Léo Chardon
Remy Barberet

Algorithmique 5I-IN1

Le chemin du petit robot
- problématique
- La méthode gloutonne:
- La méthode optimale:
- comparaison

Sac de valeur maximum
- problématique
- La méthode gloutonne
- La méthode optimale
- Comparaison

Répartition du stock
- problématique
- La méthode gloutonne
- La méthode optimale
- Comparaison

Chemin de somme maximum
- problématique
- La méthode gloutonne
- La méthode optimale
- Comparaison

Recherche d'un élément dans une matrice triée
- problématique
- Méthode gloutonne
- Méthode optimale
- Comparaison

Léo Chardon
Remy Barberet

Le chemin du petit robot

problématique

L'objectif du robot est de d'atteindre le Nord-Est d'une matrice NM le plus rapidement possible, en se déplaçant uniquement vers la droite, vers le haut ou les deux à la fois, sa position initiale étant en bas à gauche. Pour cela on a deux algorithmes, un représentant la méthode dite gloutonne et un autre optimisé. Nous allons donc comparer les deux algorithmes à travers ce rapport ainsi que des tests.

On rappelle que le déplacement vers une direction à un coût.

La méthode gloutonne:

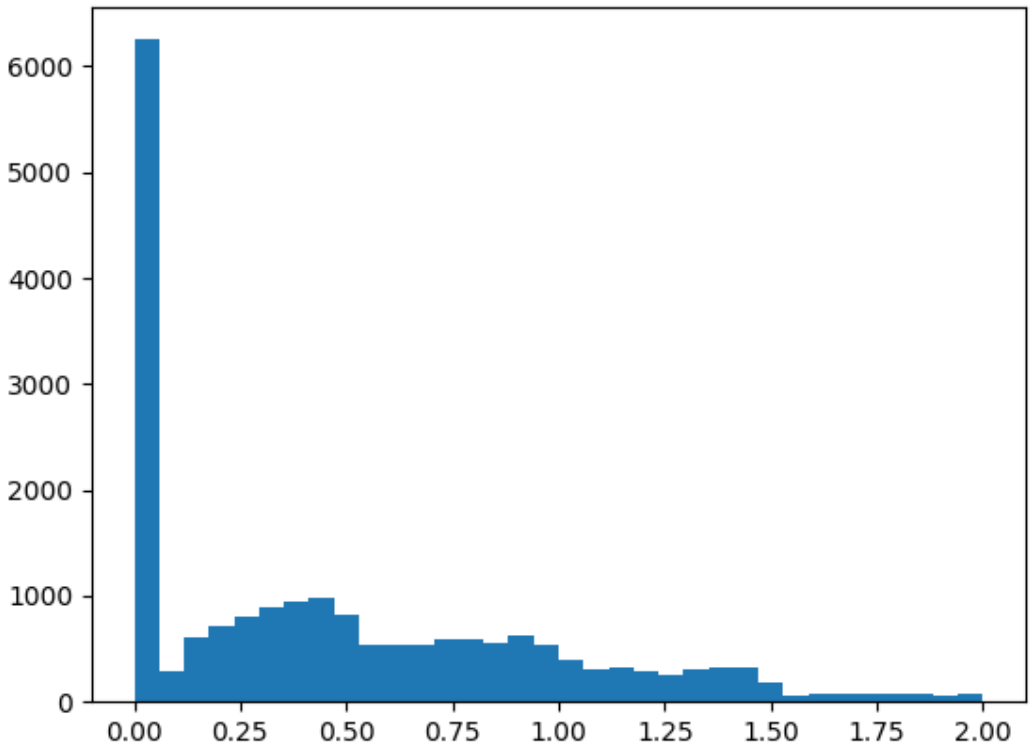
Le robot se déplace simplement vers le chemin le moins coûteux pour lui. Il ne test pas les différentes possibilités

La méthode optimale:

La méthode optimale est celle reprise du cours, le robot teste les différentes possibilités et regarde lesquelles sont les plus basses.

