Chapitre ITC5 : Graphes

Exercices

Exercice ITC5.1 : Graphe des diviseurs [*]

On considère un graphe dont les sommets sont les entiers de 2 à 12 et les arcs représentent la relation de division : un arc de x vers $y \neq x$ signifie « x divise y ».

1. Dessinez le graphe associé.
2. Indiquez le degré entrant et sortant des nœeuds 2, 4, 7 et 12.
3. Comment s'appelle (en termes mathématiques) le voisinage sortant d'un nœud ? et son voisinage entrant ?

Exercice ITC5.2 : Jeu d'échecs [*]

On souhaite représenter le jeu d'échecs par un graphe : les nœuds sont les états possible du jeu et les arcs indiquent qu'on peut atteindre un autre état et jouant une pièce.

1. Le graphe est-il orienté ?
2. Donner l'ordre de grandeur du nombre de nœuds.

Exercice ITC5.3 : Réseau d'amis [*]

On considère le tableau suivant traduisant les relations d'amitiés sur un réseau social :

	Jean	Emma	Fred	Luca	Lila	Stan	Anne
Jean		x	x			x	x
Emma	x		x				x
Fred	x	x		x		x	
Luca			x		x		
Lila				x			
Stan	x		x				x
Anne	x	x				x	

1. Tracez le graphe correspondant.
2. Écrire la matrice d'adjacence, puis la liste d'adjacence de ce graphe.
3. On appelle *distance* entre deux nœuds, le nombre minimal d'arcs/arrêtes à suivre pour aller d'un nœud à l'autre.
   Quelle est la distance entre Jean et Lila ?
4. On appelle *excentricité* d'un nœud le maximum des distances entre ce nœud et tous les autres nœuds.
   Donner pour chaque nœud son degré d et son excentricité e.
5. On appelle *diamètre* l'excentricité maximale et *rayon* l'excentricité minimale. On appelle *centre* du graphe tout nœud dont l'excentricité est égale au rayon.
   Déterminez ces caractéristiques pour ce graphe.
6. *Bonus* : on constate qu'il existe 2 personnes qui ont le même degré. Justifiez que c'est toujours le cas pour un graphe non orienté.

Exercice ITC5.4 : Résolution d'un problème à l'aide d'un graphe [**]

On cherche à résoudre le problème suivant : on dispose d'une réserve d'eau très grande, et de deux seaux, l'un de 3L, l'autre de 5L de capacité. On souhaite prélever exactement 4L d'eau. Pour cela, on effectue des opérations qui consistent toujours :

— soit à plonger un récipient dans un récipient plus grand et à le remplir jusqu'à ras bord

— soit à verser la totalité d'un récipient dans un autre à condition que celui-ci puisse le contenir.

Par exemple, au début, on peut par exemple remplir le seau de 5L à ras bord grâce à la réserve. Ensuite, on

peut soit le reverser dans la réserve, soit prélever 3L avec le petit seau (et il reste 2L dans le grand seau) ; etc.
On va représenter ce problème grâce à un graphe où

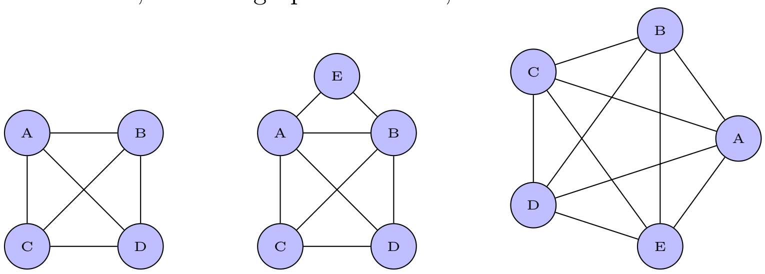
— les états possibles sont les nœuds, notés par un couple (x,y) donnant la quantité d'eau dans le grand seau puis dans le petit seau

— un arc de a à b indique qu'on peut passer d'un état a à un état b

1. En considérant par exemple un état où le grand seau contient $2L$, justifiez que le graphe est forcément orienté, c'est-à-dire qu'il existe des transformations qu'on peut faire dans un sens mais pas dans l'autre.
2. Tracez le graphe associé à ce problème.
3. Déduisez-en les étapes à effectuer pour prélever 4L d'eau dans le grand seau.

