BACHARI Rayyan

YU Marie

Lien git : https://github.com/RayyanLeVrai/TP3_Algo_Graphe.git

Compte rendu TP3 Algo : Graphes

afficher_graphe_largeur

Chaque sommet est marqué comme non visité (couleur = 0). On utilise une structure de file pour gérer l'ordre des sommets à visiter. Le sommet de départ est enfilé après sa recherche par son étiquette.Tant que la file n'est pas vide, on défile un sommet, on le marque comme visité (couleur = 1) et on enfile ses voisins non visités. On affiche l'étiquette de chaque sommet au fur et à mesure de sa visite.

afficher_graphe_profondeur

Comme pour le parcours en largeur, les couleurs des sommets sont réinitialisées pour indiquer qu'ils n'ont pas été visités. Une pile est utilisée pour gérer les sommets à traiter, commençant par le sommet de départ. Tant que la pile n'est pas vide, on dépilote un sommet, et si celui-ci n'a pas été visité, on le marque comme tel et on empile tous ses voisins non visités. L'étiquette de chaque sommet est affichée lorsqu'il est visité.

dijkstra

Tous les sommets sont initialisés avec une distance infinie, sauf le sommet source qui a une distance de 0. Une file à priorité est utilisée pour toujours traiter le sommet ayant la plus petite distance provisoire. Pour chaque sommet traité, on met à jour les distances des sommets adjacents si un chemin plus court est trouvé via ce sommet. Après avoir traité tous les sommets, les distances finales sont disponibles et peuvent être utilisées pour déterminer les plus courts chemins.

elementaire

Les sommets du chemin sont marqués comme non visités. On parcourt les sommets du chemin, marquant chaque sommet comme visité. Si on rencontre un sommet déjà visité, le chemin n'est pas élémentaire. Les couleurs des sommets sont remises à zéro pour ne pas affecter les opérations futures.

simple

Les arcs entre les sommets consécutifs du chemin sont stockés dans un tableau. On compare chaque arc avec les autres pour détecter les répétitions. Si une répétition est trouvée, le chemin n'est pas simple.

eulerien

On liste tous les arcs du graphe puis on les compare pour vérifier que tous les arcs du graphe sont inclus dans le chemin sans omissions ni répétitions.

hamiltonien

Le chemin doit contenir tous les sommets du graphe. On assure que chaque sommet du graphe est présent dans le chemin.

graphe_eulerien

On calcule le degrés entrants et sortants. On les compare pour chaque sommet, car les arcs entrants doivent égaler les arcs sortants. Puis on vérifie que le graphe est connecté par un parcours.

graphe_hamiltonien

On vérifie que pour chaque paire de sommets non adjacents, la somme de leurs degrés doit être au moins égale au nombre total de sommets.

distance

On initialise tous les sommets sont à une distance infinie, sauf le sommet de départ. On apllique Disjktra en mettant à jour les distances des sommets adjacents jusqu'à traiter tous les sommets.

excentricite

On applique Dijsktra à partir du sommet ciblé pour obtenir la distance à tous les autres sommets. L'excentricité est la distance maximale obtenue.

Fonction diametre

On Calcule l'excentricité pour chaque sommet puis on cherche le maximum : la plus grande de ces valeurs est le diamètre.