Université virtuelle du Burkina Faso (UV-BF)

Option : Master Fouille de données et Intelligence Artificielle

Projet Théorie des graphes

Azise Oumar Diallo, Ph.D.
azise.oumar.diallo@gmail.com
22/05/2025

Résumé du projet

L'objectif de ce projet est de résoudre un problème concret en utilisant la modélisation par graphes. Vous devez :

1. Formuler le problème en termes de graphes (définir précisément les questions à résoudre).
2. Apporter des réponses en vous appuyant sur les notions fondamentales de la théorie des graphes : définitions, propriétés, théorèmes, et algorithmes adaptés.

Mise en œuvre technique : Pour chaque problème, une **implémentation algorithmique** sera développée dans le langage de programmation de votre choix. Chaque programme devra être accompagné d'un **exemple de test minimal** illustrant son fonctionnement.

Thématiques abordées : Chaque sujet correspond à un thème classique de la théorie des graphes : connexité, plus court chemin, arbre couvrant minimal.

Organisation du travail :

- Le projet se réalise en groupes prédéfinis (entre 4 et 5 personnes, réparties en 8 groupes).
- **Répartition des sujets :**
  - Deux groupes traiteront le même sujet.

- L'attribution des sujets a été réalisée aléatoirement par l'enseignant (voir document Google Sheets partagé).

— **Échéance et rendu :**
- Date limite : le **22 juin 2025 avant 23h59**.
- Contenu à déposer sur Moodle : Un fichier ZIP au nom du groupe et contenant :
  - Le **code source** du projet.
  - Un fichier `README` expliquant comment utiliser votre programme.
  - Un **rapport synthétique** (5 pages maximum) décrivant votre travail.
  - Le support de présentation (PPT, PDF...)
- **Dans la semaine du 23 juin 2025** : Présentation des projets en raison d'une durée de 10/15 min par groupe.

Format du fichier d'entrée du graphe

Votre programme devra lire un graphe orienté, pondéré, avec d'éventuelles boucles, encodé sous forme de fichier texte.

— Le graphe est décrit dans un fichier texte de $n+1$ lignes, où n est le nombre de sommets :
- La première ligne contient un entier n, le nombre de sommets.
- Les n lignes suivantes décrivent la matrice d'adjacence pondérée : la ligne i contient n entiers séparés par des espaces, où la j-ème valeur indique le **poids de l'arête** allant de i vers j. Une valeur 0 signifie qu'il n'y a pas d'arête.

— Les poids sont supposés être des **entiers**.

— Exemple de fichier correspondant au graphe de la Figure 1 :

```
6
0  0  0  2  0  0
20  0  8  0  7  0
2  0  0  7  0  0
0  11  0  0  0  0
0  0  5  0  4  0
0  0  0  0  0  0
```

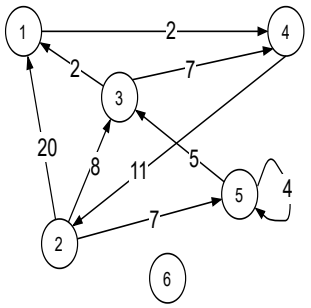

Figure 1 – Exemple de graphe correspondant au fichier ci-dessus

Conseils et recommandations

- **Visualisation des résultats :** Pensez à représenter graphiquement vos résultats (ex. : arbres couvrants, chemins trouvés). N'hésitez pas à faire preuve de créativité dans vos affichages.
- **Prétraitement :** dans certains cas, il peut être utile d'**éliminer les boucles** avant d'appliquer un algorithme.
- **Vérification des hypothèses :**
  - Identifiez les hypothèses nécessaires au bon fonctionnement de vos algorithmes.
  - Que se passe-t-il si ces hypothèses ne sont pas respectées ?
  - Tentez d'adapter votre programme pour vérifier automatiquement ces hypothèses ou, mieux encore, le rendre robuste à des cas plus généraux.