Exercice ITC5.5 : Chemins eulériens [*]

Un chemin eulérien dans un graphe est un chemin qui parcourt toutes les arrêtes une et une seule fois.
Déterminez, dans les graphes suivants, s'il existe des chemins eulériens :



Remarque : on peut démontrer que, si tous les nœuds ont degré pair, on peut tracer un chemin eulérien en partant de n'importe quel nœud. Si deux nœuds seulement ont un degré impair, il suffit de partir d'un nœud de degré impair et d'aller à l'autre nœud, puis de continuer ; dans les autres cas, c'est impossible.

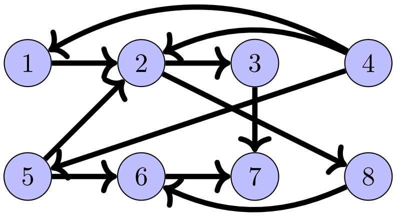
Exercice ITC5.6 : Parcours d'un graphe [**]

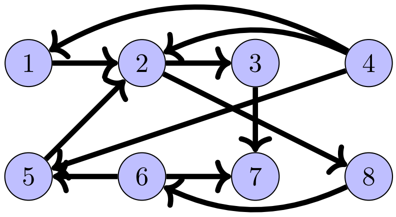
On considère le graphe contenant 10 sommets de A à J, dont la liste d'adjacence est :
`A=[[B,C],[A,D,E],[A,F,G],[B],[B,H,I],[C,I],[C],[E,J],[E,F],[H]]`
et les poids associés à chaque arc/arrête valent
`P=[[2,3],[2,4,2],[3,1,6],[4],[2,3,3],[1,4],[6],[3,2],[3,4],[2]]`

1. Ce graphe est-il orienté ?
2. Dessinez ce graphe en plaçant les poids sur les arrêtes. Est-il connexe ?
3. Effectuez un parcours en largeur de ce graphe à partir du nœud A.
4. Effectuez un parcours en profondeur de ce graphe à partir du nœud A.
5. Lors d'un parcours en profondeur, si on rencontre au moins 1 fois lors de la recherche des voisins d'un nœud, un nœud qui a déjà été exploré, alors cela signifie que ce graphe contient un cycle.
Est-ce le cas ici ? Quel est le cycle ?
6. Déterminez visuellement deux chemins permettant d'aller de A à J, et indiquez leur poids.

Exercice ITC5.7 : Tri topologique [**]

Parfois, quand une série littéraire ou audiovisuelle a du succès, les auteurs en profitent pour ajouter toutes sortes de nouveaux épisodes situés parfois avant, parfois au milieu, parfois après la série initiale, de sorte que le lecteur ou le spectateur ne sait plus dans quel ordre il faut les regarder.
On va représenter cela par un graphe orienté dans lequel les nœuds représentent les tomes ou les épisodes, et les arcs signifient la précédence : s'il y a un arc de i vers j, c'est qu'il faut avoir lu/vu i avant j.
On considère deux graphes représentant une telle situation :





1. Si le graphe possède un cycle, alors il y a un problème. Déterminez lequel de ces deux graphes n'est pas cyclique.
2. Pour le graphe en question, on va effectuer un *tri topologique* sur les nœuds. Il s'agit de trier les nœuds dans un ordre tel que, pour tout arc allant de i vers j, alors i se trouve avant j. Ainsi, ce tri nous donnera l'ordre dans lequel lire ou voir les nœuds.

