TRAVAUX PRATIQUES
IA : PROGRAMMATION PAR CONTRAINTE EN ROBOTIQUE
5 ETI

Objectifs pédagogiques des TPs

Nous souhaitons vous faire découvrir en quoi une thématique de l'intelligence artificielle, *la programmation logique* et plus particulièrement *la programmation par contrainte*, permet de concevoir des solutions originales à des problématiques de la robotique.

Pour cela, nous vous proposons deux TPs très applicatifs de découvertes.

La programmation logique

Ce premier TP vous fera découvrir l'univers de la programmation logique avec notamment une partie de Prolog [1].

Nous travaillerons sur un jeu de données réel provenant de la RobotCup.

Nous ferons ensuite le lien entre prolog et un mécanisme d'intégration simple en Python.

Pyke et la programmation logique

Pyke [2] est une librairie Python avancée intégrant de très nombreuses fonctionnalités permettant d'implémenter des programmes utilisant la programmation logique.

Nous découvrirons quelques–unes de ces fonctionnalités, la structure d'une solution basée sur Pyke, ainsi que quelques autres aspects intéressants (traitement simple des interactions en langage naturel par exemple).

Nous appliquerons concrètement cette librairie sur les travaux réalisés lors du TP précédent.

1. Un des premiers solveurs permettant d'exprimer des contraintes, une référence du domaine de la programmation logique, développé par l'INRIA
2. http://pyke.sourceforge.net

Première partie

Prolog et la programmation logique

Exercice 1 : Prise en main de prolog

Il existe deux implémentations principales de prolog : swi-prolog ou gprolog.

Les exemples seront donnés en gprolog.

Le tutoriel http://pcaboche.developpez.com/article/prolog/presentation/ devrait suffire pour le commencer.

0.1.1 Chargez le fichier de règles `bases.pl` avec gprolog

```
gprolog

| ?- consult('bases').
```

N'oubliez pas de terminer toute instruction prolog par un point (.).

0.1.2 Lisez le fichier de règles et expliquez chacune des règles présentes

0.1.3 Demandez l'étude des contraintes suivantes :

1. 'Bruno' est-il une personne ?
2. 'Donald' est-il une personne ?
3. Qui est une personne ?
4. existe-t-il au moins une personne ?
5. existe-t-il au moins un animal(X) ?
6. existe-t-il une relation `objet` ?

Résultats attendus :

1. 'Bruno' est-il une personne ? `yes`
2. 'Donald' est-il une personne ? `no`
3. Qui est une personne ?

   ```
   X = 'Bruno' ? ;
   X = 'Alain'
   yes
   ```

   ou

   ```
   X = ['Bruno','Alain']
   yes
   ```

   si vous utilisez le prédicat `findall(,contrainte,X).`
   voir http://pcaboche.developpez.com/article/prolog/findall-bagof-setof/
4. existe-t-il au moins une personne ?

   ```
   true ? ;
   yes
   ```

5. existe-t-il au moins un animal ? `uncaught exception: error(existence_error(procedure ,animal/1),top_level/0)`
6. existe-t-il une relation `objet` ? `uncaught exception: error(existence_error(procedure ,objet/0),top_level/0)`

0.1.4 Créez quelques nouvelles règles simples que vous testerez dans le solveur

Ajoutez par exemple la règle atomique "objet".

Exercice 2 : Règles non atomiques

Consultez le fichier `regles.pl`.

0.2.1 Réalisez le même travail de test et lecture que pour l'exercice précédent

0.2.2 Comment s'exprime la négation ?

Exercice 3 : Etude conjointe d'un système contraint : exemple du zèbre

Etude de l'énigme du zèbre :

1. Le norvégien habite la première maison,
2. La maison à coté de celle du norvégien est bleue,
3. L'habitant de la troisième maison boit du lait,
4. L'anglais habite la maison rouge,
5. L'habitant de la maison verte boit du café,
6. L'habitant de la maison jaune fume des kools,
7. La maison blanche se trouve juste après la verte,
8. L'espagnol a un chien,
9. L'ukrainien boit du thé,
10. Le japonais fume des cravens,
11. Le fumeur de old golds a un escargot,
12. Le fumeur de gitanes boit du vin,
13. Le voisin du fumeur de Chesterfields a un renard,
14. Le voisin du fumeur de kools a un cheval.

Qui boit de l'eau ? Qui a le zèbre ?

Exercice 4 : Un cas concret de robotique

L'objectif de la fin du TP est de réaliser une base de connaissance commune sur les données issues de la RobotCup (https://github.com/RoboCupAtHome/Leipzig2016/).

Exemples de tâches et de scenarii : https://github.com/RoboCupAtHome/Leipzig2016/find/master

Objets et catégories : pdf dans le répertoire ressources

Lieux : csv dans le répertoire ressources.

0.4.1 Lisez différents types de scenarii

0.4.2 Discussion générale : quels types de problèmes pourrait-on résoudre ?

0.4.3 Modélisation et répartition des tâches

0.4.4 Utilisation de python avec prolog

voir pour le moment PyLog de Christophe Delord : http://cdsoft.fr/pylog/index.html