NSI · PREMIÈRE / COURS

Les algorithmes gloutons

Astrolab Cours · astrolabcours.fr

01 Le problème

Rendre la monnaie avec le moins de pièces possible, remplir un sac de randonnée avec les objets les plus utiles sans dépasser son poids, planifier un maximum de réunions dans une salle. Essayer toutes les combinaisons est souvent beaucoup trop long. Existe-t-il un raccourci ?

01 L'idée : le plus gros bonbon d'abord

Un enfant devant un bocal prend toujours le plus gros bonbon qui reste, sans réfléchir à la suite. C'est rapide, et souvent pas mal. Mais parfois, en prenant le plus gros, il laisse un bocal où rien ne va ensemble. Un **algorithme glouton**, c'est cet enfant.

01 Le principe

> Un **algorithme glouton** construit une solution étape par étape en faisant à chaque fois le **meilleur choix immédiat**, sans jamais revenir en arrière.

C'est rapide, mais le choix localement le meilleur ne mène pas toujours à la meilleure solution globale.

01 Le rendu de monnaie

On rend la plus grosse pièce possible, tant qu'il reste quelque chose à rendre :

```
def rendu(somme, pieces):
    """pieces est triée par ordre décroissant."""
    res = []
    for p in pieces:
        while somme >= p:
            res.append(p)
            somme = somme - p
    return res

rendu(68, [50, 20, 10, 5, 2, 1])   # [50, 10, 5, 2, 1]
```

Avec les pièces en euros, le glouton donne toujours le minimum de pièces. Avec des pièces de 1, 3 et 4, pour rendre 6, il donne 4 + 1 + 1 alors que 3 + 3 suffit.

© Astrolab Cours · astrolabcours.fr/nsi/premiere/algorithmes-gloutons/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

Le sac à dos

Un sac supporte 10 kg. Objets : A (7 kg, valeur 42), B (5 kg, 30), C (5 kg, 30). Stratégie gloutonne : prendre d'abord les objets de meilleur rapport valeur/poids. Les trois ont un rapport de 6. Si on prend A en premier, il reste 3 kg : valeur totale 42. Mais B + C donnent 60.

Exemple guidé

Rendre 8 € avec des pièces de 1, 4 et 6 € en utilisant l'algorithme glouton.

```
def rendu(somme, pieces):
    res = []
    for p in sorted(pieces, reverse=True):
        while somme >= p:
            res.append(p)
            somme = somme - p
    return res

rendu(8, [1, 4, 6])   # [6, 1, 1]
```

1. Le glouton prend d'abord la plus grosse pièce possible : 6, il reste 2.
2. Puis 1 et 1 : **3 pièces**.
3. Or $4 + 4$ fait 8 avec **2 pièces** : le glouton n'a pas trouvé l'optimum.

Avec les euros (1, 2, 5, 10, 20, 50…), le glouton donne toujours le minimum : le système est dit **canonique**.

Retour au problème

Le glouton est un raccourci : à chaque étape, il fait le meilleur choix immédiat, sans revenir en arrière. C'est très rapide, mais pas toujours optimal. Il faut vérifier, ou prouver, qu'il convient au problème.

Les pièges

- Croire que le glouton est toujours optimal.
- Oublier de trier les pièces ou les objets avant de commencer.
- Le glouton ne revient jamais sur un choix déjà fait.

© Astrolab Cours · astrolabcours.fr/nsi/premiere/algorithmes-gloutons/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits