GAMBIRASIO Florian, DEHAINE Alexandre, PINTO Cristobal. Groupe C6

SAE S2.02 -- Rapport pour la ressource Graphes

Version 1: un seul moyen de transport

Présentation d'un exemple

Présenter un exemple concret de problème (données complètes pour la plateforme avec tous les moyens de transport, préférences de l'utilisateur qui comprennent le moyen de transport choisi, le critère d'optimisation, et nombre d'itinéraires demandés).
Donner la solution du problème du point de vue de l'utilisatrice, càd quels sont les itinéraires possibles, quels sont les meilleurs itinéraires et pourquoi.
Pour l'instant on ne parle pas de graphes; on peut éventuellement faire des schémas.

Modèle pour l'exemple

Donner le graphe modélisant l'exemple ci-dessus.
Donner la solution du problème (càd les meilleurs itinéraires) en tant que chemins dans le graphe.

Modélisation pour la Version 1 dans le cas général

Expliquer de manière abstraite comment, étant donné un problème de recherche d'itinéraire (plateforme avec tous types de lignes, moyen de transport choisi, critère d'optimisation, nombre d'itinéraires demandés) on peut construire un graphe permettant de résoudre le problème de recherche d'itinéraire. C'est à dire:
- quels sont les sommets du graphe par rapport aux données du problème,
- quelles sont ses arêtes, par rapport aux données du problème,
- comment sont définis les poids des arêtes,
- quel algorithme sur les graphes permet de résoudre le problème d'itinéraire (nom de l'algorithme, arguments).

Utiliser un vocabulaire précis sur les graphes.

Les sommets du graphe représentent les villes du réseau, tandis que les arêtes correspondent aux trajets allant d'un sommet A à un sommet B, avec comme poids la modalité choisie par l'utilisateur (prix, pollution, temps de trajet, etc.).
À partir de ce graphe, on peut alors déterminer le ou les trajets idéaux grâce à un algorithme des plus courts chemins, prenant en paramètre la ville de départ, la ville d'arrivée et les modalités choisies afin de générer le graphe nécessaire à la résolution du problème.

Implémentation de la Version 1

Écrire une classe de test qui reprend l'exemple, définit toutes les données de la plateforme, construit le graphe et calcule la solution.
Votre classe peut utiliser des assertions (test unitaire) ou bien afficher la solution.

*Donner ici le **nom complet de la classe**, **la date et l'identifiant du commit à regarder** et un **lien vers la page de cette classe sur gitlab qui correspond au bon commit**.*

On insiste sur l'importance de spécifier le commit. En effet, quand vous commencerez la Version 2, le code utilisé pour le test de la Version 1 sera modifié. Il se peut que vous n'ayez pas le temps de finaliser la Version 2 et vous retrouver avec un code qui ne marche pas même pour la Version 1. C'est pourquoi il est important de rédiger le rapport au fur et à mesure et de donner ici un lien vers la version de votre code qui marche pour la Version 1 du projet.

Nom de la classe test : **PlatefomeTest** (https://gitlab.univ-lille.fr/sae2.01-2.02/2024/C6/-/blob/main/src/PlateformeTest.java) qui test certaines fonction de la classe Plateforme et **UsePlateforme (https://gitlab.univ-lille.fr/sae2.01-2.02/2024/C6/-/blob/main/src/UsePlateforme.java)** qui test l'exécution de l'application avec les donnée en exemple.

Version 2 : multimodalité et prise en compte des correspondances

Cette section explique la solution pour la Version 2 du projet.

