Correction TD 13

PROBLEME D'AFFECTATION :

On veut affecter six programmes a,b,c,d,e,f à six programmeurs A,B,C,D,E,F en minimisant la somme des temps de travail et de façon à ce qu'un programmeur fasse exactement un programme. Le tableau ci-dessous rapporte les différents temps de travail. Pour résoudre ce problème d'affectation, on va appliquer l'algorithme hongrois qui permet plus généralement d'affecter n individus à n travaux en minimisant la somme des coûts d'affectation.

	a	b	c	d	e	f
A	14	6	18	16	63	15
B	41	78	44	73	70	25
C	44	81	36	80	80	78
D	46	74	5	25	83	3
E	72	32	55	51	3	81
F	69	76	12	99	83	30

1) Montrer qu'on ne change pas la solution du problème en soustrayant ou ajoutant une valeur numérique δ à tous les éléments d'une rangée (ligne ou colonne). Seule la valeur de la solution est modifiée.

Le faire sur l'exemple en soustrayant le plus petit élément de chaque rangée en commençant par les lignes et en terminant par les colonnes, (l'intérêt est de faire apparaître des zéros).

2) On associe au nouveau tableau un graphe biparti reliant les programmeurs aux programmes quand le coût réduit (obtenu après soustraction des minima en lignes et en colonnes) vaut 0. Les arcs du graphe biparti sont dits admissibles.

On introduit aussi un réseau de transport en reliant la source s aux programmeurs par des arcs de capacité 1, le puits p aux programmes par des arcs de capacité 1. Les arcs admissibles sont de capacité infinie.

a) Construire le réseau de transport.

b) Calculer un flot maximal sur ce réseau et en déduire un couplage maximal de coût nul.

c) Le marquage de Ford-Fulkerson à l'optimum permet de marquer les lignes X* de X et les colonnes Y* de Y.

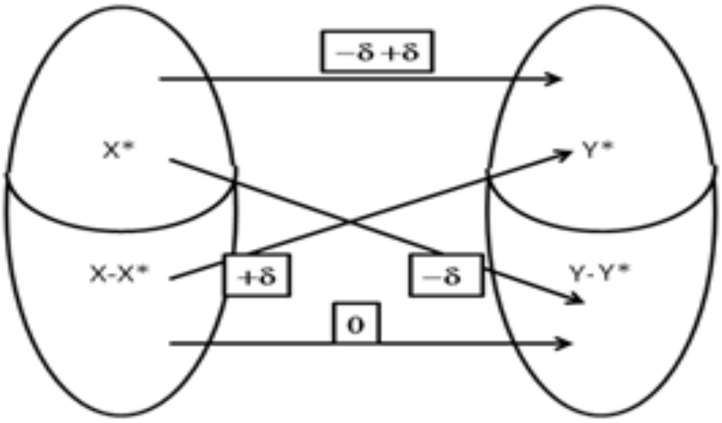
Vérifier sur l'exemple qu'il n'y a pas d'arc admissible entre X* et Y-Y*.

Vérifier sur l'exemple que les arcs entre X-X* et Y* ont un flux nul.

Justifier les deux propriétés.

d) Soit δ le plus petit coût réduit d'affectation d'un programmeur de X* à un programme de Y-Y*. Calculer δ. Ajouter δ aux colonnes de Y* et soustraire δ aux lignes de X*. Pourquoi ces opérations sont-elles licites ?

e) Montrer que la procédure pratique suivante permet de calculer la nouvelle matrice.



Procédure pratique.

-rayer toute ligne non marquée ;

-rayer toute colonne marquée ;

-soit δ le plus petit élément non rayé ;

-retrancher δ aux éléments non rayés ;

-ajouter δ aux éléments rayés deux fois.

f) On obtient une nouvelle matrice de coûts. Vérifier que les arcs de flux non nuls restent admissibles. Quelle est la propriété des arcs qui étaient admissibles et qui ne le sont plus ?

Quelle est la propriété des arcs devenus admissibles? Quelle est la capacité de l'ancienne coupe minimale ? Montrer que le nouveau marquage contient le précédent ?

g) Calculer le nouveau flot maximal ?

3) Validité de la méthode Hongroise.