Léo Chardon
Remy Barberet

comparaison



histogramme de la distance relative avec 20000 runs

On observe une différence assez importante entre les deux algorithmes grâce au premier pique.

```
100% (20000 of 20000) |###################| Elapsed Time: 0:00:36 Time:  0:00:36
Greedy:
    mean:221.26195
    std:102.85526594296229
Optimal:
    mean:188.0
    std:63.936724249448375
```

On observe aussi que l'algorithme optimal à une moyenne de parcours beaucoup plus faible que celle de l'algorithme glouton, on peut l'expliquer par le fait que l'optimal va regarder les combinaisons de valeur contrairement au glouton.

Léo Chardon
Remy Barberet

Sac de valeur maximum

problématique

On possède deux listes, une liste de valeur et une liste qui correspond à l'encombrement que possède chaque valeur de la liste précédente. On cherche à former un sac d'une valeur la plus grande sans dépasser l'encombrement qui est donné.

La méthode gloutonne

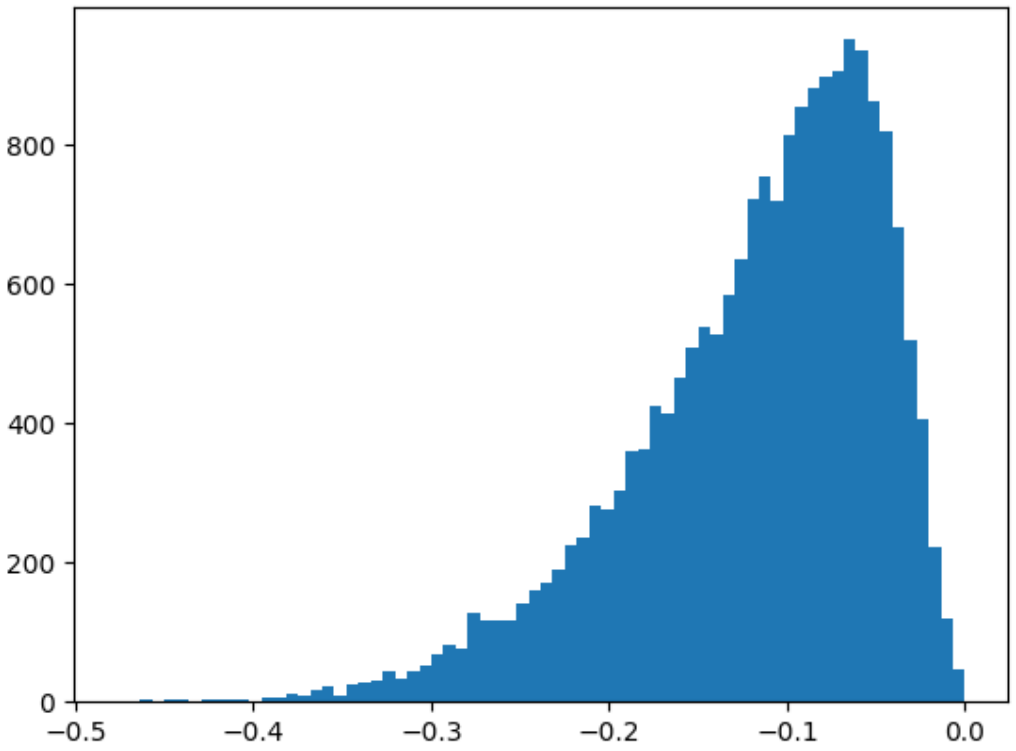
On va associer dans une liste de tuple la valeur ainsi que son encombrement, on va ensuite trier dans l'ordre décroissant cette liste de tuple par les valeurs, donc le premier élément des tuples.
Si l'encombrement le permet on va ensuite ajouter la valeur dans le sac, on a donc notre sac de valeur maximum de façon gloutonne

La méthode optimale

Algorithme récupéré par le cours, il va calculer le sac optimal de façon récursive et grâce à des sous ensemble.

