# Éléments de théorie des graphes

*Ce compte rendu est issu d'une présentation réalisée par Éric Sopena, professeur à l' I.U.T de Technologie de l'Université Bordeaux 1.*

*Ce document, portant sur les fondements de la théorie des graphes, constitue une approche théorique à l'usage des enseignants. Il n'est pas destiné à être directement exploité avec les élèves ; d'autres ressources seront prochainement proposées dans ce sens.*

*(Pour d'éventuelles questions, il est possible de joindre M. Sopena :* sopena@labri.fr)

# I Préambule

La théorie des graphes est utilisée dans un grand nombre de disciplines (mathématiques, physique, économie, etc.). Les recherches en théorie des graphes sont essentiellement menées par des informaticiens, du fait de l'importance des aspects algorithmiques (recherche de solutions).

Il s'agit essentiellement de modéliser des problèmes :

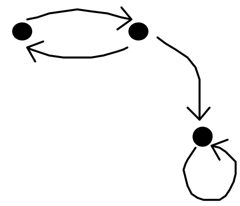
- on exprime un problème donné en termes de graphes ;
- il devient alors un problème de «théorie des graphes » que l'on sait le plus souvent résoudre car il rentre dans une catégorie de problèmes connus.

Les solutions de problèmes de graphes peuvent être :

- faciles et efficaces (car le temps nécessaire pour les traiter par informatique est raisonnable car il dépend polynomialement du nombre de sommets du graphe) ;
- difficiles (car le temps de traitement est exponentiel) ; dans ce cas on utilisera une heuristique, c'est-à-dire un processus de recherche d'une solution – pas forcément la meilleure.

# II Définitions et terminologie

### *Graphe orienté GO*



Il se représente par des points et des flèches entre les points : les points sont les sommets du graphe, les flèches des arcs (orientés) qui relient certains sommets entre eux. D'un point de vue mathématique, si S est l'ensemble des sommets, un graphe représente une relation binaire entre des éléments de S.

### *Graphe non orienté GNO*

Un graphe non orienté n'est qu'un graphe orienté symétrique ; si un arc relie le sommet *a* au sommet *b*, un autre arc relie le sommet *b* au sommet *a* : on ne trace alors qu'un trait entre *a* et *b* que l'on appelle une « arête ».

### *Degré d'un graphe*

Dans le cas d'un GO, le « degré sortant » d'un sommet *x* est le nombre d'arcs qui partent de *x*, noté $d^+(x)$, et son « degré entrant », noté $d^-(x)$, est le nombre d'arcs arrivant au sommet *x*.
On a la relation : $\Sigma\, d^+ = \Sigma\, d^- = |A|$ (nombre d'arcs).
Dans le cas d'un GNO, le degré est le nombre d'arêtes rattachées au sommet *x* (les boucles comptent pour 2).
On a la relation : $\Sigma\, d = 2|A|$.

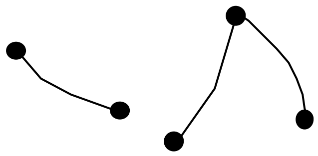
### *Chemin ou chaîne*

C'est une succession d'arcs parcourus dans le même sens. Le nombre d'arcs parcourus s'appelle la longueur du chemin.
On parle de « chaîne » si l'on ne tient pas compte de la direction des arcs ; on parle ainsi de chaîne dans les GNO.
Si un chemin revient à son point de départ, on parle de «circuit » dans un GO, ou de « cycle » dans un GNO.
L'adjectif «élémentaire » s'applique quand le chemin ou le cycle parcourt des sommets distincts.
La « distance » entre deux sommets est la longueur du plus court chemin entre ces deux sommets.
Enfin le « diamètre » d'un graphe est la plus grande distance séparant deux sommets de ce graphe.

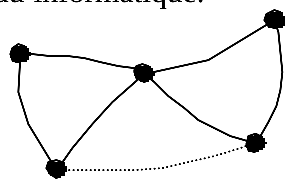
### *Connexité*

On dit qu'un GNO est « connexe » s'il deux éléments quelconques de ce graphe sont reliés par au moins une chaîne.
On appelle « composante connexe » d'un GNO, un sous-ensemble maximal de sommets tels qu'il existe une chaîne entre deux sommets quelconques.
Un GO est « fortement connexe » si, quels que soient deux sommets *x* et *y*, il existe un chemin reliant *x* à *y* et un reliant *y* à *x*.

Un graphe est « $k$-connexe » si on peut le déconnecter en retirant $k$ sommets et qu'on ne peut pas en en supprimant $k - 1$ ; son « degré de connexité » est alors $k$. Cela permet, par exemple, de mesurer la résistance aux pannes d'un réseau informatique.



Graphe à deux composantes connexes

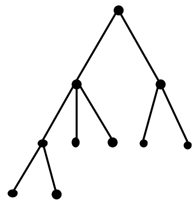


Graphe 1-connexe (traits pleins)
Avec les pointillés, il devient 2-connexe.

On définit de façon analogue la « $k$-arête connexité » qui concerne les arêtes au lieu des sommets.

### *Types de graphes*

#### Arbre



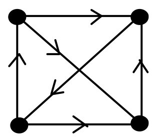
Ce sont des GNO connexes sans cycle.
Ou encore des GNO connexes dont le nombre de sommets est égal au nombre d'arêtes plus 1.

#### Graphe complet (clique)

C'est un GNO avec toutes les connexions possibles ; un graphe complet à $n$ sommets est noté $K_n$.

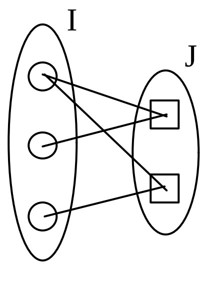
$K_1$ $K_2$ $K_3$ $K_4$

Dans le cas d'un graphe complet $K_n$, le nombre d'arêtes est $\frac{n(n-1)}{2}$.



On appelle « tournoi » un graphe complet orienté.
Une flèche peut alors représenter la relation « a gagné sur ».

#### Graphe biparti

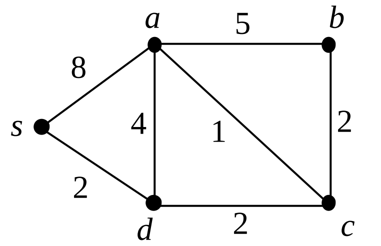



Un ensemble I de sommets d'un graphe est « indépendant » si aucun élément de I n'est connecté à un autre élément de I.
Un graphe « biparti » est un graphe qui représente des relations entre deux ensembles indépendants I et J.

On définit de même un graphe multiparti.

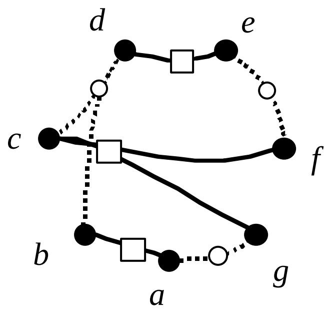
### Graphe valué ou pondéré

Quand les arêtes représentent un coût (en argent, en temps…), on leur attribue un nombre ; ce qui donne un graphe valué ou pondéré.

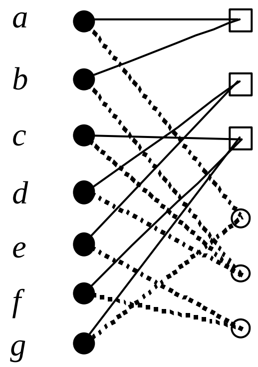



### Hypergraphe

Quand les relations ne sont pas toutes binaires, on parle d'hypergraphe ; on peut alors le transformer en graphe biparti.