   La technique est la suivante :

   — on prépare une pile p initialement vide

   — on part d'un nœud quelconque (celui de plus petit numéro, par exemple) et on effectue un parcours en profondeur (on choisira toujours les voisins dans l'ordre numérique). Lors du parcours, à chaque fois qu'un nœud est totalement exploré (c'est-à-dire lorsque l'algorithme « rebrousse chemin », on empile dans p le nœud en question

   — on choisit alors un nœud pas encore exploré, et on recommence de même, mais sans explorer les nœuds déjà explorés précédemment

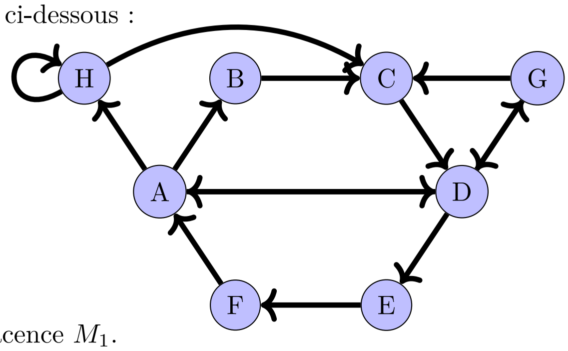
   — on continue tant qu'il reste des nœuds

   — on obtient alors l'ordre de tri topologique en dépilant p

   Appliquez cette technique au graphe acyclique et déterminez un ordre de tri topologique (il n'est pas unique).

Exercice ITC5.8 : Nombre de chemins [**]

On considère le graphe orienté ci-dessous :



1. Écrivez sa matrice d'adjacence M_1.
2. Effectuez un parcours en largeur depuis A.
3. Effectuez un parcours en profondeur depuis A.
4. On pose $M_2 = M_1^2$. Justifiez que $M_2[i,j]$ représente le nombre de chemin de longueur 2 allant de i à j.
5. Généralisez cette propriété à $M_n = M_1^n$.

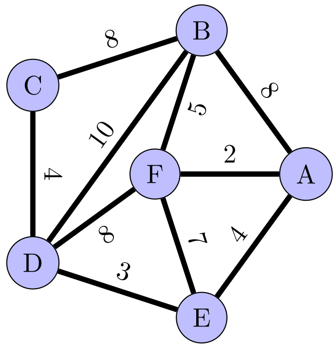
   Un ordinateur donne les résultats suivants :

$$M_2 = \begin{pmatrix} 1 & 0 & 2 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 2 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} M_3 = \begin{pmatrix} 0 & 1 & 2 & 4 & 0 & 1 & 0 & 2 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 2 & 0 & 1 & 0 & 1 \\ 3 & 0 & 2 & 1 & 2 & 0 & 2 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 2 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \end{pmatrix} M_6 = \begin{pmatrix} 16 & 2 & 14 & 10 & 12 & 2 & 12 & 9 \\ 1 & 3 & 3 & 7 & 1 & 2 & 1 & 4 \\ 9 & 1 & 8 & 5 & 7 & 1 & 7 & 5 \\ 6 & 9 & 13 & 24 & 5 & 7 & 5 & 14 \\ 5 & 0 & 3 & 2 & 4 & 0 & 4 & 2 \\ 2 & 5 & 6 & 12 & 2 & 4 & 2 & 7 \\ 10 & 4 & 11 & 12 & 8 & 3 & 8 & 9 \\ 5 & 4 & 7 & 11 & 4 & 3 & 4 & 7 \end{pmatrix}$$

   Interprétez les valeurs de : $M_2[0,2]$; $M_3[0,2]$; et $M_6[0,2]$
6. Combien existe-t-il de boucles à 6 arcs ?

Exercice ITC5.9 : Application simple de l'algorithme de Dijkstra [**]

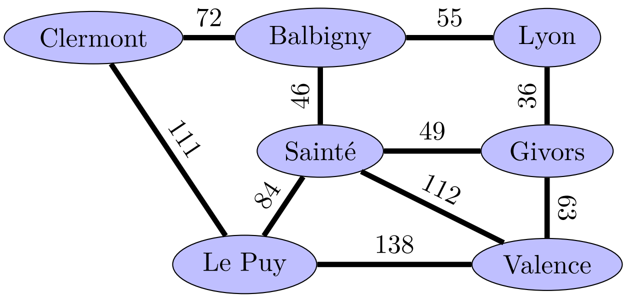
On considère le graphe pondéré suivant :



Appliquez l'algorithme de Dijkstra pour trouver le plus court chemin de E à B.