Léo Chardon
Remy Barberet

Comparaison



histogramme de la distance relative avec 20000 runs

Différence flagrante entre les algorithmes

```
100% (20000 of 20000) |####################| Elapsed Time: 0:00:43 Time:  0:00:43
Greedy:
    mean:297.39845
    std:73.63881169327422
Optimal:
    mean:333.7661
    std:64.98134417500148
```

Même si l'algorithme glouton ne se tiens pas loin de l'algorithme optimale, l'optimale n'en reste pas moins meilleur (valeur plus élevé) dû au fait qu'il effectue des combinaison de valeur et ne fait pas de parcours linéair contrairement au glouton

Léo Chardon
Remy Barberet

Répartition du stock

problématique

On a un nombre d'entrepôt représenter dans une matrice par le nombre de ligne et le nombre de stock représentés par les colonnes, on cherche à repartir un stock sur les entrepôts et on connaît le gain que l'on fait dès que l'on répartit une unité, on cherche à avoir la meilleur répartition du stock sur tous les entrepôts.

La méthode gloutonne

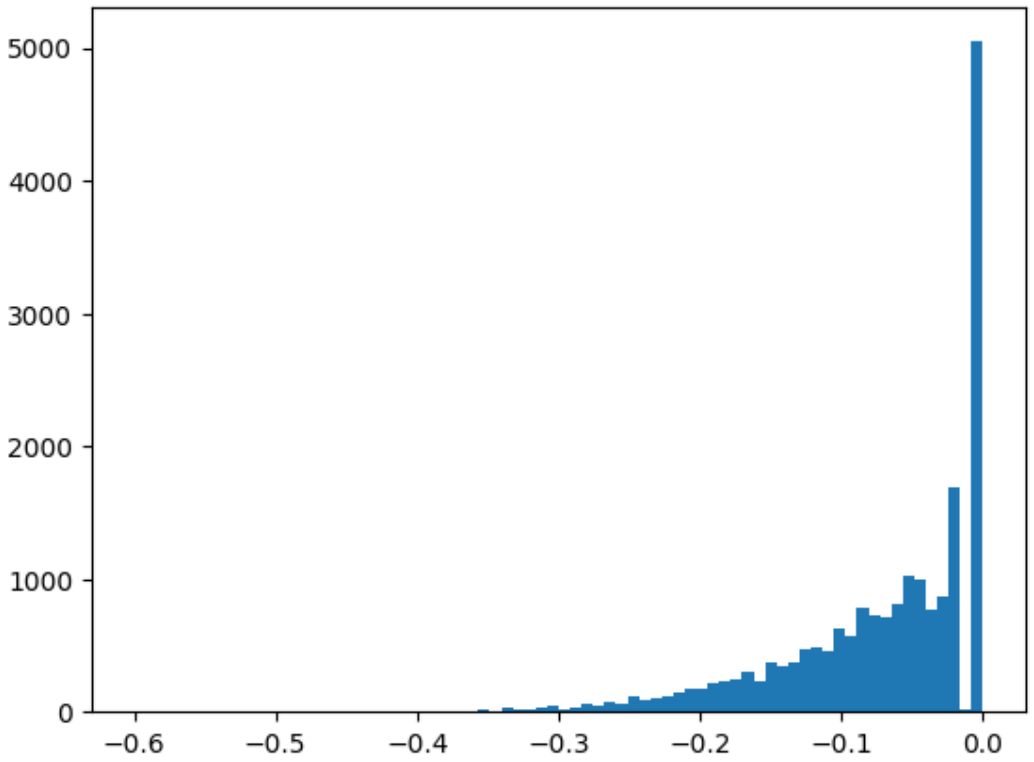
On observe pour chaque ligne le plus gros gain effectué en donnant une unité (entre la colonne initiale et celle suivante), on mémorise ensuite l'indice de cette ligne pour ne pas se tromper et effectuer de nouveau le calcul sur celle-ci, on avance donc l'indice uniquement pour cette ligne et on recommence. On obtient donc ensuite la répartition du stock la plus bénéfique sur les entrepôts

La méthode optimale

Méthode reprise de celle du cours.

