Algorithmique
TD n° 7 : Programmation dynamique

Exercice 1 :

Le problème PGC (plus grand carré) consiste à chercher dans une matrice de 0 et 1 (un code QR) le plus grand carré formé uniquement de 1. Il est formellement défini comme suit :
Donnée : Une matrice M de taille $n \times n$ dont les coefficients valent 0 ou 1.
Sortie : la longueur maximale k du côté d'un carré dans la matrice ne contenant que des 1.
Exemple :

Pour $M = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$ on a $k = 2$ à cause de $\begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & \mathbf{1} & \mathbf{1} & 0 \\ 0 & \mathbf{1} & \mathbf{1} & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$ ou de $\begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & \mathbf{1} & \mathbf{1} & 1 \\ 1 & \mathbf{1} & \mathbf{1} & 0 \end{bmatrix}$

Une matrice $n \times n$ sera représentée par un tableau bidimensionnel $M[1..n, 1..n]$.

1. (Cette question est indépendante des suivantes)
   Une première version d'algorithme serait la suivante :

```
LPCG=0;
Pour i de 1 à n
  Pour j de 1 à n
    Pour k de 1 à min(n-i+1, n-j+1)
      si le carré de côté de long. k et coin haut gche (i,j) ne contient que des 1
      alors LPGC=max(LPCG,k);
retourner LPGC;
```

   Supposons que tester si un carré de côté de longueur k ne contient que des 1 coûte $O(k^2)$. Quelle sera alors à votre avis la complexité de cet algorithme ? Borner par O la complexité du pire cas en fonction de n. Justifier brièvement, une preuve complète n'est pas requise. (2 points)

2. On cherche maintenant un algorithme plus efficace que le précédent. On pose $PGC(i, j)$ la valeur maximale du côté d'un carré ne contenant que des 1 et dont le coin haut gauche est (i, j).

   Pour la matrice M ci-dessus, $PGC(1, 1) = 1, PGC(1, 2) = 0, PGC(2, 1) = 1, PGC(2, 2) = 2$. Complétez la matrice des valeurs de PGC pour la matrice M ci-dessus :

   $PGC = \begin{bmatrix} 1 & 0 & ? & ? \\ 1 & 2 & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$ (1 point)

   Si on connait la matrice PGC, il suffira ensuite de chercher son élément maximal pour résoudre le problème.

3. Si $M[i, j] = 0$, que vaut $PGC(i, j)$? (1 point)

4. Soit j un entier, $1 \leqslant j \leqslant n$. Que vaut $PGC(n, j)$, en fonction de $M[n, j]$? (Nous sommes donc dans une case de la dernière ligne). Même question avec $PGC(i, n)$, pour $1 \leqslant i \leqslant n$. (Nous sommes donc dans une case de la dernière colonne) (1 point)

5. (Note : même si vous ne savez répondre à cette question, il est quand même possible de répondre aux questions suivantes). Soit maintenant $1 \leqslant i, j < n$ et $M[i, j] = 1$ (dernier cas qui reste a considérer et le plus général).
   Calculer la dépendance :
   $$PGC(i, j) = \phi(PGC(i+1, j), PGC(i, j+1), PGC(i+1, j+1))$$
   qui exprime $PGC(i, j))$ en fonction de $PGC(i+1, j), PGC(i, j+1)$ et $PGC(i+1, j+1)$:
   - dans le cas où $PGC(i+1, j) \neq PGC(i, j+1)$
   - dans le cas où $PGC(i+1, j) = PGC(i, j+1)$

   Il est conseillé de faire un dessin dans chacun des deux cas. (4 points)
6. On veut déduire de ce qui précède un algorithme de programmation dynamique en $O(n^2)$ pour résoudre le problème PGC. L'algorithme devra remplir la table $PGC[1..n, 1..n]$ qui contiendra les valeurs $PGC(i, j)$. Quel sera un bon ordre de remplissage de la table ? (1 points)
7. Ecrire l'algorithme de programmation dynamique en $O(n^2)$. Si vous n'avez pas su répondre à la question 5, dans votre programme vous pourrez faire référence à la fonction ϕ même si vous n'en connaissez pas l'expression. (2 points)