Programmation efficace

TP6 : Trouver des équipes !

Résumé du problème

Il y a n joueurs et un arbitre. On cherche à répartir efficacement les n joueurs dans p équipes.

Les joueurs d'une même équipe doivent communiquer entre eux : chaque joueur de l'équipe doit envoyer un message à chaque autre joueur de son équipe.

Le réseau de communication est un graphe orienté pondéré .

Le but est de répartir les joueurs dans les p équipes afin d'avoir un coût de communication globale minimal.

Pour résoudre le problème

Langage choisi : JAVA

- Pour le confort avec le langage
- Pour l'orienté objet et la facilité de traiter le problème via des objets

Utilisation de Dijkstra

- Graphe orienté pondéré
- Recherche du coût de communication global minimal (= recherche d'un chemin avec arcs de poids minimaux)

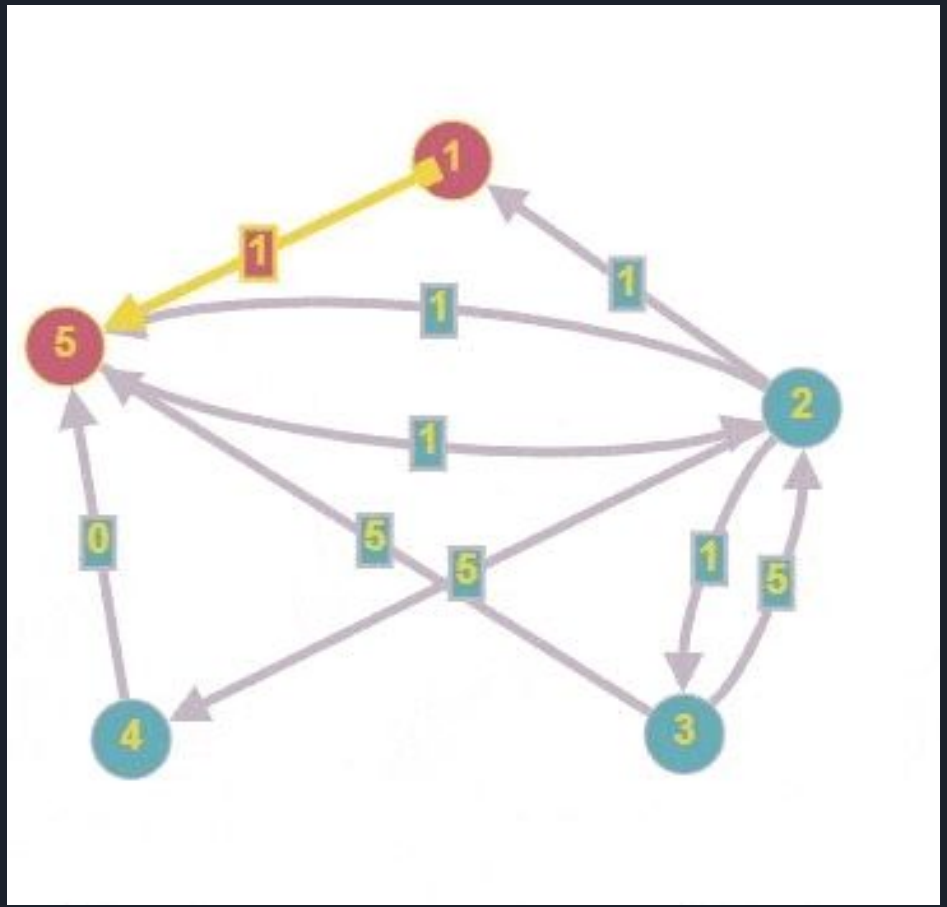
L’algorithme

Création des sommets

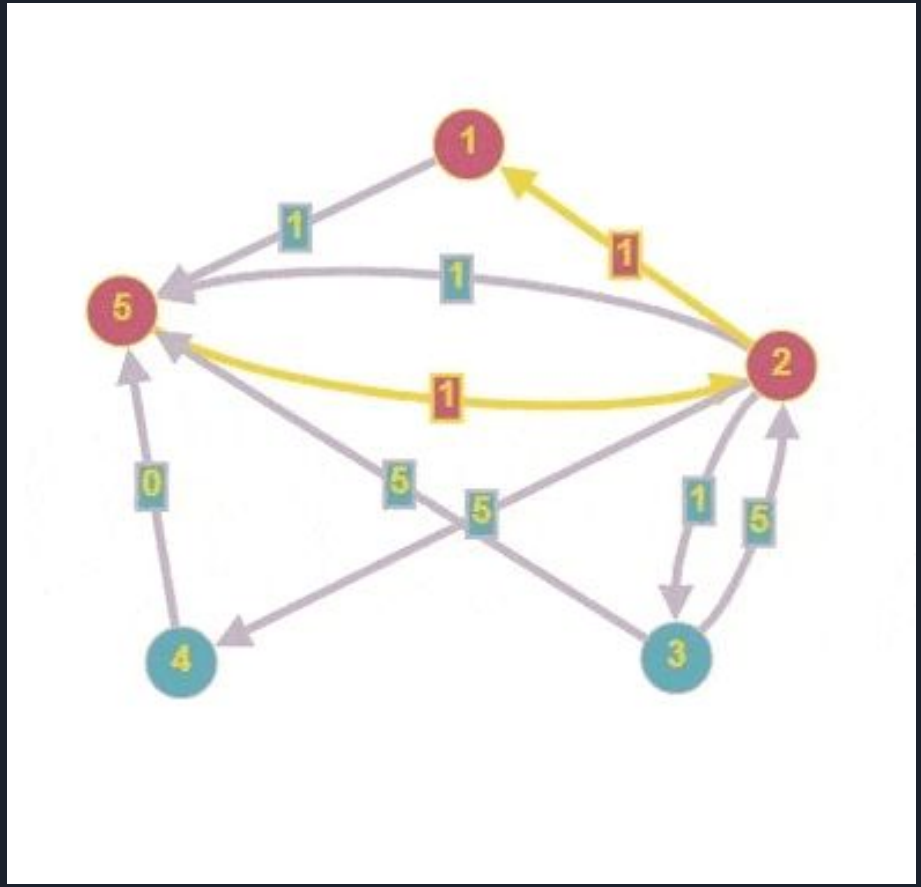
Chaque sommet possède :

- un id
- un boolean qui distingue les joueurs de l'arbitre
- un liste de voisins sous forme de map < idVoisin, distance>

Dans un premier temps on calcule grâce à Dijkstra la distance de chaque joueur vers l'arbitre :



Puis de l’arbitre à chaque joueur:



On obtient deux tableaux:

Arbitre vers joueurs

joueur 1	joueur 2	joueur 3	joueur 4	arbitre 5
2	**1**	**2**	**6**	**0**

Joueurs vers arbitre

joueur 1	joueur 2	joueur 3	joueur 4	arbitre 5
