clement.rouvroy@ens.psl.eu

Diriger la foule grace aux graphes

Consignes Le candidat respectera le langage imposé (C) et ne devra pas utiliser de librairie hors-programme. Il est invité à faire des schémas clairs et précis de ses algorithmes lors des appels au correcteur, c'est à la fois un gain de temps pour les deux et une manière de montrer qu'il a compris ce qu'il manipule. Les preuves écrites doivent être formelles, sauf si la consigne précise que ce n'est pas nécessaire, la rigueur sera évaluée. L'examinateur ne débugera pas votre code, en revanche en cas de doute ("ai-je le droit à telle ou telle librairie ?", "je suis sortie de la VM, comment y revenir ?", "ai-je le droit à une indication pour cette question ?") n'hésitez pas à poser votre question. Elle ne vous dévalorisera pas, si la réponse peut vous retirer des points, l'examinateur vous demandera avant de vous donner la réponse si vous l'acceptez (si vous n'avez pas de chance, le jour de l'oral il vous retirera les points rien que pour avoir posé la question, par exemple si vous demandez "comment trier une liste en $O(n\log(n))$?" ou un autre résultat classique du programme, ça sera sûrement retenu contre vous). Enfin, si l'examinateur n'est pas à votre portée quand vous avez besoin d'une aide / d'une question oral à donner, gardez le bras levé et lisez la suite, ne restez jamais passif.

Introduction du sujet Ce sujet traite des réseaux de flots, une branche de l'informatique issue de la théorie des graphes. Ce sujet est décomposé en trois parties :

- Une introduction aux flots et quelques questions de prises en main.
- Une présentation de l'algorithme de Ford-Fulkerson et une preuve de sa correction (à ne traiter qu'à la fin de la partie 4)
- Une application de l'algorithme de Ford-Fulkerson dans le cas des mouvements de foule

I Pendant que ça charge

Question 1 de cours

1. Définir ordre bien fondé.
2. Expliquer le principe de l'algorithme de Dijkstra.
3. Donnez la complexité de l'algorithme de Floyd-Warshal.

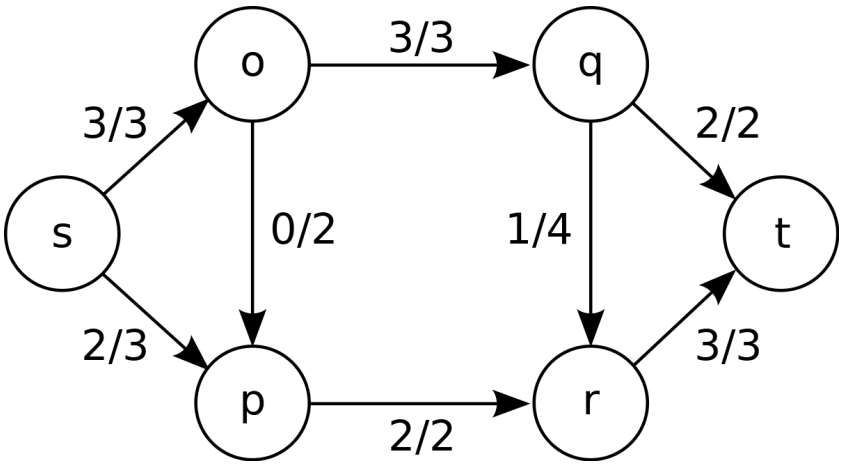
II Introduction

Définition 2.1 Réseau de flots

Un réseau de flots est un graphe (V, E) orienté étiqueté (par deux valeurs) tel que :

- Il existe un sommet $s \in V$ et un sommet $t \in V$ tels qu'il n'existe pas d'arête de la forme (i, s) ou (t, i). On dit que s est la source et t le puits.
- Chaque arête a deux étiquettes : une capacité (dans $\mathbb{N}$, notée c) et un flot (dans $\mathbb{N}$ aussi, noté f)
- Deux conditions soient respectées :
  - Contrainte de capacité : $\forall (u, v) \in E, f(u, v) \leq c(u, v)$

▸ Conservation du flot : $\forall u \in V, \sum_{w \in V} f(u,w) = \sum_{w \in V} f(w,u)$



