BAUCHER Quentin
JOUNIOT Constantin

Projet d'Algorithmique

Comparaison entre la méthode locale et la méthode globale, sur divers cas et exemples

ESIEE Paris - E5FI - Année 2021/2022

Nous avons étudié divers problèmes d'optimisation durant nos cours d'algorithmique qui nous ont permis de trouver une méthode qui trouverait la meilleure solution possible à notre problème.
Par exemple, si nous devions trouver comment remplir un sac de course en ayant la meilleure combinaison possible d'objets en fonction de leurs valeurs et de la place qu'ils prennent dans le sac, notre algorithme serait capable de donner cette combinaison.
Cependant, cette méthode peut souvent s'avérer coûteuse en temps sur des problèmes de très grandes tailles. C'est pourquoi il existe une autre solution, appelée solution locale ou gloutonne, où l'algorithme va prendre les valeurs qui l'intéresse le plus lorsqu'il a le choix entre plusieurs valeurs. Tout cela sans se soucier d'avoir le meilleur résultat global.
Dans ce rapport, nous expliquerons, dans différents cas, comment fonctionne cette fonction locale et la comparaison avec la fonction optimale.

Sommaire

Exécution du programme

Afin d'exécuter notre programme, veuillez utiliser une version de Python supérieure ou égale à la version suivante : **3.8.8**

Ensuite veuillez faire attention à avoir les librairies suivantes d'installées :

```
argparse
random
matplotlib
numpy
os
```

Enfin, pour exécuter notre programme, effectuez les tests entre les différentes méthodes optimales et les méthodes locales, souvent appelées "gloutonne". Vous pouvez exécuter le script de différentes manières :

Exécute tous les tests de tous les exemples

```
py Algorithmie_projet_BAUCHER_JOUNIOT_global.py  --all
```

Exécute les tests du robot sur la grille

```
py Algorithmie_projet_BAUCHER_JOUNIOT_global.py  --robot_grille
```

Exécute les tests du le répartitions de valeur maximale dans un sac

```
py Algorithmie_projet_BAUCHER_JOUNIOT_global.py  --sac_max
```

Exécute les tests du gain max sur la répartitions de stocks dans des entrepôts

```
py Algorithmie_projet_BAUCHER_JOUNIOT_global.py  --entrepots
```

Exécute les tests du chemin de somme maximum

```
py Algorithmie_projet_BAUCHER_JOUNIOT_global.py  --chemin_max
```

Exécute les tests du robot sur le graphe

```
py Algorithmie_projet_BAUCHER_JOUNIOT_global.py  --robot_graphe
```

Les différents exemples sont aussi sous la forme de Notebook Jupyter afin d'exécuter étape par étape les différentes parties du code (voir le répertoire du projet).

Le robot sur une grille

Fonction locale

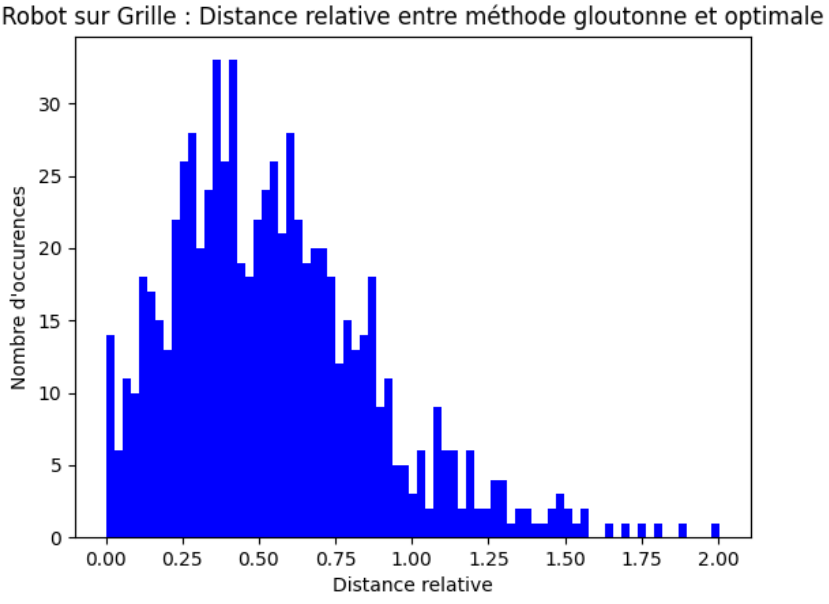
Imaginons une grille, de N*M dimension. Un robot est placé à la case (0,0) de cette grille et cherche à atteindre la position (N-1,M-1) de la grille. Le robot peut se déplacer vers le Nord (N), l'Est (E), et le Nord-Est (NE) afin d'arriver à son but.

