Intelligence Artificielle et Prolog

Programme d'Aide au Voyageur

Binôme :
Antônio Guilherme Ferreira Viggiano
Bruno Brizida Dreux

Professeur :
Catherine Jazzar
Pascal Prea

28 juin 2012

Table des matières

1 Présentation du sujet

Le contexte choisi pour le projet est la suggestion de destinations de voyage. En se basant sur les informations entrées par l'utilisateur, qui disent respect à lui même et à ses goûts en voyage, le système expert propose des destinations adaptées. Ce contexte a été choisi non seulement parce que voyager c'est une passion pour nous, mais aussi parce que il peut aboutir en une application commerciale. Une possibilité, par exemple est un site que, après proposer la destination, propose des liens vers des sites des entreprises partenaires, d'achat de billets, réservation d'hôtels, etc.

2 Le Programme

Dans cette section, nous allons présenter le programme, ses algorithmes plus importants, les représentations internes et l'interface graphique.

2.1 Les algorithmes

2.1.1 Chaînage Avant

Le chaînage avant doit, à partir des informations confirmés (base de faits) et des règles universelles (base de règles), aboutir à des conclusions pas évidentes au début. Pour le faire, il parcourt plusieurs fois la base de règles, et il ajoute les conclusions obtenues à chaque passage à la base de faits. À la nouvelle passage, les conclusions peuvent être différentes, parce que les faits le sont. Il doit s'arrêter quand il parcourt toutes les règles et qu'il n'y a pas de conclusion nouvelle.

L'algorithme du chainage avant est le suivant :

```
chainageAvant(BaseDeFaits, BaseDeRegles) : procédure
boolean conditionVrai = Vrai
entier numeroChanges = 1
entier resultat : tableau
tant que numeroChanges!=0:
      numeroChanges = 0
      pour tout i en BaseDeRegles
           si conclusion(regle[i]) n'appartient pas à BaseDeFaits
                conditionVrai = Vrai
                pour tout j en conditions de regle[i]
                     si condition[j] de regle[i] n'appartient pas à BasedeFaits
                          conditionVrai = Faux
                     fin si
                fin pour
                si conditionVrai est Vrai
                     ajouter la conclusion de la regle[i] à resultat
                     ajouter 1 à numeroChanges
                fin si
           fin si
      fin pour
fin tant que
fin
```

2.1.2 Chainage Arrière

Le chaînage arrière doit, à partir d'une base de règles et d'une base de faits et d'une conclusion, vérifier si cette conclusion s'applique à la base de faits. Pour le faire, il parcourt toutes les règles qui aboutissent à la conclusion recherchée et vérifie, de manière récursive, si avec les conditions de départ, on peut satisfaire toutes les conditions nécessaires.

L'algorithme du chainage arrière est le suivant :

```
chainageArriere(BaseDeFaits, BaseDeRegles, but:entier) : boolean
boolean conditionVrai = Vrai
si but appartient à BaseDeFaits
      renvoyer Vrai
fin si
Pour i en toutes les règles dont la conclusion est but
      conditionVrai = Vrai
      Pour j en toutes les conditions de règle[i]
           conditionVrai = conditionVrai et chainageArriere(BaseDeFaits,
           BaseDeRegles, condition[j] de regle[i])
      fin pour
      si conditionVrai = Vrai
           renvoyer Vrai
      fin si
fin pour
renvoyer Faux
fin
```

2.2 La présentation de la réalisation

2.2.1 Répresentation interne des mots par des entiers

Pour faciliter le traitement de l'information, on a décidé de coder tous les mots propres au programme en entiers. Un exemple d'une telle traduction serait :

```
1 Jeune
2 En couple
3 Rome
```

L'utilisation d'une classe "Traduction" simplifie énormément le problème d'acquisition des données. On se limite à programmer la partie logique du moteur d'inférences, qui peut être réutilisée dans n'importe quel contexte. Ce programme est donc tout à fait *général*.

Pour changer le "thème" du programme (par exemple, si on décide de faire un serveur qui aide le client à choisir son plat au lieu d'une "aide au voyageur"), il suffit de récrire la Base de Règles et d'ajouter une Base de Faits, suivant les conventions établies dans le programme. Cette traduction est entièrement interne au programme, et donc aucune action de l'utilisateur est nécessaire à ce propos.

Pour le programme, une règle est un objet de la classe "Regle", qui contient un entier, la conclusion, et un tableau d'entiers, qui contient les conditions. Une base de règles est simplement donc un tableau de règles.

2.2.2 Les fichiers

Pour stocker les informations propres au programme (Base de Règles, Base de Faits, Traduction), nous avons utilisé des fichiers texte. En début de programme, le fichier de traductions est chargé automatiquement, ainsi comme un fichier de la Base de règles "default". Les fichiers de bases de faits et de bases des données peuvent êtres chargés, modifiés et enregistrés par l'utilisateur depuis le programme.

Nous avons créé trois extensions de fichier spécifiques : ".trd", pour les traductions, ".bdf", pour les bases de faits et ".bdr", pour les bases de règles.

Un fichier de traduction contient, à chaque ligne, un couple entier-mot correspondant. Pour que le programme puisse identifier si le mot est une destination de voyage, on a décidé de réserver les entiers multiples de 10 pour les destinations.