Léo Chardon
Remy Barberet

Comparaison



histogramme de la distance relative avec 20000 runs

```
100% (20000 of 20000) |##################| Elapsed Time: 0:00:01 Time:  0:00:01
Greedy:
    mean:41.88665
    std:7.897860582303285
Optimal:
    mean:45.05435
    std:7.125061127983394
```

L'algorithme optimale est légèrement meilleur que celui du glouton, on peut expliquer cette différence par le fait que l'algorithme optimale va tester plusieurs possibilités de répartition des stocks même si un chemin d'une colonne à la suivante à fait un petit gain, contrairement à la méthode gloutonne qui ne retient uniquement les plus gros gain.

Léo Chardon
Remy Barberet

Chemin de somme maximum

problématique

Le but est de trouver le chemin nous renvoyant le plus grand nombre en parcourant un triangle, chaque niveau du triangle possède un nombre ayant une valeur et autant de nombre que le numéro de son niveau, les éléments ont donc un fils droit et/ou fils gauche. Si ils sont des feuilles, alors ils n'ont ni fils droit ni gauche.
La structure de donnée est représentée par une liste où le premier élément est le sommet du triangle et où l'élément +1 et +2 sont respectivement les fils gauche et droit.

La méthode gloutonne

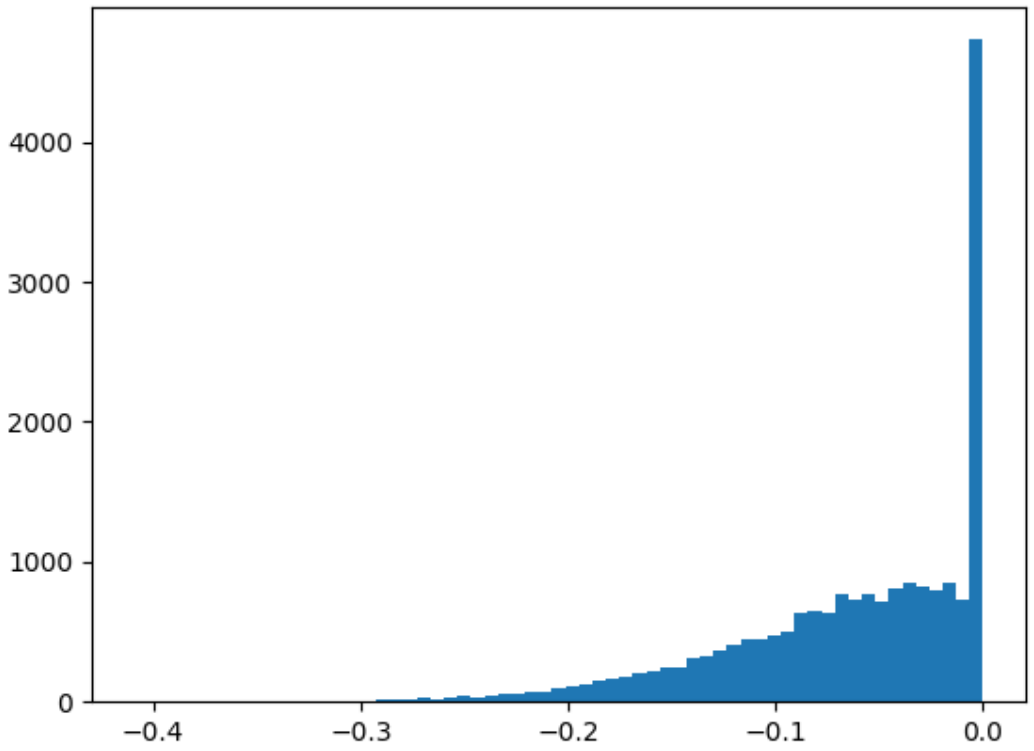
On parcourt le triangle en allant là où la valeur est la plus grande entre le fils droit et gauche, on additionne cette valeur à celle que l'on a initialement (0) et on s'arrête lorsque l'on tombe sur une feuille.

