Structures de Données - L2, S4
TP Noté

Le but de ce TP noté c'est de manipuler des relations binaires et un ensemble de relations binaires construit inductivement. On rappelle qu'une relation binaire R sur V est un sous-ensemble de $V \times V$ pour un certain ensemble V. Les primitives qui nous intéressent dans la manipulation d'une relation binaire sont :

— `adjacents (R,x,y)` : Étant donnés $x, y \in V$, tester si $(x, y) \in R$,
— `voisins(R,x)` : Étant donné $x \in V$, retourner l'ensemble $\{y \in V \mid (x, y) \in R\}$.

Mise à part le code source, vous rendrez un rapport - de maximum 10pages - expliquant les choix faits pour les implémentations des différentes fonctionnalités demandées (et contenant les réponses aux questions de complexité). Le choix du langage n'est pas fixé, mais vous choisirez un langage que votre chargé de TP connait (à vérifier avec ce dernier). Le travail peut être fait à **2**, **3** ou **4**, mais pas plus. Vous rendrez le travail dans un fichier archivé - zip ou tar.gz - (contenant le code source et le rapport). La clarté du code ainsi que les explications seront prises en compte dans la notation. Vous expliquerez également les différentes commandes pour tester les codes [1]. Le fichier rendu ne doit pas dépassé $2Mo$.

Une relation binaire R est dite

— *indépendante* si $R = \emptyset$. La relation indépendante sur $\{1\}$ est notée $\mathbf{1}$.
— *complète* si $R = \{(x, y) \in V \times V \mid x \neq y\}$,
— *arborescente* si la cloture transitive de R est complète et $|R| = |V| - 1$.
— P_4 si $|V| = 4$ et on peut énumérer V en $\{1, 2, 3, 4\}$ tel que $R = \{(1, 2), (2, 1), (2, 3), (3, 2), (3, 4), (4, 3)\}$.
— *co-graphe* si pour tout $V' \subseteq V$ et $|V'| = 4$, alors $R \cap (V' \times V')$ n'est pas un P_4.

Exercice 4.1 *Echauffement*
On veut comparer les représentations par liste d'adjacence et matrice d'adjacence des relations binaires. Pour cela, pour chacune des deux représentations,

1. Donnez la complexité en espace de la représentation.
2. Écrivez un module qui permet de manipuler les relations binaires. On supposera que V sera toujours $\{1, \ldots, n\}$ et qu'à la création d'une relation binaire on spécifiera la taille de V.
3. Donnez la complexité en temps de chacune des primitives de base.

Exercice 4.2 *Utilisation*

1. Montrez qu'une relation binaire complète ou indépendante est toujours un co-graphe, et qu'il existe des relations binaires arborescentes qui ne sont pas des co-graphes.

1. Par exemple si c'est fait en C, on peut imaginer que vous avez un fichier qui s'appelle `test.c` dans lequel il y a un main permettant de tester les différentes implémentations à travers un menu par exemple.

2. Proposez une fonction qui teste si une relation binaire est indépendante. De même pour les autres types de relation évoqués ci-dessus.
3. Analysez la complexité en temps de vos fonctions en fonction de la représentation choisie.

Soient $R_1 \subseteq V_1 \times V_1$ et $R_2 \subseteq V_2 \times V_2$ deux relations binaires. On note $R_1 \oplus R_2$ et $R_1 \otimes R_2$ les relations binaires sur l'union disjointe $V_1 \uplus V_2$ (si $V_1 \cap V_1 \neq \emptyset$, on prend une copie de V_2 disjointe de V_1) tel que

$$R_1 \oplus R_2 := R_1 \cup R_2$$
$$R_1 \otimes R_2 := R_1 \cup R_2 \cup \{(x,y),(y,x) \mid x \in V_1, y \in V_2\}.$$

Par exemple, si $R_1 = \{(1,2),(2,1)\}$ et $R_2 = \{(4,3),(3,4)\}$, alors

$$R_1 \oplus R_2 = \{(1,2),(2,1),(4,3),(3,4)\}$$
$$R_1 \otimes R_2 = \{(1,2),(2,1),(4,3),(3,4),(1,3),(3,1),(1,4),(4,1),(2,3),(3,2),(2,4),(4,2)\}.$$

Notez que $\mathbf{1} \oplus \mathbf{1}$ est la relation indépendante sur $\{1,2\}$, et $\mathbf{1} \otimes \mathbf{1}$ est la relation compléte sur $\{1,2\}$.

