1

C.
- cat: S
- mood: ind
- tense: pres
- dg1:
  - cat: NP
  - agr: <3>
    - num: sg
    - pers: 1
- dg2:
  - cat: VP
  - agr: <3>

D.
- cat: S
- mood: <1> ind
- tense: <2> pres
- dg1:
  - cat: NP
  - agr: <3>
- dg2:
  - cat: VP
  - mood: <1>
  - tense: <2>
  - agr: <3>
    - num: sg
    - pers:1

E.
- cat: S
- mood: ind
- tense: pres
- dg1:
  - cat: NP
  - agr: <3>
- dg2:
  - cat: VP
  - mood: ind
  - tense: pres
  - agr: <3>

F.
- cat: S
- mood: <1> ind
- tense: <2> pres
- dg1:
  - cat: NP
  - agr:
    - num: pl
    - pers: 1
- dg2:
  - cat: VP
  - mood: <1>
  - tense: <2>
  - agr:
    - num: sg
    - pers:1

4.4 – Unification grammars

- CFG can be enriched by associating FS with non terminals in all rules to be transformed into Unification Grammars.

- A possible way of doing it is to associate a FS in the following form with any rule $A \longrightarrow B_1 \dots B_n$:

$$\begin{bmatrix} \text{cat: } A \\ \vdots \\ \text{dg}_1\text{: } \begin{bmatrix} \text{cat: } B_1 \\ \\ \end{bmatrix} \\ \vdots \\ \vdots \\ \text{dg}_n\text{: } \begin{bmatrix} \\ \text{cat: } B_n \end{bmatrix} \end{bmatrix}$$

- Chart parsing algorithms for CFG, such as the Earley algorithm, can be enriched to deal with FS : identification of non terminals is replaced with unification of FS and addition of items to the chart is controlled by subsumption.

4.4 – Unification grammars

1. Consider the following CFG :

   S ⟶ NP VP

   NP ⟶ Det N | NP PP | mary

   PP ⟶ Prep NP

   VP ⟶ V NP

   Det ⟶ the

   N ⟶ man | road | forest

   V ⟶ knows | know | comes

   Prep ⟶ on | to

   Transform the grammar into a unification grammar so that the sentence *“mary knows the man on the road to the forest”* can be parsed but not *“* mary know the man on the road to the forest”* and *“* mary comes the man on the road to the forest”*