Problème : chaque déplacement a un coût et le robot cherche à avoir le coût le moins élevé possible lorsqu'il aura atteint son but.
Afin de résoudre ce problème via une solution locale, le robot testera à chaque nouvelle case quel est le déplacement qui lui coûtera le moins entre les directions N, E, et NE. Et cela jusqu'à être arrivé à la case (N-1,M-1).

Cet algorithme va donc fonctionner au cas par cas, à chaque case que le robot parcourt.
Ce n'est pas la fonction optimale car le robot pourrait choisir par exemple une direction qui coûte 2 puis ensuite serait obligé de prendre une direction qui coûte 20, résultant en un coût total de déplacement de 22.
Alors qu'une autre option aurait été de prendre une direction qui coûte 4 (au lieu de prendre celle qui coûte 2), pour avoir ensuite une direction qui coûte 10, résultant en un coût total de 14, au lieu de 22.

Comparaison avec l'optimale

En effectuant la comparaison des distances relatives entre les deux fonctions nous trouvons cet histogramme :



Une grande distance montre une différence de coût élevé entre les deux fonctions. Plus il y a de pics après 0 dans le graphique, plus on peut en conclure que la méthode optimale est

meilleure que la méthode gloutonne, car cette répartition de pics montrent que les deux fonctions donnent un résultat différent.
Dans le cas du robot sur sa grille, nous pouvons remarquer une forte répartition des distances relatives entre les deux fonctions, montrant que la méthode gloutonne n'est pas efficace.

Le sac de valeur maximum

Fonction locale

Le problème du sac de valeur maximum est assez simple. Nous sommes au supermarché et une promo spéciale arrive : "Vous avez le droit d'avoir un sac, de contenance C, remplit de n'importe quel produit de notre magasin gratuitement".

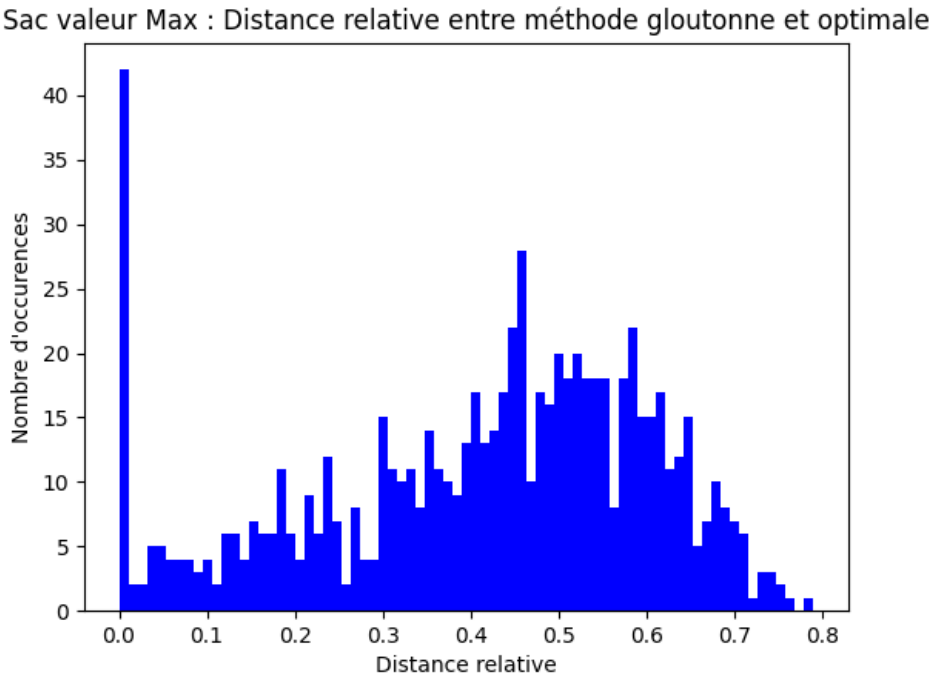
Evidemment, les derniers mois furent un peu difficiles et vous souhaitez donc avoir un sac de plus grande valeur mais un produit i prend une certaine place Ei dans votre sac. Lorsque vous rajoutez un objet à votre sac, il perd de sa contenance, qui devient alors : C = C - Ei. Vous ne pouvez pas faire rentrer un objet dans votre sac si votre sac n'a plus de place.
L'approche locale consiste à récupérer les objets qui ont la plus grande valeur et de les mettre en priorité dans le sac.

