Faculté des Sciences et Technologies
Dpt Informatique

LIF064 - Optimisation – TD5

Voyageur de commerce : algorithme de Little

Exercice 1

Un voyageur de commerce doit effectuer une tournée passant par plusieurs villes. Les durées pour se rendre d'une ville à l'autre sont les suivantes :

	A	B	C	D	E	F
A	∞	1	7	3	14	2
B	3	∞	6	9	1	24
C	6	14	∞	3	7	3
D	2	3	5	∞	9	11
E	15	7	11	2	∞	4
F	20	5	13	4	18	∞

En utilisant l'algorithme de Little, déterminez la tournée optimale (la plus courte) passant par toutes les villes.

Une solution exacte au Problème du Voyageur de Commerce (PVC ou Traveller Saleman Problem TSP) est donnée par l'algorithme de Little. Il travaille directement sur la matrice de distance et est un algorithme d'énumération par séparation et évaluation (branch and bound). La première étape est une évaluation de la longueur minimale du chemin, par la soustraction du plus petit élément de chaque ligne puis de chaque colonne. *(Dessiner graphe pour sous-ensemble ABC)*

	A	B	C	D	E	F	min
A	∞	1	7	3	14	2	1
B	3	∞	6	9	1	24	1
C	6	14	∞	3	7	3	3
D	2	3	5	∞	9	11	2
E	15	7	11	2	∞	4	2
F	20	5	13	4	18	∞	4

	A	B	C	D	E	F
A	∞	0	6	2	13	1
B	2	∞	5	8	0	23
C	3	11	∞	0	4	0
D	0	1	3	∞	7	9
E	13	5	9	0	∞	2
F	16	1	9	0	14	∞
min	0	0	3	0	0	0

	A	B	C	D	E	F
A	∞	0	3	2	13	1
B	2	∞	2	8	0	23
C	3	11	∞	0	4	0
D	0	1	0	∞	7	9
E	13	5	6	0	∞	2
F	16	1	6	0	14	∞

La première évaluation minimale de la longueur du chemin est donc égale à $13 + 3 = 16$ (somme des min des lignes et des colonnes).

On passe ensuite à la phase de séparation, où on choisit si un couple de ville est ou non sans doute dans le chemin optimal. Par exemple, si le chemin contient AB le coût augmente de zéro (valeur AB dans le tableau). Par contre si le chemin ne le contient pas (noté $\overline{AB}$), le coût est la somme des min sur la ligne A et la colonne B, i.e. ici $1 + 1 = 2$. On fait ce calcul pour tous les coûts minimaux (c.-à-d. à zéro) présents dans le tableau. Le couple de ville choisi est celui qui a la plus grande somme des min.

	A	B	C	D	E	F
A	∞	0 (2)	3	2	13	1
B	2	∞	2	8	0 (6)	23
C	3	11	∞	0 (0)	4	0 (1)
D	0 (2)	1	0 (2)	∞	7	9
E	13	5	6	0 (2)	∞	2
F	16	1	6	0 (1)	14	∞

Ici le couple BE a la plus grande valeur : 6 qui donnerait un chemin d'au moins $16 + 6 = 22$ si on ne le prend pas, et est donc choisi. Si on ajoute BE au chemin final optimal, on supprime la ligne B et la colonne E du tableau (impossible de partir de B vers autre chose que E, et impossible d'arriver à E depuis une autre ville que B), et on met le coût de EB à l'infini.

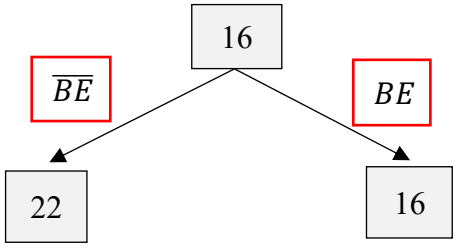
	A	B	C	D	F
A	∞	0	3	2	1
C	3	11	∞	0	0
D	0	1	0	∞	9
E	13	∞	6	0	2
F	16	1	6	0	∞

On recommence à soustraire les min des lignes puis des colonnes pour évaluer le coût de prendre le couple de ville BE. On a :

	A	B	C	D	F	min
A	∞	0	3	2	1	0
C	3	11	∞	0	0	0
D	0	1	0	∞	9	0
E	13	∞	6	0	2	0
F	16	1	6	0	∞	0
min	0	0	0	0	0	0

Donc l'évaluation du chemin si on prend BE n'augmente pas le coût total et reste à 16.