Soit $\mathcal{CO}$ l'ensemble de relations binaires défini inductivement (appelé *co-arbre*) :

(B) $\mathbf{1} \in \mathcal{CO}$,
(I) $R_1 \oplus R_2 \in \mathcal{CO}$, $R_1 \otimes R_2 \in \mathcal{CO}$ si R_1 et R_2 sont dans $\mathcal{CO}$.

Par la définition de $\mathcal{CO}$, si $T \in \mathcal{CO}$, alors la relation R représentée par T est sur l'ensemble $\{1,\ldots,n\}$ où n est le nombre de feuilles de T.

Exercice 4.3 *Création de $\mathcal{CO}$*
On se propose de représenter les objets dans $\mathcal{CO}$ par des arbres binaires. Les noeuds internes seront étiquetés par les opérations $\oplus$ et $\otimes$, et les feuilles par $\mathbf{1}$. Voir Figure 1 pour un exemple. Proposez un module `co-arbre` qui permet

1. de créer la relation indépendante $\mathbf{1}$,
2. de créer le co-arbre $T_1 \oplus T_2$ à partir des co-arbres T_1 et T_2,
3. de créer le co-arbre $T_1 \otimes T_2$ à partir des co-arbres T_1 et T_2
4. de construire la matrice d'adjacence de la relation binaire représentée par T.
5. de construire la liste d'adjacence de la relation binaire représentée par T.

Pour les deux dernières questions, vous aurez besoin d'écrire une fonction qui donne un numéro entre 1 et n à chaque feuille d'un co-arbre T sachant que n est le nombre de feuilles de T.

Dans un co-arbre T le *plus petit ancêtre commun* de deux noeuds x et y, noté $NCA(x,y)$, est défini inductivement par :

$$NCA(x,y) := \begin{cases} x & \text{si } x \text{ est ancêtre de } y \\ y & \text{si } y \text{ est ancêtre de } x \\ NCA(pere(x),y). & \end{cases}$$

Exercice 4.4
Nous allons maintenant manipuler les co-arbres.

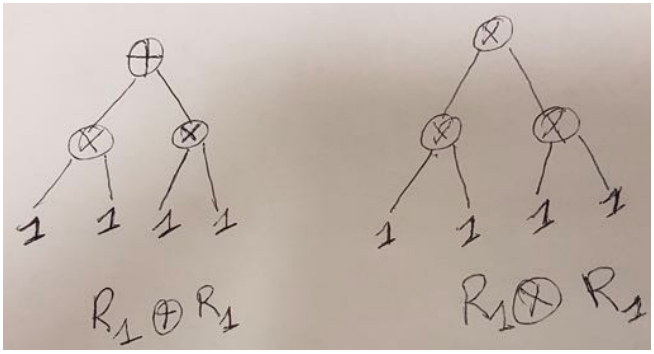

FIGURE 1 – Exemples de co-arbre représenté sous forme d'arbre.

1. Ecrire une fonction qui prend deux numéros de deux feuilles x et y, et retourne $NCA(x, y)$.
2. On peut vérifier par induction que si T est un co-arbre, et x et y sont les numéros de deux feuilles, alors $(x, y) \in R$ (avec R la relation représentée par T) ssi $NCA(x, y)$ est un noeud étiqueté $\otimes$.

   Ecrivez les fonctions `adjacents` et `voisins` lors la relation R est donnée par un co-arbre le représentant.
3. Ecrivez une fonction qui vérifie si la clôture transitive d'une relation (donnée par un co-arbre le représentant) est complète.
4. Comparer les temps de complexité entre les trois représentations, ainsi que les complexités en espace. Que pouvez-vous en déduire ?