Exercice ITC5.10 : Plus court chemin [**]

On considère le graphe non-orienté et pondéré (les poids sont les temps en minutes indiquées par un GPS) :



On classe les sommets dans l'ordre d'importance des villes : Sainté, Lyon, Clermont, Valence, Le Puy, Givors, Balbigny.

1. Écrivez sa liste d'adjacence.
2. Appliquez l'algorithme de Dijkstra pour déterminer le chemin le plus court de Clermont-Ferrand à Valence.

Exercice ITC5.11 : Séparation en deux groupes des pixels d'une image [**]

Pour reconnaître automatiquement des caractères dans une image, on commence par binariser l'image, c'est-à-dire par passer tous les pixels à 2 couleurs : noir ou blanc. Pour cela, il suffirait d'appliquer un seuil : tous les pixels dont la luminosité dépasse une certaine valeur deviennent blancs, les autres sont noirs ; cependant, le choix de la luminosité seuil est dépendant des propriétés de l'image. On va voir ici un exemple d'algorithme un peu abstrait mais qu'on va appliquer ; cet exercice est inspiré de questions de l'épreuve CCINP PC 2023.
Considérons l'image ci-dessous à 4 pixels ; chaque pixel a une valeur comprise entre 0 (noir) et 255 (blanc) avec des nuances de gris.

117	28
46	217

On va créer un graphe contenant un nœud par pixel, plus un nœud source S et un nœud puit P. Le but est de faire passer un fluide imaginaire de la source au puits en passant par des tuyaux imaginaires entre les nœuds, c'est-à-dire des arcs pondérés par 3 nombres : un débit maximal, un débit utilisé et un débit restant :

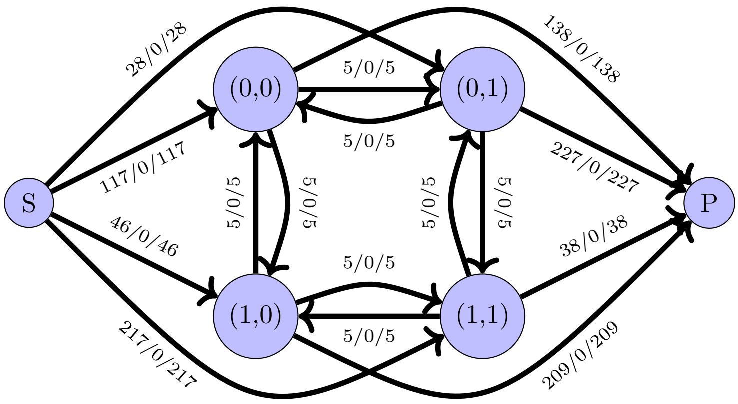
— on place un arc de S vers chaque nœud représentant un pixel, avec comme débit maximal la valeur du pixel

— on place un arc de chaque nœud représentant un pixel vers P, avec comme débit maximal 255 - (valeur du pixel)

— on place un arc de chaque nœud représentant un pixel vers ses pixels voisins (haut/bas/droite/gauche) avec comme débit maximal 5

— on initialise les débits utilisés à 0 ; le débit restant est toujours calculé comme (débit maximal) - (débit utilisé)

On obtient alors le graphe suivant pour l'image choisie :



1. On dit qu'un arc est saturé lorsque son débit restant est nul.

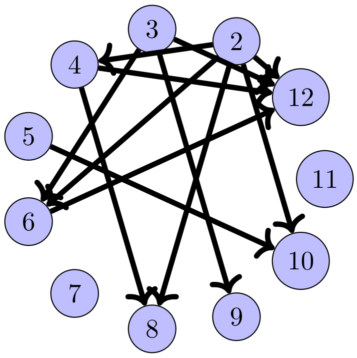
   Cherchez, parmi les chemins permettant d'aller de S à P sans passer par des arcs saturés, les plus courts (ceux qui passent par le moins de nœuds possible) ; parmi ces chemins, sélectionnez celui dont le minimum des débits restants des arcs parcourus par ce chemin est maximal (c'est-à-dire, pour chaque chemin, évaluez le débit maximal que vous pouvez faire passer dans ce chemin en respectant les contraintes de débit restant ; et choisissez le chemin qui permet un débit le plus grand possible).