- Version imagée Fondamentalement, un réseau de flots représente une quantité qui passe dans un réseau qui est limité (comme des tuyaux d'eau par exemple). On s'intéresse ainsi à la quantité totale qui rentre dans le réseau, ie la quantité que le réseau doit gérer, c'est cette valeur qu'on cherche à optimiser (la quantité d'eau par seconde dans des tuyaux, d'humains dans une foule, et j'en passe)

Question 2 à l'écrit

Montrez (informellement) que la quantité rentrant dans le réseau (sortant de s) est égale à la quantité quittant le réseau (rentrant dans t)

Pour la suite du TP, récupérez le fichier `flots.c`. Il contient des définitions et le squelette du TP, l'intégralité de votre code devra être fait dedans (pensez à en faire une copie en cas de mauvaise manipulation). Les sommets `0` et `nb_sommets - 1` sont respectivement la source et le puits.

Question 3 code

Implémentez les fonctions `initialize_graphe`, `free_graphe`, `ajoute_arete` et `modifie_flot`.

Question 4 code

Implémentez la fonction `quantite_circulante` qui renvoie la valeur circulant de la source vers le puit dans le réseau passé en argument (noté $val(f)$ pour valeur du flot).

Question 5 code

Implémentez un parcours en profondeur (fonction `dfs`)

III Algorithme de Ford-Fulkerson

Le but d'un flot est d'optimsier (ie maximiser) le résultat de votre fonction `quantite_circulante`, on pourraît essayer de le faire à la main mais ce n'est pas très efficace, surtout quand les réseaux deviennent assez grands (car il y a beaucoup de possibilités). Heureusement, nous possédons un algorithme très efficace dont la preuve est accessible (et vous sera demandé si vous finissez le sujet) : l'algorithme de Ford-Fulkerson.

Ne pas traiter les questions en rouge avant d'avoir fini la partie IV: Elles sont faisables mais longues et trop théoriques pour un TP. Cependant retenez bien les définitions et l'algorithme proposée à la fin, car il vous est demandé de le coder.

Question 6 à l'écrit

Dîtes si le réseau de flots proposé en exemple peut-être augmenté (avoir une plus grande valeur de flot). Si oui, proposer un nouveau flot augmenté, sinon justifiez brièvement.

Définition 3.1 Coupe d'un graphe

On appelle coupe d'un graphe une partition de ses sommets en deux ensembles distincts

Définition 3.2 Flot d'une coupe

Soit une coupe S, T d'un graphe (V, E), on note son flot $f_{\frac{S}{T}}$ et on a :

$$f_{S/T} = \sum_{i \in S, j \in T, (i,j) \in E} f_{i,j} - \sum_{j \in T, i \in S, (j,i) \in E} f_{j,i}$$

Question 7 à l'écrit

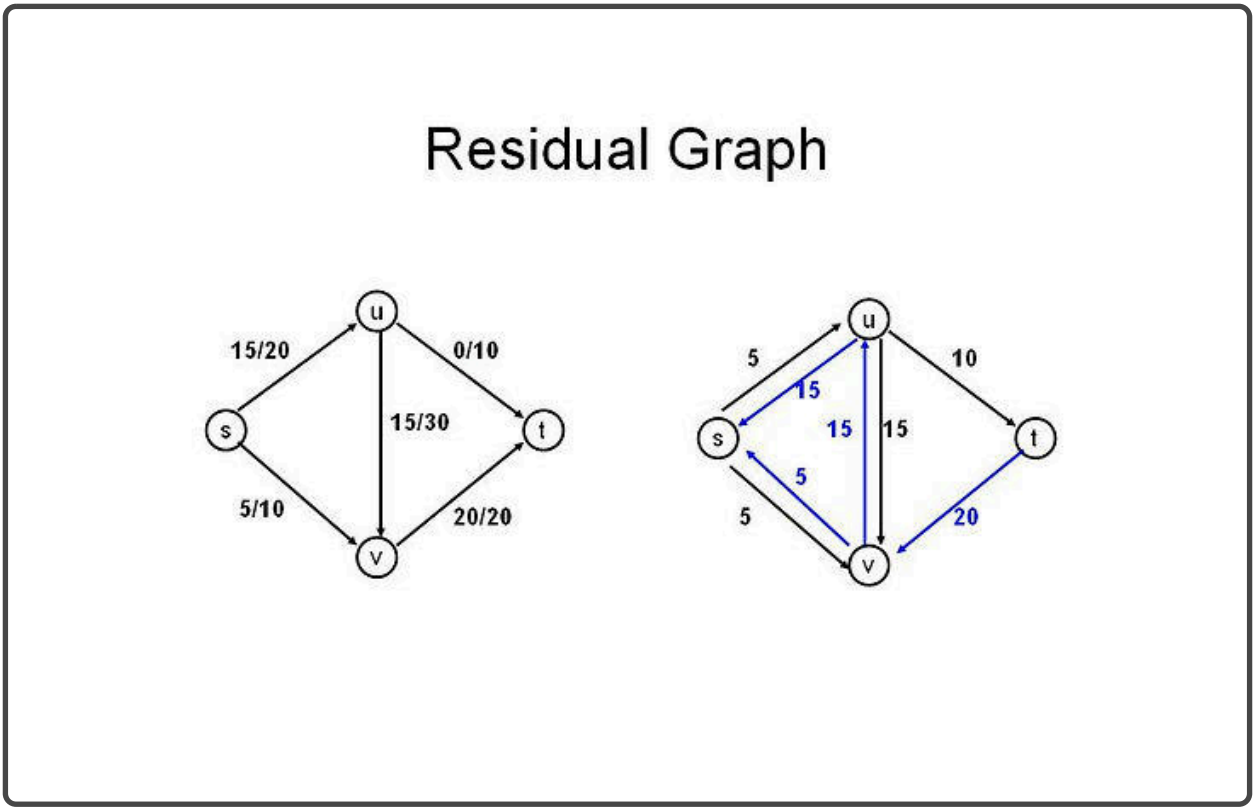
A NE TRAITER QU'UNE FOIS LA PARTIE IV TERMINEE

