| TD no12 | **Résolutions des problèmes d'IA en les décrivant** |
|---|---|

*Objectif de la séance :*
Continuer à résoudre des problèmes d'IA en décrivant uniquement le problème et les données

## Traitement automatique de la langue

Le traitement automatique de la langue est l'un des domaine de recherche de l'IA. Les problèmes à résoudre sont la traduction de document, le résumé de document, la recherche d'information, etc.

Il existe principalement deux approches pour résoudre ces problèmes, l'une numérique (utilisant des algorithmes apprentissages statistiques) et l'autre symbolique. C'est ce dernier point de vue que l'on va aborder dans cet exercice.

L'approche symbolique repose sur des étapes de traitements :

1. l'analyse lexicale qui identifie les mots du texte.

   Par exemple la phrase « Le chat boit du lait » est composée du mot « le », suivi du mot « chat », etc.

2. l'analyse syntaxique qui à l'aide d'une grammaire formelle vérifie si cette suite de mots est syntaxiquement bien formée et qui indique le type grammatical de chaque mot (article, nom, verbe, etc.). ou de chaque groupe de mots (groupe nominal, groupe verbal, etc.).

   Par exemple la phrase « Le chat boit du lait » est syntaxiquement correcte et le mot « le » est un article, le mot « chat » est un nom, etc. et que « le chat » est un groupe nominal, et « boit du lait » est un groupe verbal.

   Alors que la phrase est « Le chat boivent du lait » n'est pas syntaxiquement correcte.

3. l'analyse sémantique vérifie si les phrases ont un sens.

   Par exemple la phrase « Le chat boit du lait » est sémantiquement correcte, alors que la phrase « Le chat court du lait » ne l'est pas.

4. l'analyse pragmatique repositionne les phrases dans un contexte (culturel, temporel, géographique, etc.).

   Par exemple dans les deux phrases « Felix est dans la cuisine, il boit du lait », Felix fait très certainement référence à un chat, et le pronom « il » fait référence à ce chat.

Nous allons nous intéresser ici à l'analyse syntaxique pour des phrases affirmatives simples (sujet, verbe, complément d'objet direct) en français : à partir d'une liste de mots représentant une phrase affirmative. L'objectif est, si cette phrase est syntaxiquement correcte, de produire un arbre indiquant le rôle de chaque mot.

Par exemple l'analyse syntaxique de la phrase « Le petit chat noir boit le lait » donne l'arbre présenté par la figure 1, qui indique que la phrase est composée d'un groupe nominal suivi d'un groupe verbal, qui est composé d'un verbe et d'un groupe nominal.

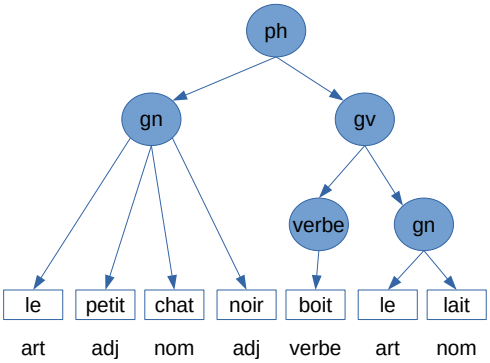


FIGURE 1 – Arbre syntaxique de la phrase "le petit chat noir boit le lait"

L'objectif de cet exercice est de développer un prédicat *analyse/2* qui réalise cette tâche :

```
?- analyse([le,petit,chat, noir, boit, le, lait],Arbre).
Arbre = ph(gn(art(le), adj(petit), nom(chat), adj(noir)), gv
   (verbe(boit), gn(art(le), nom(lait)))) ;
false.
```

swi-prolog propose nativement le prédicat *append/3* tel que `append(Liste1, Liste2, Liste)` est vrai lorsque `Liste` est la concaténation de `Liste1` avec `Liste2`.

L'archive `tal.zip` disponible sur moodle contient les fichiers prolog suivants :

- `pronoms.pl` qui contient des pronoms représentés par les faits *pronom/5* avec pour premier paramètre le pronom, comme deuxième paramètre son genre (`masculin` ou `feminin`), puis son nombre (`singulier` ou `pluriel`), puis son rang (`1`, `2` ou `3`) et comme dernier paramètre le type de pronom personnel (`sujet`, `cod` et `coi`).
- `noms.pl` qui contient un corpus de noms communs représentés par les faits *nom/3* avec pour premier paramètre le nom, puis son genre et enfin son nombre ;
- `adjectifs.pl` qui contient un corpus d'adjectifs représentés par les faits *adjectif/4* avec pour premier paramètre l'adjectif, comme deuxième son genre, comme troisième son nombre et en dernier s'il se positionne généralement `avant` ou `apres` un nom commun.
- `verbes.pl` qui contient un corpus de verbes conjugués représentés par les faits *verbe/4* avec pour premier paramètre la conjugaison d'un verbe, puis son temps de conjugaison, puis son nombre et enfin son rang ;
- `tal.pl` qui importe ces différents fichiers.

Complétez le fichier `tal.pl` en développant les prédicats :

1. *gr_nominal/3* tel `gr_nominal(Mots,Nombre,Arbre)` est vrai lorsque `Arbre` est l'arbre syntaxique du groupe nominal `Mots` tel `Nombre` vaut `singulier` lorsque `Mots` est une suite de mots au singulier, `pluriel` sinon. Par exemple :

   ```
   ?- gr_nominal([le,chat],Nombre,Arbre).
   Nombre = singulier,
   Arbre = gn(art(le), nom(chat)) ;
   false.

   ?- gr_nominal([le,chat,noir],Nombre,Arbre).
   Nombre = singulier,
   ```

```
Arbre = gn(art(le), adj(noir), nom(chat)) ;
false.

?- gr_nominal([le,petit,chat],Nombre,Arbre).
Nombre = singulier,
Arbre = gn(art(le), adj(petit), nom(chat)) ;
false.

?- gr_nominal([les,petits,chats,noirs],Nombre,Arbre).
Nombre = pluriel,
Arbre = gn(art(les), adj(petits), nom(chats), adj(noirs)
    ).
```

2. *gr_verbal/4* tel `gr_verbal(Mots,Nombre,Rang,Arbre)` est vrai lorsque `Arbre` est l'arbre syntaxique du groupe verbal `Mots`, tel qu `Mots` commence par un verbe conjugué, suivi d'un groupe nominal et que `Nombre` et `Rang` caractérisent la conjugaison de ce verbe (`singulier` ou `pluriel` pour `Nombre` et `1`, `2` ou `3` pour `Rang`). Par exemple :

```
?- gr_verbal([boit,le,lait],Nombre,Rang,Arbre).
Nombre = singulier,
Rang = 3,
Arbre = gv(verbe(boit), gn(art(le), nom(lait))) ;

?- gr_verbal([boivent,le,lait],Nombre,Rang,Arbre).
Nombre = pluriel,
Rang = 3,
Arbre = gv(verbe(boivent), gn(art(le), nom(lait))) ;
Nombre = pluriel,
Rang = 3,
Arbre = gv(verbe(boivent), gn(art(le), nom(lait))) ;
false.
```

3. *analyse/2* tel présenté précédemment : cas où le sujet est un groupe nomimal (le `Rang` du verbe conjugué est alors obligatoirement `3`).
4. Améliorer le prédicat *analyse/2* afin que le sujet puisse aussi être un pronom, par exemple :

```
?- analyse([je,conduis,une,voiture],Arbre).
Arbre = ph(pr(je), gv(verbe(conduis), gn(art(une), nom(
    voiture)))) ;
false.
```