TP 8 : Graphes – partie 2

Nous allons nous intéresser dans ce TP à quelques algorithmes « classiques » de parcours de graphes.

Pour cela, nous considérons le graphe `G=(X,U)` suivant :

- `X= A, B, C, D, E, F, G, H}`
- `U={((A,C), (A,H), (C,D), (C,E), (D,B), (D,E), (E,G), (E,F)}`

Descendants

On se propose de rechercher les descendants de chacun des sommets d'un graphe G. On dit que x_j est le descendant de x_i s'il existe dans G un chemin allant de x_i à x_j.

Tout chemin étant une succession d'arcs, il peut être construit de proche en proche au moyen des successeurs de chaque sommet. On utilisera, pour rechercher les descendants, une structure qui fonctionnera comme une file.

Cette structure possèdera <u>trois champs</u> :

- Le corps de la file
- Un pointeur d'entrée dans la file par la droite (P_e)
- Un pointeur de sortie par la gauche (P_s)

La partie active de la file, si elle est non vide, va de la position P_s jusqu'à la position P_e comprise. La partie morte (lorsqu'elle existe) va de la 1ère position jusqu'à la position (P_s-1). La condition de vie d'une telle file est ($P_s <= P_e$).

Ecrire la fonction `Descendants(G,Ix,F)` qui remplit la structure de File avec les descendants du sommet d'indice `Ix`.

Quasi-Forte Connexité Inférieure

Un graph est quasi-fortement connexe inférieurement s'il possède une racine c'est-à-dire :

- Un seul sommet de degré inférieur nul (il n'admet aucun prédécesseur)
- Et ce sommet admet tous les autres sommets comme descendants (il existe au moins un chemin allant de la racine jusqu'à n'importe quel sommet)

Définir la méthode qui permet de caractériser cette propriété et écrire la fonction `QFCI(G)` qui renvoie vrai si le graphe est quasi-fortement connexe inférieurement et faux sinon.

Enumération de chemins

Exprimer une méthode permettant de parcourir tout le graphe en utilisant une stratégie en « **profondeur d'abord** » partant de la racine `A`. Ecrire la fonction `Enumere_Chemins(G,Ix,Iy)` qui affiche tous les chemins allant de `Ix` à `Iy` dans le graphe `G`.

Détection de circuits

On ajoute au graphe `G` l'arc `(F,D)`. Modifier la procédure précédente pour prendre en compte la nouvelle situation qui évite tout circuit dans l'énumération.

Ecrire maintenant une méthode puis la procédure `Dectecte_Et_Affiche_Circuits(G,Ix)` qui affiche toute possibilité de circuit ayant une racine d'indice `Ix`.