Exemple : nous regardons l'objet de plus grande valeur dans notre magasin, peut-il rentrer dans le sac ? Si oui, on le fait rentrer dans le sac, on réduit la contenance de notre sac, puis on passe au deuxième objet de plus grande valeur.
Est-ce que le deuxième objet de plus grande valeur du magasin peut rentrer dans le sac ? Si oui, on fait comme pour le premier objet. Sinon, on passe au troisième objet, et ainsi de suite.

Cette méthode n'est pas optimale car on pourrait avoir un objet d'une grande valeur, mais prenant une trop grande place et empêchant de prendre à la place plusieurs objets de petite taille mais avec des valeurs acceptable, qui pourraient ensemble peser plus que le grand objet seul.
Exemple : on a C = 10, on a un objet de taille 10 et de valeur 10 et trois objets de taille 3 et de valeur 4. Notre fonction locale, dite gloutonne, va mettre le premier objet et aura une valeur globale de 10 alors que la fonction optimale retournera le sac avec les trois objets de taille 3 et de valeur 4, donnant une valeur du sac égale à 12.

Comparaison avec l'optimale

En effectuant les mêmes opérations que pour le Robot sur une grille, nous arrivons à cet histogramme :



On a une forte différence entre la méthode gloutonne et la méthode optimale sur ce cas. On remarque surtout cette différence par la présence d'un grand pic en 0, montrant que la distance relative est majoritairement de 0, soit aucune différence entre les choix d'objets du glouton et de l'optimale.
Cependant, il y a des variations montrant qu'il y a quand même une différence importante lorsque les deux méthodes ne trouvent pas les mêmes résultats, montrant par la dispersion la partie milieu-droite de notre graphique.

Répartition de stocks dans des entrepôts

Fonction locale

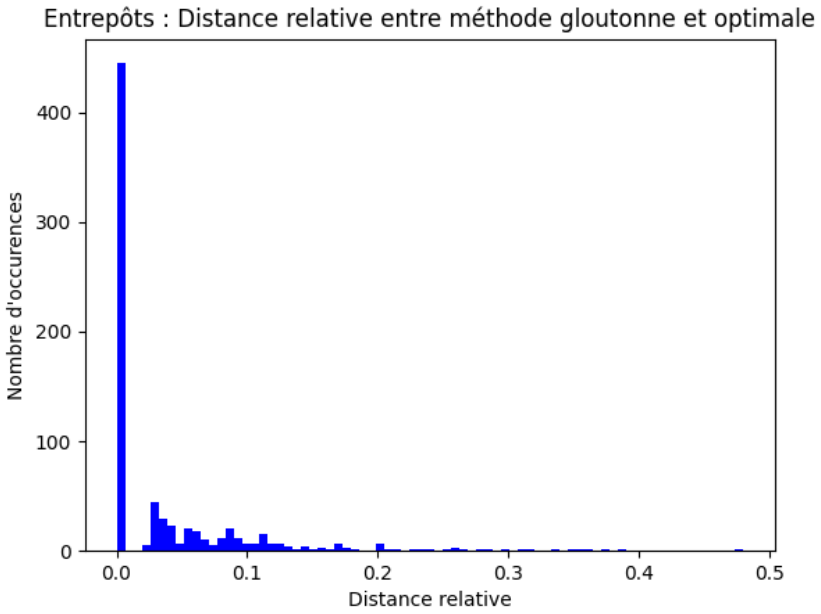
Le cas est le suivant : nous sommes en 1994 et Jeff Bezos, fondateur d'Amazon, cherche à répartir des stocks de livres dans plusieurs entrepôts pour ses livraisons.
Malheureusement, nous ne sommes pas encore à l'époque où les machines vont faire le travail pour nous, il faut justement les créer !

Il faut savoir que pour un stock S de livres, un entrepôt X aura un certain gain (en argent) G(X,S) et Jeff Bezos cherche avoir un gain maximal quand il va répartir ses stocks.
Dans l'approche locale, notre algorithme va chercher à disposer chaque stockage 1 à 1 dans les entrepôts disponibles. Imaginons que sur deux entrepôts 1 et 2, l'entrepôt 1 nous donne plus de gain si nous lui donnons un niveau de stock nécessaire,
soit G(1,S+1) - G(1,S) > G(2,S+1) - G(2,S)
Nous allons répartir ce stock sur l'entrepôt 1, et nous allons procéder de telle sorte jusqu'à la fin des stocks disponibles.