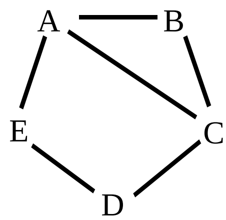

se transforme en :



# III Problèmes de coloration

Famille de problèmes qui se représentent par des GNO dont les arêtes sont des contraintes d'incompatibilité. On cherche alors à décomposer l'ensemble des sommets du graphes en sous-ensembles indépendants (compatibles). Dans la coloration d'une carte de géographie, deux pays ayant une frontière commune ne doivent pas être coloriés dans la même couleur ; on trace donc une arête entre ces deux pays (représentés par des sommets du graphe).

Exemple : un examen comporte cinq matières A, B, C, D, E, et des étudiants doivent passer plusieurs de ces matières. Si chaque épreuve dure une demi-journée, combien la session doit-elle comporter au minimum de demi-journées ?
Il faudra trois demi-journées au minimum par exemple : {A , D}, {B , E} et {C}.

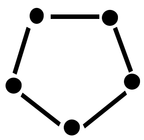



Une « $k$-coloration » est une application qui permet de colorer des sommets avec $k$ couleurs différentes (deux sommets voisins ont des couleurs différentes). Pour chaque couleur $c$, l'ensemble des sommets de couleur $c$ est alors un ensemble indépendant. Le plus petit nombre $k$ possible s'appelle le « nombre chromatique » du graphe, noté Khi(G).
Le nombre chromatique du graphe de l'exemple précédent est 3.

Remarques
1. $G = K_n \Leftrightarrow \text{Khi}(G) = n = |S|$ (nombre de sommets).
2. G contient $K_n \Rightarrow \text{Khi}(G) \geq n$.

Théorème d'Erdös

Pour tout $k$, il existe un graphe sans triangle qui nécessite au moins $k$ couleurs.



Exemple
Ce graphe ne comporte aucun triangle mais nécessite quand même trois couleurs.

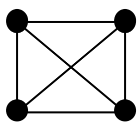
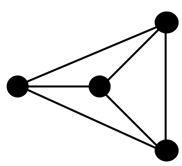
Théorème de Brooks (1941)

a) Khi(G) ≤ D(G) + 1 (où D(G) est le degré maximal du graphe).
b) Khi(G) = D(G) + 1 ⇔ G complet ou G est un cycle impair.

Théorème des 4 couleurs (Appel et Haken 1977)

Tout graphe planaire (qui peut être dessiné sans croisement d'arêtes) est 4-coloriable.

Ce théorème n'a été démontré que grâce à l'utilisation d'ordinateurs, tant le nombre de cas à étudier est grand.

Attention : ce graphe  est planaire car on peut le représenter ainsi : 

Remarque : dans le cas d'un graphe planaire sans triangle, trois couleurs suffisent.

### *Résolution du problème de coloration*

Tester la 1-colorabilité ou la 2-colorabilité est un problème facile.
Si on passe à la 3-colorabilité, on tombe dans des problèmes difficiles car le temps de traitement devient exponentiel.

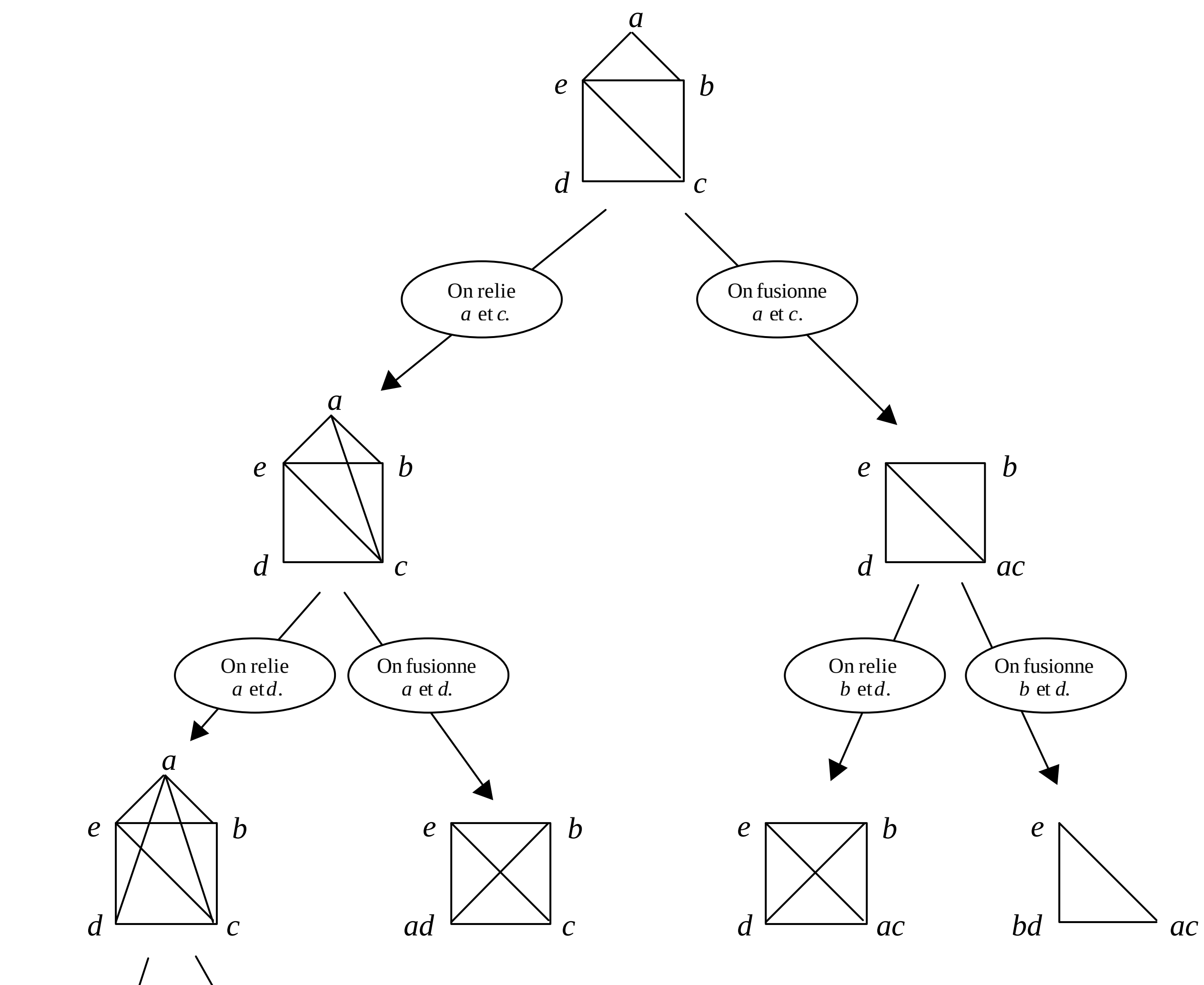
### *Méthode pour trouver toutes les colorations possibles*

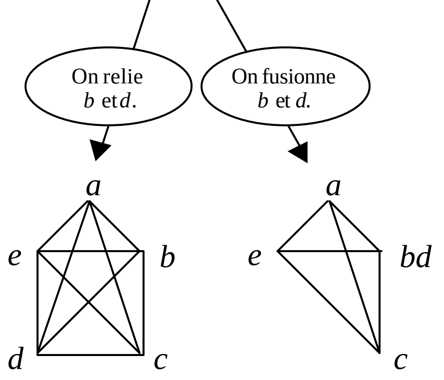
Si G = $K_n$ alors il faut $n$ couleurs.