   Pour ce chemin, on choisit de faire passer un débit égal au débit maximal possible par ce chemin ; on ajoute alors ce débit au débit utilisé de chaque arc parcouru, et on met à jour les débits restants (au moins 1 débit restant doit passer à 0 : l'arc devient saturé).
2. Recommencez le principe de la question précédente jusqu'à ce qu'il n'y ait plus de chemin possible.
3. On choisit alors comme pixels blancs tous ceux qui sont accessibles depuis la source en passant par des arrêtes non saturées ; les autres pixels sont noirs.

   Déterminez l'image binarisée obtenue.

Corrections des exercices

Corrigé de l'exercice ITC5.1 : Graphe des diviseurs [*]

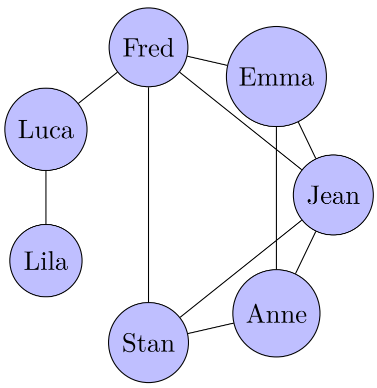
1. 

2. Pour 2 : $d^- = 0$ et $d^+ = 5$
   Pour 4 : $d^- = 1$ et $d^+ = 2$
   Pour 7 : $d^- = 0$ et $d^+ = 0$
   Pour 12 : $d^- = 4$ et $d^+ = 0$
3. Le voisinage entrant représente les diviseurs, le voisinage sortant représente les multiples.

Corrigé de l'exercice ITC5.2 : Jeu d'échecs [*]

1. Il est orienté car s'il existe des coups pour lesquels on peut revenir en arrière, ce n'est pas toujours le cas (si on mange un epièce, ou si on avance un pion).
2. Il y a 32 pièces sur 64 cases donc au maximum $\frac{64!}{32!}$ états soit environ 5.10^{53} états. En réalité, tous ne sont pas possibles, mais le nombre d'états reste gigantesque.

Corrigé de l'exercice ITC5.3 : Réseau d'amis [*]

1. 

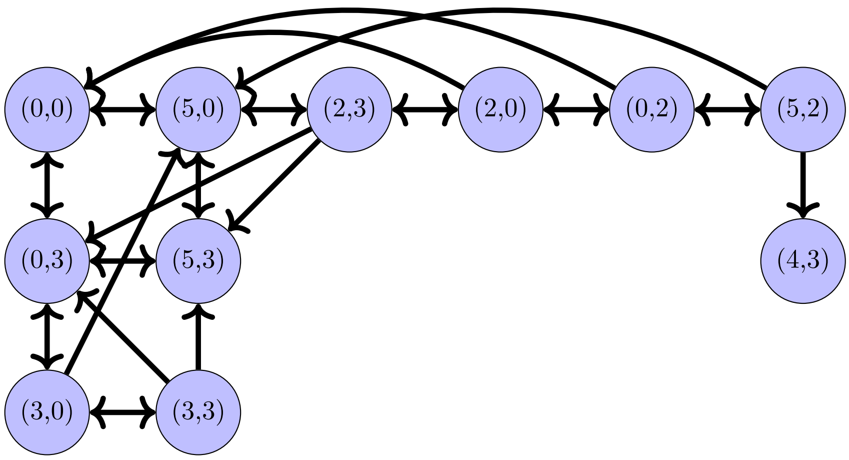
2. Le tableau proposé est déjà une matrice d'adjacence : $\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$
   Par les listes : [[Emma,Fred,Stan,Anne],[Jean,Fred,Anne],[Fred,Emma,Luca,Stan],[Fred,Lila],[Luca],[Jean,Fred,Anne],[Jean
3. La distance de Jean à Lila vaut 3.
4. Lila : $d = 1$ et $e = 4$ (Anne est la plus éloignée)
   Luca : $d = 2$ et $e = 3$ (Anne est la plus éloignée)
   Fred : $d = 4$ et $e = 2$ (Lila et Anne)
   Emma : $d = 3$ et $e = 3$ (Lila)

