M1Info/BIA/2010-2011
TP5/6 PROLOG
(PROgrammation LOGique)

Illustration du cours Système à Base de Connaissances : Etude pour la réalisation d'un moteur d'inférence en PROLOG

Avertissement : Ce TP est noté. Vous devez rendre un CR correspondant au plan proposé et un code commenté avant le 17 décembre (le tout sous format électronique : .PDF ou .DOC pour le CR et .TXT ou .PL pour les sources commentés). Pénalités après le 19 décembre.

Objectif des deux séances de TP

Il s'agit d'étudier les mécanismes d'un moteur d'inférence en chaînage avant simple, arrière et mixte. Les tests seront fait sur des requêtes sur la base de connaissances fournie.

On donne une base de règles à titre d'exemple :

r1 :	**Si** fleur **et** graine	**Alors** phanérogame
r2 :	**Si** phanérogame **et** graine_nue	**Alors** sapin **et** ombre
r3 :	**Si** phanérogame **et** 1cotylédone	**Alors** monocotylédone
r4 :	**Si** phanérogame **et** 2cotylédone	**Alors** dicotylédone
r5 :	**Si** monocotylédone **et** rhizome	**Alors** muguet
r6 :	**Si** dicotylédone	**Alors** anémone
r15 :	**Si** joli	**Alors non** rhizome
r7 :	**Si** monocotylédone **et non** rhizome	**Alors** lilas
r8 :	**Si** feuille **et non** fleur	**Alors** cryptogame
r9 :	**Si** cryptogame **et non** racine	**Alors** mousse
r10 :	**Si** cryptogame **et** racine	**Alors** fougère
r11 :	**Si** non feuille **et** plante	**Alors** thallophyte
r12 :	**Si** thallophyte **et** chlorophylle	**Alors** algue
r13 :	**Si** thallophyte **et non** chlorophylle	**Alors** champignon **et non** bon
r14 :	**Si non** feuille **et non** fleur et **non** plante	**Alors** colibacille

1 Moteur en chaînage avant

On veut réaliser un moteur d'inférence d'ordre zéro, fonctionnant en chaînage avant, en régime irrévocable et monotone, et qui ne constitue pas d'ensemble de conflit (dès qu'une règle est déclenchable, elle est déclenchée).

• Définir la base de règles grâce à un prédicat regle :

regle(ri) :- si(Liste de prémisses), alors(Liste de conclusions).

• Définir un prédicat permettant à l'utilisateur d'initialiser la base de faits. On utilisera le prédicat assert pour ajouter vrai(Fait) pour les faits positifs et faux(Fait) pour les faits négatifs.

• Définir un prédicat qui sature la base de règles et produit une trace de son fonctionnement (on pourra s'inspirer du troisième moteur d'inférence présenté en cours).

Exemple d'exécution :

```
?- faits([fleur,graine,un_cotyledone,non(rhizome)]).
Yes

?- saturer.
r1
non(rhizome) fleur graine un_cotyledone phanerogame
r3
non(rhizome) fleur graine un_cotyledone phanerogame monocotyledone
r7
non(rhizome) fleur graine un_cotyledone phanerogame monocotyledone lilas
Yes
```

2 Moteur en chaînage arrière

• Écrire un moteur d'inférence d'ordre 0 fonctionnant en chaînage arrière.
On représentera la base de règles et la base de faits comme dans la première partie.

Exemple d'exécution :

```
?- faits([fleur,graine,un_cotyledone,joli]).
Yes

?- satisfait(muguet).
fleur dans la base de faits
graine dans la base de faits
phanerogame satisfait grace a r1
un_cotyledone dans la base de faits
monocotyledone satisfait grace a r3
No

?- satisfait(lilas).
fleur dans la base de faits
graine dans la base de faits
phanerogame satisfait grace a r1
un_cotyledone dans la base de faits
monocotyledone satisfait grace a r3
joli dans la base de faits
non(rhizome) satisfait grace a r15
lilas satisfait grace a r7
Yes
```

3 Chaînage mixte ou autre amélioration

Dans cette troisième partie du TP, nous vous proposons d'effectuer une amélioration du fonctionnement du moteur d'inférence. Vous pouvez choisir une amélioration qui vous intéresse et la réaliser. Si vous n'avez pas d'idée, nous vous proposons de réaliser un moteur fonctionnant en chaînage mixte selon la description ci-dessous.

On distingue deux catégories de faits :
- les faits « terminaux », qui ne figurent jamais en partie gauche d'une règle,
- les faits « observables », qui ne figurent jamais en partie droite d'une règle.

• Définir le prédicat `terminal(F)` (respectivement `observable(F)`) vrai si le fait F est terminal (resp. observable).

• Il est possible que la base de faits fournie par l'utilisateur ne permette pas en chaînage avant de conclure sur un fait terminal. Dans ce cas, on n'a pas répondu à l'utilisateur sur le végétal qu'il

observe (dans notre exemple). On veut donc lui poser des questions afin de conclure sur un fait terminal.

Pour cela, on procède en deux temps :
- On cherche une règle « presque déclenchable », c'est-à-dire dont toutes les prémisses sont vraies sauf une portant sur un fait qui est inconnu.
- On essaie de prouver ce fait inconnu en chaînage arrière. Si le chaînage arrière aboutit à essayer de prouver un fait observable inconnu, on pose une question à l'utilisateur sur ce fait observable. L'utilisateur peut répondre « oui », « non », ou « je ne sais pas ». Dès qu'on a une réponse « oui » ou « non » à une de nos questions, on peut ajouter un fait à la base de faits et relancer le moteur. Si l'on n'arrive toujours pas à conclure sur un fait terminal, il faut réitérer le processus. Attention à ne pas poser deux fois la même question à l'utilisateur.

Exemple d'exécution :

```
?- faits([fleur,graine,un_cotyledone]).
Yes
?- go.
r1
fleur graine un_cotyledone phanerogame
r3
fleur graine un_cotyledone phanerogame monocotyledone
j essaie de prouver graine_nue
Est-ce que graine_nue ? (o/n/i)
|: n.
j essaie de prouver deux_cotyledone
Est-ce que deux_cotyledone ? (o/n/i)
|: i.
j essaie de prouver rhizome
j essaie de prouver dicotyledone
phanerogame dans la base de faits
j essaie de prouver joli
Est-ce que joli ? (o/n/i)
|: o.
r15
non(graine_nue) non(rhizome) fleur graine un_cotyledone phanerogame monocotyledone joli
r7
non(graine_nue) non(rhizome) fleur graine un_cotyledone phanerogame monocotyledone joli lilas
Yes
?- raz.
Yes
?- faits([graine,graine_nue,deux_cotyledone]).
Yes
?- go.
j essaie de prouver phanerogame
Est-ce que fleur ? (o/n/i)
|: o.
graine dans la base de faits
phanerogame satisfait grace a r1
r1
graine graine_nue deux_cotyledone fleur phanerogame
r4
graine graine_nue deux_cotyledone fleur phanerogame dicotyledone
r6
graine graine_nue deux_cotyledone fleur phanerogame dicotyledone anemone
r2
graine graine_nue deux_cotyledone fleur phanerogame dicotyledone anemone sapin ombre
Yes
```