ALGORITHMES GLOUTONS

1 – LES ALGORITHMES GLOUTONS

1.1 – Présentation

Les algorithmes gloutons constituent une grande famille d'algorithmes et sont très utilisés dans les **problèmes d'optimisation**. Un algorithme glouton est une méthode de recherche d'une **solution locale** qui va apporter une **solution globale non nécessairement optimale**. Le problème se ramène à chaque étape à un problème plus simple et chaque étape sélectionne l'une des meilleures possibilités et ne remet jamais en cause les choix précédents.

On utilise un algorithme glouton lorsqu'i faut effectuer une **sélection** sur un ensemble d'objets en cherchant à **maximiser ou minimiser** une grandeur tout en respectant certaines **contraintes**.

La sélection est effectuée en appliquant à chaque étape une **règle de choix** définie à l'avance qui effectue ce qui semble être le meilleur choix sur le moment en espérant arriver à la fin à la solution optimale.

1.2 – Exemple 1

On cherche à sélectionner cinq nombres dans le tableau ci-dessous (**sélection**) en cherchant à avoir leur somme la plus grande possible (**maximiser une grandeur**) et en s'interdisant de choisir deux nombres voisins (**contrainte**).

15	**4**	**20**	**17**	**11**	**8**	**11**	**16**	**7**	**14**	**2**	**7**	**5**	**17**	**19**	**18**	**4**	**5**	**13**	**8**

<u>Règle de choix de l'algorithme glouton</u> : A chacune des cinq étapes, choisir le plus grand nombre possible dans les choix restants (Basée sur l'idée que plus on choisit un grand nombre à chaque étape, plus on aura un grand résultat à la fin)

Déroulé de l'algorithme glouton :

– Étape 1 : on choisit le 20 et on élimine le 4 et le 17 voisins (contrainte).

15	4	20	17	11	8	11	16	7	14	2	7	5	17	19	18	4	5	13	8

– Étape 2 : on choisit le 19 et on élimine le 17 et le 18 voisins (contrainte).

15	4	20	17	11	8	11	16	7	14	2	7	5	17	19	18	4	5	13	8

– Étape 3 : on choisit le 16 et on élimine le 11 et le 7 voisins (contrainte).

15	4	20	17	11	8	11	16	7	14	2	7	5	17	19	18	4	5	13	8

– Étape 4 : on choisit le 15.

15	4	20	17	11	8	11	16	7	14	2	7	5	17	19	18	4	5	13	8

– Étape 5 : on choisit le 14 élimine le 2 voisin.

15	4	20	17	11	8	11	16	7	14	2	7	5	17	19	18	4	5	13	8

La **maximisation gloutonne** obtenue a pour valeur $\mathbf{20 + 19 + 16 + 15 + 14 = 84}$.

L'algorithme glouton est-il un "bon" algorithme sur cet exemple ?

Argument pour répondre Non	La meilleure solution est $\mathbf{20 + 19 + 17 + 16 + 15 = 87}$
Argument pour répondre "Oui"	Sur de longues listes, les algorithmes permettant de trouver la meilleure solution (on dit optimale) auraient une complexité beaucoup plus grande que notre algorithme glouton. De plus, on peut démontrer que notre algorithme glouton fournit dans tous les cas une solution "**presque optimale**".

1.3 – Exemple 2

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 algorithme gloutons différents.

<u>Règle de choix de l'algorithme glouton A :</u>
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).

<u>Règle de choix de l'algorithme glouton B</u> :
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).

Déroulé de l'algorithme glouton A :

Etape	1	2	3	4	5	6	7	8	9	10
Début	**10h**	**12h30**	**14h**	**15h**	**16h15**	**16h30**	**18h45**	**21h30**	**0h**	**1h30**
Spectacle	**3A**	**2B**	**3B**	**2C**	**3D**	**2D**	**4D**	**3F**	**3G**	**2G**

Déroulé de l'algorithme glouton B :

Etape	1	2	3	4	5	6	7	8	9	10	11	12
Début	**11h**	**11h45**	**13h30**	**14h45**	**16h**	**16h30**	**17h30**	**18h30**	**20h**	**23h**	**1h30**	**3h**
Spectacle	**4A**	**2A**	**4B**	**3B**	**2C/3C**	**3D**	**5D**	**5E**	**5F**	**3F**	**3G**	**2G**

L'algorithme glouton A abouti à une moins bonne solution que l'algorithme glouton B (10 spectacles contre 12 spectacles). En effet, l'algorithme A choisi 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).