Sinon, il existe au moins deux sommets $a$ et $b$ non reliés.

> Les colorations de G pour lesquelles les sommets $a$ et $b$ sont de même couleur sont exactement les colorations du graphe $G_1$ obtenu à partir de G en identifiant $a$ et $b$.
> Les colorations de G pour lesquelles les sommets $a$ et $b$ ont des couleurs distinctes sont exactement les colorations du graphe $G_2$ obtenu à partir de G en rajoutant l'arête $ab$. On détermine ensuite les colorations de $G_1$ et $G_2$ selon le même principe.

Exemple :



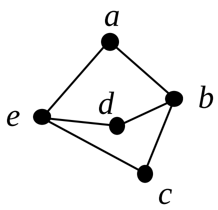

Le nombre minimal de couleur est donc 3 : on colorie $a$ et $c$ avec une couleur, $b$ et $d$ avec une seconde et $e$ avec une troisième.

On aurait pu démarrer le processus avec les sommets $a$ et $d$ et on aurait trouvé les mêmes graphes complets à l'arrivée.

### *Heuristique*

Quand les problèmes sont trop compliqués (cf. exercice 18 du programme de TES), on définit un processus qui donnera une solution (pas forcément la meilleure).
On range les sommets dans l'ordre décroissant de leurs degrés : $s_1$, $s_2$, $s_3$ … $s_n$.
On colorie ces sommets dans l'ordre précédemment défini avec pour règle de donner à chaque sommet la couleur la plus petite, en fonction des sommets voisins qui sont déjà colorés.

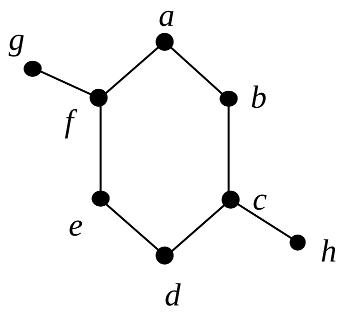
On range les sommets dans l'ordre décroissant de leurs degrés puis on attribue les couleurs :




| Degrés | 3 | 3 | 2 | 2 | 2 |
|---|---|---|---|---|---|
| Sommets | *b* | *e* | *a* | *c* | *d* |
| Couleurs | 1 | 1 | 2 | 2 | 2 |

Deux couleurs sont donc ici nécessaires (et c'est optimal).

Mais cette méthode ne donne pas forcément la solution optimale. Le graphe ci-contre peut manifestement être colorié en deux couleurs : une pour *a, c, e, g* et une autre pour *b, d, f , h*.




Si on applique la méthode précédente :

| Degrés | 3 | 3 | 2 | 2 | 2 | 2 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|
| Sommets | *f* | *c* | *a* | *b* | *e* | *d* | *g* | *h* |
| Couleurs | 1 | 1 | 2 | 3 | 2 | 3 | 2 | 2 |

Avec cette méthode, il faut trois couleurs.

### *Applications*

- Exercices du programme TES 9 à 18
- Téléphonie mobile, avec la difficulté supplémentaire que la coloration devient dynamique du fait du déplacement des téléphones en cours de communication.
- Réseaux de communication par fibre optique où il faut gérer les conflits entre les chemins.

# IV Problèmes de chemins

On dira qu'une chaîne est « eulérienne », si elle passe une fois et une seule par chaque arête.

Théorème d'Euler

Soit G un GNO possédant au moins une arête.
Ce graphe G contient une chaîne eulérienne (donc englobant toutes les arêtes de G) si et seulement si G est connexe et contient zéro ou deux sommets de degré impair.
S'il y a deux sommets $a$ et $b$ de degrés impairs, le chemin va de $a$ vers $b$. S'il n'y a pas de sommet de degré impair, il s'agit d'un cycle eulérien (retour au point de départ).

Démonstration

On suppose qu'il existe une chaîne eulérienne commençant au sommet $a$ et finissant au sommet $b$ avec éventuellement $a = b$.
G est évidemment connexe puisque deux sommets quelconques sont reliés par un morceau de la chaîne eulérienne de départ.
D'autre part, tout sommet $x$ autre que $a$ et $b$ est de degré pair (car à toute arête entrant dans $x$ correspond une arête sortant de $x$).
De plus, si $a \neq b$, $a$ est impair et $b$ aussi, donc en tout cela fait deux sommets impairs.
Par contre si $a = b$, alors ce sommet $a$ est pair ; donc il n'y a aucun sommet impair.

On suppose que G est connexe et qu'il y a 0 ou 2 sommets impairs (notés alors $a$ et $b$).

On va raisonner par récurrence sur le nombre d'arêtes de G. Le résultat est immédiat si G ne possède qu'une seule arête (G a deux sommets $a$ et $b$ de degré 1 et le chemin $ab$).
Supposons le résultat vrai pour les graphes ayant au plus $m - 1$ arêtes et considérons un graphe G à $m$ arêtes.

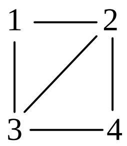
On construit dans un premier temps un chemin C, ou un cycle, qui ne sera pas nécessairement eulérien. Celui-ci sera ensuite complété.

Pour construire ce chemin, on part du sommet impair $a$ ou, si tous les sommets sont pairs, d'un sommet quelconque que l'on appellera également $a$. On progresse ensuite en choisissant à chaque étape une arête non encore utilisée. Le graphe étant fini, ce processus se termine nécessairement. Il est aisé d'observer qu'on termine nécessairement sur le sommet $b$ si $a$ est impair, ou sur le sommet $a$ si tous les sommets sont pairs (on a alors un cycle).
Le chemin ou cycle C ainsi obtenu peut, par chance, être eulérien. Dans le cas contraire, il est clair que le graphe $G - C$ (obtenu en supprimant les arêtes de C) est un graphe dont tous les sommets sont pairs.

Chaque composante connexe $CC_i$ de $G - C$ satisfait les hypothèses de récurrence et contient donc un cycle eulérien $C_i$. Le chemin C rencontre naturellement chacun de ces cycles $C_1$, $C_2$, etc. (car le graphe G est connexe).

Le chemin (ou cycle) eulérien de G s'obtient alors de la façon suivante : on parcourt C et, chaque fois que l'on rencontre pour la première fois un sommet d'un cycle $C_i$, on parcourt $C_i$ avant de poursuivre C… On « absorbe » ainsi tous les cycles $C_i$, et donc toutes les arêtes de G.

### *Matrice d’adjacence*




Le graphe G dessiné ci-contre a pour matrice d’adjacence : $M = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$.

On écrit 1 si les sommets sont en relation, 0 sinon.
La matrice M donne le nombre de chemins de longueur 1.
On s’intéresse souvent à l’existence et non pas au nombre de chemins.
Pour avoir les chemins de longueur 2, on calcule $M^2$ ; etc.

### *Plus court chemin*

Si on travaille sur des graphes valués, on recherche un coût minimum ; si on travaille sur des graphes non valués, on recherche alors un nombre d’arêtes minimum.

On trouve ce genre de problème chaque fois que l’on travaille sur un graphe dont les sommets représentent des « configurations » et les arêtes des « opérations » permettant de passer d’une configuration à une autre. C’est notamment le cas en théorie des jeux (Rubik’s cube, problèmes du style « loup, chèvre et chou », problèmes de remplissage de jarres, etc.) où l’on cherche à atteindre une configuration « gagnante » à partir d’une configuration de départ donnée.