Jean : $d = 4$ et $e = 3$ (Lila)
Anne : $d = 3$ et $e = 4$ (Lila)
Stan : $d = 3$ et $e = 3$ (Lila)

5. Le diamètre vaut donc 4, le rayon vaut 2 et le centre est Fred ; c'est donc à lui d'organiser les fêtes.
6. Soit un graphe à n nœuds. Les degrés des nœuds varient entre 0 et $n-1$. Supposons que tous les nœuds aient des degrés différents, ils vont donc de 0 à $n-1$. Alors il existe un nœud de degré 0 (lié à personne) et un nœud de degré $n-1$ (lié à tout le monde) ce qui est absurde.
   Si le graphe est orienté, c'est différent.

Corrigé de l'exercice ITC5.4 : Résolution d'un problème à l'aide d'un graphe [**]

1. Si on verse les 2L dans la réserve, on ne peut plus les reprendre.
2. Il faut partir de (0,0) et lister toutes les actions possibles : c'est une sorte de parcours de graphe.



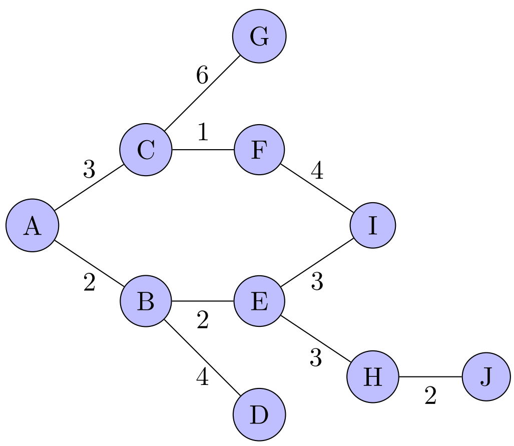
3. Le chemin à suivre est donc $(0,0) \to (5,0) \to (2,3) \to (2,0) \to (0,2) \to (5,2) \to (4,3)$

Corrigé de l'exercice ITC5.5 : Chemins eulériens [*]

Pour le premier : non.
Pour le deuxième : oui en partant de C ou de D ; par exemple C-D-B-E-A-C-B-A-D.
Pour le troisième : oui en partant de n'importe quel nœud, par exemple A-B-C-D-E-A-C-E-B-D-A

Corrigé de l'exercice ITC5.6 : Parcours d'un graphe [**]

1. On constate qu'à chaque fois qu'il y a un arc de x vers y, il y a aussi un arc de y vers x : ce graphe est donc non-orienté.



2.

3. En largeur, on parcourt : A,B,C,D,E,F,G,H,I,J
4. En profondeur, on parcourt : A,B,D,E,H,J,I,F,C,G

5. En arrivant en C, on trouve comme voisin A qui a déjà été exploré, donc il y a un cycle : A,B,E,I,F,C
6. A,B,E,H,J a pour poids 9.
   A,C,F,I,E,H,J a pour poids 16.

Corrigé de l'exercice ITC5.7 : Tri topologique [**]

1. Le graphe de droite contient le cycle 5-2-8-6-5.
2. On part du nœud 1, le parcours en profondeur donne 1-2-3-7-8-6 et la pile vaut p=7,3,6,8,2,1.
   On part de 4, le parcours est 4-5 donc p=7,3,6,8,2,1,5,4.
   Un ordre de tri topologique est donc 4,5,1,2,8,6,3,7.

Corrigé de l'exercice ITC5.8 : Nombre de chemins [**]