La méthode optimale

Grâce à la récursivité, on va calculer le maximum parmis tous les chemins depuis un niveau donnée, l'origine en l'occurrence, plus le triangle est grand et plus la complexité va être élevé car il faut calculer à chaque fois l'additions de toutes les valeurs, contrairement à la méthode gloutonne où l'on se dirige automatiquement vers le plus grand nombre.
Cette méthode est préférée pour sa fiabilité car on ne peut passer outre un nombre grand qui serait profond dans le triangle, contrairement au glouton qui peut complètement passer à côté.

Léo Chardon
Remy Barberet

Comparaison



histogramme de la distance relative avec 20000 runs

```
100% (20000 of 20000) |###################| Elapsed Time: 0:28:57 Time:  0:28:57
Greedy:
    mean:176.74145
    std:32.72076866299904
Optimal:
    mean:188.69475
    std:32.94022726754477
```

Encore une fois, l'algorithme optimal bat l'algorithme glouton car il teste l'ensemble des possibilités, et un grand nombre bas dans le triangle n'est pas à écarter.

Léo Chardon
Remy Barberet

Recherche d’un élément dans une matrice triée

problématique

On cherche un élément dans une matrice de nombre triée. L’élément se trouve ou pas dans la matrice. Ici on s’intéresse au nombre d’itération possible que l’on va réaliser pour trouver notre nombre. La plus grande itération possible peut être largeur de la matrice x sa longueur.

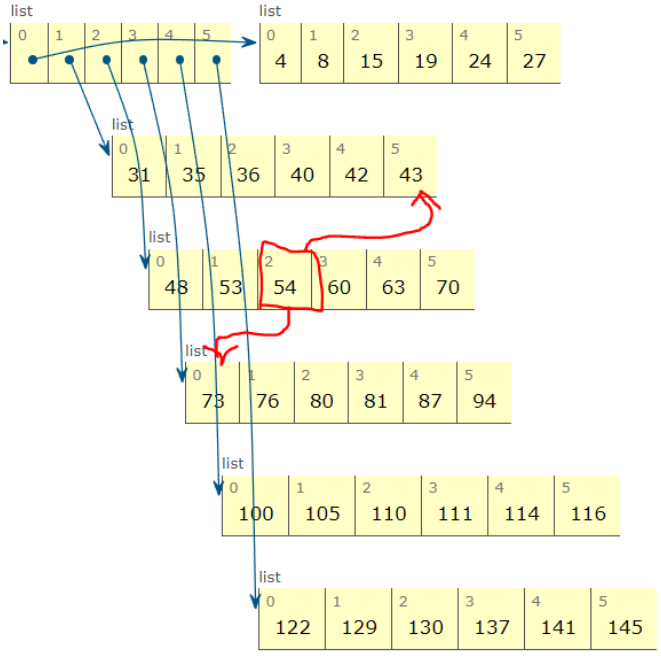
Méthode gloutonne

La méthode gloutonne consiste simplement à parcourir la matrice et tester si la position dans laquelle nous nous trouvons correspond à notre nombre. La complexité va donc très vite augmenter si notre nombre se trouve en fin de matrice.

Léo Chardon
Remy Barberet

Méthode optimale

La méthode optimale consiste à appliquer le principe de la dichotomie mais sur la matrice, donc à la fois sur les lignes et les colonnes



