Représentation et Modélisation des connaissances
Prolog – TP2

Objectif:

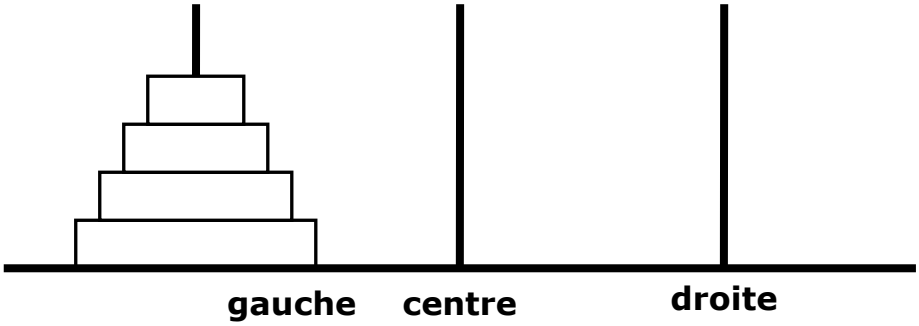
- formalisation de problèmes en Prolog
- planification
- les listes en Prolog

Exercice 1 – Listes et trace

- Testez les différents prédicats sur les listes et arithmétique vues en cours et en TD.
- Comparez quelques traces avec les arbres de résolution.

Exercice 2 – Les tours de Hanoï

Cet exercice consiste à programmer un « classique » de la récursivité : les « tours de Hanoï ». Ce jeu consiste à transférer tous les disques du poteau de gauche vers le poteau de droite, en ne déplaçant qu'un disque à la fois, et en ne plaçant jamais un disque sur un autre disque plus petit que lui :



Vous aurez besoin d'un prédicat **transferer(N,A,B,I)** où :

- **N** est le nombre de disques à transférer ;
- **A** est le poteau où les disques sont placés ;
- **B** est le poteau de destination ;
- **I** est le poteau intermédiaire.

Et d'un prédicat **deplacer(A,B)** qui ne fera qu'afficher :

- « On déplace un disque de … vers … ».

Les questions que vous devez vous poser sont :

- Que faire s'il n'y a qu'un seul disque ?
- Que faire s'il y en a davantage ?
- Quelle est la condition d'arrêt ?

Pour vous aider, disons que pour transférer N disques, il suffit de disposer d'un prédicat qui sait déplacer N-1 disques. Selon la figure, il faut transférer trois disques de la gauche vers le centre, de déplacer le grand disque de la gauche vers la droite, puis de transférer les trois disques du centre vers la droite.

Exemple d'exécution pour N=3 :

?- **transfer(3, gauche, droite, centre).**

On déplace un disque de gauche vers droite
On déplace un disque de gauche vers centre
On déplace un disque de droite vers centre
On déplace un disque de gauche vers droite
On déplace un disque de centre vers gauche
On déplace un disque de centre vers droite
On déplace un disque de gauche vers droite
Yes

Exercice 3 – Automates finis

On représente les mots sur un alphabet A par des listes, ainsi le mot « abaa » par [a, b, a, a] sur A = {a, b}. Un automate sur A est un ensemble d'états Q = {q0, q1, ...} ayant un état initial q0, un (ou plusieurs) état final, et une relation de transition donnée par des triplets (q, x, q') où x est une lettre de l'alphabet A et q, q' sont des états de Q. Un mot m est reconnu par l'automate s'il existe une suite de transitions de l'état initial à un état final au moyen des lettres de ce mot.

- Ecrire les clauses générales pour cette reconnaissance.
- Ecrire les clauses particulières décrivant l'automate à deux états q0, q1, et les transitions définies par tr(q0, a) = q1, tr(q0, b) = q1, tr(q1, a) = q0, tr(q1, b) = q1 avec q0 est l'initial et q1 est l'état final.
- Décrire l'automate reconnaissant les mots contenant le sous-mot « iie » sur l'alphabet A = {a, e, i, o, u}.

Exercice 4 – Le loup, la chèvre et le chou

Le problème *« du loup, de la chèvre et du chou »* est un exemple typique de planification par recherche dans un graphe d'états. Nous ne mettrons pas en œuvre d'heuristique car l'espace de recherche, assez restreint, ne provoque pas d'explosion combinatoire.

Un batelier doit transborder un loup, une chèvre et un chou de la rive gauche à la rive droite d'un fleuve. À chaque traversée, il peut prendre au maximum un seul des trois personnages. De plus, pour éviter des disparitions, il ne peut laisser dans surveillance ni le loup et la chèvre ensemble, ni la chèvre et le chou ensemble. En effet, en son absence, le loup mangerait la chèvre et la chèvre le chou.