Le problème consiste à chercher le plus court chemin entre :

1. un sommet de départ donné et un sommet d’arrivée donné ;
2. un sommet et tous les autres ;
3. *n* sommets de départ et *m* sommets d’arrivée.

En voici un algorithme de résolution (méthode de Dijsktra).

Les données sont : un graphe G, un sommet de départ *s*. On associe à chaque sommet *x* le coût du meilleur chemin connu appelé poids(*x*). On mémorise également, pour chaque sommet, le voisin par lequel on « arrive » pour réaliser le meilleur chemin connu.
Soit S l’ensemble de tous les sommets et Π l’ensemble des sommets optimaux.

Initialisation

poids($s$) $\leftarrow$ 0

poids($x$) $\leftarrow +\infty$ pour $x \neq s$

$\Pi \leftarrow \emptyset$

début

Tant que $\Pi \neq S$

choisir un sommet $x \notin \Pi$ de poids minimum

$\Pi \leftarrow \Pi \cup \{x\}$

pour tout voisin $y$ de $x$ n'appartenant pas à $\Pi$

si poids($x$) + valeur($x$ $y$) < poids($y$)

alors poids($y$) $\leftarrow$ poids($x$) + valeur($x$ $y$)

mémoriser en $y$ que l'on vient de $x$

fin si

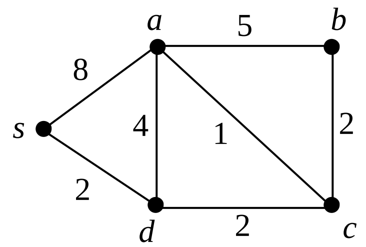
fin pour tout

fin tant que

Cet algorithme donne tous les plus courts chemins de $s$ vers tous les autres sommets.

Exemple

Voici la suite des résultats obtenus en faisant tourner cet algorithme sur le graphe ci-contre :




| sommets | $s$ | $a$ | $b$ | $c$ | $d$ | |
|---|---|---|---|---|---|---|
| début | 0 | $+\infty$ | $+\infty$ | $+\infty$ | $+\infty$ | on garde $s$ |
| étape 1 | | 8($s$) | $+\infty$ | $+\infty$ | 2($s$) | on garde $d$ |
| étape 2 | | 6($d$) | $+\infty$ | 4(d) | | on garde $c$ |
| étape 3 | | 5($c$) | 6($c$) | | | on garde $a$ |
| étape 4 | | | 6($c$) | | | on garde $b$ |

D'où les chemins :
- de $s$ vers $d$ direct de coût 2 ;
- de $s$ vers $c$ de $s$ vers $d$ puis de $d$ vers $c$ (total 4) ;
- de $s$ vers $b$ de $s$ vers $d$ puis de $d$ vers $c$ puis de $c$ vers $b$ (total 6) ;
- de $s$ vers $a$ de $s$ vers $d$ puis de $d$ vers $c$ puis de $c$ vers $a$ (total 5).

# V Problèmes d'ordonnancement

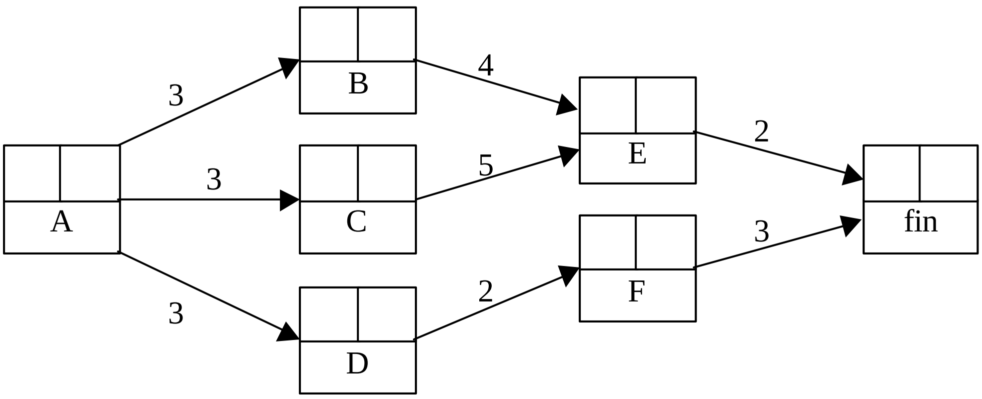
Il s'agit de savoir planifier l'exécution de tâches qui ont une certaine durée, et qui ont entre elles des relations d'antériorité (par exemple, dans les révisions qu'il faut faire avant de passer le baccalauréat, il y a des chapitres qu'il faut revoir avant d'autres).

Exemple On remplira la colonne «antériorité » avec les tâches qui doivent être exécutées avant celle considérée. On n'utilisera que les « antériorités immédiates », c'est-à-dire que si la tâche E doit être traitée après la tâche B et que la tâche B doit être traitée après la tâche A, on ne marquera pas l'antériorité A dans la ligne consacrée à E. Les antériorités non-immédiates ont été écrites entre parenthèses.

| Tâche | Durée en semaines | Antériorité |
|---|---|---|
| A | 3 | |
| B | 4 | A |
| C | 5 | A |
| D | 2 | A |
| E | 2 | (A), B, C |
| F | 3 | (A), D |

### *Méthode française MPM (Méthode Potentiel Métra)*

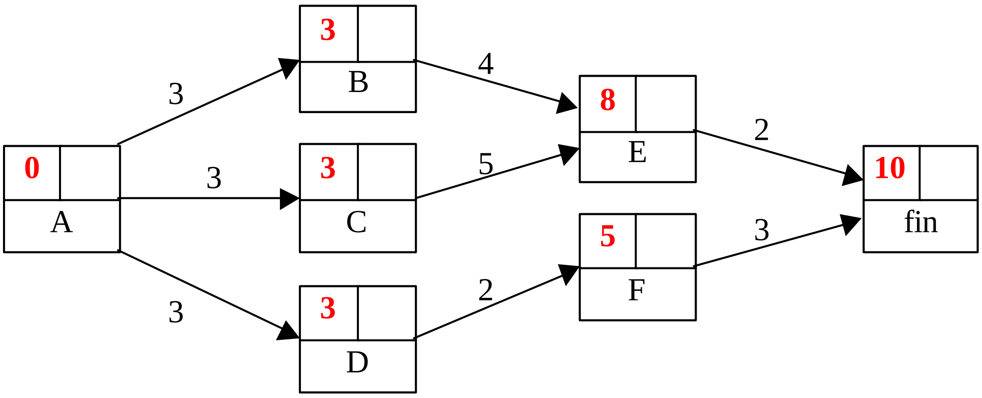
On crée un graphe orienté dont les sommets sont les tâches ; on crée une tâche fictive qui est la tâche « fin » (sous-entendu « du processus »).
Les arcs sont les relations d'antériorité immédiate ; ils sont valués par la durée de la tâche source.




Des algorithmes de parcours de graphes permettent de calculer des chemins vérifiant certaines propriétés.

