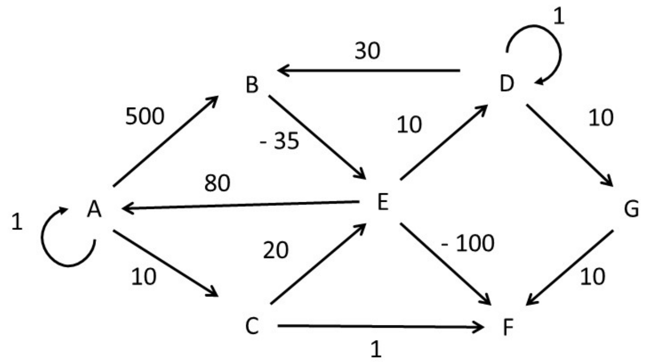
Q1 – Algorithme de Bellman

Déroulez l'algorithme de Bellman pour rechercher les chemins de valeurs minimales partant du sommet A dans le graphe ci-dessous :



Votre copie doit indiquer sous la forme que vous voulez, mais de façon suffisamment claire :

- les valeurs initiales en début de calcul,
- les résultats intermédiaires tout au long du déroulement de l'algorithme,
- les résultats en fin d'exécution,
- la raison pour laquelle l'algorithme s'arrête.

k	A	B	C	D	E	F	G
0	0 A						
1	0 A	500 A	10 A				
2	0 A	500 A	10 A		30 C	11 C	
3	0 A	500 A	10 A	40 E	30 C	-70 E	
4	0 A	70 D	10 A	40 E	30 C	-70 E	50 D
5	0 A	70 D	10 A	40 E	30 C	-70 E	50 D

Chaque case « itération k / sommet » contient la valeur π courante ainsi que le prédécesseur du sommet ayant fourni cette valeur. Les cases vides correspondent à la valeur « ∞ » pour π.
Les valeurs initiales sont dans la ligne k=0. Les valeurs finales sont dans la ligne k=5.
L'algorithme s'arrête car toutes les valeurs de π sont identiques pour deux itérations successives k=4 et k=5.

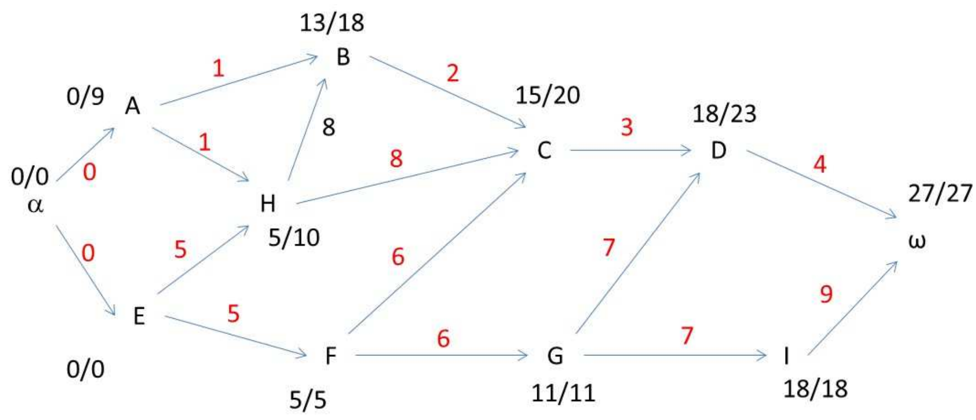
Q2 – Graphe d'ordonnancement

2.1- Construisez le graphe d'ordonnancement pour le tableau de contraintes suivant :

Tâche	A	B	C	D	E	F	G	H	I
Durée	1	2	3	4	5	6	7	8	9
Contraintes	aucune	A H	B F H	C G	aucune	E	F	A E	G

Interprétation (exemples) : A n'a aucune contrainte pour pouvoir démarrer ; H ne peut démarrer que lorsque A et E sont terminées.

2.2- Calculez le calendrier au plus tôt et le calendrier au plus tard. Vous considérerez que la date au plus tard de fin de projet est égale à sa date au plus tôt.



α et ω représentent respectivement le début du projet et la fin du projet.
Les valeurs en rouge sont celles associées aux arcs.
Les couples de valeurs à côté des sommets correspondent aux « date au plus tôt / date au plus tard » pour chaque tâche.

