TP : Conception d'une application de TALN

(activité d'équipe de 2 à 4 membres)

IFT-2003 Intelligence artificielle I - H2016
Laurence Capus

Travail pratique : dialogue avec un automate d'une compagnie de transport par autocar

Ce travail compte pour 5% de la note finale.
Échéance : le 21 avril 2016 (23h59)

Objectifs d'apprentissage :

À l'issue de ce travail, un étudiant devrait être capable de :

- résoudre un problème concret de traitement automatique du langage naturel;
- identifier les enjeux du traitement du langage naturel sur un problème concret;
- analyser les limites d'une solution à un problème concret de traitement du langage naturel.

Description du contexte du travail pratique :

On vous demande de développer un module de questions/réponses en langage naturel permettant à un utilisateur d'interagir plus facilement avec un automate d'une compagnie de transport par autocar. Celui-ci posséderait un ensemble d'informations permettant d'assurer un dialogue du type :

- À quelle heure part le prochain autocar pour Montréal ?
- 16h
- Combien de temps faut-il pour aller de Québec à Rimouski ?
- 4h30
- Quel est le prix d'un trajet Québec-Montréal ?
- 50 $

Votre tâche consisterait donc à construire un programme en langage Prolog qui permet de :

- comprendre la question de l'utilisateur exprimée en langage naturel;
- rechercher la réponse à la question de l'utilisateur en utilisant les informations que possèdent l'automate;
- formuler une réponse.

L'annexe 2 donne la définition du prédicat lire/2 qui transforme une phrase en une liste de mots. Cette liste est plus facilement exploitable par un analyseur.

Travail à réaliser :

1- **Base d'informations (10 %)** : Constituer un ensemble d'informations à donner à l'automate. Expliquer chaque prédicat utilisé pour représenter ces informations. Par exemple, le prédicat trajet(Départ, Destination, HD, HA) signifie qu'un autocar quitte le lieu Départ à l'heure HD, vers le lieu Destination et son heure d'arrivée prévue est HA. Il est possible d'utiliser un ou plusieurs prédicats selon les informations retenues. Implanter ces informations en Prolog.

2- **Questions (10 %) :** Déterminer les questions auxquelles l'automate sera capable de répondre, au moins 5 questions de types différents. L'objet sur lequel porte la question doit être différent de même que le format de la question. Par exemple, les questions « Combien coûte un trajet entre Québec et Montréal ? » et « Combien coûte un trajet entre Rimouski et Québec ? » sont considérées identiques.

3- **Analyse sémantique (30 %) :** Donner la grammaire attribuée capable d'analyser les questions indiquées en (2) et de retourner la sémantique contenue dans les questions (sens des questions). Implanter cette grammaire en langage Prolog à l'aide du prédicat question/3.
Par exemple :

```
?- question(Q, [combien, coûte, un, trajet, entre, québec, et, montréal], []).
Q=prixTrajet(québec, montréal, Variable)
```

4- **Réponse aux questions (10 %) :** Implanter un programme en Prolog pour obtenir les réponses de l'automate à partir de ses propres informations, à l'aide du prédicat reponse/2. Afficher un texte « La réponse est : » suivi de la réponse à la question posée.
Par exemple :

```
?- question(Q, [combien, coûte, un, trajet, entre, québec, et, montréal], []), reponse(Q,R).
La réponse est : 50 $
```

5- **Interface (5 %) :** Ajouter un prédicat « lancer/0 » qui permet de démarrer le dialogue avec l'automate.

6- **Résultats (25%) :**

- Faire des tests pour montrer le niveau d'interprétation des questions (à donner dans le rapport).
- Discuter les résultats obtenus : Êtes-vous satisfait de votre travail ? Comment pourriez-vous l'améliorer ? Est-ce que de nouvelles questions pourraient facilement être interprétées ?

Les 10% restants sont dédiés à l'appréciation globale du travail remis (présentation et expression écrite).
Ne pas oublier de valider votre travail à l'aide de la grille d'autoévaluation fournie à l'annexe 1.

<u>Modalités de remise de ce travail</u> :

Partie logicielle (tous les éléments de la partie logicielle doivent être documentés et dans 1 seul fichier PL) :

- la base d'informations de l'automate;
- le prédicat question/3;
- le prédicat reponse/2;
- le prédicat lancer/0 (seul prédicat d'interface).