### Dates « au plus tôt »

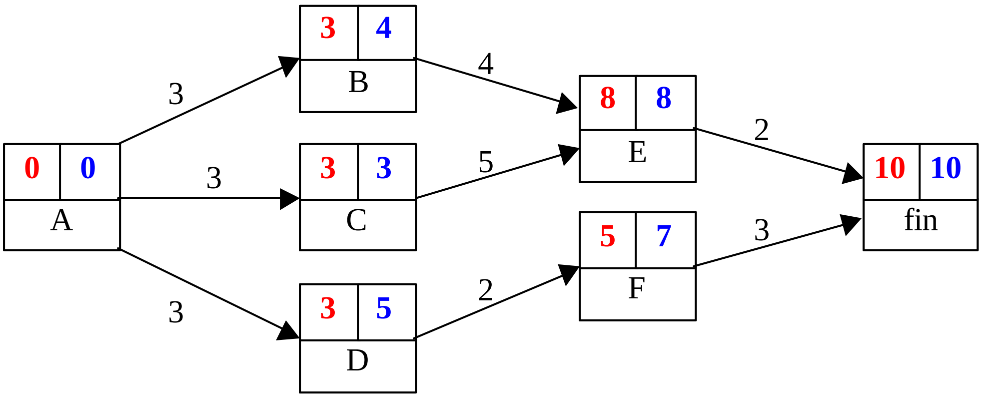
On traite les sommets par niveaux en partant du début.
Pour chaque sommet $i$ on note la date $t_i$ qui est la longueur du plus **long** chemin de la tâche initiale à la tâche $i$.




Le travail ne pourra donc pas être terminé avant 10 semaines.
La tâche E ne pourra pas commencer avant 8 semaines, la tâche F avant 5 semaines, etc.

### Dates « au plus tard »

On traite les sommets en partant de la fin (en marquant **10** pour le sommet « fin »).
Pour chaque sommet on note la date $t_i^*$ qui est la longueur du plus **court** chemin de la tâche $i$ à la tâche « fin ».




Pour effectuer l'ensemble des tâches en 10 semaines, il faudra avoir commencé la tâche E au bout de 8 semaines, commencé la tâche F au bout de 7 semaines, etc.

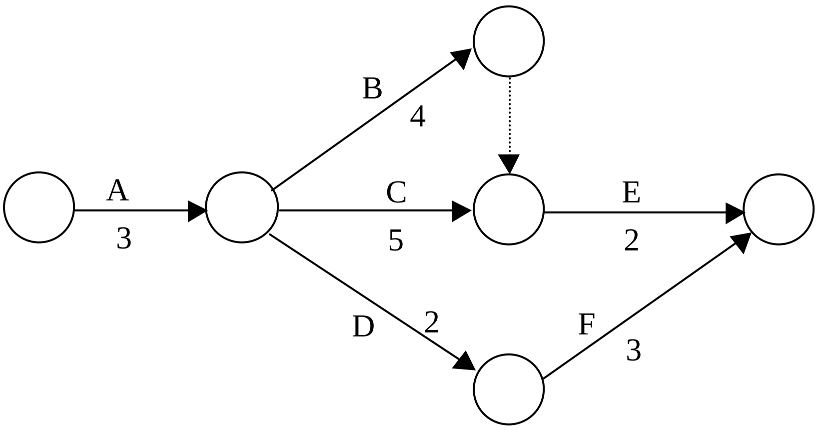
### Exploitation de ce graphe

Il y a des tâches critiques, celles pour lesquelles on a : $t_i = t_i^*$ : la tâche E devra être effectuée en 8 semaines (ni plus ni moins) pour que le processus soit achevé au bout des 10 semaines. Les tâches critiques définissent un ou plusieurs chemins critiques composés de tâches dont l'exécution ne doit connaître aucun retard pour que le projet soit achevé au plus tôt.

Par contre il y a de la latitude pour les tâches qui ne sont pas critiques : la tâche F pourra être démarrée entre la semaine 5 et la semaine 7.
De même il y a un chemin critique : A – C – E – fin (il y a toujours un chemin critique dans un graphe MPM).

### *Méthode américaine : PERT*

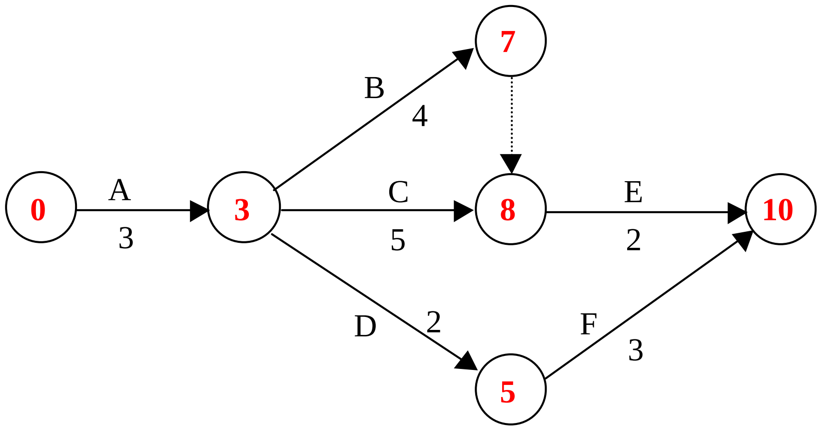
On construit un graphe dont les arcs représentent les tâches ; ils sont valués par la durée.
Intuitivement, un sommet ayant un arc entrant étiqueté X correspond à « la tâche X est terminée », un sommet ayant deux arcs entrants X et Y à « les tâches X et Y sont terminées ». Selon les relations d'antériorité immédiate, on peut être amené à rajouter des arcs fictifs de coût 0 (voir exemple ci-dessous).

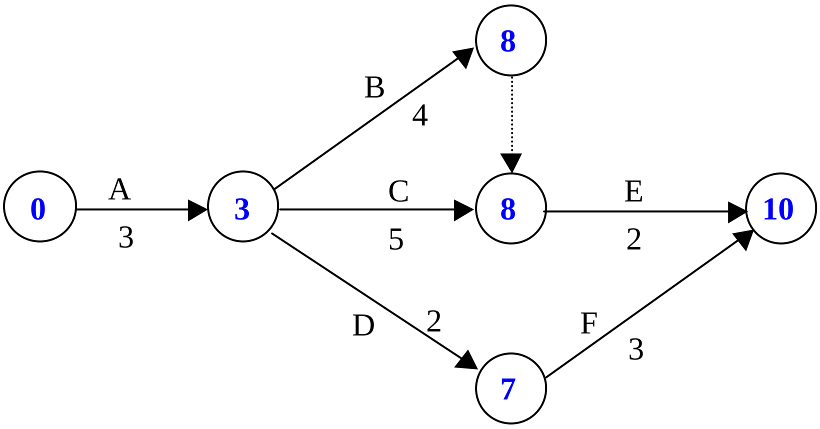



On est obligé de rajouter un arc fictif de coût 0 (en pointillés) ), car la tâche E ne peut démarrer que lorsque B et C sont terminées.

Au niveau de chaque sommet, on calcule la longueur du plus **long** chemin du sommet source à ce sommet.

On a donc les dates de fin « au plus tôt ».




À l'envers, on calcule la longueur du chemin le plus **court** conduisant au sommet « fin ».

On a donc les dates « au plus tard ».

Remarque : une tâche critique est représentée par un arc où les dates de début et de fin « au plus tôt » en rouge sont égales aux dates de début et de fin « au plus tard » en bleu ; ici, les tâches critiques sont A, C et E.