Montrez que si on a une coupe en deux ensembles (S, T) et que $s \in S$, alors on a $val(f) = f_{S/T}$

Définition 3.3 Graphe résiduel

On construit un graphe noté G_f associé au flot f (le graphe résiduel) de cette manière :

- Pour chaque arête dans G on ajoute deux arêtes dans G_f : une dans le même sens de valeur $cap - flot$ (ce qu'on peut ajouter sans dépasser la capacité) et une dans l'autre sens de valeur $flot$ (ce qu'on peut retirer sans tomber en-dessous de 0)
- On n'ajoute pas une arête étiquettée à 0 (à retenir pour une future preuve)



Il faut voir le graphe résiduel comme le graphe des possibilités (attention il est étiqueté avec une seule valeur, qu'on appellera capacité, il n'y a pas de "flot" sur les arêtes du graphe résiduel, comme sur le dessin proposé dans la définition)

Question 8 à l'écrit

A NE TRAITER QU'UNE FOIS LA PARTIE IV TERMINEE

Montrez le résultat suivant :

Théorème 3.1 Augmentation liée à un chemin dans G_f

Soit c un chemin de s à t dans le graphe résiduel de G. Si on ajoute (ou retire selon l'orientation) la plus petite valeur sur le chemin au flot de G, alors f' (ainsi obtenue) est bien un flot et $val(f') = val(f) + \min_{p \in c} cap(p)$ et

Question 9 à l'écrit

A NE TRAITER QU'UNE FOIS LA PARTIE IV TERMINEE

Montrez que s'il n'existe pas de chemin de s à t dans G_f, alors il existe une coupe de capacité $val(f)$

Définition 3.4 Capacité d'une coupe

On appelle capacité d'une coupe S, T la valeur $c_{S/T} = \sum_{(i,j)\in S\times T, (i,j)\in E} c_{i,j}$

Question 10 à l'écrit

A NE TRAITER QU'UNE FOIS LA PARTIE IV TERMINEE

Montrez que $val(f) \leq c_{S/T}$ pour toute coupe

Question 11 à l'écrit

A NE TRAITER QU'UNE FOIS LA PARTIE IV TERMINEE

Concluez sur un lien entre l'existence d'un chemin de s à t dans G_f et l'optimalité du flot f.

On peut ainsi obtenir l'algorithme de Ford-Fulkerson :

Définition 3.5 Algorithme de Ford-Fulkerson

- On construit G_f, le graphe résiduel associé au flot f
- tant qu'il existe un chemin de s à t dans G_f,
  - On trouve α (la plus petite capacité de ce chemin)
  - On augmente chaque arcs sur le chemin de α (attention, le signe est différent selon que l'arc soit dans le sens du flot ou dans le sens inverse)

On obtient alors un flot optimal d'après la série de questions que vous ne traiterez qu'après la partie IV terminée.