Un fichier de base de règles contient, à chaque ligne, plusieurs entiers. Par souci de simplicité, nous avons choisi de mettre d'abord la conclusion, et ensuite les conditions, dans la forme :

```
ConditionALORS ConditionSI ConditionSI ConditionSI ...
```

Un fichier de base de faits contient, à chaque ligne, un entier, qui répresente le fait en question. Pour pouvoir utiliser les mêmes fonctions de gestion de fichier et la classe "Regle" pour les faits, il faut mettre une valeur de contrôle dans la variable entière alors. Nous avons choisi de mettre -1, et donc chaque ligne du fichier d'une base de faits commence par un -1.

3 Interface Graphique

La principale raison pour laquelle nous avons choisi de faire ce projet sous Java a été sa facilité de développer une interface graphique amie de l'utilisateur et cohérente.

En suivant une logique usuelle de construction de composants et division d'espace, on a décidé de mettre une barre de menus (`JMenu`) dans le haut de la fenêtre, et une aire de text (`JTextArea`) dans la partie du bas, qui actualise constamment l'avancement du programme (messages d'annulation, d'échec, etc.). Dans la partie centrale, où toute "l'action" du programme se déroule, il y a le panneau principal qui, à son tour, est composé d'une barre de roulage (`JScrolledPane`) qui prend les autres panneaux et les affiche quand nécessaire.

Tout cela est géré dynamiquement, au travers d'une classe principale `GUI`, pour *Graphic User Interface*, qui coordonne tous les autres panneaux (ceux de création d'une Base de Règles ou d'une Base de Faits, par exemple) et qui manipule ces bases de données.

Les points clés de ce projet d'Intelligence Artificielle, les chainages Avant et Arrière, bien évidamment n'ont pas été oubliées, et des boutons indicatifs ont été mis dans une barre d'outils (`JToolBar`) juste en dessous de la barre de menus.

Dans le cadre des ajouts de règles ou de faits d'un nombre à priori non déterminé (par exemple, l'utilisateur peut créer des règles de n'importe combien de conditions), nous avons choisi la solution de le lui demander avant de l'affichage des cases à compléter (`JComboBox`). Cette méthode évite de gérer des boutons qui doivent être actualisés constamment (on pourrait supposer qui l'utilisateur veuille d'abord ajouter une seule règle, ensuite trois autres – cela serrait assez difficile de gérer se dynamiquement).

En ce qui concerne la partie esthétique du projet, le manager de l'interface de l'utilisateur propose trois options de visualisation graphiques de la fenêtre, aussi connus comme *look and feel*. Celui de démarrage est le "Windows", mais on peut aussi jouer avec les `LookAndFeel` Métal ou le futuriste "Nimbus".

4 Manuel de l'utilisateur

4.1 Installation et consultes rapides

Le logiciel ne doit pas être installé. Pour le lancer, il suffit d'aller dans le répertoire "Aide au voyageur" et lancer le ficher "aide_voyageur_1_0".

Le programme développé possède une interface graphique standard, comme la majorité des logiciels aujourd'hui. De ce fait, l'utilisation de cet outil ne doit pas être difficile pour une personne qui a les connaissances minimales pour utiliser un ordinateur, l'interface étant très intuitive.

Pour un utilisateur débutant, qui veut consulter des destinations de voyage, il suffit d'aller dans le menu déroulant "Base de Faits > Créer", entrer ses informations et goûts, et après appuyer sur "Chainage Avant". Si, à la place, il veut vérifier si une destination spécifique est adaptée pour lui, il faut appuyer sur "Chainage Arrière" après avoir créé la Base de Faits.

Celles sont les applications plus courantes du logiciel, mais beaucoup plus de possibilités sont disponibles. Elles seront traitées en détaille dans la suite.

4.2 Les menus déroulants

4.2.1 Le menu Fichier

- Enregistrer : enregistre la Base de Faits et la Base de Règles
- Enregistrer sous : comme "Enregistrer", sauf qu'il permet aussi de choisir le répertoire où le fichier sera enregistré.

4.2.2 Les menus Base de Faits et Base de Règles

Les menus Base de Faits et Base de Règles possèdent les mêmes options, pour les deux types de base de données, qui sont :

- Charger : permet de charger une base de données qui a été enregistré avant. [1]
- Afficher : permet d'afficher la base de données chargé actuellement.
- Créer : permet la création d'une base de données.
- Modifier : permet de modifier la base de données actuelle, soit en ajoutant ou enlevant des éléments, soit en modifiant les éléments eux mêmes.

4.2.3 Les menus "Look et Feel" et "Help"

Le menu Look et Feel permet de choisir entre trois styles graphiques différents pour la fenêtre. Le menu Help, a travers le item "About", permet d'afficher les informations relatives au programme (créateur, version, etc.). A partir de l'item "Manuel de l'utilisateur", il est possible d'accéder le manuel de l'utilisateur, qui explique en détail le fonctionnement du programme.

4.3 Les boutons

Les boutons permettent de lancer les fonctions les plus importantes du logiciel : le chaînage avant et le chainage arrière. Lors de l'appui du bouton chaînage avant, il commence automatiquement le calcul. Pour le chaînage arrière, il demande d'abord à l'utilisateur quel destination il veut consulter.

4.4 La barre de statut

La barre en bas de la fenêtre graphique permet, en tout moment, de vérifier la situation du programme. Elle indique s'il y a eu des erreurs, si le programme est en train de calculer les réponses, s'il a réussi, etc.

1. Une base de données "default" est chargé au début de l'exécution du programme.