# VI Automates

Quelques définitions et notations relatives aux langages formels.

alphabet A — Ensemble fini d'éléments appelés « lettres ».

mot — Suite de lettres. Exemple u = aba où $a \in A$ et $b \in A$.
$A^*$ désigne l'ensemble de tous les mots que l'on peut former avec A (cet ensemble est infini.
|u| désigne la longueur du mot u.
$|u|_a$ désigne le nombre de lettres a dans le mot u.
Le mot vide est noté $\varepsilon$.

langage L — Sous-ensemble de $A^*$.

opérations — Sur les mots, on définit la concaténation ; si u = aba et v = aabb, on définit le mot uv = abaaabb.
Sur les langages, on définit les opérations ensemblistes :
- intersection $\cap$
- réunion notée ici + plutôt que $\cup$
- produit $L_1L_2 = \{ u = u_1u_2 \text{ où } u_1 \in L_1 \text{ et } u_2 \in L_2 \}$
- étoile * $L^* = \{\varepsilon\} + L + L^2 + L^3 + \ldots$
  $L^+ = L^* - \{\varepsilon\} = L + L^2 + L^3 + \ldots = LL^*$

Remarque : l'abus de notation a au lieu de {a} est fréquent.

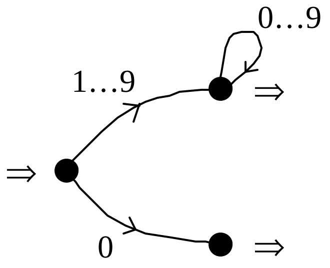
langage rationnel — Langage défini par une expression rationnelle qui est une expression finie utilisant les lettres de l'alphabet et les opérations union, produit et étoile.

Exemples

1) A = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, l'ensemble des multiples de 5 est l'ensemble des mots dont la dernière lettre est 0 ou 5 ; il s'écrit : $(0+1+2+\ldots+9)^*(0+5)$.

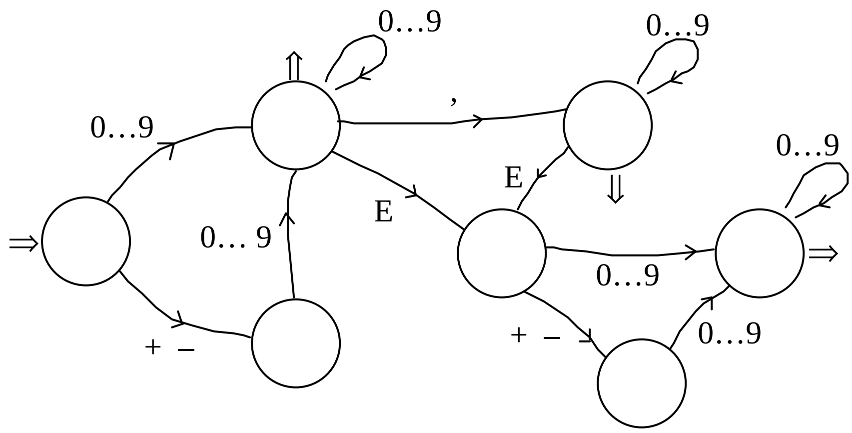
2) A = {a, b}
   - mots sans a : $b^*$ ;
   - mots qui ne contiennent pas ab : $b^*a^*$ ;
   - mots ne contenant pas aba : $b^*(a^*bb^*)^*a^*(\varepsilon+b+bb)$.

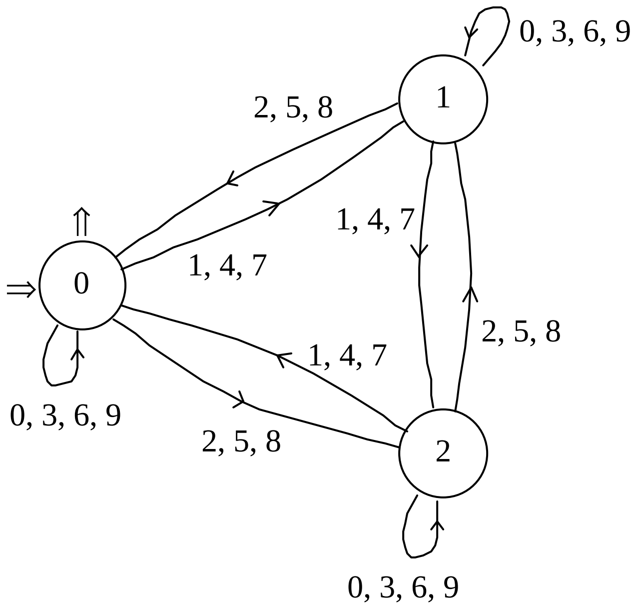
On s'intéresse aux langages reconnaissables, c'est-à-dire tels qu'il existe un automate qui puisse en reconnaître les mots. Un automate utilise un graphe dont les sommets sont des états et à chaque arc est associée la reconnaissance d'une ou plusieurs lettres. Un tel automate est obligatoirement fini ; il comporte une entrée et une ou plusieurs sorties. De plus l'automate est déterministe c'est-à-dire que, pour chaque mot entré, il n'existe qu'un parcours possible du graphe.




Exemple d'automate qui reconnaît tous les entiers dont l'écriture est normalisée (ne commençant pas par un 0).

Exemple d'automate qui reconnaît une entrée numérique dans un tableur (par exemple : 12,3 ou 08 ou –15 ou 5E12 ou 14E–3).

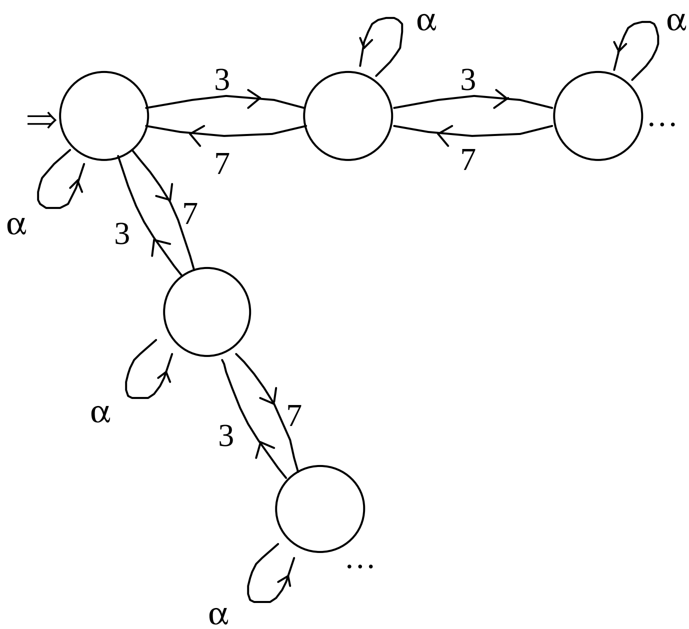



Exemple d'automate reconnaissant tous les multiples de 3.

Essayons de construire un automate qui reconnaisse des nombres ayant autant de 3 que de 7.

On a noté α tout chiffre différent de 3 et de 7.




Le nombre d'états est infini donc il n'existe pas d'automate pour reconnaître un nombre ayant autant de 3 que de 7.

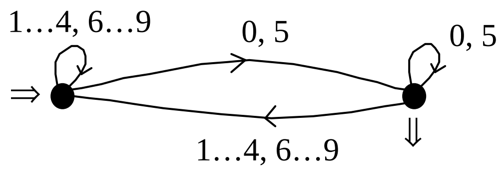
Donnons une définition formelle d'un automate.

Un automate $\mathcal{A}$ est défini par
- un alphabet A fini ;
- un ensemble fini Q d'états ;
- un état initial $q_0$ ;
- un sous-ensemble F de Q représentant les états terminaux ;
- une fonction de transition $\delta : Q \times A \rightarrow Q$

On le note : $\mathcal{A} = < A , Q , q_0 , F , \delta >$

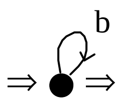
Autres exemples d'automates.

