INF8775 – Analyse et conception d’algorithmes

TP3 – Automne 2022

# Informations sur la correction

* Répondez dans le document du TP3\_A22\_Rapport.docx.
* Le date limite pour rendre ce TP est le 6 décembre à 23h59, vous devez faire une *remise électronique sur Moodle* en suivant les instructions suivantes :
  + Le dossier remis doit se nommer matricule1\_matricule2\_tp3 et doit être compressé sous format zip.
  + À la racine de ce dernier, on doit retrouver :
    - Ce rapport au format .docx
    - Un script nommé *tp.sh* servant à exécuter l’algorithme du TP. L’interface du script est décrite à la fin de ce document.
    - Le code source et l'exécutable.
* Vous avez le choix du langage de programmation utilisé. Notez que le code et l'exécutable soumis seront testés sur les ordinateurs de la salle L-4714 et doivent être compatibles avec cet environnement. En d’autres mots, tout doit fonctionner correctement lorsque le correcteur exécute votre script *tp.sh* sur un des ordinateurs de la salle.

# Mise en situation

Ce dernier travail pratique se fera dans le cadre du concours du meilleur algorithme pour la session d’automne 2020. Le travail demandé consiste à concevoir et implanter un algorithme de votre cru pour résoudre un problème combinatoire. Le classement des équipes déterminera votre note pour la *qualité de l'algorithme*. Votre algorithme sera exécuté sur 3 exemplaires de notre choix pendant 3 minutes chacun.

Ce TP porte sur un sujet qui a été aux nouvelles assez souvent récemment, les élections. Pour avoir des élections, il faut bien sûr des électeurs, mais avant même que ceux-ci puissent aller voter, il faut qu’ils soient enregistrés dans des circonscriptions (« districts », zones électorales). Ces dernières sont à leur tour composées de municipalités qui sont plutôt les zones administratives dans lesquelles les électeurs habitent et qui peuvent être plus ou moins densément peuplées. De plus, les populations changent et se déplacent, par conséquent, il faut périodiquement redessiner les zones électorales afin qu’elles demeurent représentatives de la population. Au Canada, aux niveaux fédéral et provincial, les différents facteurs (desquels le nombre d’habitants est un facteur majeur) qui sont pris en compte dans ce redécoupage sont clairement établis et ce dernier est fait par des organismes indépendants des différents partis politiques. Néanmoins, ce redécoupage des zones électorales peut être utilisé à mauvais escient, afin de favoriser un parti politique plutôt qu’un autre par exemple, dans des pays où il est un peu moins bien encadré, comme aux États-Unis et cette pratique s’appelle le « gerrymandering ». Voici la liste des sources qui ont été utilisées pour ce paragraphe et que vous pourriez consulter si vous voulez en savoir plus sur le sujet :

<https://www.elections.ca/content.aspx?section=res&dir=ces&document=part1&lang=f>

[https://www.elections.ca/content.aspx?section=res&dir=cir/list&document=index338&lang=f#list](https://www.elections.ca/content.aspx?section=res&dir=cir/list&document=index338&lang=f" \l "list)

[https://fr.wikipedia.org/wiki/Circonscriptions\_%C3%A9lectorales\_du\_Canada](https://fr.wikipedia.org/wiki/Circonscriptions_électorales_du_Canada)

<https://www.elections.ca/content.aspx?section=res&dir=eim/issue6&document=p6&lang=f>

<https://www.washingtonpost.com/news/wonk/wp/2015/03/01/this-is-the-best-explanation-of-gerrymandering-you-will-ever-see/>

<https://www.nytimes.com/2019/06/27/us/what-is-gerrymandering.html>

Le TP porte plus spécifiquement sur le « gerrymandering » et en voici les détails :

* Il y a *n* municipalités carrées de 100 habitants chacune, pour lesquelles on connaît les proportions de votes pour les deux partis, les verts et les jaunes.
* On doit diviser les municipalités en *m* circonscriptions qui sont formées de *k* municipalités chacune avec *plancher(n/m) ≤ k ≤ plafond(n/m).* (m est un paramètre d’entrée du programme)
* Une circonscription n’a pas besoin d’être connexe, mais la distance manhattan entre toute paire de municipalités dans une même circonscription doit être au plus *plafond(n/2m)*.
* Un parti remporte une circonscription si la majorité des électeurs à l’intérieur de celle-ci ont voté pour lui, autrement dit, la somme des votes à l’intérieur de toutes les municipalités de la circonscription est majoritaire.

Votre objectif est de regrouper les municipalités en circonscriptions de manière à maximiser le nombre de circonscriptions dans lesquelles les verts sont majoritaires, afin de les aider à remporter les élections.

Voici une figure qui montre différents découpages de municipalités en circonscriptions afin de changer les résultats des élections. À noter que la situation est simplifiée dans la figure, car on y fait abstraction du nombre d’électeurs qui se trouvent dans chacune des municipalités, en d’autres termes, tous les habitants d’une municipalité votent soit vert soit jaune.