On itère autant de fois que nécessaire la procédure pratique et le calcul du flot :

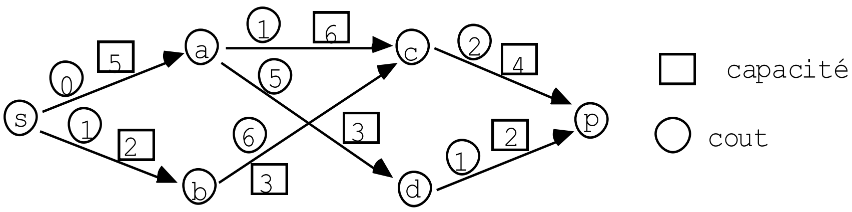
-le faire;

-à quelle condition s'arrête-t-on ?

-en déduire l'optimalité de la méthode.

EXERCICE : Algorithme de ROY:

Appliquer l'algorithme de Roy pour déterminer un flot maximal de coût minimal dans le graphe suivant.



On dessinera les graphes d'écart intermédiaires et on précisera les chaînes améliorantes successives utilisées. Calculer le coût du flot obtenu.

Problème d'affectation :

Question 1:

	a	b	c	d	e	f	
A	14	6	18	16	63	15	-6
B	41	78	44	73	70	25	-25
C	44	81	36	80	80	78	-36
D	46	74	5	25	83	3	-3
E	72	32	55	51	3	81	-3
F	69	76	12	99	83	30	-12
							=85

	a	b	c	d	e	f
A	8	0	12	10	57	9
B	16	53	19	48	45	0
C	8	45	0	44	44	42
D	43	71	2	22	80	0
E	69	29	52	48	0	78
F	57	64	0	87	71	18
	-8			-10		=18

	a	b	c	d	e	f
A	0	0	12	0	57	9
B	8	53	19	38	45	0
C	0	45	0	34	44	42
D	35	71	2	12	80	0
E	61	29	52	38	0	78
F	49	64	0	77	71	18

Le programmeur A fera obligatoirement une des tâches, donc il suffira d'ajouter la valeur retranchée sur cette ligne pour faire apparaître les zéros pour retrouver la valeur réelle. Le raisonnement est le même pour les colonnes. Le coût minimum d'affectation est donc supérieur à 85+18 = 103.

De façon plus formelle on a :

$$\left|\begin{array}{l} \text{Coût} = \sum_{(i,j)} x_{ij}\, C_{ij} \quad (\min) \\ x_{ij} \in \{0, 1\} \\ \sum_{i} x_{ij} = 1 \quad \forall j \in 1..n \\ \sum_{j} x_{ij} = 1 \quad \forall i \in 1..n \end{array}\right.$$

$$x_{ij} = \begin{cases} 1 \text{ si i est affecté à j} \\ 0 \text{ sinon} \end{cases}$$

posons : $C'_{ij} = C_{ij} - \delta_i$

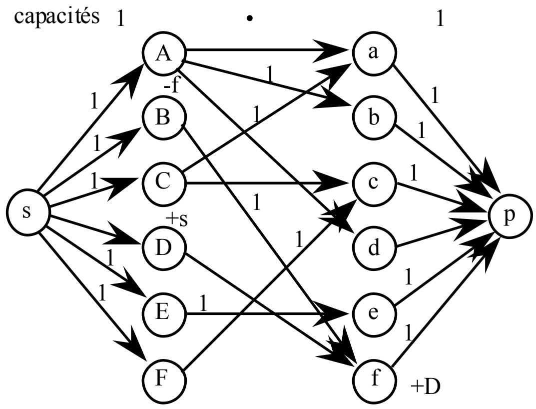
$$\text{Coût}'(x) = \sum_{(i,j)} x_{ij}\, C'_{ij} = \sum_{i,j} (C_{ij} - \delta_i) x_{ij} = \sum_{i,j} C_{ij} x_{ij} - \sum_{i,j} \delta_i x_{ij} = \text{Cout}(x) - \sum_i \delta_i \Big(\sum_j x_{ij}\Big) = \text{Cout}(x) - \sum_i \delta_i$$

$\text{Coût}' + \sum_{i} \delta_i = \sum_{(i,j)} x_{ij}\,(C'_{ij} + \delta_i) = \sum_{(i,j)} x_{ij}\, C_{ij}$ donc la valeur de chaque solution est diminuée de $\sum_{i} \delta_i$ et on raisonne de la même manière pour les colonnes en indice j.

Donc toute solution x qui minimise la fonction Cout'(x) minimisera la fonction Cout(x) et vice-versa.

Question 2 :

a/



b/ Algorithme du flot complet :

chemins améliorants :

(s, A, a, p) +1; (s, B, f, p) +1; (s, C, c, p) +1; (s, E, e, p) +1.

