LOGICIEL DE MANIPULATION DE GRAPHES ETIQUETES

Devoir de PROLOG

Auteurs :

Quang Minh NGUYEN
10700214

Kevin VIMALANATHAN
11002789

Enseignants :

Joseph LE ROUX

Henry SOLDANO

Dans le but de mettre en pratique nos connaissances en PROLOG, il nous a été demandé de réaliser un petit logiciel pour la « manipulation de graphes étiquetés ».

Ce logiciel devra permettre d'entrer des requêtes associées à des motifs, de construire les sous-graphes associés aux requêtes et de construire les abstractions de ces sous-graphes.

1. Pour gérer les entrées-sorties de graphes, on appelle 2 types de prédicats : **lire/0** et **ecrire/0.**

Le prédicat lire/0 permet de charger une base de données de graphes. Ici on a simplement fixé pour consulter le fichier *'test.pl'* qui se trouve dans le même dossier, si on veut importer un fichier externe, il suffit de le mettre dans le même répertoire puis on entre la requête **consult('nom_fichier').** pour charger cette base de donnée.

Prédicat **ecrire/0** permet de transformer la base de données et créer un nouveau fichier sous un format ***resultat.dot*** de manière à pouvoir ensuite le visualiser. (Astuce, sous terminal la commande sera *dot –Tpng resultat.dot -o graphe.png*).

Le prédicat **ecrire/0** est composé des prédicats suivants : **tell/1**, **parcourGraphe/1**, **writeln/1 dessinerRelation/1** et **told/0**.

Tell/1 permet d'ouvrir ou créer le fichier *resultat.dot*.

Writeln/1 permet d'écrire une ligne de code. Ici on a mis en forme le fichier *resultat.dot* par écrire le nom du graphe G.

ParcourGraphe/1 est un prédicat qui appelle récursivement des autres prédicats : **ecrireSommet/1** afin de parcourir la liste des sommets puis écrire un par un dans le fichier *resultat.dot*.

DessinerRelation/1 fait aussi des appels récursifs de prédicat **parcourirLiaison/1** appelle lui-même **listeAjacent/2** et **ecrireLiaison/2**, pour parcourir la liste des sommets et leurs sommets adjacents pour former des liaisons (les arcs entre deux sommets).

Told/0 sert à sauvegarder et fermer le fichier *resultat.dot*.

2. Ensuite, on explore le graphe via des prédicats ci-dessous :

listeSommet/1 renvoie une liste des sommets du graphe.

listeMotif/1 renvoie une liste des motifs/étiquettes du graphe. Ici on peut utiliser soit **setof/3** soit **bagof/3** mais pas **findall/3** car avec **findall/3** va donner une liste des motifs mais avec des doublants.

sommetAdjacent/2 comme son nom du l'indique, renvoie la liste des sommets adjacent (successeurs) d'un sommet donné.

listeAdjacent/2 a la même fonctionnalité que **sommetAdjacent/2** mais avec une autre façon de codage

liaison/2 vérifie si 2 sommets sont liés.

Appartient/2 vérifie si le motif appartient à tel sommet ou à l'inverse quel motif est dans quel(s) sommet(s).

chercherMotif/2 décrit presque la même fonction mais avec une liste des motifs et liste des sommets.

composantConnex/1 vérifie si les sommets dans la liste sont bien connectés entre eux.

sousGraphe/2 c'est pour répondre à la question 2 du projet, avec une liste de motifs précise, on peut trouver une liste des composants connectés.

degreSommet/2 pour vérifie le degré d'un sommet.

3. Suite à notre incompréhension du sujet et faute de temps après l'explication tardive pour le résoudre, nous n'avons pas eu le temps de faire correctement la réponse à la 3ème question.