Esiee-Paris - cours d'algorithmique IGI-2102

Projet de l'unité à faire en binôme

Optimisations locales versus optimisations globales

rene.natowicz@esiee.fr - 24/03/2022

Version du 8 mai 2022 : bonus « Chain Reactions », Google challenge 2022

Ce projet demande de comparer les valeurs de stratégies d'optimisation locale, souvent appelées stratégies gloutonnes, avec les valeurs optimales calculées par programmation dynamique.

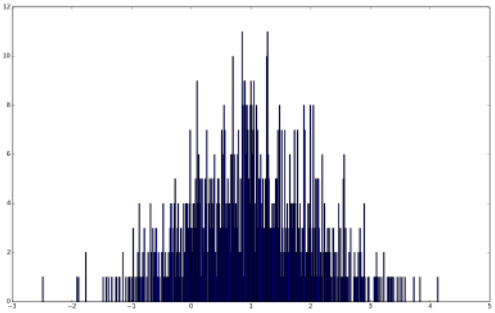
Exemples :

- dans le cas du calcul du chemin du petit robot (TD 5, exercice 6) la stratégie gloutonne consiste, en toute case, à se diriger dans la direction dont le coût en cette case est minimum ;
- dans le cas d'un sac de valeur maximum, la stratégie "gloutonne par valeur" consiste à mettre dans le sac les objets dans l'ordre des valeurs décroissantes, et la stratégie "gloutonne par densité de valeurs" consiste à les y mettre par ratios "valeur/taille" décroissants ;
- dans le cas de la répartition d'un stock sur un ensemble d'entrepôts, l'affectation gloutonne alloue une unité de stock à l'un des entrepôts pour lesquels l'augmentation de gain résultant de cette livraison est maximum ;
- de même pour la répartition optimale d'un temps de travail sur un ensemble d'unités : les heures sont progressivement allouées, chacune à l'unité pour laquelle l'augmentation de note estimée est maximum ;
- dans le cas d'un chemin de somme maximum – `https://projecteuler.net/archives` problèmes 18 et 67 – la stratégie gloutonne consiste à se diriger vers le descendant gauche si sa valeur est supérieure à celle du descendant droit, et réciproquement.

Pour comparer une stratégie gloutonne et la stratégie optimale, on génère un problème avec des valeurs choisies au hasard et on le résout avec chacune des deux stratégies. Soient v^* et g les valeurs optimale et gloutonne. On calcule la distance relative entre les deux solutions : $\frac{v^*-g}{v^*}$ pour une maximisation, $\frac{g-v^*}{v^*}$ pour une minimisation.
On répète cette expérience pour un grand nombre de problèmes (on parle de *runs*), puis on affiche l'histogramme de la distribution des distances relatives, et on donne la médiane, la moyenne, et l'écart-type des distances relatives.

Ci-dessous un exemple de construction et d'affichage d'un histogramme en Python.

```
% python
Python 2.7.16 (default, Aug 30 2021, 14:43:11)
[...]
>>> import matplotlib.pyplot as plt
>>> import random
>>> N = 1000
>>> Deltas = [random.gauss(1,1) for i in range(N)]
>>> num_bins = len(Deltas)//2
>>> h = plt.hist(Deltas, num_bins)
>>> plt.show()
```



Comparer les stratégies gloutonnes et optimales des problèmes des cinq exemples ci-dessus.

Pour chacun des exemples, la comparaison statistique sera sur 5000 *runs*.

Langage de programmation : Java.

On demande un rapport au format PDF incluant pour chacun des exemples l'histogramme, la médiane, la moyenne et l'écart-type des distances relatives. En annexe du rapport : les programmes dument commentés.

Le fichier pdf du binôme aura pour nom ceux des deux membres du binôme. Exemple : le rapport du binôme Juliette Capulet - Roméo Montaigu sera dans le fichier `Capulet-Montaigu.pdf`.

