Rapport TP1 Technologie de la langue

Marin Julien

3 mars 2018

1 Fonctionnalités

Le programme final a les fonctionnalités suivantes :

— construction de phrase depuis une grammaire et un vocabulaire (et donc vérification si phrase bien construite)
— accord en genre, nombre des éléments concernés.
— distinction de si les objets sont animés (une pomme ne dort pas)
— analyse syntaxique d'une phrase
— enrichissement du vocabulaire *pendant* l'exécution
— proposition d'enrichissement lorsqu'une phrase n'est pas reconnue.

2 Partie faite en cours

Il a progressivement été implémenté la grammaire, l'accord de genre et de pluriel et d'animation avec les objets, et l'analyse syntaxique. Détaillons ici les éléments qui nous semble d'intérêt.

La grammaire est, simplement, celle qui a été indiquée en cours.

Le genre (`masc`, `fem`), la pluralité (`sing`, `pluriel`) et le caractère animé d'un objet (`animé`, `nonanimé`) sont renseignés comme des termes dans nos prédicats *np* (nom propre), *nc* (nom commun), etc, là où la différenciation est pertinente : le caractère animé n'importe pas exemple pas pour les déterminants.

À des fins de debug, une fonction `phrases.` a été codé, pour obtenir l'ensemble des phrases bien formées selon la grammaire et les mots renseignés.

Nous avons implémenté la relation qui (marie qui mange des pommes dort) , qui permet d'inclure un groupe verbal dans un groupe nominal. Pour éviter que cela donne lieu à des scénarios de boucle infini (un GN qui génère un GV qui génère un GN. . .), nous avons différencié deux groupe nominaux possibles :

— Le groupe nominal "général", qui inclus la relation qui
— Le groupe nominal "Sans récursion" qui en est une sous catégorie, et qui n'inclus pas cette relation.

Finalement, pour l'analyse syntaxique nous avons ajouté à chacun de prédicats représentants nos syntagmes (et classe grammaticales) un terme identifiant le noeud de l'arbre syntaxique correspondant ; par exemple "*phrase*" pour la racine de la phrase.

3 Bonus

Ont été ajoutées aux programmes les features suivantes :

— Enrichir le vocabulaire connu par le programme pendant l'exécution

— Demander un enrichissement de vocabulaire lorsqu'une phrase n'est pas comprise

Nous avons découpé l'implémentation de cette partie en deux programmes dans un soucis de modularité, la quantité de code s'agrandissant.

Divers commentaires ont été rédigés pour expliquer le code, de nombreux affichages `writeln` pour expliquer son déroulement ont été produits. Les deux modules possèdent une fonction `main.` pour mettre en évidence ces nouvelles fonctionnalités et comment les mettre en oeuvre.

3.1 Automodification

Le premier module, `automodif.pl`, permet la prise en charge de la modification par des prédicats dynamiques et l'implémentation de ces deux features. Des fonctions pour chaque classe grammaticale ont été créés pour pouvoir enrichir le vocabulaire en saisissant les accords nécessaires (genre, nombre, caractère animé) tels que `ajoutNP(Mot)`, `ajoutNC(Mot)`, ... Une autre fonction, `ajoutMot(Mot)` a alors été rajouter de manière à pouvoir saisir le type grammatical d'un mot et appeler la fonction d'ajout associé.

On créé un prédicat `dico(Mot,ClasseGrammaticale)` qui contiendra l'ensemble de notre vocabulaire. Cela permettra, pour chaque phrase, de vérifier si ses mots sont connus par la fonction `parcoursVerifDico`. Si un mot n'est pas reconnu par la fonction, elle proposera alors son ajout.

3.2 Interfaçage

Le deuxième programme `TP1_automodif.pl` consiste en l'interfaçage de ce dernier module au travail réalisé en cours. Il nécessite le premier module pour son fonctionnement. Nous avons choisi de faire le peuplement initial de notre vocabulaire par les fonctions crées dans `automodif.pl` dans un soucis de lisibilité, en utilisant des fonctions tel que `ajoutNP(Nom,Genre).`. On peut ainsi peupler nom propres, noms communs, déterminants, verbes intransitifs et transitifs. La reconnaissance de phrase vérifie (1) si la phrase n'est pas reconnue (grammaticalement par manque de vocabulaire), et (2) s'il manque du vocabulaire (grâce à `parcoursVerifDico` proposant ainsi d'ajouter le vocabulaire manquant).

4 Perspective

Les adjectifs et adverbes n'ont pas été ajoutés par manque de temps, mais cela ne devrait pas demander beaucoup d'effort de les ajouter à l'architecture actuelle, les modifications n'étant principalement que sur la grammaire, qui ne fait actuellement qu'une vingtaine de lignes.

On imagine (1) un syntagme adjectival comprenant adverbe et adjectifs, (2) une distinction pour les adjectifs épithète se positionnant avant le nom (e.g. une *grande* montagne) et ensuite (e.g. une montagne *magnifique*), et (3) la prise en charge d'adjectif épithète dans les groupe nominaux par inclusion dans ceux ci de syntagme adjectival.

D'avantage de travail serait nécessaire pour (4) inclure des adjectifs attribut (e.g. la voiture *est* très belle) dans la grammaire, nécessitant une distinction parmis les verbes transitifs de verbe d'état (être, paraître, sembler ...)