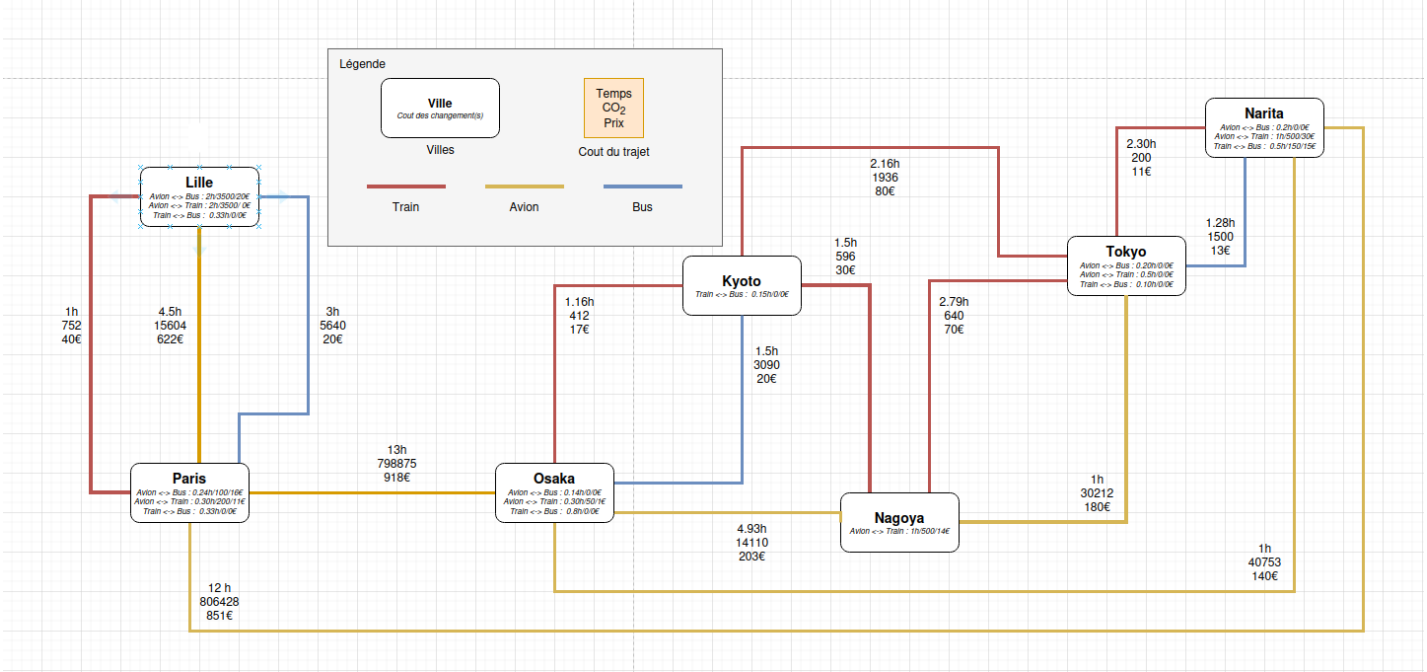
Présentation d'un exemple

Présenter un exemple concret (plateforme, coûts de correspondance, critère d'optimalité).
Donner la solution du problème du point de vue de l'utilisatrice (quels sont les itinéraires possibles, lesquels sont optimaux et pourquoi).
Il est possible d'utiliser le même exemple que pour la Version 1 ou le modifier si pertinent.

Nous avons opté pour un graphe correspondant en un lien entre la France et le Japon. Il relient les villes Lille, Paris, Osaka, Kyoto, Nagoya, Tokyo et Narita. C'est un exemple intéressant car certaines villes ne sont relié que par 1 ou 2 moyens de transport.

Modèle pour l'exemple

Donner le graphe modélisant l'exemple ci-dessus.
Donner la solution du problème (càd les meilleurs itinéraires) en tant que chemins dans le graphe.



Modélisation pour la Version 2 dans le cas général

Mêmes questions que pour la section correspondante de la Version 1, mais cette fois-ci les données d'entrée contiennent aussi des coûts de correspondance.
Vous pouvez expliquer l'entièreté de la solution pour la Version 2, ou bien indiquer ***clairement*** *les différences par rapport à la solution proposée pour la Version 1.*

Implémentation de la Version 2

Écrire une classe de test qui reprend l'exemple, définit toutes les données de la plateforme, construit le graphe et calcule la solution.
Votre classe peut utiliser des assertions (test unitaire) ou bien afficher la solution.
Donner ici le ***nom complet de la classe****,* ***la date et l'identifiant du commit à regarder*** *et un* ***lien vers la page de cette classe sur gitlab qui correspond au bon commit****.*
En particulier, il peut s'agir de la même classe que celle donnée pour la Version 1, mais un commit différent.

Nom de la classe test : **PlatefomeTest** (https://gitlab.univ-lille.fr/sae2.01-2.02/2024/C6/-/blob/main/src/PlateformeTest.java) qui test certaines fonction de la classe Plateforme et **UsePlateforme (https://gitlab.univ-lille.fr/sae2.01-2.02/2024/C6/-/blob/main/src/UsePlateforme.java)** qui test l'exécution de l'application avec les donnée en exemple.

Version 3 : optimisation multi-critères

Suivre le même plan que pour les deux autres sections.
Pour l'exemple, veillez à spécifier toutes les données des problèmes. En particulier, on ajoute ici l'expression des préférences d'optimisation de l'utilisatrice.
Comme précédemment, il est possible d'utiliser le même exemple et simplement l'enrichir.

Fin du rapport

Barème sur 30 pts

Toute question sur le barème est à adresser à iovka.boneva@univ-lille.fr

- Rapport non rendu à temps -> note 0
- **(7, décomposé comme suit)** Divers
  - **(1,5)** Respect de la structure du rapport
  - **(1,5)** Section Version 1 rendue pour le 18/05/2024. Cette version peut contenir les parties en italique.
  - **(1,5)** Section Version 2 rendue pour le 08/06/2024. Cette version peut contenir les parties en italique.
  - **(1)** Utilisation de vocabulaire précis sur les graphes (termes vu en cours, noms des algorithmes, etc.)
  - **(1,5)** Style d'écriture fluide et compréhensible

- **(8, décomposé comme suit)** Solution pour la Version 1
  - **(2)** Exemple pertinent (illustre tous les aspects du problème) et lisible (en particulier, ni trop grand ni trop petit, bien présenté)
  - **(4)** Le modèle de l'exemple permet de trouver la solution sur l'exemple. La modélisation pour le cas général permet de résoudre le problème posé
  - **(2)** L'implémentation de l'exemple est correcte et fonctionnelle

- **(6, décomposé comme suit)** Solution pour la Version 2
  - **(1)** Exemple pertinent
  - **(4)** le modèle de l'exemple permet de trouver la solution sur l'exemple. La modélisation pour le cas général permet de résoudre le problème posé
  - **(1)** L'implémentation de l'exemple est correcte et fonctionnelle

- **(3)** Qualité de la description de la solution (concerne les sections "Modèlisation dans le cas général" pour les Versions 1 et 2):
  - La modélisation pour le cas général est décrite de manière abstraite mais précise et complète. Pour vous donner une idée, un·e étudiant·e de BUT qui a validé les ressources Graphes et Dev devrait être en mesure d'implémenter votre solution d'après la description que vous en faites, sans avoir à trop réfléchir.

- **(6)** Solution pour la Version 3: mêmes critères que pour la Version 2