Théorie des graphes

Alexandre Duret-Lutz
adl@lrde.epita.fr

17 mars 2012

Huitième partie VIII

Graphes cordaux

Qui a tué le Duc de Densmore ? (1/2)

Il s'agit du titre d'une nouvelle policière[1] écrite par Claude Berge[2]
L'intrigue est la suivante :
Le Duc de Densmore est retrouvé carbonisé après l'explosion de son vieux château de l'île privée de White. L'enquête indique que l'explosion est due à une bombe placée dans la cave de château.
Dans la période qui précéda l'explosion, le Duc avait invité ses 8 ex-femmes. Le capitaine du bateau qui faisait la navette entre l'île et le continent se souvient d'avoir transporté chacune des 8 femmes pour un aller-retour, mais sans se souvenir des dates et heures.

---

1. Bibliothèque Oulipienne n°67, 1994, Réédition Castor Astral, 2000.
2. C. Berge (1926–2002) est un mathématicien français considéré comme l'un des fondateurs de la théorie des graphes telle que nous la connaissons. Son livre *Théorie des Graphes et ses Applications* (1954) a été écrit pour unifier une théorie qui existait uniquement de façon disparate à l'étranger.

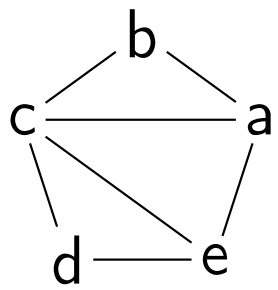
Qui a tué le Duc de Densmore ? (2/2)

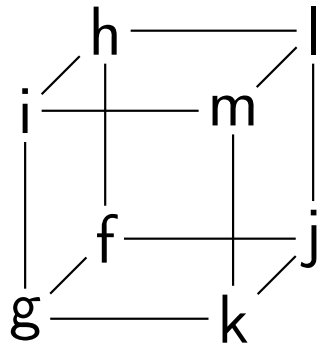
Les femmes ne se souviennent pas non plus des dates et heures, mais se rappellent s'être croisées. Ainsi :

- Anne a rencontré Félicie, Cynthia, Georgia, Emilie et Betty
- Betty a rencontré Cynthia, Anne et Hélène
- Cynthia a rencontré Anne, Emilie, Diane, Betty et Hélène
- Diane a rencontré Cynthia et Emilie
- Emilie a rencontré Félicie, Cynthia, Diane et Anne
- Félicie a rencontré Emilie et Anne
- Georgia a rencontré Anne et Hélène
- Hélène a rencontré Cynthia, Georgia et Betty

On sait d'autre part que l'une de ces femmes était désavantagée par le testament. Ce dernier a malheureusement disparu dans l'incendie.

Graphe cordal

$G_1 =$ 

$G_2 =$ 

Un cycle élémentaire est un cycle qui ne passe pas plusieurs fois par le même sommet.

(a, b, c, d, e) est un cycle élémentaire, mais pas (b, c, d, e, c, a).

Une corde d'un cycle élémentaire est une arête qui relie deux sommets non consécutifs du cycle.

(a, c) est une corde de (a, b, c, e) ; (c, e) une corde de (a, c, d, e) ; toutes deux sont aussi des cordes de (a, b, c, d, e).

Un graphe est cordal (ou *triangulé*) si tout cycle de longueur ≥ 4 possède une corde.

G_1 est cordal, mais pas G_2. Les graphes complets (K_n) et les arbres sont des graphes cordaux.

Triangulation d'un graphe (1/2)

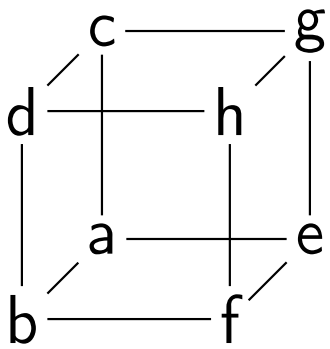
Il s'agit d'ajouter des cordes à un graphe pour le rendre cordal.
Approche par élimination sur un graphe $G = (V, E)$.
Pour i allant de 1 à $|V|$:

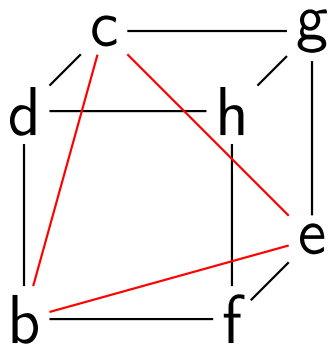
- Choisir un sommet x_i de G
- Considérer tous les voisins $N(x_i)$ de x_i, et ajouter les arcs $E_i = N(x_i) \times N(x_i)$.
- Supprimer le sommet x_i (et les arcs incidents) de G .

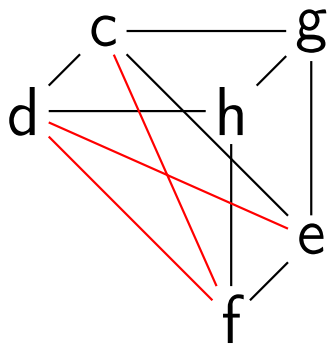
Le graphe $G' = (V, E \cup E_1 \cup E_2 \cup \ldots \cup E_{|V|})$ est cordal.