1.4 – Conclusion

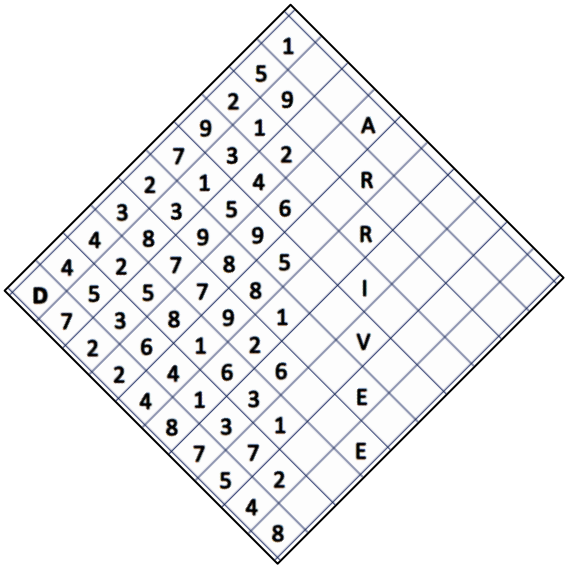
Les algorithmes gloutons s'utilisent lorsqu'on cherche à effectuer une sélection en optimisant une quantité et en respectant certaines contraintes. Ils se basent sur une règle de choix définie à l'avance qui effectue ce qui semble être le meilleur choix sur le moment.

Dans certaines situations comme l'algorithmes B de l'exemple 2, ils permettent de trouver systématiquement la meilleure solution. Dans d'autres situations comme l'exemple 1, sans trouver la meilleure solution, ils permettent toutefois de trouver une "bonne" solution sans effectuer "beaucoup de calculs".

2 – EXERCICES

2.1 – Exercice 1

Sur la grille ci-contre, en partant de la case tout à gauche marquée de la lettre D, nous souhaitons atteindre les cases vides sur la partie droite en se déplaçant de case en case. Lorsque nous sommes sur une case, il est possible de se déplacer sur une des deux cases adjacentes situées sur la droite. On note S la somme de toutes les cases traversées.



Par exemple il est possible d'effectuer la trajectoire suivante : **D – 7 – 5 – 3 – 5 – 7 – 9 – 8 – 9 – 6** qui conduit à **S = 59**.

L'objectif de l'algorithme recherché est d'effectuer **la trajectoire qui rend la somme S la plus petite possible.**

Question 1 : Indiquez ce que nous cherchons à sélectionner, quelles sont les contraintes et quelle est l'optimisation recherchée.

Question 2 : Déterminez une règle de choix qui permet de définir un algorithme glouton. Préciser alors quelle trajectoire et quelle somme S obtenez-vous sur cette grille

Question 3 : La trajectoire optimale, sur cette grille donne une somme S = 23. Vérifiez si votre algorithme glouton a trouvé la trajectoire optimale. Indiquez dans le cas d'une réponse négative si la trajectoire trouvée est "proche" de la trajectoire optimale.

Question 4 : Pour obtenir la solution optimale de façon certaine une solution consiste à trouver toutes les trajectoires possibles et à calculer pour chacune d'elles la somme associée. Déterminer le nombre de trajectoires possibles et justifier que cette méthode est coûteuse en termes de calculs par rapport à l'algorithme glouton.

2.2 – Exercice 2

On souhaite charger des containers de marchandises sur des wagons qui peuvent transporter 60 tonnes chacun. Les masses (en tonnes) des containers sont représentées dans un tableau T de nombres entiers. On suppose qu'on peut charger autant de containers qu'on le souhaite sur un wagon tant que la masse des containers déposés ne dépasse pas 60 tonnes. Par exemple avec le tableau T des 18 masses suivantes :
`T = [32, 1, 4, 11, 16, 38, 30, 15, 40, 20, 26, 5, 25, 14, 44, 17, 7, 6]`

Il possible de charger les 18 containers en les répartissant sur N = 7 wagons en les répartissant ainsi dans N=7 sous-tableaux : `W1 = [32, 20, 4]` ; `W2 = [30, 26]` ; `W3 = [11, 44]` ; `W4 = [40, 15, 5]` ; `W5 = [38, 17]` ; `W6 = [14, 16, 25, 1]` et `W7 = [6, 7]`.

Nous recherchons un algorithme glouton qui optimise la **répartition des containers de T dans le plus petit nombre N de wagons**.

Question 1 : Dans cet exemple la sélection effectuée est une "répartition" du tableau T en N sous-tableaux, précisez alors quelle est la contrainte et quelle est l'optimisation recherchée.

Pour charger un wagon à partir des containers qui restent à charger, on considère la règle de choix suivante : **Tant la masse ne dépasse pas les 60 tonnes, choisir le container le moins lourd** et le charger sur le wagon (Basée sur l'idée que plus on cherche à mettre de containers sur un wagon, moins on utilisera de wagons). Nous obtenons alors la répartition suivante :

`W1 = [1, 4, 5, 6, 7, 11, 14]` ; `W2 = [15, 16, 17]` ; `W3 = [20, 25]` ; `W4 = [26, 30]` ; `W5 = [32]` ; `W6 = [38]` ; `W7 = [40]` et `W8 = [44]`.

Question 2 : Précisez si cette répartition est-elle une solution optimale.

Question 3 : Appliquez cette règle sur le tableau ci-dessous. Précisez si la solution est optimale.
`T2 = [4, 14, 32, 9, 31, 42, 12, 5, 31, 29, 39, 1, 3, 2, 5, 12, 38]`

Question 4 : Justifiez pourquoi cette règle de choix est n'est pas très efficace lorsqu'il y a beaucoup de containers de plus de 30 tonnes.