Ford Fulkerson :

chaînes améliorantes : [s, F, c, C, a, A, b, p] +1

marquage au blocage de l'algorithme : D(+s); f(+D); B(-f).

c/ Vérification sur l'exemple.

$X^* = \{B, D\}$; $X\text{-}X^* = \{A, C, E, F\}$

$Y^* = \{f\}$; $Y\text{-}Y^* = \{a, b, c, d, e\}$

A la fin de l'algorithme de Ford Fulkerson le marquage des sommets définit une coupe de capacité minimale.

<u>Il n'existe pas d'arcs admissibles entre X^* et $Y\text{-}Y^*$</u> :

raisonnons par l'absurde : soit $i \in X^*$ et $j \in Y\text{-}Y^*$ et il existe un arc admissible entre i et j. La capacité d'un arc admissible étant infinie on a $f_{ij} < C_{ij}$, donc j aurait été nécessairement marqué.

C'est donc impossible d'avoir des arcs admissibles entre X^* et $Y\text{-}Y^*$

Les arcs sortant d'une coupe sont nécessairement saturés.

<u>Les arc entre $X\text{-}X^*$ et Y^* ont un flux nul :</u>

raisonnons par l'absurde : soit $i \in X\text{-}X^*$ et $j \in Y^*$ et $f_{ij} > 0$; puisque j est marqué et $f_{ij} > 0$ alors i le sera nécessairement par l'algorithme de Ford Fulkerson donc contradiction.

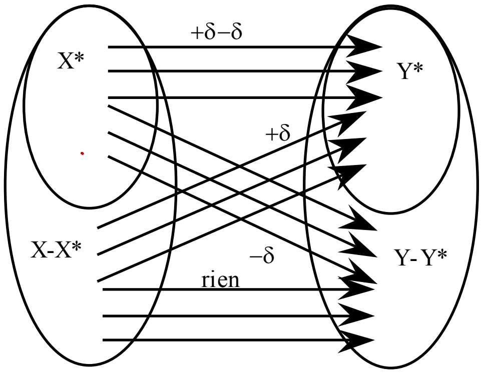
Les arcs entrant dans une coupe ont nécessairement un flux nul sinon il sont dans la coupe.

	a	b	c	d	e	$f(Y^*)$	
A	0	0	12	0	57	9	
$B(X^*)$	8	53	19	38	45	0	-2
C	0	45	0	34	44	42	
$D(X^*)$	35	71	2	12	80	0	-2
E	61	29	52	38	0	78	
F	49	64	0	77	71	18	
						+2	

	a	b	c	d	e	$f(Y^*)$
A	0	0	12	0	57	11
$B(X^*)$	6	51	17	36	43	0
C	0	45	0	34	44	44
$D(X^*)$	33	69	0	10	78	0
E	61	29	52	38	0	80
F	49	64	0	77	71	20

1/ c'est licite de retrancher et ajouter une valeur constante sur une ligne ou une colonne, et cela ne modifie pas la solution optimale (Cf. Q1).

2/ Cela revient à retirer δ à toutes les valuations des arcs entre X^* et $Y\text{-}Y^*$; et à ajouter δ aux valuations des arcs entre $X\text{-}X^*$ et Y^* (Cf. dessin). Sur les valuations des arcs entre X^* et Y^* on ajoute et retranche δ donc le bilan est nul (on conserve les zéros). Enfin on ne fait aucune opération sur les arcs entre $X\text{-}X^*$ et $Y\text{-}Y^*$.



e/

<u>Les arcs de flux non nul restent admissibles :</u>

• les arcs sont en dehors de la coupe : comme on n'ajoute ni ne retranche δ il restent admissibles;

• les arcs sont dans la coupe : on leur a retranché et ajouté δ il restent admissibles.

NB : les arcs entrant dans la coupe ont un flux nul et il n'y a pas d'arcs admissibles sortant de la coupe (Cf. c/).

<u>Les arcs qui étaient admissibles et qui ne le sont plus :</u>

ils correspondent aux arcs entre les sommets X-X* et Y* ; or on a vu que leur flux étaient nuls car il entraient dans la coupe. On ajoute la quantité δ ils ne sont donc plus admissibles.

