TD2 (préparation)

QUESTION D'EXAMEN

PLUTOT BRUTE OU PLUTOT GLOUTONNE ?

On se propose de comparer la complexité de deux stratégies de construction d'algorithmes (force brute et algorithme glouton) sur un même problème.

Questions préliminaires (5 points)

Pour le moment, restons sur des généralités.

1. Pour quel(s) type(s) de problème utilise-t-on un **algorithme glouton** ? Quel est son principe général ?

2. Un **algorithme glouton** peut être considéré comme construit en 2 parties, la première effectue un traitement sur les données selon un certain critère, la seconde construit la solution ; donnez cet algorithme (choisissez la façon de l'écrire qui vous semble la plus opportune)

3. Que pouvez-vous dire de la complexité d'un **algorithme glouton** en général ?

4. Quel est le principe d'un **algorithme de force brute** ?

5. Que pouvez-vous dire de la complexité d'un **algorithme de force brute** en général ?

Application à un problème (8 points)

Enoncé

On considère le problème des pièces de monnaies.
Soient différentes pièces de monnaie exprimées en centimes (par exemple 2, 5, 2, 20, 10, 5 et 2) et une somme à rendre (par exemple 32). La question est de savoir s'il est possible de rendre la monnaie à partir des pièces disponibles (ici une première solution est 2,5,20 et 5 ; une autre 2, 20 et 10)

a) Approche Force brute ***(2+1 points)***

Donner l'algorithme de force brute appliquée à ce problème SANS CHERCHER à minimiser le nombre de pièces à rendre.

Complexité dans ce cas précis (on notera N le nombre de pièces)

b) Approche gloutonne **(1+2+1+1 points)**

Il vous est demander d'appliquer à ce problème une approche gloutonne. Attention, nous appliquons strictement l'approche gloutonne (nous verrons après si l'optimum est atteint).

Pour cela nous allons utiliser comme critère la valeur des pièces.

Les données initiales sont 2, 5, 2, 20, 10, 5 et 2

A quoi correspondent ces données après la première phase de l'algorithme, quelle est la classe de complexité de cette première phase ?

Comment s'applique ensuite la deuxième partie de l'algorithme glouton, dérouler « manuellement » son exécution ici à partir des données de la question précédente
Important : si vous ne savez pas répondre à la question précédente, vous utiliserez comme données 2, 5, 2, 20, 10, 5 et 2 pour montrer que vous savez quand même appliquer un algo glouton.

Quelle est la complexité de cette 2ème partie ?

Quelle est la complexité de l'algorithme glouton appliqué à ce problème de pièce de monnaie (on notera N le nombre de pièces)?

TD2

MISE EN ŒUVRE DE FORCE BRUTE

QUESTIONS DE COURS/EXAMEN COMPLEXITE

Approche force brute (10 points)

On souhaite résoudre le problème suivant (pas la classe des problèmes de ce type) avec une approche « force brute ».

DEUX+CINQ=SEPT

```
    D   E   U   X
+   C   I   N   Q
  -------------------
    S   E   P   T
```

Chaque chiffre est remplacé par une lettre, ce problème peut alors se ramener à un système de relations entre lettres ou entre mots.

Attention rien ne dit qu'un même chiffre n'est pas associé à deux lettres différentes.

Il s'agit de trouver à quoi corresponde chaque lettre en terme de chiffre pour que la somme soit juste. Une solution évidente est que toutes les lettres correspondent à la valeur nulle.

On souhaite résoudre le problème en faisant appel à un algorithme « force brute ».

Questions introductives (3 points)

1. Qu'est-ce qu'un algorithme de force brute ? Quel est son principe général ?

1. Quelles sont les lettres représentant l'espace d'états (par opposition à celle se déduisant du choix de deux autres) ? Combien y en a t-il ? (Dit autrement quelles sont les inconnues du système)

3. Combien de valeurs peuvent prendre chacune de ces lettres ?

4. Quelle est la taille de l'espace d'état, la complexité d'un algorithme de force brute sur le problème ?

Construction de l'algorithme (1+3+3 points)

5. On représente chaque lettre du problème à trouver par une variable du même nom. Par exemple la lettre I sera associé à une variable I (ou i) dans l'algorithme. Comment itérer sur cet espace d'états (cf questions 1 et 2) ?
   Remarque : la manière la plus simple n'est peut être pas la meilleure mais si elle permet de répondre à la question elle est satisfaisante *ici*.

5. On suppose l'existence d'une fonction OK qui, étant données les valeurs de ces lettres (cf point 1) représentant l'espace d'états, retourne un booléen indiquant si ces valeurs sont solutions du problème ou non.

   Donner l'algorithme de force brute qui utilise cette méthode/fonction

7. Ecrire la méthode ok (définir les paramètres et leur type, le type du résultat et l'algorithme associé, ...)

Coder cette solution.

Force brute (7 points)

Soit le problème qui demande à équilibrer une répartition de poids sur une balance à 2 plateaux dont l'énoncé est rappelé ci-dessous.

On dispose d'un ensemble de n poids w_1, w_2,..,w_n que l'on doit répartir en deux sous-ensembles de masses égales.

On pose $x_i = 0$ si la masse i est à gauche, et $x_i=1$ sinon.

On souhaite savoir si il existe des valeurs de x_i telles que

$$\sum_{i=1}^{n} x_i w_{i=} \sum_{i=1}^{n} (1 - x_i) w_i$$

Complexité (1+2 points)

Qu'est-ce qui représente une solution au problème ?

Qu'est-ce qui représente l'espace d'état/l'espace des solution possibles? Combien de valeurs peuvent prendre chacun des éléments d'une solution et donc quelle est la taille de l'espace d'état de ce problème ?

Résolution (1 +3)

Quel est le principe d'un algorithme de force brute et comment ce principe s'applique ici (on pourra se contenter de la mise en œuvre la plus simple) ?

Donner maintenant l'algorithme de force brute qui résout le problème de l'équilibrage des plateaux.

Coder cette solution.

Passer à sa généralisation

Bonus

On souhaite généraliser le problème à k plateaux (ie on souhaite répartir les poids parmi k sous-ensembles).

Solution exacte : la différence entre les masses de chaque plateau est nulle

Solution approchée : la différence entre les masses de chaque plateau est nulle ou inférieure à un seuil

Solution optimisée : on minimise le majorant de la différence