1 Sujet 1 : Connexité

1.1 Algernon et le labyrinthe

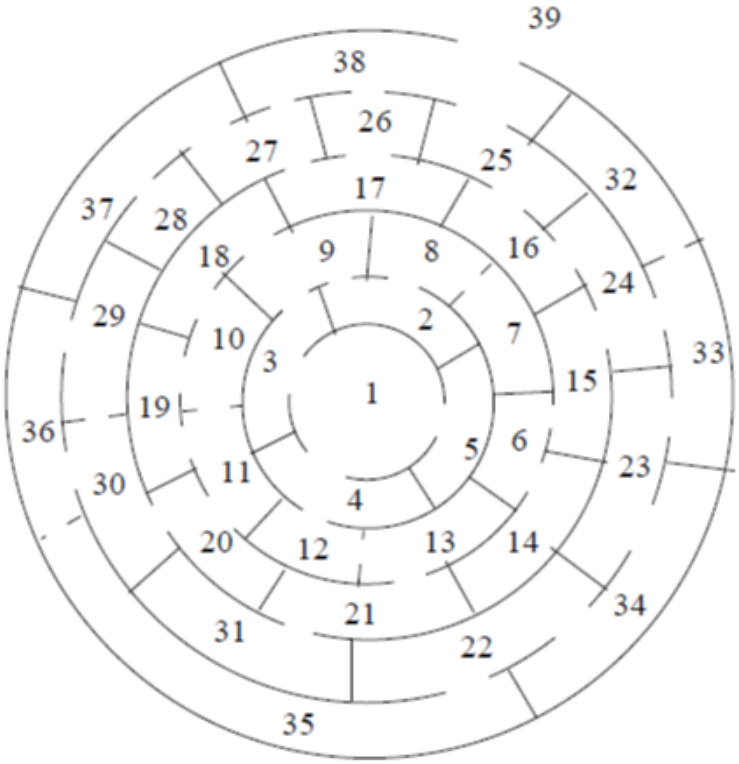

FIGURE 2 – Algernon et le labyrinthe

Algernon, petite souris blanche de laboratoire, apprend à sortir de labyrinthes. Les chercheurs cherchent donc à lui construire des labyrinthes selon plusieurs critères :

1. Algernon est placée au centre du labyrinthe. Est-ce qu'elle pourra en sortir ? Est-ce qu'elle a plusieurs solutions ?
2. Pour les essais suivants, Algernon est placée à un nouvel endroit du labyrinthe à chaque fois. Peut-elle sortir du labyrinthe à tous les coups ?
3. Algernon est maline, elle a réussi à trouver un chemin depuis le centre du nouveau labyrinthe et s'en souvient. Les chercheurs ferment donc toutes les portes par lesquelles elle est passée. Peut-elle encore sortir du labyrinthe depuis le même point de départ ?