Multiples de 5.

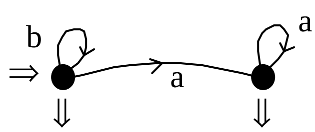



Dans un langage ne contenant que les deux lettres a et b.

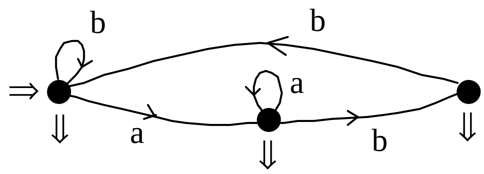
Mots sans la lettre a



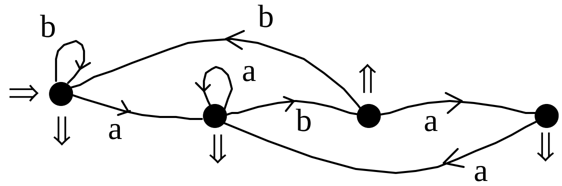

Mots sans ab




Mots sans aba




Mots sans abab

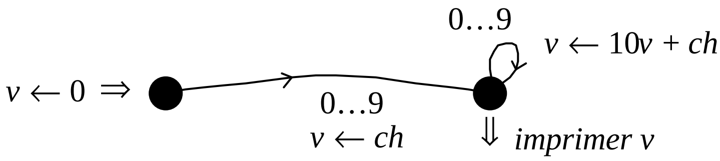



Théorème : il y a équivalence entre langages rationnels et langages reconnaissables.

### *Automates avec actions*

L'idée est ici de pouvoir associer une action (par exemple exécuter un « petit » algorithme) à chaque « opération » de l'automate : entrée, transition d'un état à l'autre, sortie. Chaque mot traité par l'automate peut ainsi produire un résultat…

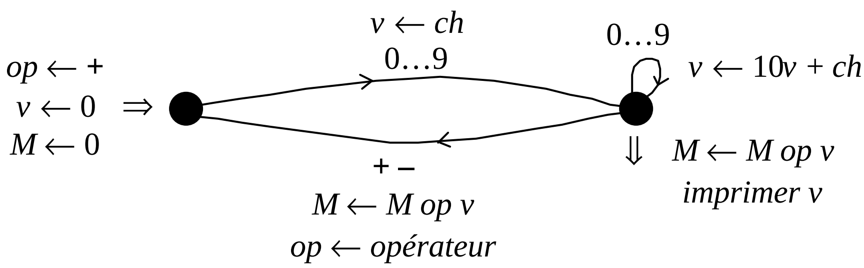
Automate de reconnaissance d'entiers.




*v* désigne la valeur de l'entier lu

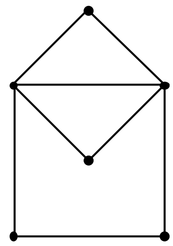
initialisation
    $v \leftarrow 0$
entrée du premier chiffre *ch*
    $v \leftarrow ch$
tant qu'un chiffre *ch* est entré
    $v \leftarrow 10v + ch$
fin du tant que
    impression de *v*

Reconnaissance et évaluation d'expressions arithmétiques sans parenthèses ni virgule comme par exemple : 183 + 51 – 12



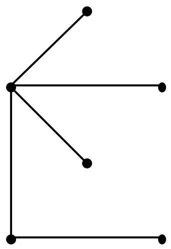
# VII Arbres couvrants

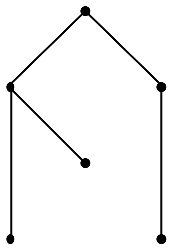
On considère un graphe non orienté connexe quelconque :



On appellera « arbre couvrant » de ce graphe un arbre qui aura les mêmes $n$ sommets mais qui n'aura que $n - 1$ arêtes ; on cherchera à retirer au graphe le plus possible d'arêtes en faisant en sorte que le graphe obtenu reste encore connexe.

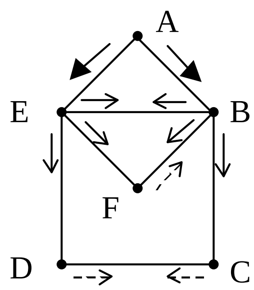
En voici deux exemples :





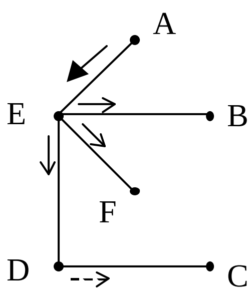
Un arbre est un graphe connexe dans lequel il n'y a pas de cycle. On peut dire aussi qu'un arbre est un graphe connexe ayant un sommet de plus que d'arêtes.

Imaginons que le graphe représenté plus haut soit celui d'un réseau et que A veuille envoyer un message à tous les autres membres de ce réseau.
Les machines ne connaissent pas la structure du réseau.
On applique les règles suivantes : A envoie un message à tous ses voisins et, lorsqu'un sommet reçoit le message pour la première fois, il le diffuse à tous ses autres voisins (et ne le renvoie pas à celui qui lui a envoyé).
Il y a donc 3 étapes et 11 messages envoyés.

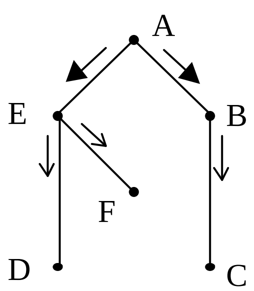



Si on fait la même chose avec un arbre couvrant, il n'y a plus de doublon et il n'y a plus qu'un message qui circule par arête. Le nombre d'étapes dépend de l'arbre choisi :

5 messages en 3 étapes.




5 messages en 2 étapes.




Le nombre minimum d'étapes sera obtenu avec les arbres construits en largeur (voir ci-dessous).

La diffusion d'informations par arbre couvrant est donc plus performante et moins redondante que par graphe.
Une question se pose donc : comment construire un arbre couvrant à partir d'un graphe connexe ?

Construction en largeur (par niveaux) :

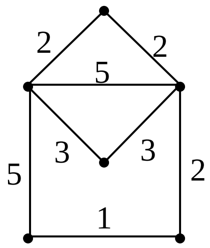
on prend les voisins de niveau 1 ;
les voisins non pris donnent le niveau 2 ;
etc.

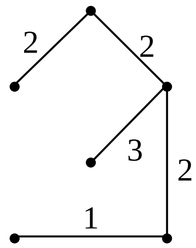
Construction en profondeur (peu intéressante pour les réseaux)

on avance tant qu'on peut (si on ne peut plus avancer, on recule comme dans l'exploration d'un labyrinthe)

ce qui donne en partant de A : A B F E D C
ou en partant de E : E B A (B) F (E) (B) C D.

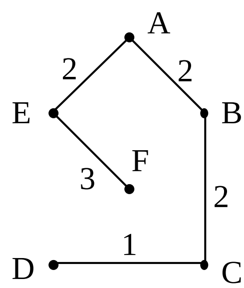
Si le graphe est valué, c'est-à-dire s'il y a un coût sur chaque arête, on cherchera un arbre couvrant de coût minimal.




Arbre de coût 10.

Un algorithme de recherche de l'arbre couvrant de coût minimal (PRIM) est : tant qu'on n'a pas pris $n - 1$ arêtes, on choisit une arête autorisée de poids minimal (autorisée signifiant : en faisant en sorte qu'il n'y ait pas de cycle). C'est ainsi qu'a été obtenu l'arbre de coût 10 ci-dessus.
Un autre algorithme (KRUSKAL) permet de rechercher un arbre de coût minimum.