Les prochaines questions ont pour but d'implémenter l'algorithme de Ford-Fulkerson :

Question 12 code

Implémentez la fonction `residual_graph` qui prend en entrée un graphe et renvoie le graphe résiduel associé

Question 13 code

1. (dans votre tête)Comment trouver un chemin de s à t si il existe et être sûr qu'il n'existe pas sinon ?
2. Implémentez votre stratégie et créez une nouvelle fonction `ford_fulkerson` qui prend en entrée un graphe G correspondant à un flot valide et applique l'algorithme.

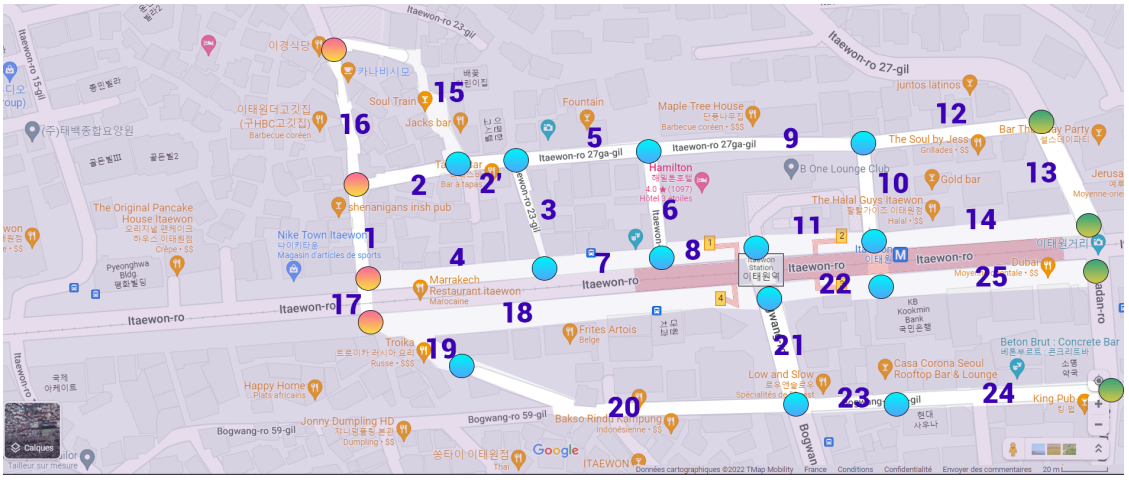
   On fera attention à la manière d'ajouter selon l'orientation de l'arête dans G_f par rapport à l'orientation dans G

IV Application à Itaewon (mouvement de foule)

Des sujets classiques vous proposerez des applications au commerce ou à des réseaux routiers de camions, ce sujet va vous proposer une étude de cas d'un mouvement de foule.

Contexte Nous sommes le samedi 29 Octobre 2022 dans le quartier d'Itaewon à Séoul, les sud-coréens sortent dans la rue pour une marche festive organisée par la ville. C'est la fin du covid et tout le monde saute sur l'occasion de retrouver un contact avec les autres. Problème : certaines rues sont trop étroites, la marche dégénère et au total, c'est plus de 150 morts. Dans cette fin de sujet (qui ne sert que d'exemple d'application, pas de panique la partie théorique est terminée) nous allons appliquer l'algorithme de Ford-Fulkerson pour obtenir une trajectoire à donner aux individus dans la marche garantissant qu'il n'y ait pas d'accident

Données Pour cela, il nous faut un réseau de flots, voici comment nous sommes passés d'Itaewon à un réseau de flots :



Définition 4.1 Super-source et super-puit

Les points oranges sont des super-sources et les points verts sont des super-puits : on connecte une "vraie" source sans contrainte (1000000 par exemple) à chacune des super-sources et tous les super-puits sont connectés à un sommet sans contrainte (100000 par exemple), ce qui permet de partir de plusieurs endroits et d'arriver de plusieurs endroits, vous n'avez pas à y prêter attention puisque le graphe vous est déjà fournie.

Désormais, il faut donner une capacité à chaque arête, pour cela je me base sur un travail que j'ai effectué pour mon TIPE (disponible en annexe) et les données vous sont donc fournises (fichier `itaewon.c`). Brièvement, la capacité représente ici des `humains par seconde` donc si l'algorithme nous affiche 100 pour valeur de flots sur une arête, ça veut dire que la rue conseille d'accueillir 100 personnes par seconde pour maintenir la sécurité.