1	**1**	**5**	**0**	**0**

Répartition des joueurs

Plusieurs idées envisagées :

1) Placement aléatoire
2) Tester toutes les combinaisons
3) En fonction du coût d'ajout d'un joueur dans une équipe
4) En fonction du coût d'ajout d'un joueur dans une équipe avec ordre d'ajout aléatoire

Répartition des joueurs

Plusieurs idées envisagées :

- Placement aléatoire
- Tester toutes les combinaisons

3) En fonction du coût d'ajout d'un joueur dans une équipe
4) En fonction du coût d'ajout d'un joueur dans une équipe avec ordre d'ajout aléatoire

Quelques résultats des deux variantes

Répartition / Fichier	En fonction du coût d'ajout d'un joueur dans une équipe (3)	En fonction du coût d'ajout d'un joueur dans une équipe avec ordre d'ajout aléatoire (4)
Exemple sujet: exemple 1 exemple 2	**13** **26**	**13** **24**
04.in (coût de chaque arête à 10000)	**24300078713504**	**-7489261602832**
09.in (plus de sommets que de joueurs)	**147058911306**	**147058667951**
30.in (coût de 0 ou 1)	**305036**	**305111**

Au final

Sur les 53 fichiers de tests:

- 7 fichiers avec résultats identiques pour les deux répartitions
- 2 fichiers pour lesquelles la répartition (3) est meilleure
- 44 fichiers pour lesquelles la répartition (4) est meilleure

La répartition en fonction du coût d'ajout d'un joueur dans une équipe avec ordre d'ajout aléatoire semble être un bon choix