Recherche Opérationnelle :
TP 4 : programmation dynamique

Sandra U. Ngueveu, Corentin Boennec, Arthur Clavière, Aloïs Duguet
ngueveu@laas.fr

2021

Le but est de résoudre le problème du sac à dos et le problème du plus court chemin dans un graphe à l'aide d'un algorithme de programmation dynamique que vous aurez implémenté.
Rendus attendus (à soumettre sur moodle à l'emplacement dédié) : archive .zip contenant :
- le notebook en julia permettant de résoudre les instances fournies et d'autres de votre choix
- un rapport synthétique (3 pages max) au format pdf s'il n'est pas déjà inclu dans le notebook

1 Programmation dynamique

Implémenter en Julia l'algorithme de programmation dynamique permettant de résoudre le problème du sac à dos.

Tester sur plusieurs instances de taille différentes fournies dans le TP2-TP3, donner les solutions obtenues et valeurs de fonction-objectif. Comparer avec celles obtenues avec le branch-and-bound implémenté lors des TP2-3.

2 Rapport de TP attendu : Instructions

Ce rapport de 3 pages maximum pourra être inclus dans un notebook. Il participera à la note globale de TP du cours de Recherche Opérationnelle. Il est à rendre en binôme du même groupe de TP.

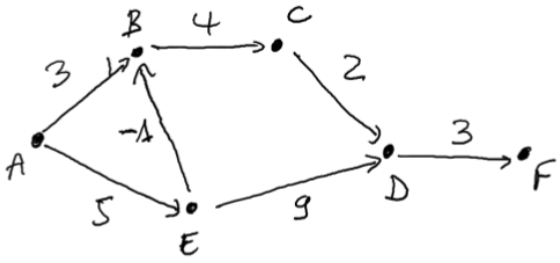
Le fichier sera à rendre au plus tard le **vendredi 7 janvier 23h55** sur moodle. Donner, par exemple :

- Quelques détails sur les points clés et non triviaux de votre implémentation
- Une courte argumentation de l'adéquation du résultat avec l'instance résolue
- Quelques éléments d'analyse, par exemple :
  - faire une analyse comparative des résultats et performances obtenus avec ceux du branch-and-bound implémenté lors des TP2-3.

3 BONUS : Calcul du plus court chemin entre deux sommets d'un graphe

Implémenter en Julia l'algorithme de Bellman-Ford vu en cours permettant de calculer le plus court chemin entre un sommet source s et tous les autres sommets d'un graphe quelconque. A titre d'exemple vous pouvez vous baser sur l'exemple donné en cours et rappelé en annexe. Tester ensuite avec différents jeux de données de votre choix.

4 ANNEXE : rappel de l'exemple du cours



itération	A	B	C	D	E	F
0	0	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
1	0	3^A	$+\infty$	$+\infty$	5^A	$+\infty$
2	0	3^A	7^B	14^E	5^A	$+\infty$
3	0	3^A	7^B	9^C	5^A	17^D
4	0	3^A	7^B	9^C	5^A	12^D
5	0	3^A	7^B	9^C	5^A	12^D

Données

- Ce graphe a $n = 6$ sommets (A, B, ..., F) et $m = 7$ arcs ((AB), (AE), ... (DF))
- Chaque arc ij a un coût c_{ij} (par exemple $c_{AB} = 3$)

Relation de récurrence

- Soit f_i^k la valeur du plus court chemin du sommet de départ (A) et un sommet i calculé à l'itération k
- La relation de récurrence qui s'applique est $f_i^k = \min_{\forall j \in \text{Pred}(i)} f_j^{k-1} + c_{ji}$ avec $i \in \{1, ..., n\}$ et $k \geq 1$

Condition d'arrêt

- L'algorithme de Bellman-Ford s'arrête dès l'itération k qui vérifie soit $\forall i$ $f_i^k = f_i^{k-1}$, soit $k >= n + 1$.
- Remarque : Si le cas de figure $k = n + 1$ se produit, alors cela démontre l'existence d'un cycle de longueur négative dans le graphe. Selon ce que modélise ce graphe, il se peut que cela permette de détecter une incohérence.

Solution obtenue

- Le plus court chemin entre A et F est A-B-C-D-F. Ce chemin a un coût de 12.

FIGURE 1 – Exemple : graphe et tableau des valeurs de f_i^k obtenu lors de l'application de l'algorithme de Bellman-Ford pour calculer le plus court chemin entre les noeuds A et F du graphe fourni