Un arbre est construit à partir de ces choix.



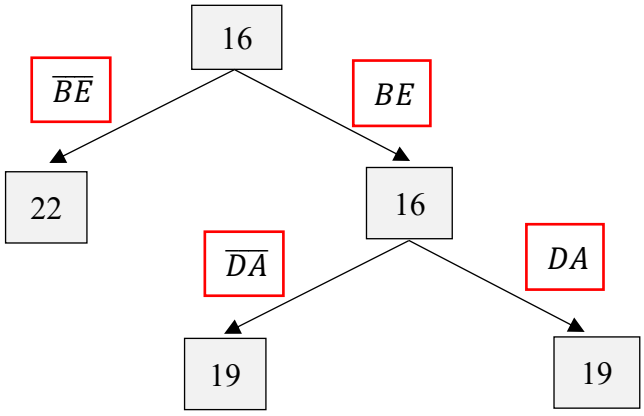
On choisit de continuer sur la branche donnant le coût le plus petit (on reviendra éventuellement sur l'autre branche si on aucun chemin de coût plus petit que 22 n'est trouvé). Ceci fini la première itération de l'algorithme de Little, on fait cela jusqu'à obtenir une matrice nulle.

	A	B	C	D	F
A	∞	0 (2)	3	2	1
C	3	11	∞	0 (0)	0 (1)
D	0 (3)	1	0 (3)	∞	9
E	13	∞	6	0 (2)	2
F	16	1	6	0 (1)	∞

On choisit DA qui aura un coût $16 + 3 = 19$ si on ne le prend pas. Pour évaluer le coût de le prendre on regarde les min des lignes et des colonnes après suppression de la ligne D, de la colonne A et de l'élément AD.

	B	C	D	F	min
A	0	3	∞	1	0
C	11	∞	0	0	0
E	∞	6	0	2	0
F	1	6	0	∞	0
min	0	3	0	0	3

Cette fois le coût de prendre DA est de 3 (colonne C), donc le chemin total est $16 + 3 = 19$. On a l'arbre :



En cas d'égalité de deux branches, il est conseillé de choisir celle où le nombre de couples de villes pris est maximal, étant donné que l'on se rapproche ainsi potentiellement plus vite d'un chemin complet.

On applique cette méthode jusqu'au bout, en obtenant les tableaux suivants.

	B	C	D	F	min
A	0 (1)	0 (3)	∞	1	0
C	11	∞	0 (0)	0 (1)	0
E	∞	3	0 (2)	2	0
F	1	3	0 (1)	∞	0
min	0	3	0	0	3

Sans $AC = \overline{AC} = 19 + 3 = 22$

	B	D	F	min
C	11	∞	0	0
E	∞	0	2	0
F	1	0	∞	0
min	1	0	0	1

Avec $AC = 19 + 1 = 20$
Et comme on a déjà DA, on ne peut avoir CD donc on met CD à l'infini.

La branche $\overline{DA}$ est plus prometteuse (à 19) donc on y retourne (tableau à l'itération précédente en mettant $DA = \infty$).

	A	B	C	D	F	min
A	∞	0 (2)	3	2	1	0
C	0 (10)	11	∞	0 (0)	0 (1)	0
D	∞	1	0 (4)	∞	9	0
E	10	∞	6	0 (2)	2	0
F	13	1	6	0 (1)	∞	0
min	3	0	0	0	0	3

Sans $CA = \overline{CA} = 19 + 10 = 29$

	B	C	D	F	min
A	0	∞	2	1	0
D	1	0	∞	9	0
E	∞	6	0	2	0
F	1	6	0	∞	0
min	0	0	0	1	1

Avec $CA = 19 + 1 = 20$

La branche précédente avec AC est plus prometteuse (même coût mais plus de couples de villes sélectionnés), on la continue donc.