En appelant X un ensemble de sommets avec $\varnothing \leftarrow X$ initialement, on choisit un sommet x que l'on place dans X : $X \leftarrow \{x\}$.
Tant qu'on n'a pas tous les sommets, on choisit une arête ab sortant de X de coût minimum avec $a \in X$ et $b \notin X$.
On rajoute alors b dans X : $X \leftarrow X \cup \{b\}$.

Sur l'exemple X={A}, puis X={A ; E}, puis X={A ; E ; B}, puis X={A ; E ; B ; C}, puis X={A ; E ;B ; C ; D} et enfin X={A ; E ; B ; C ; D ; E ; F}

## VIII Graphes planaires

Un graphe est « planaire » s'il peut être dessiné sur un plan ou sur une sphère sans croisement d'arêtes.

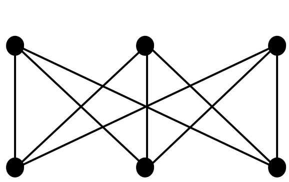
Soit F le nombre de faces d'un graphe (on compte la face extérieure infinie), A son nombre d'arêtes et S son nombre de sommets.

Théorème (Formule d'Euler)

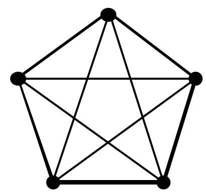
Si un graphe est planaire connexe, alors : $F - A + S = 2$.

On démontre cette formule en effectuant une récurrence sur $A - S$.

1) La formule est vraie pour $A - S = -1$ car, dans ce cas, le graphe est un arbre donc il n'a qu'une seule face, donc $F = 1$, donc $F - A + S = 1 - (-1) = 2$.
2) On suppose la formule vraie jusqu'à $A - S = k$.
Ce graphe connexe contient un cycle G. Si on retire une arête e à ce cycle, on peut alors appliquer au graphe $G - \{e\}$ la formule de récurrence. Or, si $G - \{e\}$ possède f faces, a arêtes et s sommets avec $f - a + s = 2$, alors G possède $f + 1$ faces, $a + 1$ arêtes et s sommets, donc on obtient bien ce qu'on voulait : $(f + 1) - (a + 1) + s = 2$.



$K_{3,3}$



$K_5$

Théorème de Kuratowski : Un graphe G est planaire si et seulement s'il ne contient pas de sous-graphe partiel « de type » $K_{3,3}$ ou $K_5$ (c'est-à-dire de sous-graphe que l'on peut obtenir en remplaçant les arêtes de $K_{3,3}$ ou $K_5$ par des chemins disjoints).
(Démonstration difficile)

Remarque : Dans un graphe simple, chaque face est incidente à trois arêtes au moins et, d'autre part, chaque arête délimite au moins deux faces, donc $F \le \frac{2A}{3}$.

Remarque : L'inégalité précédente peut servir à prouver que $K_5$ , qui est connexe, n'est pas planaire. En effet, s'il l'était, avec $A = 10$ et $S = 5$, la formule d'Euler $F - A + S = 2$ entraînerait $F = 7$ ; mais alors, l'inégalité $F \le \frac{2A}{3}$ s'écrirait $7 \le \frac{20}{3}$, ce qui est absurde.

Lemme : Dans tout graphe planaire, on peut trouver un sommet de degré inférieur ou égal à 5.

En effet, supposons par l'absurde que tous les sommets sont de degré $\ge 6$ ; comme une arête est incidente à 2 sommets et qu'ici chaque sommet est incident à au moins 6 arêtes, on a donc $S \le \frac{2A}{6}$. Comme de plus $F \le \frac{2A}{3}$, on déduit $F - A + S \le \frac{2A}{3} - A + \frac{2A}{6}$ soit $F - A + S \le 0$ ; ceci contredit la formule d'Euler.

### *Coloration de graphes planaires*

Avant de démontrer le théorème des 4 couleurs (en utilisant l'informatique), on avait démontré les théorèmes suivants :

Théorème des 6 couleurs : tout graphe planaire est coloriable avec au maximum 6 couleurs.

> Soit G le plus petit contre-exemple ("plus petit" en nombre de sommets).
> Soit $x$ un sommet de G, tel que le degré ($x$) soit inférieur ou égal à 5.
> G – {$x$} est coloriable avec 6 couleurs (puisque G est le plus petit contre-exemple),
> Or $x$, qui est relié à au plus cinq autres sommets a donc au plus cinq couleurs interdites, donc G est finalement coloriable avec 6 couleurs.

Théorème des 5 couleurs : tout graphe planaire est coloriable avec au maximum 5 couleurs.

La démonstration se fait par récurrence. Le résultat est naturellement vrai pour les graphes ayant au plus cinq sommets…
Soit G un graphe ayant au moins six sommets. Nous avons vu précédemment que tout graphe planaire contient un sommet de degré au plus cinq. Soit x un tel sommet dans G.

- si x est de degré au plus quatre, alors toute coloration de G-x (le graphe obtenu en supprimant le sommet x dans G) peut être étendue à une coloration de G (car nous disposons de cinq couleurs et au plus quatre sont utilisées pour colorier les voisins de x).
- si x est de degré cinq, toute coloration de G-x n'utilisant qu'au plus quatre couleurs pour les voisins de x peut être de la même façon étendue à une coloration de G.
- il reste donc à traiter le cas où x est de degré cinq et, dans la coloration de G-x, cinq couleurs sont utilisées pour les voisins de x. Soient $y_1,\ldots,y_5$ les voisins de x (pris dans l'ordre trigonométrique) et $c_1,\ldots,c_5$ leurs couleurs respectives.

Il est maintenant nécessaire de modifier la couleur de l'un des voisins de x pour « libérer » une couleur pour x… Si l'on change la couleur de $y_1$ et $c_3$, cela peut poser un problème si $y_1$ avait un voisin colorié $c_3$… Il faudrait alors également modifier la couleur de ce voisin et ainsi de suite…

Plus formellement, soit $H_{1,3}$ le sous graphe de G-x composé des sommets coloriés $c_1$ ou $c_3$. Si l'on échange les couleurs des sommets de la composante connexe de $H_{1,3}$ qui contient $y_1$, on a toujours une coloration valide !…
Mais il reste un problème : il est possible que $y_3$ appartienne à la même composante et soit donc recolorié en $c_1$… Dans ce cas, nous n'avons rien gagné car cinq couleurs sont toujours utilisées par les voisins de x…
L'idée « magique » est alors la suivante : on peut essayer, soit de recolorier $y_1$ en $c_3$, soit de recolorier $y_2$ en $c_4$… or, il est impossible qu'à la fois nous ayons $y_1$ et $y_3$ dans la même composante de $H_{1,3}$ et $y_2$ et $y_4$ dans la même composante de $H_{2,4}$ ! (en effet, si c'était le cas, nous aurions deux chemins ayant des couleurs distinctes, l'un reliant $y_1$ à $y_3$, l'autre $y_2$ à $y_4$, qui doivent nécessairement se couper en un sommet… qui aurait ainsi deux couleurs !).
Donc, il est toujours possible de recolorier $y_1$, ou $y_2$, de façon à n'avoir plus que quatre couleurs utilisées par les voisins de x. Cette coloration peut alors être étendue à une coloration de G…

On utilise par exemple les graphes planaires dans la confection de circuits imprimés dans lesquels on ne peut pas s'autoriser trop de croisements.