1.2 Circulation à sens unique

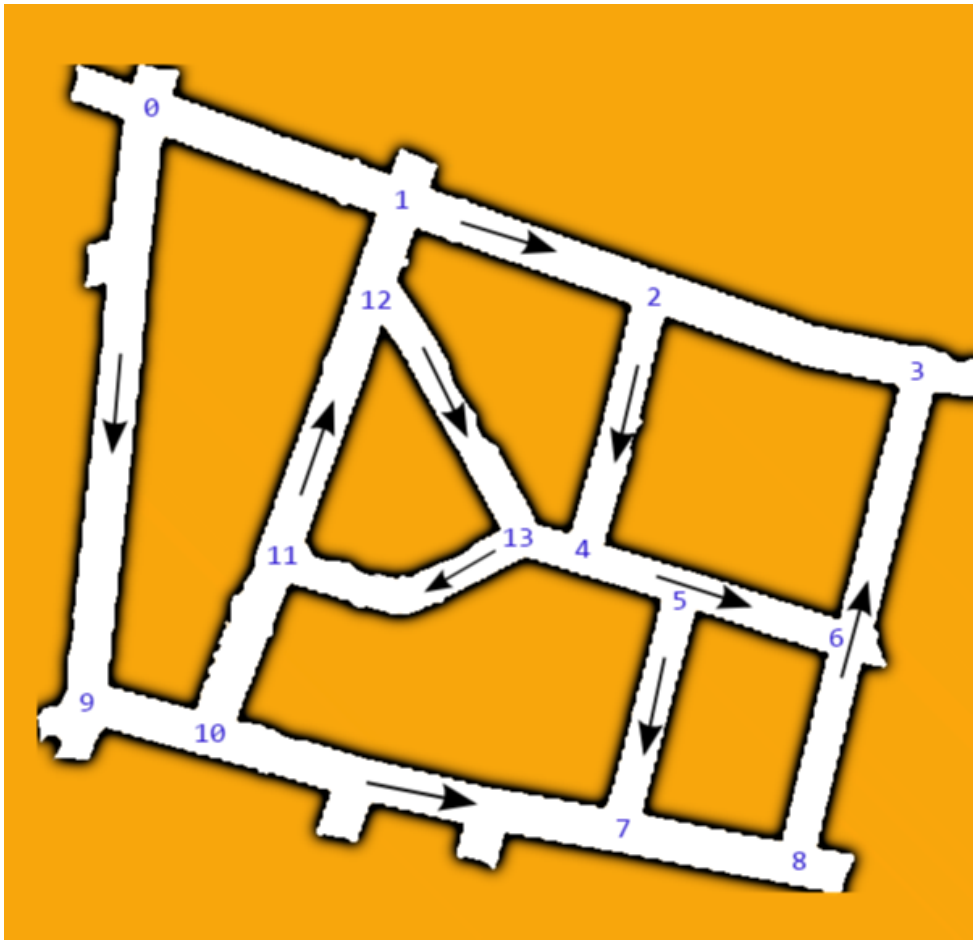

Figure 3 – Circulation à sens unique

Monsieur **Voie** est chargé d'installer des sens uniques dans tout un quartier afin que la circulation soit plus fluide (les rues étant trop étroites, le stationnement et la circulation posent un problème). Voici le plan avec le sens de circulation de chaque rue (Figure 3). M. **Voie** se demande si on peut vraiment se déplacer partout dans le quartier sans en sortir.

1. Graphe fortement connexe
   (a) Modéliser le plan sous forme de graphe.

(b) Le plan de M. Voie est-il fortement connexe ?

(c) Quelles sont les rues qui doivent être mises en double sens ?

(d) Peut-elle sortir du labyrinthe à tous les coups ?

2. Méthode 1 : deux parcours

(a) Qu'observe-t-on lorsque l'on considère le graphe transposé et la propriété que le graphe initial devrait respecter ?

(b) Donner le principe de construction des composantes fortement connexes utilisant un double parcours et écrire la fonction correspondante.

(c) Et si on veut uniquement les sommets appartenant à la même composante qu'un sommet donné ?

3. Méthode 2 : Tarjan

(a) Appliquer l'algorithme de "Tarjan" pour trouver les composantes du graphe de la Figure 3.

(b) Écrire la fonction qui détermine les composantes fortement connexes d'un graphe orienté.

(c) Et si on veut uniquement les sommets appartenant à la même composante qu'un sommet donné ?

2 Sujet 2 : Plus court chemin

M. **Clément B.** veut se faire construire une résidence secondaire à Ouagadougou. Dans le tableau ci-dessous la liste des tâches à effectuer. Pour chaque tâche sont indiqués son libellé, sa durée et ses prédécesseurs (les tâches qui doivent être terminées avant la réalisation de la tâche courante). Les tâches peuvent être réalisées simultanément. Le but est ici d'aider C. B. à réaliser son projet le plus rapidement possible.