Ceci n'est pas l'approche optimale car dans l'entrepôt 2, le gain pourrait être de 0 en ajoutant S=1 de stock puis de 10 en étant à S=2, mais étant donné que notre algorithme ne mettra jamais de stock S=1 dans l'entrepôt, il n'ira jamais chercher le gain de 10 au stock S=2.

Comparaison avec l'optimale

En reprenant le modèle de comparaison des anciens cas et exemples, nous trouvons cet histogramme :



Les résultats sont déjà bien plus parlants ! On peut remarquer une forte présence du pic en 0. Montrant que les résultats ne sont pas très différents entre la méthode gloutonne et la méthode optimale.
Il y a certains cas où la méthode optimale montrera son efficacité, mais on peut voir l'intérêt de prendre la méthode gloutonne, pour des calculs de très grande échelle.

Chemin de somme max dans une pyramide

Fonction locale

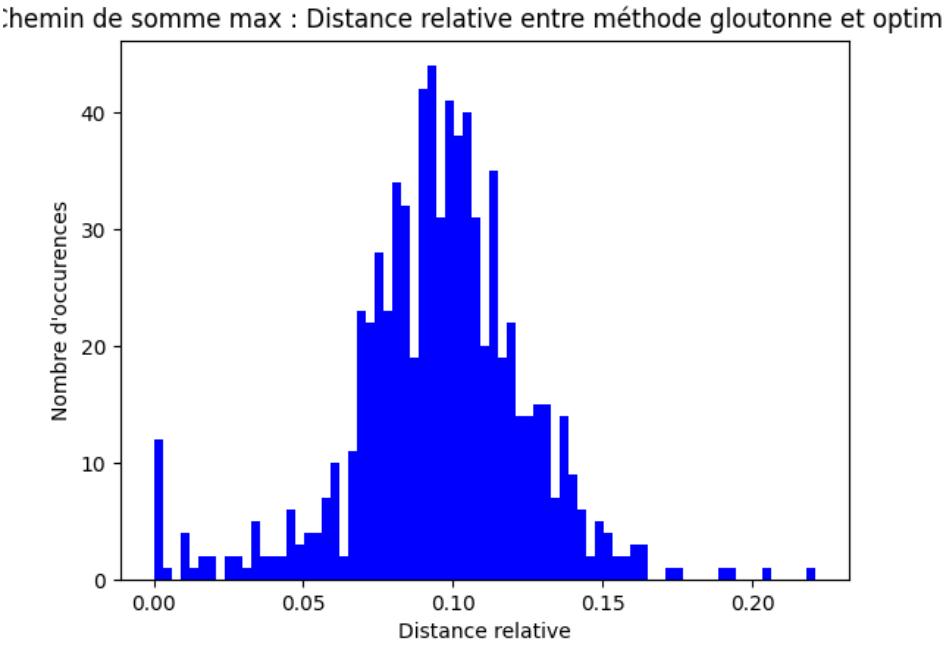
Notre exemple est le suivant : vous êtes Indiana Jones, et malheureusement, lors d'une poursuite en avion, vous tombez de ce dernier et vous retrouvez au sommet d'une pyramide. Vous devez descendre de cette pyramide tout en récupérant les pièces sur chaque pierre sur lequel vous passez. Bien évidemment, vous voulez récupérer le plus de pièces possibles afin de vous enrichir.

Vous possédez donc un tableau représentant votre pyramide et le nombre de pièces sur chaque pierre de la pyramide. Vous cherchez donc une approche pour descendre tout en bas de la pyramide avec le plus de pièces possible.
L'approche locale va donc procéder ainsi : "J'ai le choix entre deux pierres en dessous de moi, je vais prendre la pierre possédant le plus de pièces afin d'avoir le gain maximal". Et donc l'approche locale va procéder de descendre ainsi en prenant le maximum entre les deux seuls choix dont elle dispose.

Cette méthode n'est pas la méthode optimale car l'approche locale va peut-être passer à côté de d'un chemin où la somme de pièces serait plus conséquente. Étant donné qu'elle choisie au cas par cas, elle ne saura jamais que l'autre chemin était plus bénéfique pour elle.