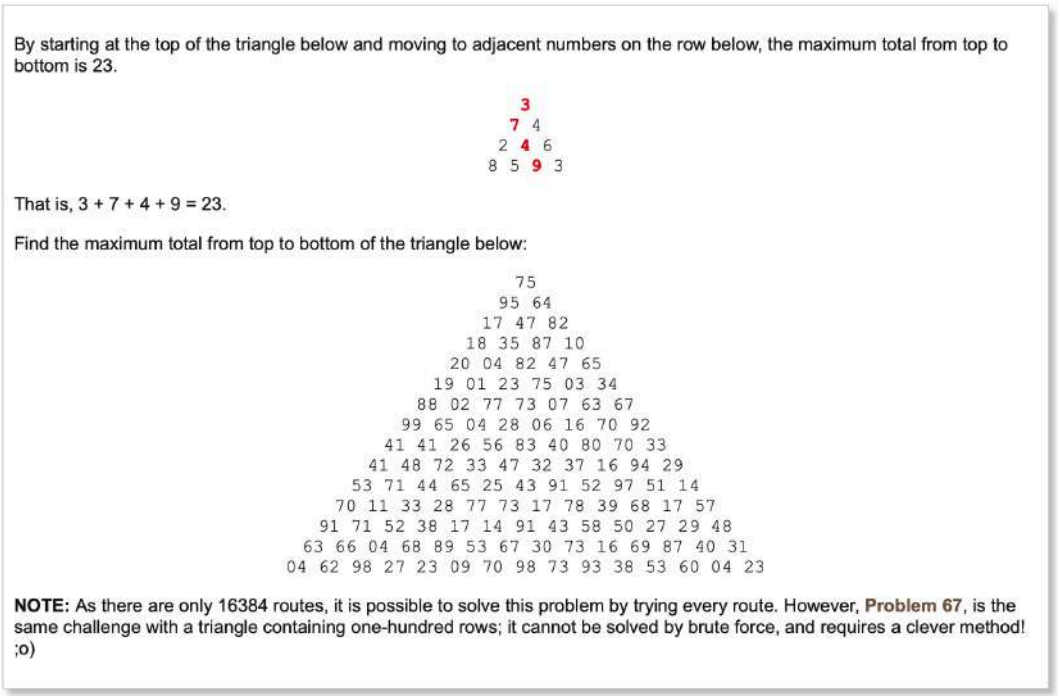
Les k-nômes, $k > 2$, ne sont pas autorisés. Si le nombre d'étudiants est impair, il y aura un monôme.

Les rapports sont à m'envoyer par courrier électronique.
Objet du message : `ALGORITHMIQUE : RAPPORT DU BINOME X Y`.
Exemple : `ALGORITHMIQUE : RAPPORT DU BINOME CAPULET MONTAIGU`

Date limite de remise des projets : vendredi 10 juin 2022.

Chemin de somme maximum.

Il s'agit des problèmes 18 et 67 du site `ProjectEuler.com` : voir `https://projecteuler.net/archives`



By starting at the top of the triangle below and moving to adjacent numbers on the row below, the maximum total from top to bottom is 23.

```
   3
  7 4
 2 4 6
8 5 9 3
```

That is, 3 + 7 + 4 + 9 = 23.

Find the maximum total from top to bottom of the triangle below:

```
                75
               95 64
              17 47 82
             18 35 87 10
            20 04 82 47 65
           19 01 23 75 03 34
          88 02 77 73 07 63 67
         99 65 04 28 06 16 70 92
        41 41 26 56 83 40 80 70 33
       41 48 72 33 47 32 37 16 94 29
      53 71 44 65 25 43 91 52 97 51 14
     70 11 33 28 77 73 17 78 39 68 17 57
    91 71 52 38 17 14 91 43 58 50 27 29 48
   63 66 04 68 89 53 67 30 73 16 69 87 40 31
  04 62 98 27 23 09 70 98 73 93 38 53 60 04 23
```

NOTE: As there are only 16384 routes, it is possible to solve this problem by trying every route. However, **Problem 67**, is the same challenge with a triangle containing one-hundred rows; it cannot be solved by brute force, and requires a clever method! ;o)

Représentation du triangle dans un tableau T[0:n]. Considérons un triangle à m niveaux, numérotés de 0 à $m-1$. Combien y a-t-il de valeurs dans ce triangle ?

- Au niveau 0 : 1 valeur
- au niveau 1 : 2 valeurs
- au niveau 2 : 3 valeurs
- etc.
- au niveau $m-1$: m valeurs

Un triangle à m niveaux contient donc $n = 1 + 2 + ... + m = \frac{m\times(m+1)}{2}$ valeurs.

Soit $T[0:n]$ un tableau d'entiers tel que $n = \frac{m\times(m+1)}{2}$. Pour voir ce tableau comme un triangle, nous devons disposer d'une fonction $g(i)$ qui retourne l'indice $g_i = g(i)$ du descendant gauche de l'indice i. Alors nous aurons l'indice d_i du descendant droit car $d_i = g_i + 1$.