Code tâche	Libellé	Durée (semaines)	Prédécesseurs
0	Obtention du permis de construire	2	-
1	Maçonnerie	7	-
2	Charpente de la toiture	3	0, 1
3	Toiture	1	2
4	Plomberie et électricité	8	0, 1
5	Façade	2	3, 4
6	Fenêtres	1	3, 4
7	Aménagement du jardin	1	3, 4
8	Plafonds	3	6
9	Peintures	2	8
10	Emménagement	1	5, 9

TABLE 1 – La maison de C.B.

1. Modéliser le projet sous forme de graphe.
2. Quelle est la durée minimale du projet donné en énoncé ? A quel problème de graphe s'apparente la recherche de la durée minimale du projet ?
3. Adapter le principe de Bellman pour résoudre ce problème. Appliquer au graphe de la question 1.
4. La date au plus tôt de la tâche i est la date à laquelle la tâche peut commencer au plus tôt. La date au plus tard de la tâche i est la date au plus tard où la tâche peut commencer sans entraîner de retard sur le projet. Si la date au plus tôt et la date au plus tard d'une tâche i sont identiques alors cette tâche est dite critique. Comment peut-on obtenir ces dates ?
5. Calculer les dates au plus tôt et au plus tard du projet de M. B. Quelles sont les tâches critiques ?
6. Code : L'entrée est une liste des tâches représentant un projet :
   — Les n tâches sont numérotées de 0 à $n-1$.
   — Pour chaque tâche t, à la position t dans la liste :
      — Sa durée ;
      — la liste des tâches qui doivent être terminées avant t.

La fonction à écrire $project(L)$ doit donner :
— Une liste, contenant pour chaque tâche t :
   — Sa date au plus tôt ;
   — "slack" : le retard maximum que peut prendre t sans entraîner de retard sur la suite.
— La durée minimum pour réaliser le projet.

3 Sujet 3 : Arbre couvrant minimal

Soit le réseau de routeur d'un fournisseur d'accès d'Internet. La compagnie veut minimiser le coût global du réseau. La Figure 4 représente le réseau avec le coût pour chaque liaison.

1. Quel type de graphes permet d'avoir le nombre minimum de liaisons tout en gardant tous les routeurs reliés entre eux ?
2. On intègre ici la notion de coût : Pour répondre au problème, quelles sont les contraintes qu'il faut satisfaire ?
3. Proposer une solution pour le graphe de la Figure 4 (liste des arêtes à conserver).

Nous allons maintenant essayer de construire un algorithme qui trouve une solution. L'approche la plus classique est incrémentale : on ajoute progressivement des arêtes à la solution (en partant d'un ensemble vide). À chaque étape de l'algorithme, on obtient un ensemble d'arêtes qui préservent en partie les propriétés finales attendues. En théorie, un algorithme incrémental peut redémarrer d'une solution partielle intermédiaire.