<u>Les arcs qui sont devenus admissibles :</u>

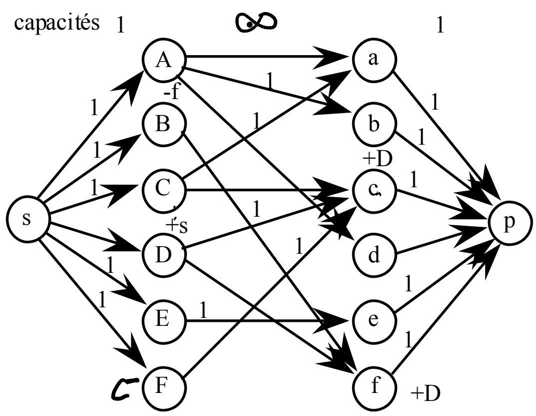
Ce sont des arcs sur lesquels on a retiré la valeur δ ; il relient X* à Y-Y*. Ils correspondent à des arcs sortant de la coupe. Donc qui vont permettre d'améliorer le couplage.

<u>La capacité de l'ancienne coupe minimale</u> est égale au cardinal de Y*+ le cardinal de X-X*.

Le nouveau marquage contient le précédent :

On a enlevé des arcs qui ne permettaient pas d'améliorer le marquage et ajouté des arcs de flux nuls entre X-X* et Y*. Donc si on applique Ford Fulkerson sur ce nouveau graphe, on marquera nécessairement les mêmes sommets que précédemment, plus éventuellement d'autres grâce aux arcs ajoutés de flux nuls.

f/ A noter que le flot initial est toujours valide sur le réseau de transport modifié car les arcs qui portaient ce flot sont toujours presents sur le réseau.



Marquage à l'arrêt de l'algorithme : B(-f); F(-c); D(+s); c(+D); f(+D).

g/

Les éléments non rayés sont ceux qui correspondent aux arcs entre X* et Y-Y* Le plus petit élément non rayé correspond bien à δ le plus petit coût réduit d'affectation d'un programmeur de X* à un programme de Y-Y*. Retrancher δ aux éléments non rayés permet donc d'enlever δ directement sur les arcs entre X* et Y-Y*. Les éléments rayés 2 fois sont ceux qui correspondent aux arcs entre X-X* et Y* quand ils existent (c'est-à-dire qu'ils sont des arcs admissibles du graphe). Ces arcs ont un flux nul sinon ils seraient marqués (Cf. c/ et e/).

La procédure est équivalente on évite en plus de soustraire et d'ajouter δ sur les arcs entre X* et Y*.

h/

à partir du graphe de f/ :

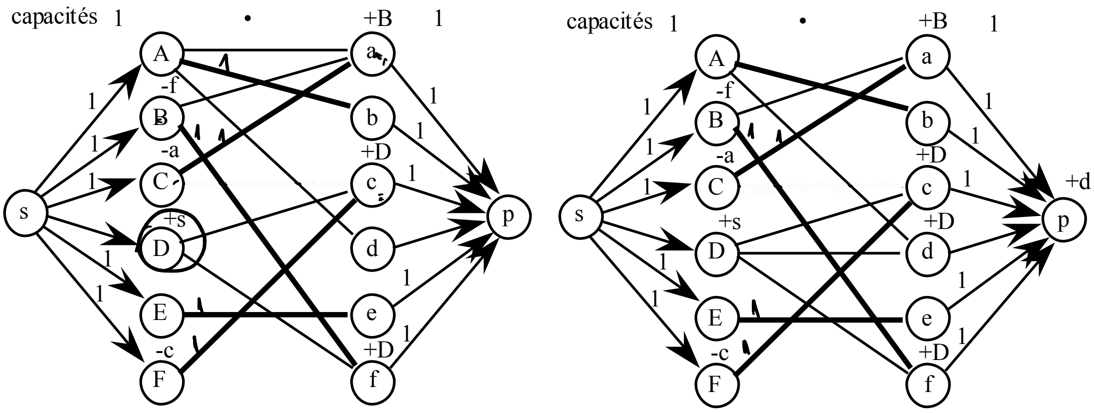
	a	b	c(Y*)	d	e	f(Y*)
A	0	0	12	0	57	11
B(X*)	***6***	51	17	36	43	0
C	0	45	0	34	44	44
D(X*)	33	69	0	10	78	0
E	61	29	52	38	0	80
F(X*)	49	64	0	77	71	20

1 δ = 6

	a(Y*)	b	c(Y*)	d	e	f(Y*)
A	0	0	18	0	57	17
B(X*)	0	45	17	30	37	0
C(X*)	0	45	6	34	44	50
D(X*)	27	63	0	***4***	72	0
E	61	29	58	38	0	86
F(X*)	43	58	0	71	65	20

2 δ = 4

avec tableau 1 : X* = (B, D, F); Y* = (c, f) avec tableau 2 : X* = (B, C, D, F); Y* = (a, c, f)



Figures allant avec le tableau 2 et 3.