	B	D	F	min
C	10	∞	0 (12)	0
E	∞	0 (2)	2	0
F	0 (10)	0 (0)	∞	0
min	1	0	0	1

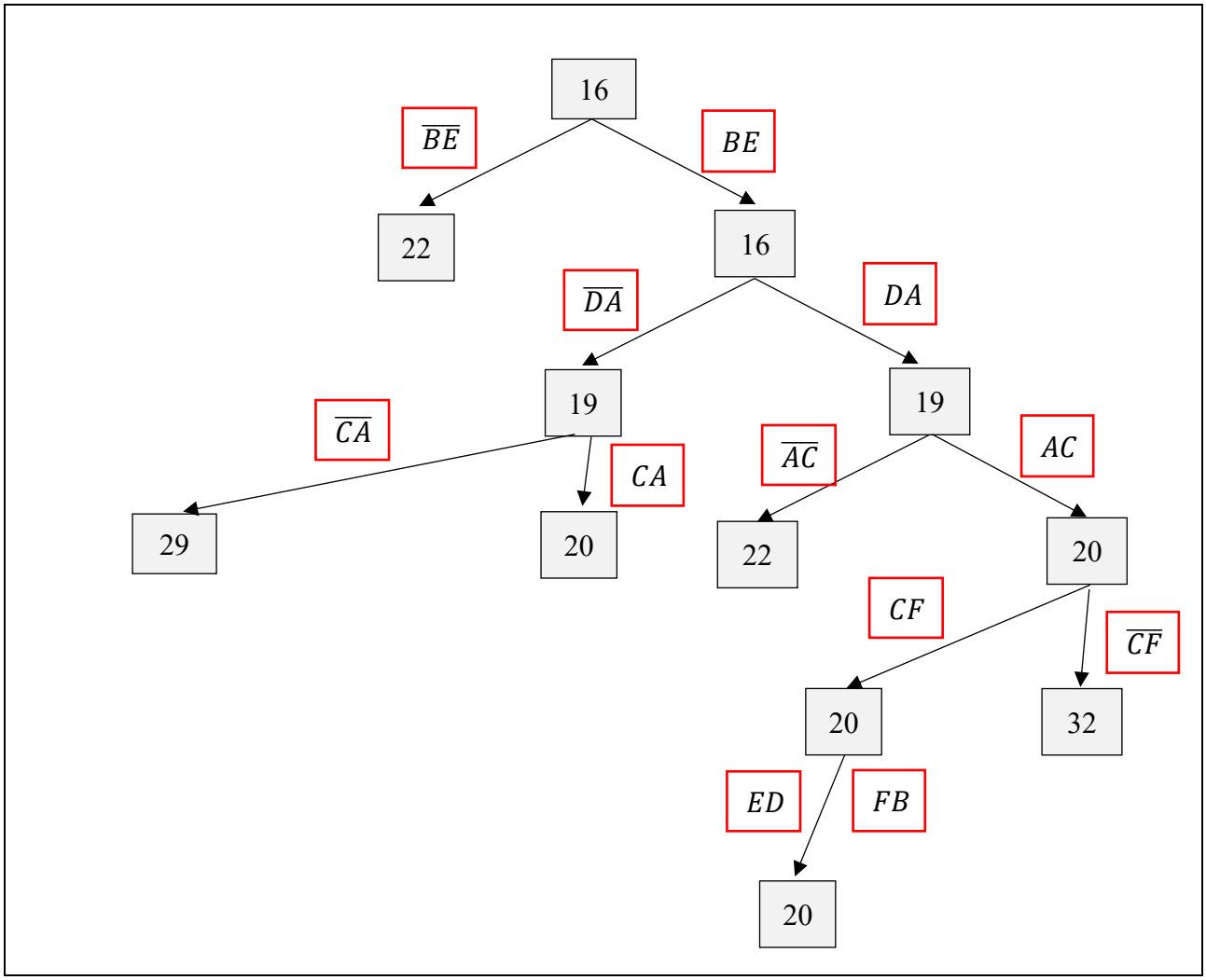
Sans $CF = \overline{CF} = 20 + 12 = 32$

	B	D	min
E	∞	0	0
F	0	∞	0
min	0	0	0

Avec $CF = 20 + 0 = 20$
Et comme on a déjà DA et AC, on ne peut avoir FD donc on met FD à l'infini.

