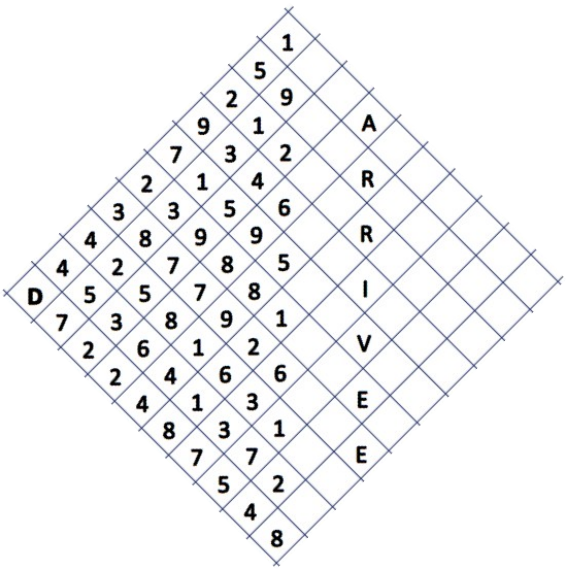
Algorithmes avancés

Algorithmes gloutons

Les algorithmes gloutons sont une grande famille d'algorithmes qui permettent de résoudre des problèmes complexes, avant de définir ce qui caractérise ce type d'algorithme voyons un exemple.

I Un exemple

Sur la grille ci-contre, on part de la case tout à gauche marquée de la lettre D. On souhaite atteindre les cases vides sur la partie droite en se déplaçant de case en case.



Lorsqu'on est sur une case on peut se déplacer sur une des deux cases voisines situées sur la droite. On note S la somme de toutes les cases traversées.

Par exemple on peut effectuer la trajectoire suivante :

D – 7 – 5 – 3 – 5 – 7 – 9 – 8 – 9 – 6

qui conduit à S = 59.

On cherche à effectuer la trajectoire qui rend la somme S la plus petite possible.

Exercice 1

- *Cherchons à comprendre la nature du problème.*
  Que cherche-t-on à sélectionner ? Quelles sont les contraintes ? Quelle est l'optimisation recherchée ?
- *Définissons un algorithme glouton*
  Définir une règle de choix qui vous donne une solution respectant les contraintes. Quelle trajectoire et quelle somme S obtenez-vous avec cette règle ?
- *Est-il optimal ?*
  Sur cette grille, en cherchant bien, la trajectoire optimale donne une somme S=23. Votre algorithme glouton a-t-il trouvé cette trajectoire optimale ?
- Pour obtenir la solution optimale de façon certaine on souhaite trouver toutes les trajectoires possibles et calculer pour chacune d'elles la somme associée.
  Quel est le nombre de trajectoires au total ? Cette méthode est-elle beaucoup plus coûteuse en nombre de calculs ?

II Définition

Définition

Un problème d'optimisation est un problème où il s'agit de déterminer un minimum, un maximum glouton est un algorithme d'optimisation.

Un algorithme glouton (*greedy algorithm* en anglais) est une méthode de résolution de ce type de problème. Un algorithme est dit glouton si à chaque étape un choix que l'on estime optimal localement est effectué selon une stratégie définie à l'avance. On espère obtenir à la fin un optimum global, mais cela n'est pas toujours le cas. Le choix fait à une certaine étape n'est jamais remis en question, il n'y a donc pas de retour en arrière permettant d'explorer d'autres voies.

Dans le premier exemple, on veut minimiser trouver la trajectoire dont la somme est minimale, L'algorithme glouton consiste à décider qu'à chaque étape on choisit la case contenant le nombre le plus petit. On a vu que cet algorithme n'était pas optimal. Chercher à savoir si un algorithme est optimal est un travail des chercheurs en sciences informatiques. Pour certains algorithmes, ces derniers ont montré qu'ils pouvaient être optimaux si certaines conditions étaient réunies. On a enfin vu que par contre l'algorithme glouton utilisé demandait beaucoup moins de calculs que celui en « force brut » qui explorerait toutes les trajectoires possibles. Si les trajectoires avaient été un peu plus longues, explorer toutes les trajectoires deviendrait vite trop coûteux en temps de calcul. Dans ce cas la démarche gloutonne est de complexité linéaire alors que parcourir l'ensemble des trajets est de complexité exponentielle.

On utilise des algorithmes gloutons lorsque l'on sait qu'ils sont efficaces dans des problèmes d'optimisation où explorer toutes les possibilités demanderait trop de calculs. Des problèmes classiques sont le rendu de monnaie, la répartition des ressources (planning), le problème du sac à dos (*knapsack problem*), la coloration d'une carte, la compression de données ou le voyageur de commerce.

Le problème du sac à dos est équivalent minimiser la quantité de matériaux dans la découpe de matériaux, charger de manière optimale une cargaison ou choisir le problème le plus rentable dans le monde de la finance. C'est un problème dit NP-complet, c'est-à-dire parmi les plus difficiles à résoudre de façon optimale en un temps raisonnable.

III Optimisation d'un planning

On dispose du planning d'un festival culturel qui propose des spectacles sur cinq scènes différentes.
Un festivalier, qui arrive à 10:00, cherche à choisir des spectacles dans ce planning (**sélection**) en cherchant à voir le plus grand nombre de spectacles possibles dans sa journée (**maximiser_une quantité**) et en s'imposant de voir chaque spectacle en entier (**contrainte**).
Il peut envisager deux algorithmes gloutons différents.

Règle de choix de l'algorithme glouton A :
À chaque étape choisir, parmi les cinq scènes, le prochain spectacle qui *commence* en premier (Basée sur l'idée que moins on attend entre deux spectacles, plus on verra de spectacles.)

Règle de choix de l'algorithme glouton B :
À chaque étape choisir, parmi les cinq scènes, le prochain spectacle qui *finit* en premier. (Basée sur l'idée que plus un spectacle finit tôt, plus il y aura de la place pour les spectacles suivants.)

Exercice 2

Faire fonctionner l'algorithme A en utilisant le planning ci-contre.
Pour cela compléter la liste dont on a déjà complété les deux premières étapes.

1. spectacle 3A
2. spectable 2B
3. …

Faire de-même avec l'algorithme B.
On donne aussi les premières étapes :

1. spectacle 4A
2. spectacle 2A
3. …

L'algorithme glouton A abouti à une moins bonne solution que l'algorithme glouton B (10 spectacles contre 12 spectacles). En effet, le A choisit parfois de très longs spectacles (comme le 3A) ce qui est contradictoire avec le fait d'en voir beaucoup.

Remarque : On peut démontrer que l'algorithme glouton B fournit systématiquement une solution optimale (ainsi, sur cet exemple, on ne peut pas trouver de solution permettant de voir plus de 12 spectacles en entier).

Sources :

- https://fr.wikipedia.org/wiki/Algorithme_glouton

Pour aller plus loin :

- https://fr.wikipedia.org/wiki/Probl%C3%A8me_du_sac_%C3%A0_dos
- https://interstices.info/le-probleme-du-voyageur-de-commerce/
- https://culturemath.ens.fr/thematiques/lycee/algorithmes-gloutons
- https://www.youtube.com/watch?v=yrOj8XvroIE