Partie rapport (en .PDF) :

Le rapport doit avoir comme d'habitude une page couverture, une introduction et une conclusion et les éléments suivants :

- la base d'informations avec les prédicats utilisés;
- les questions choisies et les réponses qui seront fournies;
- la grammaire qui permet d'analyser les questions;
- la partie Résultats.

L'ensemble du travail est à remettre en format électronique via le Portail des cours. Les travaux remis en retard ne seront pas corrigés.

Annexe 1. Grille d'autoévaluation

Conception (65%)				
Base d'informations (10%)	Une base d'informations est implantée et tous les prédicats utilisés sont expliqués.	Une base d'information est implantée mais tous les prédicats utilisés ne sont pas expliqués.	Seulement les prédicats utilisés sont expliqués.	Pas de base d'information implantée et aucun prédicat utilisé n'est expliqué.
Identification des questions et réponses (10%)	Au moins 5 types de questions sont identifiés et utilisés par le programme.	Seulement 4 types de questions sont identifiés et utilisés par le programme.	Seulement 2 types de questions sont identifiés et utilisés par le programme.	Aucune question n'a été identifiée.
Analyse des questions et implantation du prédicat question/3 (30%)	Une grammaire attribuée est proposée et permet d'analyser les questions.	Une grammaire attribuée est proposée mais ne permet pas d'analyser toutes les questions.	Une grammaire attribuée est proposée mais ne permet d'analyser aucune des questions.	Aucune grammaire n'est proposée pour analyser les questions.
	La grammaire est implantée et fonctionne sans erreur.	La grammaire est implantée mais fonctionne avec erreur.	La grammaire est implantée mais ne fonctionne pas du tout.	La grammaire n'est pas implantée.
Implantation du prédicat reponse/2 (10%)	Le prédicat est implanté et fonctionne sans erreur.	Le prédicat est implanté mais fonctionne avec erreur.	Le prédicat est implanté mais ne fonctionne pas du tout.	Le prédicat n'est pas implanté
Prédicat lancer/0 (5%)	Le prédicat est implanté correctement.	Le prédicat est implanté avec des erreurs.	Le prédicat n'est pas du tout implanté.	

Résultats (25%)				
Exemples de dialogue	Plusieurs exemples sont présentés et les résultats sont discutés.	Plusieurs exemples sont présentés mais les résultats ne sont pas discutés.	Un seul exemple est présenté et discuté.	Pas d'exemple ou un seul exemple est présenté mais non discuté.
Améliora-tions possibles	Plusieurs améliorations possibles sont proposées et illustrées par des exemples de dialogue ou autres.	Plusieurs améliorations possibles sont proposées mais plus ou moins illustrées.	Une seule amélioration est proposée et plus ou moins illustrée.	Aucune amélioration n'est proposée.
Appréciation globale (10%)				
Expression écrite	Le rapport ne contient aucune faute (vocabulaire, grammaire, syntaxe, etc.).	Le rapport ne contient pas plus d'une dizaine de fautes (vocabulaire, grammaire, syntaxe, etc.).	Le rapport ne contient pas plus de 5 fautes par page (vocabulaire, grammaire, syntaxe, etc.).	Le rapport contient plus de 5 fautes par page (vocabulaire, grammaire, syntaxe, etc.).
Présenta-tion du rapport	Tous les éléments demandés sont présents et le rapport est paginé et contient une page de garde.	Tous les éléments demandés sont présents, mais le rapport n'est pas paginé ou ne contient pas de page de garde.	Il manque 1 élément demandé.	Il y a au moins 2 éléments demandés manquants.

Annexe 2 :

La définition en Prolog du prédicat lire/2 qui permet de traduire une phrase en une liste de mots est la suivante :

```
% Le prédicat lire/2 lit une chaîne de caractères Chaine entre apostrophes
% et terminée par un point.
% Resultat correspond à la liste des mots contenus dans la phrase.
% Les signes de ponctuation ne sont pas gérés.

lire(Chaine,Resultat):- write('Entrer la phrase '), read(Chaine),
                        name(Chaine, Temp), chaine_liste(Temp, Resultat),!.

% Prédicat de transformation de chaîne en liste

chaine_liste([],[]).
chaine_liste(Liste,[Mot|Reste]):- separer(Liste,32,A,B), name(Mot,A),
chaine_liste(B,Reste).

% Sépare une liste par rapport à un élément

separer([],X,[],[]):-!.
separer([X|R],X,[],R):-!.
separer([A|R],X,[A|Av],Ap):- X\==A, !, separer(R,X,Av,Ap).
```