1. $M_1 = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$
2. A,B,D,H,C,E,G,F
3. A,B,C,D,E,F,G,H
4. Par définition, $M_2[i,j] = \sum_k M_1[i,k].M_1[k,j]$ c'est-à-dire la somme des k tels qu'il existe un arc de i vers k et un arc de k vers j, c'est donc le nombre de chemins à 2 arcs allant de i à j.
5. $M_n[i,j]$ représente le nombre total de chemine de longueur n allant de i à j.
   On lit :
   — $M_2[0,2] = 2$ car il existe 2 chemins de longueur 2 entre A et C, à savoir (A,B,C) et (A,H,C)
   — $M_3[0,2] = 2$ car il existe 2 chemins de longueur 2 entre A et C, à savoir (A,H,H,C) et (A,D,G,C)
   — $M_6[0,2] = 14$ car il existe 2 chemins de longueur 14 entre A et C, à savoir (A,D,E,F,A,B,C), (A,D,E,F,A,H,C), (A,B,C,D,A,B,C), (A,B,C,D,A,H,C), (A,H,H,H,H,H,C), (A,H,H,C,D,G,C), (A,D,A,H,H,H,C), (A,D,A,D,A,H,C), (A,D,A,D,A,B,C), (A,D,G,C,D,G,C), (A,H,C,D,A,H,C), (A,H,C,D,A,B,C), (A,D,G,C,A,B,C), (A,D,G,C,A,H,C).
6. Une boucle est un chemin qui revient au point de départ. Par exemple $M_6[0,0]$ indique qu'il existe 16 boucles de A vers A à 6 arcs ; pour avoir le nombre total de boucle, il faut sommer tous les termes de la diagonale, c'est donc $tr(M_6) = 74$.

Corrigé de l'exercice ITC5.9 : Application simple de l'algorithme de Dijkstra [**]

Doit d la liste des distances et p celle des prédécesseurs.

— initialement, d=[$\infty,\infty,\infty,\infty,0,\infty$] et p=[nul,nul,nul,nul,nul,nul]
— on explore le voisinage de E : d=[$\underline{4},\infty,\infty,\underline{3},\cancel{0},\underline{7}$] et p=[E,nul,nul,E,nul,E]
— on explore le voisinage de D : d=[$4,\infty,\underline{7},\cancel{3},\cancel{0},7$] et p=[E,nul,D,E,nul,E]
— on explore le voisinage de A : d=[$\cancel{4},\underline{12},7,\cancel{3},\cancel{0},\underline{6}$] et p=[E,A,D,E,nul,A]
— on explore le voisinage de F : d=[$\cancel{4},\underline{11},7,\cancel{3},\cancel{0},\cancel{6}$] et p=[E,F,D,E,nul,A]
— on explore alors C mais cela ne change rien.

Donc le chemin le plus court est E-A-F-B, de longueur 11.

Corrigé de l'exercice ITC5.10 : Plus court chemin [**]

1. `[[Le Puy, Givors, Balbigny], [Givors, Balbigny], [Le Puy, Balbigny], [Sainté, Le Puy, Givors], [Sainté, Clermont, Valence], [Sainté, Lyon, Valence], [Sainté, Lyon, Clermont]]`

2. On donne deux tableaux : `distance` qui donne la distance depuis Clermont, et `origine` qui indique depuis quelle ville on vient.

   Étape 0 :
   distance=[∞,∞,0,∞,∞,∞,∞]
   origine=[nul,nul,nul,nul,nul,nul,nul]

   Étape 1 : on explore le voisinage de Clermont :
   distance=[∞,∞,~~0~~,∞,111,∞,72] origine=[nul,nul,nul,nul,Clermont,nul,Clermont]

   Étape 2 : on explore le voisinage de Balbigny :
   distance=[118,127,~~0~~,∞,111,∞,~~72~~]
   origine=[Balbigny,Balbigny,nul,nul,Clermont,nul,Clermont]

   Étape 3 : on explore le voisinage du Puy :
   distance=[118,127,~~0~~,249,~~111~~,∞,~~72~~]
   origine=[Balbigny,Balbigny,nul,Le Puy,Clermont,nul,Clermont]

   Étape 4 : on explore le voisinage de Sainté :
   distance=[~~118~~,127,~~0~~,230,~~111~~,167,~~72~~]
   origine=[Balbigny,Balbigny,nul,Sainté,Clermont,Sainté,Clermont]