On choisit d'implémenter les situations-types (nœuds du graphe) à l'aide de structures de données de type liste **[A,B,C,D]** où :

- A est la position du batelier (gauche ou droite) ;
- B est la position du loup (idem) ;
- C est la position de la chèvre (idem) ;
- D est la position du chou (idem).

La situation initiale est **[gauche,gauche,gauche,gauche]**
La situation finale est **[droite,droite,droite,droite]**

Les contraintes sont les suivantes :

- **B** doit être à l'opposé de **C** ;
- **C** doit être à l'opposé de **D**.

Les règles de transformations sont représentées à l'aide d'un prédicat **transition/3** :

```
transition('Le batelier traverse seul', [X,B,C,D], [Y,B,C,D]) :-
        oppose(X, Y), oppose(B, C), oppose(C, D).
transition('Le batelier traverse avec le choux', [X,B,C,X], [Y,B,C,Y]) :-
        oppose(X, Y), oppose(B, C).
transition('Le batelier traverse avec le loup', [X,X,C,D], [Y,Y,C,D]) :-
        oppose(X, Y), oppose(C, D).
transition('Le batelier traverse avec la chèvre', [X,B,X,D], [Y,B,Y,D]) :-
        oppose(X, Y).
```

1. Écrivez le prédicat **oppose/2** sachant que les deux valeurs sont *gauche* et *droite*.
2. Explicitez ces règles de transformation en français. Comment les tester ?
3. Écrivez un prédicat récursif **planifie(A,B,C)** qui fonctionne de la façon suivante :

    - **A** est la situation de départ ;

- **B** est la situation d'arrivée ;
- ***C*** est une liste (construite dans le membre gauche du prédicat planifie) qui doit contenir au final la liste des actions décrites dans les transitions.
- Pour planifier un parcours de **A** à **B**, il suffit de trouver une transition qui permet d'aller de **A** à **C**, et de planifier un parcours de **C** à **B**.

4. Écrivez le prédicat très simple **run/0** qui lance la planification.
5. Lancez la planification. Que remarquez-vous ?
6. Ajoutez un quatrième argument au prédicat planifie afin de construire une liste de situations déjà explorées, et prenez garde de ne pas planifier de parcours de **C** à **B** si la situation **C** fait déjà partie de cette liste !

Attention : Cette liste doit être construite dans le membre droit du prédicat **planifie/4** car elle ne dépend pas d'un but, contrairement à la liste des actions !

Indication : Utilisez le prédicat de liste **is_member/2** avec une négation.

Exercice 5 – Le zèbre

Dans la rue il y a cinq maisons. Les couleurs des maisons sont toutes différentes. Les propriétaires sont de nationalités différentes, leurs prénoms sont différents et leurs boissons préférées sont différentes. Dans chaque propriété vit un animal différent.

1. L'Anglais vit dans la maison rouge.
2. Le Suédois a un chien.
3. L'habitant de la maison verte boit du café.
4. La maison de Jean est voisine de celle du chat.
5. Le Danois boit du thé.
6. La maison verte est à droite de la maison blanche.
7. Pierre a un oiseau.
8. Paul habite la maison jaune.
9. Le buveur de lait habite la maison du milieu.
10. Le Norvégien habite la première maison à gauche.
11. Paul est voisin du cheval.
12. Jacques boit de la bière.
13. L'Allemand se prénomme Hans.
14. Le Norvégien vit près de la maison bleue.
15. La personne qui boit de l'eau est voisine de Jean.

Ecrivez un programme Prolog qui répond aux questions suivantes :

- Qui a un zèbre ?
- Qui boit du lait ?

Exercice 6 – Le singe et la banane

Autre problème classique de planification : *Le singe et la banane*…

Dans une pièce il y a un singe, une boite, et une banane suspendue au milieu du plafond. Le singe est au niveau de la porte et la boite est au niveau de la fenêtre. Pour manger la banane, le singe doit donc **marcher** de la porte vers la fenêtre, **pousser** la boite de la fenêtre vers le milieu de la pièce, **grimper** sur la boite, et prendre la banane. Lorsque le singe a pris la banane, on considère que son but est atteint.

1. Formalisez la « situation type ».
2. Écrivez une base de clauses Prolog qui contient un état initial, un état final, et les transitions nécessaires à la planification de ce problème.
3. Lancez le prédicat de planification de l'exercice n°1 sur votre base de clauses.