Question 14 code

Appliquez l'algorithme de ford-fulkerson et notez les valeurs de flots obtenues (sur une feuille de brouillon par exemple).

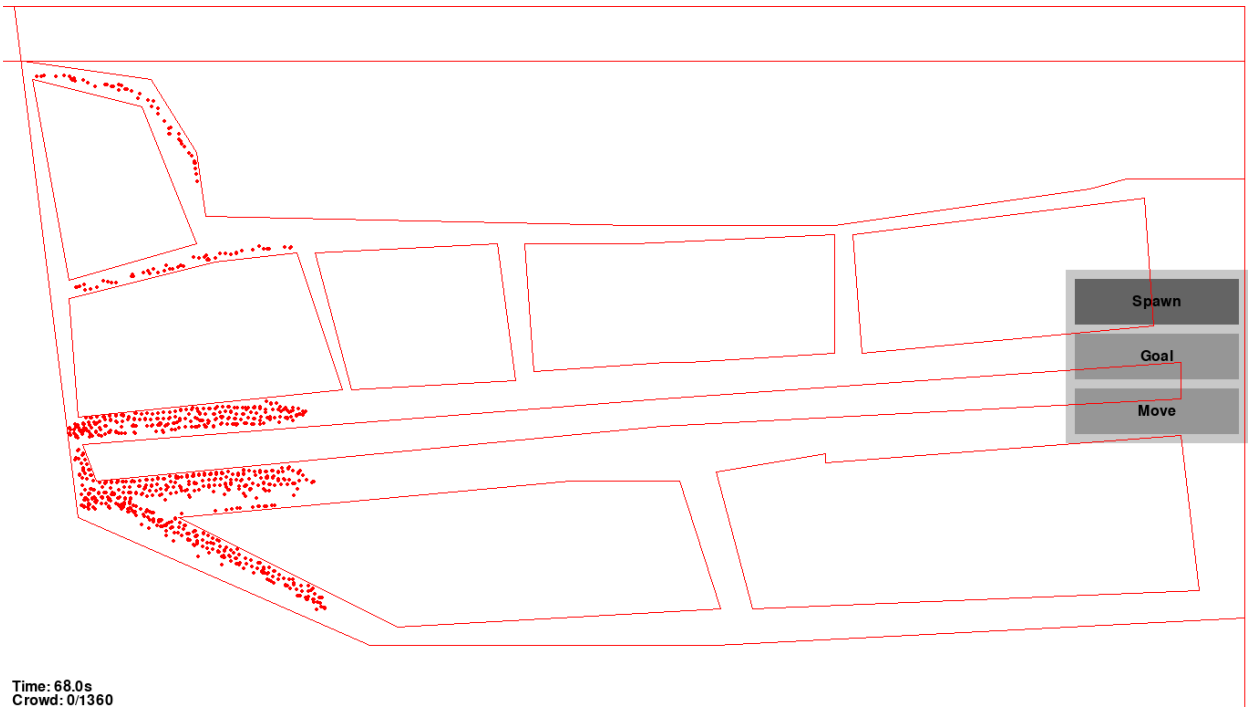
Question 15 à l'écrit

On estime à 200.000 personnes le nombre d'individus présent au mouvement de foules, avec votre itinéraire proposée, combien d'heures aurait-il fallu pour évacuer avec 0 décès ?

V Conclusion

Les réseaux de flots sont une simplification lourde (i.e. on n'applique pas de lois physiques, de principes de biologie ou de sciences comportementales) de la réalité mais si le modèle est suffisament bien calibré (ce qui peut se vérifier expérimentalement en créant des simulations qui elles se basent sur des lois physiques, cf la chaîne Youtube Fouloscopie) il peut-être utilisé en production et possiblement sauver des vies ou accélérer la producion / la livraison de biens (les transporteurs se servent souvent de réseaux de flots pour optimiser).

Si vous êtes curieux, voici à quoi ressemble les proportions que vous avez trouvé avec ce sujet si on les applique dans une simulation d'Itaewon:



En particulier, on voit qu'on ne tue personne !

Bref, fin de la pause: attaquez la démonstration de Ford-Fulkerson.