Programmation avancée

Répétition 5: Recherche exhaustive et programmation dynamique

Jean-Michel Begon

22 novembre 2019

1 100-chaîne

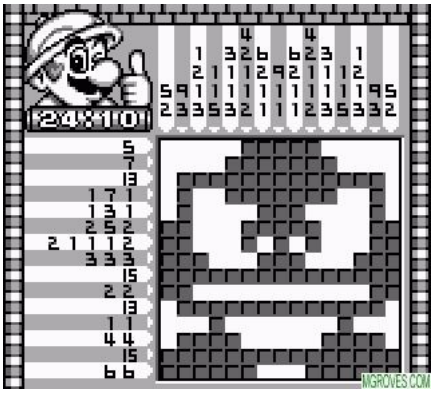
Soit la chaîne "123456789". Proposez un algorithme qui énumère les différentes manières d'insérer des "+" et des "-" devant les *nombres* de manière à obtenir un total de 100.

Par exemple, $123 + 45 - 67 + 8 - 9 = 100$.

Quelle est la complexité de votre solution ?

2 Logimage (aussi appelé picross)

Proposez un algorithme par recherche exhaustive pour résoudre un logimage :



Cette solution est-elle envisageable en pratique ? Par exemple pour une grille 15×15 ?

3 Micmaths

Dans une vidéo intitulée "La puissance organisatrice du hasard - Micmaths" (2017), Mickaël Launay exprime qu'une bille qui se déplace au hasard mais strictement vers la droite sur un treillis similaire à celui de la figure 1 va adopter une trajectoire presque droite car il y a plus de qui mènent vers les positions centrales (par exemple la position $(3, 3)$, au départ de la position $(0, 0)$.

1. Soit $T(i, j)$ $(i, j = 1, \ldots, n)$ le nombre de chemins menant à la position (i, j) en partant de la position $(0, 0)$. Formulez récursivement T.

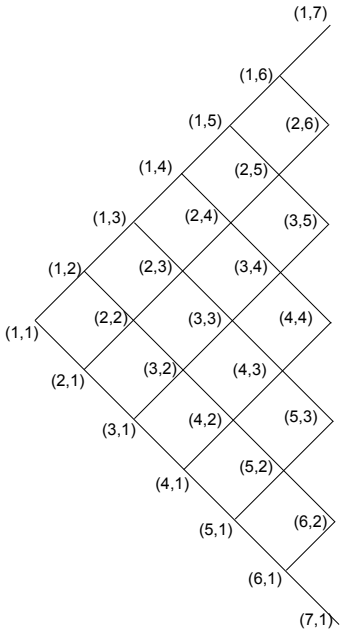


FIGURE 1 – Treillis de Launay

2. A l'aide de la mémoïsaiton, donnez le pseudo-code d'une fonction *efficace* pour calculer $T(i,j)$ $(i,j = 1,\ldots,n)$.

Remarque : si on suppose que tous les chemins sont équiprobables, on s'aperçoit que certains états "macroscopiques" (*i.e.* la position d'arrivée) sont, eux, plus probables. C'est la seconde loi de la thermodynamique.

4 B. Boigelot, 2009

On considère un terrain de jeu rectangulaire possédant $n \times m$ cases organisées en n lignes et m colonnes. Des sommes d'argent sont initialement placées sur ces cases. Un joueur traverse le terrain à partir du coin supérieur gauche jusqu'au coin inférieur droit. A chaque étape de ce trajet, le joueur collecte la somme d'argent placée sur la case où il se trouve, et a ensuite le choix de se déplacer d'une case soit vers la droite, soit vers le bas (les déplacements vers la gauche, vers le haut ou en diagonale sont interdits).

1. Proposez un algorithme permettant de calculer le gain maximum qu'un joueur peut atteindre en démarrant dans le coin supérieur gauche et en arrivant dans le coin inférieur droit. Pour ce faire :
   (a) Discutez des propriétés de sous-structure optimale et de chevauchement des sous-problèmes.
   (b) Formulez récursivement $G(i,j)$ $(1 \leq n,\ 1 \leq m)$, le gain maximum obtenu en atteignant la case i,j à partir du coin supérieur gauche. Précisez le ou les éventuels cas de base.
   (c) Donnez le pseudo-code d'une fonction *efficace* permettant de calculer $G(n,m)$.
   (d) Adaptez votre solution pour renvoyer le parcours optimal.
2. Analyzer la complexité en temps en espace de votre solution.
3. Dessinez le graphe des appels récursifs pour un problème de petite taille.

5 La carotte par les deux bouts

Au casino, un croupier vous propose le jeu suivant. Il dispose devant vous une carotte de n ($n > 1$) jetons. Chaque jeton i à une valeur donnée $V[i] > 0$. Tour à tour vous allez jouer :

— Quand c'est votre tour (vous commencez), vous avez le choix de prendre le jeton à l'une des extrémités.

— Quand c'est à son tour, le croupier prend le jeton de chaque extrémité (ou le seul jeton restant).

Le jeu se poursuit jusqu'à ce que la carotte soit vide. Vous gagnez si la somme de vos jetons est strictement supérieure à celle du croupier. Allez-vous faire vos jeux ?

1. Soit $S(i, j)$ ($1 \leq i \leq j \leq n$), la plus grande somme que vous pouvez gagner lorsque vous avez la main et qu'il reste les jetons $i, \ldots, j$ en jeu. Formulez récursivement S.
2. Donnez le pseudo-code d'un algorithme *efficace* permettant de déterminer si, étant donné une carotte V, vous pouvez gagner.
3. Adaptez votre algorithme pour renvoyer une solution gagnante s'il en existe une.
4. Quelle est la complexité au pire et meilleur cas de votre algorithme ?

6 Multiplications matricielles

Le coût de multiplication de deux matrices A et B de tailles respectives $m \times p$ et $p \times q$ est $\Theta(mpq)$. Soit le produit N matrices $A_1 \times A_2 \times ... \times A_N$.

1. Illustrez par un exemple que l'ordre des multiplications a un impact sur le coût global.
2. Proposez une solution *brute-force* pour calculer l'ordre optimal des multiplications.
3. Proposez une solution par programmation dynamique pour ce problème.
   (a) Proposez une formulation récursive du nombre minimum d'opérations à effectuer.
   (b) Déduisez-en le pseudo-code d'une fonction efficace qui optimise l'ordre des multiplications.
   (c) Quelle est la complexité de cette solution ?

7 Plus longue sous-séquence palindromique (adapté de CLRS, 15-2)

On souhaite déterminer le plus grand palindrome qui existe au sein d'un mot. Par exemple, le mot `characters` contient un palindrome de taille 5 : `carac` (il ne s'agit pas d'une sous-séquence contigüe).

1. Proposez une solution *brute-force* à ce problème.
2. Proposez une solution par programmation dynamique pour ce problème.
   (a) Formulez récursivement la taille de la plus longue sous séquence palindromique.
   (b) Ecrivez un pseudo-code pour déterminer ce palindrome.
   (c) Quelle est la complexité de cette solution ?

8 Le problème de la partition

On souhaite déterminer s'il est possible de partitionner un tableau d'entiers positifs A en deux sous-tableaux de somme identique.

1. Proposez une solution *brute-force* à ce problème. Quelle est sa complexité ?
2. Proposez un algorithme par programmation dynamique pour ce problème. Quelle est sa complexité ?

9 Le vrai parenthésage

Soit une expression booléenne composée des symboles `true`, `false`, `and` et `or`. Proposez un algorithme qui détermine le nombre de façons de parenthéser l'expression de sorte qu'elle soit évaluée à `true`.