Q3 – Modélisation

Vous voyagez dans un monde que vous ne connaissez pas. Vous partez d'un lieu que nous nommerons D, avec un budget B pour vos déplacements. A partir de ce point vous découvrirez progressivement le monde dans lequel vous êtes, de point en point, à l'aide de moyens de transport mis à votre disposition.
Lorsque vous êtes à un point P (y compris au point 'D' bien évidemment), vous disposez des informations suivantes : les autres lieux S1, S2, ... auxquels vous pouvez accéder par le moyen de transport qui vous est proposé, ainsi que les coûts nécessaires C1, C2, … pour vous y rendre. Vous décidez alors du point vers lequel vous allez vous rendre, vous payez et vous y allez.
Votre voyage peut bien entendu vous amener à revenir dans un lieu par lequel vous êtes déjà passé.
Au début de votre voyage, on vous a dit qu'à chacune de vos étapes vous pourriez trouver un message vous demandant de revenir de façon urgente à votre point de départ 'D', toujours en payant vos déplacements bien entendu ! Les coûts « aller » et « retour » sont les mêmes.
Vous devez constamment vous assurer que cela sera possible.

Avant de partir, vous avez mis au point un programme pour vous aider, et vous voyagez avec votre ordinateur.
Quel programme (utilisant la théorie des graphes bien entendu) avez-vous mis au point avant de partir ? Comment utilise-t-il les graphes ?

Soit :

à chaque endroit Pi, nous avons une réserve budgétaire de Bi

pour i=0, nous avons P0=D et B0=B

à un point Pi, nous connaissons les autres points accessibles Si1, Si2, … ainsi que les coûts associés pour un déplacement Pi-Sij : Ci1, Ci2, …

Nous disposons d'un graphe contenant les chemins découverts et les coûts de déplacement associés. Ce graphe est construit progressivement tout au long du voyage .

Au point de départ, nous connaissons P0=D et ses successeurs immédiats S01, S02, ….
(initialisation du graphe).

Lorsque nous arrivons à un point Pi, nous ajoutons ses successeurs Sij au graphe (s'ils n'y sont pas déjà), avec des arcs de valeurs Cij.

Puisque rien de spécial n'est dit à ce sujet, et parce que c'est nécessaire à la résolution du problème (aucun étudiant n'a émis l'hypothèse contraire…), on suppose que tous les déplacements peuvent se faire dans les 2 sens : le graphe est donc orienté et symétrique.

Lorsque nous arrivons en Pi :

Nous augmentons le graphe (cf ci-dessus).

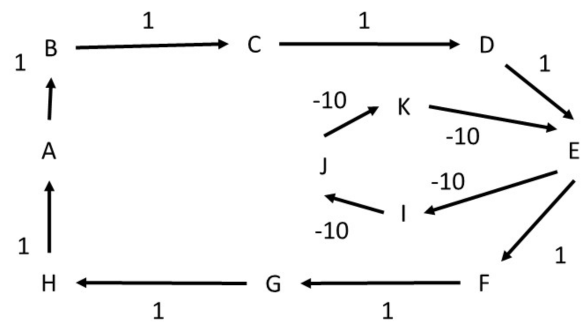
Nous déroulons un algorithme de recherche des chemins de plus faibles coûts partant de D vers tous les autres. Nous connaissons ainsi les coûts pour retourner en D depuis les endroits accessibles depuis Pi : Si1, Si2, … (le graphe est symétrique, donc pour tout sommet S le chemin le plus court de D vers S est le même que celui de S vers D).

S'il existe un Sij d'où le coût du retour en D est au plus égale à Bi-Cij (ce qu'il nous restera en poche si nous allons en Sij), alors nous y allons.

Sinon, il est préférable de retourner dès à présent en D car nous n'avons de toute façon pas d'autre choix !

Q4 – Circuit et chemin de plus faible valeur

Y a-t-il un algorithme capable de rechercher les chemins de valeurs les plus faibles en partant de A, de B, …etc vers tous les autres dans le graphe suivant :



Expliquer !

Important : pour cette question Q4, seule une réponse correcte avec une explication pourra vous apporter des points.

Non.

On note que le graphe contient le circuit A-B-C-D-E-I-J-K-E-F-G-H-A.

Donc, quel que soit le sommet de départ, il existe un chemin allant vers tous les autres et passant par le circuit E-I-J-K-E.

Ce circuit étant « absorbant » (coût négatif pour la recherche des chemins de plus faible coût), il suffit de passer une infinité de fois dans ce circuit pour relier un sommet à un autre pour faire tendre la valeur du chemin vers - ∞.

Donc il n'y a pas de « chemin le plus court ».

Q5 – Tracez un graphe le plus simple possible…

…tel que $\Gamma^{+n}(x) = \Gamma^{+(n+1)}(x)$ pour tout sommet x et tout entier n≥1.

G = ({x} , { (x,x) }) 1 sommet et 1 boucle, avec : $\forall\, n \geq 1, \Gamma^{+n}(x) = \{ x \}$

Mais on pouvait aussi dire que si le graphe est vide (G=(∅,∅)), alors il répond à la propriété énoncée… Et c'est sans doute le plus simple possible ! Mais personne n'y a pensé.

---

fin