Il n'y a plus que des coûts nuls, donc les deux couples de villes ED et FB sont ajoutés. Il ne reste plus aucune branche à explorer, la solution optimale a été trouvée. C'est le chemin qui utilise les couples de villes : BE, DA, AC, CF, ED et FB et aura un coût de 20. Par exemple si on veut partir de A, on a le chemin ACFBEDA.

On a l'arbre final de recherche :



Exercice 2

Un voyageur de commerce doit effectuer une tournée passant par plusieurs villes. Les durées pour se rendre d'une ville à l'autre sont les suivantes :

	A	B	C	D	E
A	∞	11	1	7	9
B	5	∞	3	12	3
C	7	1	∞	9	13
D	14	9	5	∞	4
E	3	12	7	1	∞

En utilisant l'algorithme de Little, déterminez la tournée optimale (la plus courte) passant par toutes les villes.

On applique la méthode de Little sur la matrice des distances.

	A	B	C	D	E	min
A	∞	10	0 (6)	6	8	1
B	0 (0)	∞	0 (0)	9	0 (0)	3
C	4	0 (9)	∞	8	12	1
D	8	5	1	∞	0 (1)	4
E	0 (0)	11	6	0 (6)	∞	1
min	2	0	0	0	0	12

Sans $CB = \overline{CB} = 12 + 9 = 21$

	A	C	D	E	min
A	∞	0	6	8	0
B	0	∞	9	0	0
D	8	1	∞	0	0
E	0	6	0	∞	0
min	0	0	0	0	

Avec $CB = 12 + 0 = 12$

	A	C	D	E	min
A	∞	0 (7)	6	8	0
B	0 (0)	∞	9	0 (0)	0
D	8	1	∞	0 (1)	0
E	0 (0)	6	0 (6)	∞	0
min	0	0	0	0	

Sans $AC = \overline{AC} = 12 + 7 = 19$

	A	D	E	min
B	∞	9	0	0
D	8	∞	0	0
E	0	0	∞	0
min	0	0	0	

Avec $AC = 12 + 0 = 12$
Comme on a AC et CB, on ne peut avoir BA.

	A	D	E	min
B	∞	9	0 (9)	0
D	8	∞	0 (8)	0
E	0 (8)	0 (9)	∞	0
min	0	0	0	

On choisit aléatoirement entre BE et ED.
Sans $ED = \overline{ED} = 12 + 9 = 21$

	A	E	min
B	∞	0	0
D	0	∞	8
min	0	0	8

Avec $ED = 12 + 8 = 20$
Comme on a obtenu 19 sans AC, on reprend là (avec $AC = \infty$).

	A	C	D	E	min
A	∞	∞	0 (2)	2	6
B	0 (0)	∞	9	0 (0)	0
D	8	0 (5)	∞	0 (0)	0
E	0 (0)	5	0 (0)	∞	0
min	0	1	0	0	7

Sans $DC = \overline{DC} = 19 + 5 = 24$

	A	D	E	min
A	∞	0	2	0
B	0	∞	0	0
E	0	0	∞	0
min	0	0	0	0

Avec $DC = 19 + 0 = 19$
Comme on a DC et CB, on ne peut avoir BD.

	A	D	E	min
A	∞	0 (2)	2	0
B	0 (0)	∞	0 (2)	0
E	0 (0)	0 (0)	∞	0
min	0	0	0	0

On choisit aléatoirement entre AD et BE.
Sans $AD = \overline{AD} = 19 + 2 = 21$

	A	E	min
B	∞	0	0
E	0	∞	0
min	0	0	0

Avec $AD = 19 + 0 = 19$
Comme on a AD, DC et CB, on ne peut avoir BA.

On obtient la matrice nulle, et on ajoute BE et EA au chemin. Nous avons donc les couples optimaux de villes CB, DC, AD, BE et EA pour un coût de 19. Si on part de la ville A, on a le chemin ADCBEA.
L'arbre de recherche est :