M. GLOUTON

ALGORITHMES GLOUTONS

- *Problème du sac à dos*
- *Problème du rendu de monnaie*

“

Le principe d'une méthode gloutonne : Avaler tout ce qu'on peut

Cela revient à construire au fur et à mesure une solution en faisant les choix qui paraissent optimaux localement.

Dans certains cas, cela donnera finalement la meilleure solution : on parlera d'algorithmes gloutons exacts.

Dans d'autres, non, on parlera d'heuristiques gloutonnes

”

Numérique et Sciences Informatiques
1ère

Support de cours :
Jean-Christophe BONNEFOY

Objectifs :

- Résoudre un problème grâce à un algorithme glouton.

1. Généralités

Optimiser un problème, c'est déterminer les conditions dans lesquelles ce problème présente une caractéristique spécifique. Par exemple, déterminer le minimum ou le maximum d'une fonction est un problème d'optimisation. On peut également citer la répartition optimale de tâches suivant des critères précis, le problème du rendu de monnaie, le problème du sac à dos, la recherche d'un plus court chemin dans un graphe, le problème du voyageur de commerce. De nombreuses techniques informatiques sont susceptibles d'apporter une solution exacte ou approchée à ces problèmes. Certaines de ces techniques, comme l'énumération exhaustive de toutes les solutions, ont un coût machine qui les rend souvent peu pertinentes au regard de contraintes extérieures imposées (temps de réponse de la solution imposé, moyens machines limités).

Les algorithmes gloutons constituent une alternative dont le résultat n'est pas toujours optimal. Plus précisément, ces algorithmes déterminent une solution optimale en effectuant successivement des choix locaux, jamais remis en cause. Au cours de la construction de la solution, l'algorithme résout une partie du problème puis se focalise ensuite sur le sous-problème restant à résoudre. Une différence essentielle avec la programmation dynamique est que celle-ci peut remettre en cause des solutions déjà établies. Au lieu de se focaliser sur un seul sous-problème, elle explore les solutions de tous les sous-problèmes pour les combiner finalement de manière optimale.

Définition :

Les algorithmes gloutons constituent une grande famille d'algorithmes.

- On peut les utiliser lorsqu'on a une sélection à effectuer sur un ensemble d'objets en cherchant à **maximiser ou minimiser** une certaine 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.

Le principal avantage des algorithmes gloutons est leur facilité de mise en œuvre. En outre, dans certaines situations dites canoniques, il arrive qu'ils renvoient non pas un optimum mais l'optimum d'un problème.

2. Premier exemple

On cherche à sélectionner 5 nombres de la liste suivante (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

Règle de choix de l'algorithme glouton :
À 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 en l'entourant et on raye le 4 et le 17 voisins (à cause de la contrainte).
Étape 2 : on choisit le 19 en l'entourant et on raye le 17 et le 18 voisins (à cause de la contrainte).
Étape 3 : on choisit le 16 en l'entourant et on raye le 11 et le 7 voisins (à cause de la contrainte).
Étape 4 : on choisit le 15 en l'entourant et on ne raye rien du tout (le 4 est déjà rayé depuis l'étape 1).
Étape 5 : on choisit le 14 en l'entourant et on raye le 2 voisin (le 7 est déjà rayé depuis l'étape 3).

Maximisation gloutonne obtenue :

20+19+16+15+14 = 84

Sur cet exemple, est-ce que l'algorithme glouton est un "bon" algorithme ?

- Non : on peut trouver une meilleure solution (20, 18, 17, 16 et 15 qui conduisent à un total de 86)
- 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"*.