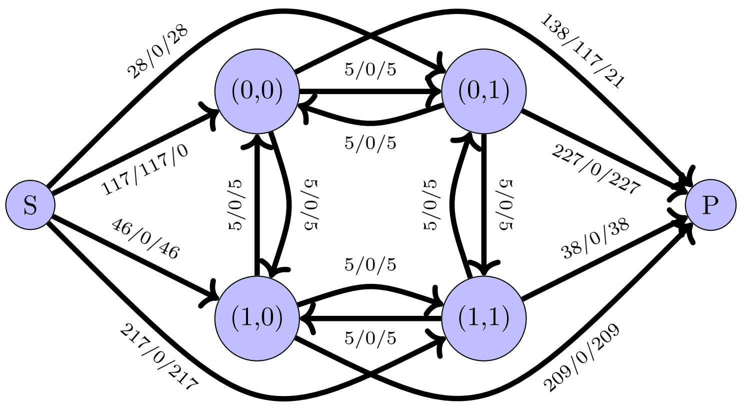
   Étape 5 : on explore le voisinage de Lyon :
   distance=[~~118~~,~~127~~,~~0~~,230,~~111~~,163,~~72~~]
   origine=[Balbigny,Balbigny,nul,Sainté,Clermont,Lyon,Clermont]

   Étape 6 : on explore le voisinage de Givors :
   distance=[~~118~~,~~127~~,~~0~~,226,~~111~~,~~163~~,~~72~~]
   origine=[Balbigny,Balbigny,nul,Givors,Clermont,Lyon,Clermont]

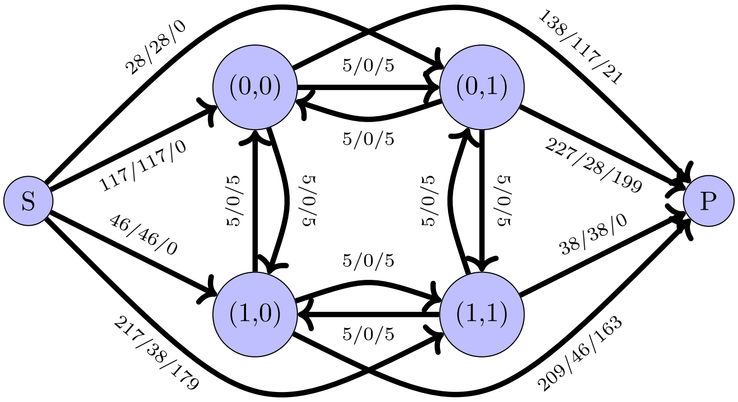
   Étape 7 : on est à Valence, donc on rembobine : de Valence on vient de GIvors, puis de Lyon, puis de Balbigny, puis de Clermont. Le chemin fait 226 minutes, sans compter les bouchons du tunnel sous Fourvière.

Corrigé de l'exercice ITC5.11 : Séparation en deux groupes des pixels d'une image [**]

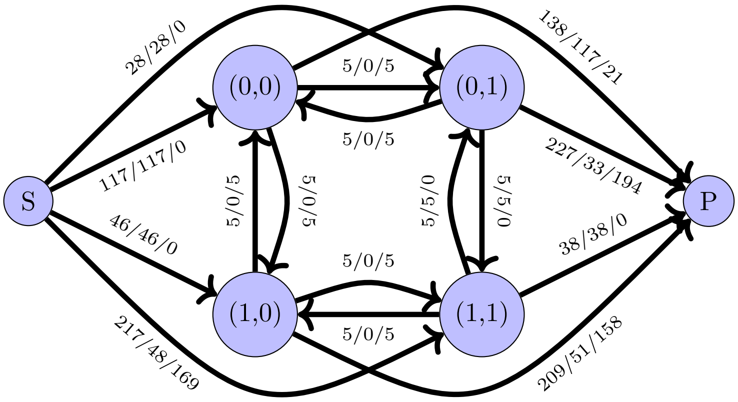
1. On choisit le chemin S-(0,0)-P de débit maximal 117 :



2. On choisit ensuite les chemins S-(1,0)-P de débit maximal 46, puis S-(1,1)-P de débit maximal 38, puis S-(0,1)-P de débit maximal 28 :



On prend ensuite les chemins à 4 nœuds commençant par (1,1), et on trouve S-(1,1)-(1,0)-P de débit maximal 5, puis S-(1,1)-(0,1)-P de débit maximal 5 :



On s'arrête car il n'y a plus de chemins possibles.

3. Seul le pixel (1,1) est accessible depuis S :