Exemple avec $n = (3 \times 4)/2 = 12/2 = 6$ et $T[0:6] = [0,10,20,30,40,50]$:

```
      00          niveau 0, indice  0, intervalle d'indices [0:1]
    10  20       niveau 1, indices 1, 2, intervalle d'indices [1:3]
  30  40  50   niveau 2, indices 3, 4, 5, intervalle d'indices [3:6]
```

```
g(0) = 1, d(0) = 2
g(1) = 3, d(1) = 4
g(2) = 4, d(2) = 5
g(3) = 6, d(3) = 7 indices hors de [0:n], donc 3 est une feuille
g(4) = 7, d(4) = 8 indices hors de [0:n], donc 4 est une feuille
g(5) = 8, d(5) = 9 indices hors de [0:n], donc 5 est une feuille
```

Comment calculer l'indice $g(i)$ du descendant situé à gauche de l'indice i ? En remarquant que la position de l'indice i dans son niveau est la position de l'indice $g(i)$ dans son niveau.

Exemples :

- $i = 0$, position $p = 0$, niveau $l = 0$: $g(0) = 1$, position $p = 0$, niveau $l + 1 = 1$
- $i = 1$, position $p = 0$, niveau $l = 1$: $g(1) = 3$, position $p = 0$, niveau $l + 1 = 2$
- $i = 2$, position $p = 1$ niveau $l = 1$, $g(2) = 4$, position $p = 1$, niveau $l + 1 = 2$.

D'où le calcul de $g(i)$:

1. trouver son niveau l dans l'arbre ;
2. ayant le niveau l de l'indice i, en déduire sa position p dans ce niveau ;
3. ayant le niveau l et la position p de l'indice i dans son niveau l, retourner l'indice $g(i)$ de son descendant gauche ;
4. ayant l'indice $g_i = g(i)$ nous avons l'indice $d_i = g_i + 1$ du descendant droit de l'indice i.

Pour tout indice i nous connaissons les indices des descendants gauche et droit. Nous pouvons donc regarder le tableau $T[0 : n]$ comme le triangle T.

Remarque : on pourrait calculer le niveau de l'indice i en résolvant une équation du second degré. On ne le demande pas. On demande de calculer $g(i)$ comme décrit ci-dessus.

Calcul de la valeur des chemins de somme maximum

Écrire une fonction `calculerM(int[] T)` qui prend en entrée le triangle $T[0 : n]$ et retourne un tableau $M[0 : n]$ de terme général $M[i] = m(i) =$ la valeur d'un chemin de somme maximum qui commence à l'indice i. La valeur $M[0] = m(0)$ est la valeur d'un chemin de somme maximum du triangle T.

Pour écrire cette fonction il vous faut l'équation de récurrence des valeurs $m(i)$.

A) Supposons le problème résolu : donner l'expression de $m(0)$.

B) Généraliser cette expression afin d'obtenir l'équation de récurrence des valeurs $m(i)$.

C) Base ... : ...

D) Cas général ... : ...

Écrire une procédure `acsm(int[] M, int[] T, int i, int n)` qui affiche un chemin de somme maximum commençant en i, où M est le tableau calculé ci-dessus et n est la taille du triangle T. L'appel `acsm(M, T, 0, n)` affiche un chemin de somme maximum qui commence en 0.

Calcul de la valeur d'un chemin chemin glouton

En chaque point du chemin on se dirige à gauche si la valeur située à gauche est plus grande que celle située à droite, et réciproquement.

Evaluation statistique.

- soit $Lmax$ le nombre de niveaux maximum. Par exemple : $Lmax = 1000$
- soit $Nruns$ le nombre de *runs* de l'évaluation statistique. Par exemple : $Nruns = 5000$
- soit $Vmax$ la plus grande valeur pouvant être présente dans le triangle. Par exemple : $Vmax = 100$

1. Définir le tableau $D[0 : Nruns]$ qui contiendra pour chaque *run* la distance relative entre la valeur du chemin de somme maximum et la valeur du chemin glouton.
2. Pour chaque run r dans l'intervalle $[0 : Nruns]$:
   - choisir le nombre de niveaux m au hasard dans l'intervalle $[1 : Lmax + 1]$,
   - soit $n = m \times (m + 1)/2$,
   - soit $T[0 : n]$ un tableau d'entiers à valeurs au hasard dans l'intervalle $[0 : Vmax + 1]$,
   - soit $v^* = m(0)$ la valeur d'un chemin de somme maximum et soit g celle du chemin glouton. La distance relative est $D[r] = \frac{v^* - g}{v^*}$.
3. Retourner D.
4. Afficher l'histogramme de D, la moyenne, la médiane et l'écart-type des valeurs de D.

BONUS : le problème « Chain Reaction » du Challenge Google 2022

https://codingcompetitions.withgoogle.com/codejam/round/0000000000876ff1/0000000000a45ef7