Marquage : B(-f); C(-a); D(+s); F(-c); a(+B); c(+D); f(+D) et δ = 4 (première figure à gauche)

	a(Y*)	b	c(Y*)	d	e	f(Y*)
A	4	0	22	0	57	21
B(X*)	0	41	17	26	33	0
C(X*)	0	41	6	30	40	50
D(X*)	27	59	0	0	68	0
E	61	29	62	38	0	90
F(X*)	43	54	0	67	61	20

Sur la figure de droite ci-dessus, on trouve la chaîne améliorante [s, D, d, p] +1. ce qui nous permet d'obtenir le couplage maximum.

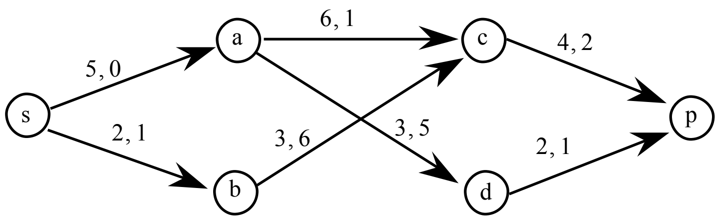
La condition d'arrêt est donc l'obtention de Y = Y*. Tous les programmes sont affectés à un programmeur.

A chaque fois que l'on fait apparaître un zéro, on n'ajoute à la valeur de la solution (Cf. Q1) que la plus petite valeur possible (sélection des δ). Donc dès que l'on obtient un zéro par ligne et colonne on peut trouver un couplage, et il est de coût minimum. Nous trouvons une solution de coût 103+2+6+4 = 115.

Correction du TD n°13

Exercice sur l'algorithme de ROY :

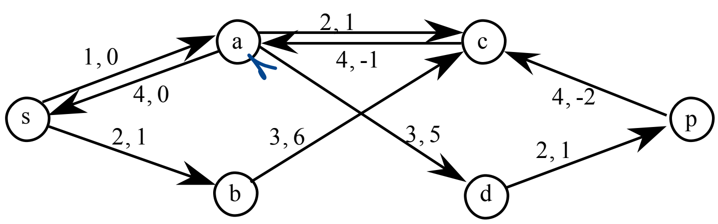
Premier graphe d'écart :



Chemins :

(s, a, c, p) -> coût = 0+1+2=3; (s, a, d, p) -> coût = 0+5+1=6; (s, b, c, p) -> coût = 1+6+2=9. On améliore le flux de 4 (coût de 4*3=12).

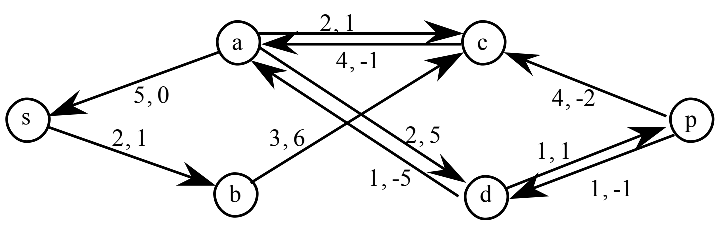
Deuxième graphe d'écart :



Chemins :

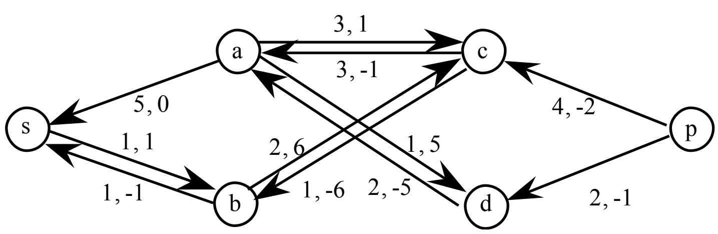
(s, a, d, p) -> coût = 0+5+1=6; (s, b, c, a, d, p) -> coût = 1+6-1+5+1=12. On améliore le flux de 1 (coût de 1*6=6).

Troisième graphe d'écart :



(s, b, c, a, d, p) -> coût = 1+6-1+5+1=12. On améliore le flux de 1 (coût de 1*12=12).

Quatrième graphe d'écart :



Pas de chemins de s à p, le flux est maximal et à coût minimal.

Résultat pour un coût de (4*3)+(1*6)+1*12)=30.