3. Le problème du sac à dos

Imaginons maintenant la situation suivante : *Un cambrioleur possède un sac à dos d'une capacité maximale de 30 Kg. Au cours d'un de ses cambriolages, il a la possibilité de dérober 4 objets A, B, C et D. Voici un tableau qui résume les caractéristiques de ces objets :*

Objet	A	B	C	D
Masse	13 Kg	12 Kg	8 Kg	10 Kg
Valeur	700 €	400 €	300 €	300 €

Le but, ici, est de déterminer les objets que le cambrioleur aura intérêt à dérober, sachant que :

- Tous les objets dérobés devront tenir dans le sac à dos (30 Kg maxi)
- Le cambrioleur cherche à obtenir un gain maximum.

Appliquons une méthode gloutonne à la résolution de ce problème appelé *problème du sac à dos* :

- Sachant que l'on cherche à maximiser le gain, commençons par établir un tableau nous donnant la "valeur massique" de chaque objet (pour chaque objet on divise sa valeur par sa masse) :

Objet	A	B	C	D
Valeur massique	54 € /Kg	33 € /Kg	38 € /Kg	30 € /Kg

- On classe ensuite les objets par ordre décroissant de valeur massique : A - C - B - D
- Enfin, on remplit le sac en prenant les objets dans l'ordre et en s'arrêtant dès que la masse limite est atteinte. C'est ici que ce fait "le choix glouton", à chaque étape, on prend l'objet ayant le rapport "valeur-masse" le plus intéressant au vu des objectifs :
  - Étape 1 : on prend A, le poids total du sac est donc de 13 Kg
  - Étape 2 : on prend C, le poids total est de 13+8=21 Kg
  - Étape 3 : on prend B, le poids total est de 13+8+12=33 Kg => impossible, on dépasse les 30 Kg, donc on repose B et l'algorithme est terminé.

Finalement, Le sac est composé de 2 objets : A et C pour une valeur totale de 1000 € et une masse totale de 21 Kg.

La solution trouvée ci-dessus est-elle optimale ?

On constate rapidement que la réponse est non, car le couple A+B permet d'atteindre une valeur de 1100 € pour une masse de 25 Kg. Dans notre problème, la méthode gloutonne ne nous donne pas une solution optimale.

4. Le problème du rendu de monnaie

Imaginons maintenant la situation suivante : *Je fais un achat dans un magasin pour une valeur de 51€, je donne un billet de 100€. La caissière doit donc me rendre 49€. Pour être le plus efficace possible, son patron lui demande de rendre la monnaie pour chaque client avec une minimum de pièces (ou billets). On prendra comme référence dans son tiroir-caisse seulement les cas suivants (avec des entiers c'est plus simple pour l'instant)*

Valeur	1 €	2 €	5 €	10 €	20 €	50 €	100 €
Type	Pièce		Billet				

En pratique, sans s'en rendre compte généralement, tout individu met en œuvre un algorithme glouton. Il choisit d'abord la plus grandeur valeur de monnaie, parmi 1, 2, 5, 10, 20 contenue dans 49 euros. La caissière va donc rendre ici d'abord deux fois un billet de 20 euros. La somme de 9 euros restant à rendre, elle choisira ensuite de rendre un billet de 5 euros, puis deux pièces de 2 euros.

Cette stratégie est gagnante pour la somme de 49 euros. L'est-elle pour n'importe quelle somme à rendre ?
On peut montrer que la réponse est positive (mais cela dépasse le cadre du programme de NSI) pour ce système monétaire (euros). Pour cette raison, un tel système de monnaie est qualifié de canonique.

Prenons maintenant le cas du système monétaire britannique d'avant 1971 suivant :

Valeur	1 p	3 p	6 p	12 p	24 p	30 p	60 p
Nom donnée à la pièce	1 Penny			1 Shilling	1 Florin	1 Demi-couronne	1 Couronne

Appliquons la méthode gloutonne pour rendre 49p. Sans trop de difficultés, nous proposons le rendu 49=30+12+6+1, soit 4 pièces alors que la solution optimale est 49=2×24+1, soit 3 pièces. La réponse à cette difficulté passe par la programmation dynamique, thème abordé en spécialité NSI de classe de terminale.