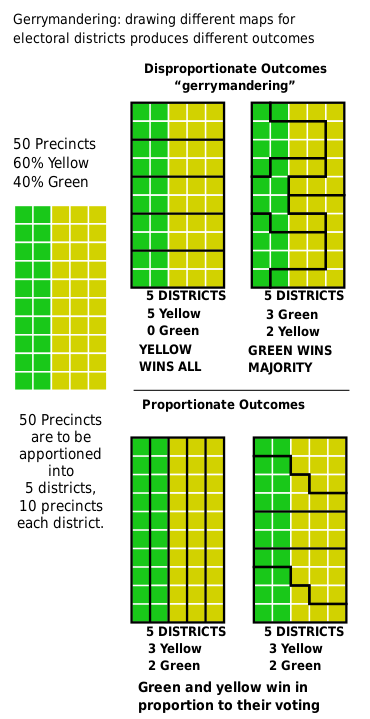


Figure 1 : Exemple de différents résultats d’élection selon les districts

source : M.boli — Travail personnel. Derived from an image by Steven nAss, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=64401739

Le rapport pour ce dernier travail pratique est assez succinct. Vous êtes encouragés à terminer ce travail assez tôt afin de ne pas compromettre la préparation à vos examens finaux.

# Jeu de données

Nous vous fournissons un générateur d’exemplaires qui fonctionne comme suit :

python[3] gen.py -x 5 -y 10

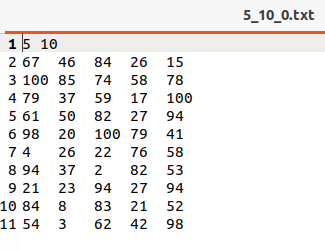
Arguments :

* x = Le nombre de municipalités le long de l’axe des x
* y = Le nombre de municipalités le long de l’axe des y

Un fichier au format x\_y\_n.txt (où n permet d’éviter d’écraser un fichier existant) est enregistré dans un dossier « ./exemplaires », comme par exemple :

Exemplaire généré après l’exécution de la ligne donnée en exemple en haut de la page :

fichier  5\_10\_0.txt



Sur la première ligne on retrouve le nombre de municipalités le long de l’axe des x et le nombre de municipalités le long de l’axe des y.

Les lignes suivantes donnent une représentation des municipalités sur un plan 2D, l’élément (i,j) de la matrice donne le nombre d’électeurs qui votent pour le parti vert (rappel : il y a 100 électeurs en tout par municipalité).

Afin de vous simplifier les choses, les coordonnées des différentes municipalités correspondent à leurs indices i et j.

## Respect de l’interface tp.sh

Utilisation :

tp.sh -e [chemin\_vers\_exemplaire] -c [nombre\_de\_circonscriptions]

Lorsque le script est exécuté sans le paramètre -p, le programme affiche uniquement le nombre de total de circonscriptions qui votent pour le parti vert sur une nouvelle ligne, à chaque fois qu’une meilleure solution est trouvée. Votre programme est sensé s’exécuter tant et aussi longtemps qu’il n’est pas manuellement interrompu.

**Argument optionnel** :

-p Chaque fois qu’une meilleure solution est trouvée, le programme affiche cette nouvelle solution (**au lieu** d’afficher le nombre de circonscriptions qui votent pour le parti vert). Le format de cette solution est décrit ci-dessous

**Important** : l’option -e doit accepter des fichiers avec des chemins absolus.

Le script tp.sh ne vous est pas fourni, mais vous pouvez facilement adapter celui du TP2.

**Format de la solution** : Pour une même solution, il faut afficher les différentes municipalités qui se trouvent dans une même circonscription sur la même ligne, une à la suite de l’autre, séparées par un espace, puis lorsqu’on passe à une prochaine circonscription, on change de ligne. Par la suite, lorsqu’une meilleure solution est trouvée, on saute deux fois la ligne. Dans l’exemple d’affichage ci-dessous nous avons d’abord une première solution avec 3 circonscriptions qui comptent 3 municipalités chacune, puis une deuxième solution est affichée à la suite de la ligne vide lorsqu’elle est trouvée. Il est à noter que les deux solutions respectent les contraintes, mais que la deuxième est meilleure (c.-à-d. que plus de circonscriptions en tout votent pour le parti vert) que la première (et que la troisième est meilleure que la deuxième et ainsi de suite) et que ce n’est que la dernière solution fournie qui sera prise en compte pour la correction.

0 0 0 1 1 0

1 1 0 2 1 2

2 0 2 1 2 2

0 0 1 0 2 0

0 1 1 1 0 2

2 1 1 2 2 2

Chaque municipalité d’une circonscription a ses coordonnées x et y séparées par un espace. Entre chaque nouvelle solution trouvée, **vous devez laisser une ligne vide**.

Un script de vérification de solution vous est fourni (check\_sol.py, les instructions d’utilisation sont dans le code source). Ce script vous indiquera si l’affichage est correct, si votre solution est valide, ainsi que le nombre de circonscriptions qui votent pour le parti vert. C’est ce script qui sera utilisé pour la correction, donc assurez-vous qu’il reconnaisse vos solutions.