La programmation dynamique

Claude-Guy Quimper

Idée

- En général, cette technique sert à résoudre des problèmes d'optimisation.
- Il existe plusieurs solutions au problème, mais on recherche celle qui optimise une fonction objectif.
- Le but est d'exprimer la solution d'une grande instance en fonction des solutions de plus petites instances.
- On calcule les valeurs objectives des solutions des plus petites instances au plus grandes, jusqu'à l'instance désirée soit résolue.
- Toutes les valeurs objectives sont maintenues dans un tableau.
- Pour trouver la solution, on remonte dans le tableau pour identifier quels choix ont mené à la valeur objective.

Le plus long palindrome

- Soit S une chaîne de caractères. Quel est le plus long palindrome que l'on peut obtenir en retirant des caractères à S?
  - kayak
  - elle
  - coloc
  - rêver
  - sexes
  - Éric notre valet alla te laver ton ciré
    (fonctionne également avec Luc)
  - eibohphobie

Le sous-problème

- Soit P[i,j] la longueur du plus long palindrome présent dans la sous chaîne S[i..j-1]

$$P[i,j] = \begin{cases} 0 & \text{si } i = j \\ 1 & \text{si } i+1 = j \\ 2 + P[i+1, j-1] & \text{si } S[i] = S[j-1] \\ \max(P[i, j-1], P[i+1, j]) & \text{sinon} \end{cases}$$

La plus longue sous-séquence

- Soit T un vecteur de nombres. Il faut supprimer le minimum d'éléments du vecteur et obtenir une séquence croissante.
- Autrement dit, il faut trouver la plus longue sous-séquence croissante.

Le sous-problème

- Soit L[i] la longueur de la plus grande sous-séquence croissante se terminant en position i.

$$L[i] = \begin{cases} 0 & \text{si } i = 0 \\ 1 & \text{si } i = 1 \\ 1 + \max_{k \in \{j \in [1,i) | t_j < t_i\} \cup \{0\}} L[k] & \text{si } i > 1 \end{cases}$$

- La plus longue séquence a une longueur de max(L[1], ..., L[n])

Le jeu des pièces d'or

- Des sacs de pièces d'or sont alignés sur une table.
- La quantité de pièces d'or que contient chaque sac est donnée par un vecteur Q.
  - Ex: Q[1..n] = [7, 10, 3, 2, 1]
- Vous êtes le premier joueur et devez prendre un sac à une extrémité de l'alignement. Votre adversaire en fera de même et le jeu se poursuivra à tour de rôle jusqu'à ce que la table soit vidée.
- Par quelle extrémité commencez-vous?

Sous-problème

- Soit P[i,j] le nombre de pièces d'or que vous pouvez prendre s'il ne reste que les pièces Q[i], Q[i+1], Q[i+2], ..., Q[j] sur la table.

$$P[i,j] = \begin{cases} Q[i] & \text{si } i = j \\ \max(Q[i], Q[j]) & \text{si } i+1 = j \\ \max(Q[i] + \min(P[i+2,j], P[i+1,j-1]), & \text{si } i+1 < j \\ \qquad Q[j] + \min(P[i+1,j-1], P[i,j-2])) & \end{cases}$$

Le jeu des oeufs

- Le problème implique n oeufs identiques et un édifice de k étages.
- Vous voulez savoir le plus haut étage duquel vous pouvez laisser tomber l'oeuf sans qu'il ne se casse.
- Quel est le nombre minimal de chutes qui garantit que vous trouvez la réponse à la question?

Le sous-problème

- Soit C[i, j] le nombre de chutes nécessaires pour trouver la réponse avec i oeufs et un édifice de j étages.

$$C[i,j] = \begin{cases} 0 & \text{si } j = 0 \\ j & \text{si } i = 1 \\ 1 + \min\limits_{1 \leq k \leq j} \max(C[i-1, k-1], C[i, j-k]) & \text{sinon} \end{cases}$$

Problème de coupe

- Une planche de longueur L doit être coupée aux positions $p = [p_1, p_2, ..., p_n]$.
- La planche doit passer dans une machine et ressort coupée à un seul endroit.
- Le temps pour couper la planche est proportionnel à sa longueur.
- Dans quel ordre faut-il effectuer les coupes?

Le sous-problème

- On ajoute 0 et L au vecteur p de la façon suivante.
  $p = [0, p_1, p_2, ..., p_n, L]$
- Soit T[i, j] le temps pour couper la planche entre p_i et p_j.
- On recherche T[0, n+1].

$$T[i,j] = \begin{cases} 0 & \text{si } i+1 = j \\ \min\limits_{i<k<j} p_j - p_i + T[i,k] + T[k,j] & \text{si } i+1 < j \end{cases}$$

Multiplication de matrices

- Soit $D = [d_0, d_1, ..., d_n]$ un vecteur d'entiers.
- Vous devez calculer le produit des matrices suivantes: $A_0A_1A_2...A_{n-1}$
- La matrice A_i est de dimensions $d_i \times d_{i+1}$
- Avec l'algorithme classique, multiplier une matrice de dimensions $a \times b$ avec une matrice de dimensions $b \times c$ prend abc multiplications.
- Dans quel ordre faut-il multiplier les matrices?

Le sous-problème

- Soit M[i, j] le nombre minimum de multiplications pour multiplier les matrices de A_i à A_{j-1}.
- On recherche M[0,n]

$$M[i,j] = \begin{cases} 0 & \text{si } i+1=j \\ \min\limits_{i<k<j} d_i \times d_k \times d_j + M[i,k] + M[k,j] & \text{sinon} \end{cases}$$