Full explanation of my resolution

Ma solution (barbare) est la suivante : créer tous les chemins possibles et comparer les distances de ceux qui contiennent les noeuds X et Y qui doivent être visité

Dans un premier temps, on récupère les informations, le nombre de noeuds, ceux de départs et d'arrivés ainsi que la distance séparant deux noeuds.

On commence alors les chemins en partant du noeud 1. Il faut ensuite chercher les noeuds auxquels peut se rendre 'Jack' pour se rendre à l'école.

Glossaire des fonctions

maFunction : retourne un tableau de tuples : [(chemins, distances)]

La fonction 'maFunction' est appelée. Elle renvoie, après un tas de processus, un tableau contenant tous les chemins et leurs distance sous forme de tuple : (chemin, distance). On a donc : [(chemin1, distance1), (chemin2, distance2),..]

listeNoeudsSuivants : liste des noeuds suivants à partir d'un noeud donné

La fonction 'listeNoeudsSuivants' est donc appeller. Cette fonction à pour rôle de renvoyer un tableau (liste) des noeuds auxquels peut se rendre Jack en quittant le noeud qui est passé en paramètre à la fonction.

unChemin : renvoie un tableau de tous les chemins

La fonction 'unChemin' est ensuite solliciter. A labase charger de renvoyer un seul chemin, cette fonction renvoie un tableau de chemins à partir d'un début de chemin qui lui est passé en paramètre. A la fin de la fonction on a : [chemin1, chemin2, chemin3,...] Avec chemin sous la forme : [noeud_de_depart, noeud_suivant1, noeud_suivant2,..., noeud_d_arrive]

Pour arriver à fonctionner, unChemin fait appel à divers fonctions :

- lierNoeuds

Comme son nom l'indique, elle est charger de lier un noeud (ou chemin) avec ses suites potentielles. Par exemple si au noeud 2 Jack peut se rendre aux noeuds 3 et 4, et que le chemin en cours est [1, 2] par exemple, la fonction renverra [[1, 2, 3], [1, 2, 4]]

- destructionDeListe

Prend deux paramètres, deux tableaux. Le premier est la liste des chemins intermédiaires d'un noeud donné et le second la liste des chemins totals et ajoute les chemins intermédiaires au chemins totaux

calculerDistance

La fonction 'calculerDistance' comme son nom l'indique permet de calculer la distance d'un chemin et le renvoie

returnTuple

Renvoie un tuple, un couple de données. Dans notre cas il s'agit de : (chemin, distance)

Vu que ma solution m'oblige à trouver tous les chemins, je suis obligé de vérifier ceux qui contiennent les noeuds X et Y par lesquels doit passer Jack

verificationOfXY

Fait cette vérification et renvoie les chemins qui les contient

Et enfin,

minCostPath

La fonction mère demandée. Elle cherche la plus petite distance parmi les chemins qui contiennent X et Y et la renvoie

Insuffisances de la solution

- Présence de beaucoup tropnde functions, complexité du code
- Impossible de commencer par un autre noeud que 1
- Mon programme stocke également dans le tableau final les chemins intermédiares avant l'arrivé au noeud final, je suis donc obliger de faire un tri à ce niveau avant d'avoir les vrais chemins

EJAD