Comparaison avec l'optimale

Comme à notre habitude, nous reprenons notre comparaison avec la méthode optimale qui va calculer tous les chemins et en ressortir le meilleur. Nous avons pris environ un nombre aléatoire entre 5 et 400 étages et effectué 750 runs afin de tester différentes combinaisons de notre fonctionnement optimal.
Nous établissons un histogramme de la distance relative, soit la différence entre les résultats des deux méthodes :



La méthode optimale montre quand même une plus grande efficacité, vu la répartition des pics, montrant qu'elle reste assez utile pour les problèmes de ce cas

Le robot sur un graphe

Fonction locale

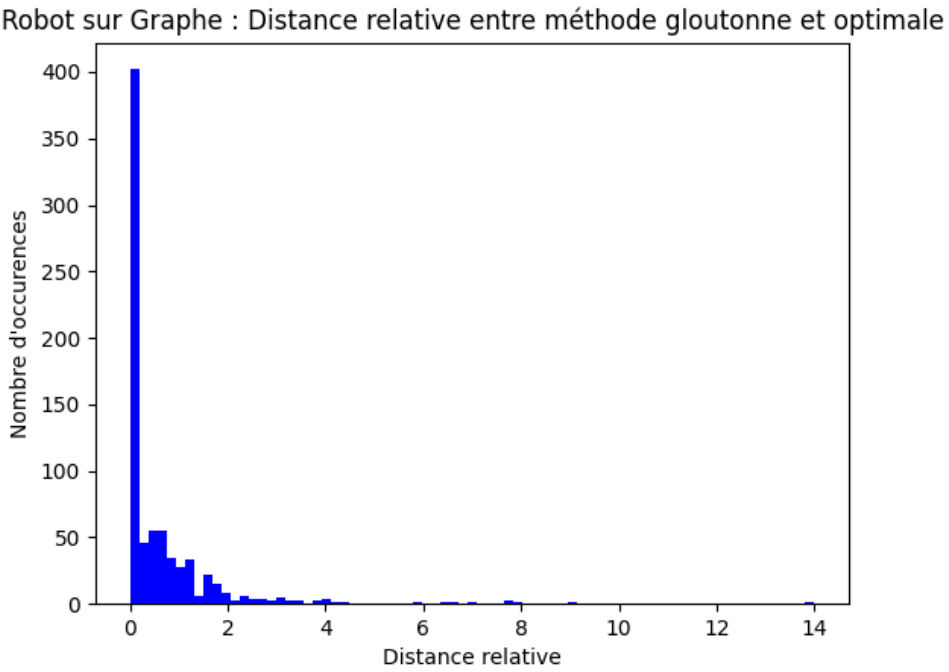
Megaman a besoin de votre aide ! Le professeur Wily, antagoniste de la série de jeux Megaman, l'a enfermé dans une prison dont les portes le téléporte dans une autre cellule. Cependant, chaque porte qui le téléporte coûte de l'énergie à Megaman, et il doit trouver le chemin qui va lui consommer le moins d'énergie possible !
Megaman trouve une carte de la prison, présentée sous la forme d'un schéma de graphes, soit un nuage de nœuds qui sont reliés entre eux pour certains, et pas pour d'autres. Sur chaque liaison entre les nœuds, on peut y apercevoir l'énergie, ou le coût, pour se transporter d'un nœud à un autre.

L'approche locale afin de trouver le chemin de coût minimal est de toujours prendre le chemin qui est de coût moindre lorsque nous sommes sur un nœud.
Imaginons que nous soyons sur le premier nœud, nous avons le choix entre un chemin qui coûte 1, 3, et 5. Nous allons choisir le chemin qui coûte 1 afin d'utiliser le moins d'énergie possible.

Cependant cette méthode n'est pas la méthode optimale car elle reprend la même problématique que le chemin de sommes maximale, mais sur une structure de données différente. Nous pourrions passer à côté d'un chemin moins coûteux sur le long terme en prenant un choix de coût minimum à court terme.

Comparaison avec l'optimale

En reprenant notre méthode de comparaison, entre l'approche locale, dite gloutonne, et l'approche optimale, nous arrivons à cet histogramme de comparaison de distances relatives.



Nous remarquons donc une forte présence d'un pic à 0, montrant que la différence entre les deux méthodes n'est pas importante.
Cependant, pour les problèmes de petite à moyenne taille, la méthode optimale reste meilleure car la répartition des pics reste assez élevée, avec une distance relative entre les deux méthodes allant jusqu'à 14.