Algorithme de Prim

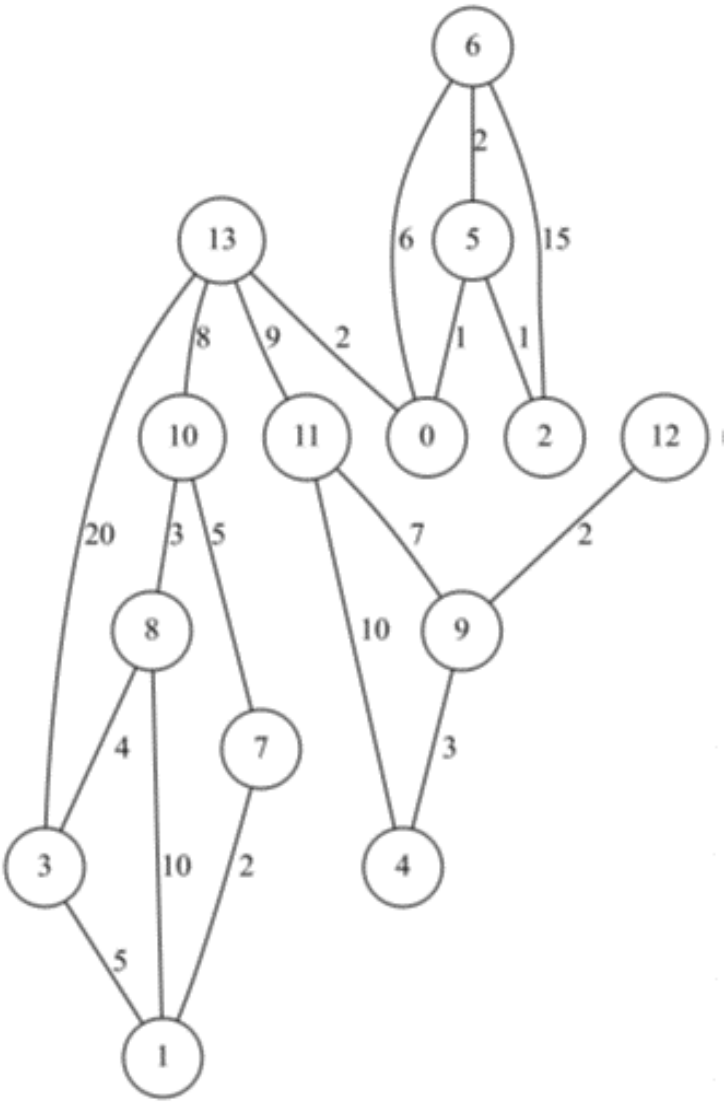

Figure 4 – Réseau de routeurs

1. Quelle propriété est maintenue à chaque itération de l'algorithme de Prim ?
2. Le principe :
   (a) On suppose déjà construit une partie de la solution, comment choisir une arête du graphe d'origine à ajouter à la solution pour que le sous-graphe partiel ainsi formé reste connexe et sans cycle ?
   (b) On veut maintenant garantir que lorsque l'algorithme termine, la solution construite soit minimale. Comment choisir le sommet de départ de la prochaine arête à ajouter ?
   (c) En déduire le principe de l'algorithme de Prim.
3. Appliquer ce principe au graphe de la Figure 4.
4. Implémentation :
   (a) Comment représenter simplement la liste des arêtes à conserver ?

(b) L'algorithme que l'on vient de décrire ressemble à un algorithme de recherche de plus court chemin, lequel ? En déduire la structure de données la plus efficace pour choisir les sommets à traiter.

(c) Écrire la fonction Prim(G) qui retourne un ARPM (un graphe) de G.

Algorithme de Kruskal

Dans un premier temps, on considère que l'on travaille avec un ensemble d'arêtes déjà trié.

1. Quelle propriété est maintenue à chaque itération de l'algorithme de Kruskal ?
2. Le principe :
   (a) Soit A l'ensemble des arêtes du graphe et T une solution partielle minimale.
   (b) Comment choisir une arête e de l'ensemble $A - T$ à ajouter à T pour que l'ensemble $T \in \{e\}$ reste minimal ?
   (c) On considère maintenant le problème des cycles : comment déterminer que la nouvelle arête n'introduit pas de cycle ?
   (d) En déduire le principe de l'algorithme de Kruskal.
   (e) Appliquer ce principe au graphe de la Figure 4.
3. Appliquer ce principe au graphe de la Figure 4.
4. Implémentation :
   (a) L'algorithme nécessite d'avoir l'ensemble des arêtes trié, mais dans notre cas nous ne disposons pas vraiment de cet ensemble. Quelle structure de données pourrait être utilisée pour construire cet ensemble et récupérer efficacement l'arête minimale à chaque étape.
   (b) Écrire la fonction $Kruskal(G)$ qui retourne un ARPM (un graphe) de G.