Recherche Opérationnelle : TP 3-4

Omar NAIM
Saad MDAA
Younes EL BOUZEKRAOUI

Département Sciences du Numérique - Deuxième année
2020-2021

1 Programmation dynamique avec Bellman-Ford

1.1 Calcul du plus court chemin entre deux sommets d'un graphe

Dans ce tp, et pour modéliser les graphes : nous nous sommes basés sur une matrice qui ressemble à la matrice d'adjacence sauf qu'au lieu qu'elle soit remplie seulement de 0, 1 et de -1, nous avons rempli chaque G(i,j) par le poids de l'arc (i,j) et de 0 s'il n'y a pas d'arc entre i et j.

La matrice de la figure 1 du sujet, est ainsi représenté par la matrice G ci-dessous:

$$G= \begin{pmatrix} 0 & 3 & 0 & 0 & 5 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & (-1) & 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Ensuite pour la programmation, nous avons défini une matrice **"dis"** de nombre de sommets lignes et de deux colonnes qu'on a remplit au fur et à mesure de l'algorithme de Belleman-Ford, où chaque **"dis(i,1)"** : représente la plus petite distance entre le sommet à qui on veut appliquer l'algorithme et le sommet i , tandis que **"dis(i,2)**: représente le prédécesseur du sommet i pour lequel on a la distance minimal.

Le résultat obtenu dans la matrice **"dis"** correspond à la dernière ligne du tableau de l'annexe qui donne les coûts minimaux vers chaque noeud ainsi que son prédécesseur, on retrouve alors le résultats souhaité qui correspond au trajet

A-B-C-D-F et qui coûte : 12

1.2 Calcul du plus long chemin entre deux sommets d'un graphe

Pour calculer le plus long chemin, nous avons gardé le même algorithme réalisé précédemment auquel, nous avons rapporté quelques modifications. Tout d'abord on a changé l'initialisation de la matrice dis(:,1), où chaque élément était initialisé à +l'infini, maintenant dans ce nouveau algorithme, dis(:,1) a été initialisé à - l'infini, de plus nous avons aussi changé l'inégalité de la boucle de l'algorithme qui était un inférieur strictement, pour prendre la plus petite distance, qui est devenu un supérieur strictement, pour prendre le plus long chemin.
La nouvelle relation de récurrence devient: $\boxed{f_j^k = max_{\forall j \in Pred(i)}\ f_j^{k-1} + c_{ji}}$

De la même manière que précédemment on retrouve dans la matrice **"dis"** le résultat

A-E-D-F et qui coûte : 17

Ce qui correspond au résultat attendu.

2 Extensions et adaptations

2.1 Construction d'un réseau de transmission à vitesse maximale

Dans cette deuxième partie, il fallait déterminer la vitesse maximale entre le sommet initial et chaque sommet i, sachant que la vitesse pour un chemin donné est la vitesse minimal des arrêtes par lesquels on passe. Pour cela, nous avons gardé à peu prés le même algorithme de Bellman-Ford, sauf qu'au lieu de prendre le minimum, nous avons pris le maximum des minimums, pour symboliser la relation de récurrence ci-dessous:

$$\boxed{f_j^k = max_{\forall j \in Pred(i)}\ min(f_j^{k-1}, G(i,j))}$$

2.2 Fiabilité de procédé de fabrication de semi-conducteurs

2.2.1 (a)

Dans cette partie là, il faut déterminer le chemin le plus sur vers chaque sommet, c'est à dire maximiser le produit des probabilités de succès, et en passant au logarithme, et sachant que le logarithme est croissant, il faut ainsi maximiser la somme des log des probabilités, par suite on remplace tout simplement le graphe G par le log G , et on applique le deuxième algorithme qu'on a définit et qui détermine le plus long chemin entre deux sommets.

2.2.2 (b)

Dans cet algorithme, nous allons nous inspirer de l'algorithme qui calcule le plus long chemin, sauf qu'au lieu d'avoir une somme dans la boucle, nous aurons un produit avec la probabilité correspondante, ainsi on aura la relation de récurrence suivante : $\boxed{f_j^k = max_{\forall j \in Pred(i)}\ f_j^{k-1} * p_{ji}}$
Et normalement on s'attend à avoir le même résultat que celui trouvé sur la question précédente et c'est bien ce qu'on trouve.

3 Bonus

Nous souhaitons minimiser la consommation du PAC :
Pour cela, nous proposons la modélisation suivante :

Les données :

- MaxInstants : le temps maximal
- $P_{Demande}(t)$: Puissance total demandée à chaque instant t
- P_{max}, P_{min}: Puissance minimale et maximale de la PAC
- LB : Limite de la batterie

Les variables :

- $\mathbf{E} = (E_{Batterie})$ Niveaux d'énergie du batterie à chaque instant t, ainsi la différence entre deux instants successifs / 1s , nous donne la puissance.

Fonction-objectif :

- $Min \sum_{t \in [1 MaxInstants]} FonctionCoutMoteur(PMoteur(t))$

Les contraintes :

- $\forall t \in [1 MaxInstants]\ \ P_{PAC}(t) + P_{ReelBatterie}(t) = P_{Demandé}(t)$
- $\forall t \in [1 MaxInstants]\ \ P_{ReelBatterie}(t) = P_{FourniBatterie}(t) * Efficacite$
- $\forall t \in [1 MaxInstants]\ \ P_{FourniBatterie}(t) = \frac{E_{Batterie}(t+1) - E_{Batterie}(t)}{(t+1)-t}$
- $\forall t \in [1 MaxInstants] \sum_{t \in [1 MaxInstants]} P_{FourniBatterie}(t) <= LB$
- $\forall t \in [1 MaxInstants]\ \ P_{PAC}(t) <= P_{max}$
- $\forall t \in [1M]\ \ P_{PAC}(t) >= P_{min}$