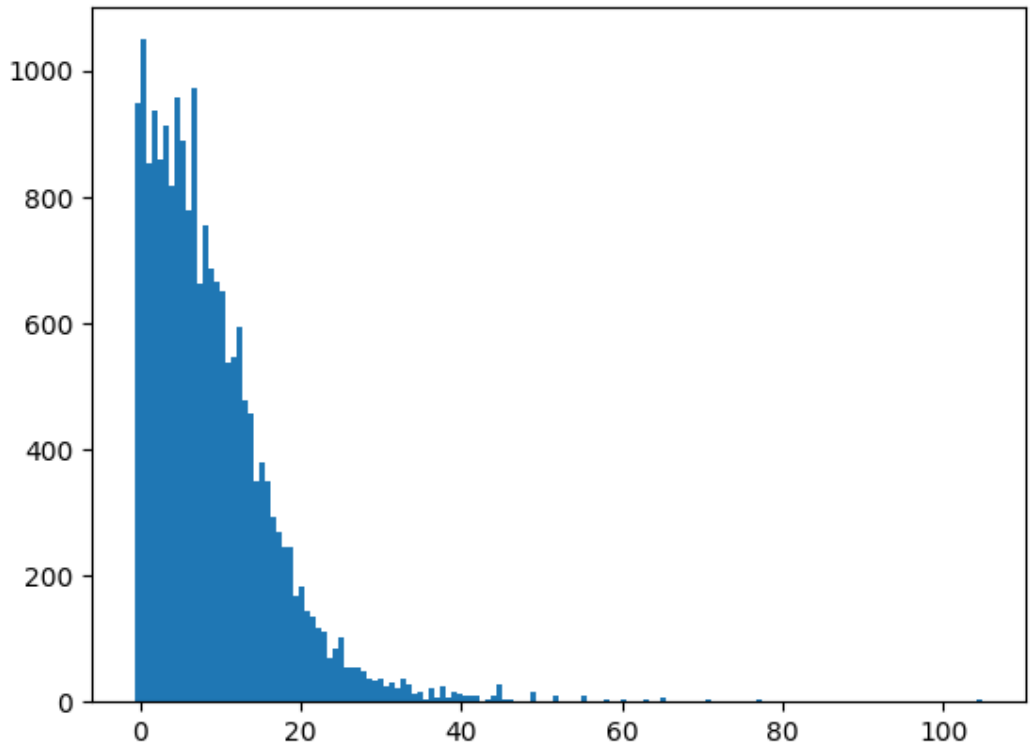
Admettons que nous cherchons le nombre 19.
Nous avons notre matrice, on commence par regarder au centre de celle-ci, donc 5 divisé sans reste par 2 il reste 2. On a notre ligne, maintenant pareil pour les colonnes.
On regarde donc notre numéro 54 et on va regarder si le nombre recherché est plus petit que le nombre de droite de la ligne d'en haut, soit 43. Si il est plus petit alors on applique la dichotomie dans la matrice et on répète l'opération. Si il est plus grand alors on applique la dichotomie dans la liste où nous sommes et nous trouvons notre nombre.

Si 54 avait été trop petit comparé à un nombre recherché, alors on aurait regardé 73 et répété l'opération comme avant, soit on avance dichotomiquement dans la matrice soit on effectue la dichotomie dans la liste actuelle.

On rappelle que la dichotomie consiste à regarder au centre d'une liste triée et à se diriger avant l'élément ou après suivant ce que l'on cherche, on raccourcit donc notre liste à chaque étape puis on regarde encore au centre.
L'avantage de cette méthode est qu'il est très facile pour l'ordinateur de rechercher un nombre grand dans une liste triée sans pour autant parcourir tous les éléments car il va éliminer les nombres trop petits automatiquement.
Le désavantage est plus humain car l'algorithme peut parfois être vastidieux comme c'est le cas pour une matrice (double dichotomie). Il faut aussi préalablement que la liste soit triée

Léo Chardon
Remy Barberet

Comparaison



histogramme de la distance relative avec 20000 runs

Ici la différence est assez flagrante, l'algorithme dichotomique met beaucoup d'itération avant de trouver le nombre. Cela s'explique par le fait qu'il coupe la matrice et les listes pour faire sa recherche.

```
100% (20000 of 20000) |###################| Elapsed Time: 0:00:01 Time:  0:00:01
Greedy:
    mean:48.94185
    std:37.24074876499532
Optimal:
    mean:5.24935
    std:1.4946820991434935
```

En moyenne l'algorithme optimal va mettre 5 itérations pour trouver son nombre, contre 49 pour l'algorithme glouton, 49 doit sûrement correspondre au maximum de nombre présent dans la matrice.