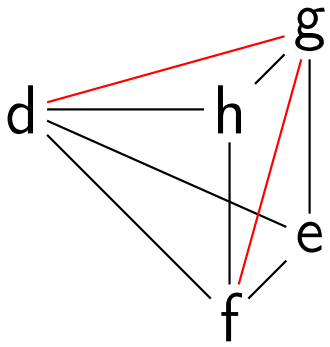
Note : le graphe obtenu diffère selon l'ordre d'élimination des états.

Triangulation d'un graphe (2/2)

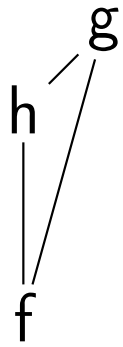
$G =$ 

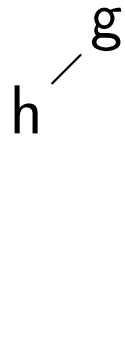
$x_1 = a$ 

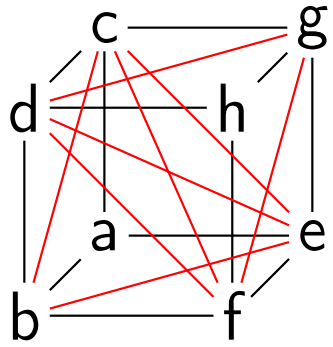
$x_2 = b$ 

$x_3 = c$ 

$x_4 = d$ 

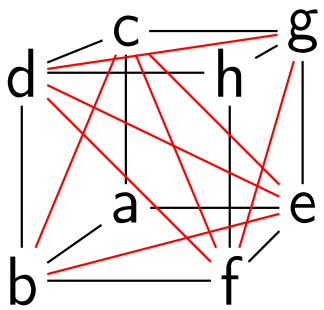
$x_5 = e$ 

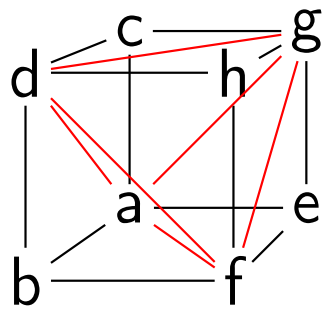
$x_6 = f$ 

$x_7 = g \ldots$ $x_8 = h \ldots$ $G' =$ 

Triangulation minimale

Soit $G = (V, E)$ un graphe, et $G' = (V, E \cup F)$ une triangulation de G. (F représente les arcs ajoutés pour triangulariser le graphe.) G' est une **triangulation minimale** de G si il n'existe pas d'autre triangulation $(V, E \cup F')$ avec $|F'| < |F|$.

$G' =$ 

$G'' =$ 

G' montre une triangulation de 8 arcs. Elle n'est pas minimale car on peut retirer l'arc (d, e) (ou (c, f) mais pas les deux car le cycle (c, d, f, e) serait sans corde). Sans cet arc, la triangulation ajoute 7 arcs.

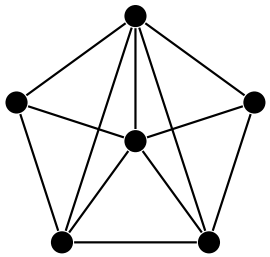
G'' montre une triangulation de 6 arcs, qui est minimale. (Il faut ajouter au moins un arc par face du cube.)

Trouver une triangulation minimale est un problème NP-complet.

Pause exercices

Exercice 1 Un arbre est un graphe cordal. Appliquer l'algorithme d'élimination présenté précédemment risque de rajouter des arêtes superflues. Pouvez-vous trouver un ordre d'élimination des sommets qui n'ajoute aucune arête ?

Exercice 2 Le graphe suivant est cordal. Pouvez-vous trouver un ordre d'élimination qui n'ajoute aucune arête ?



(Indice : vous avez 2 chances sur 6 de bien commencer.)

L'objectif de la suite et de caractériser les sommets que l'on peut retirer ainsi (les sommets **simpliciaux**) pour arriver à un algorithme de reconnaissance de graphes cordaux.

Stable et clique

Nous avons déjà parlé de stable en définissant le noyaux d'un graphe (pour chercher une stratégie gagnante dans un jeu) ou le problème de coloration des sommets d'un graphe.

Un stable dans un graphe $G = (V, E)$ est un sous-ensemble de sommets $S \subseteq V$ deux à deux non adjacents (i.e., $(S \times S) \cap E = \emptyset$).

Une clique est un sous-ensemble S des sommets d'un graphe tels que tous les sommets de S soit adjacents deux à deux. (Le graphe $G[S]$ induit par ce sous-ensemble est complet). On parle de n-clique lorsque la clique possède n sommets.

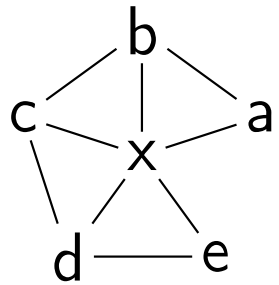
Le complémentaire d'un graphe $G = (V, E)$ est le graphe $\overline{G} = (V, (V \times V) \setminus E)$.

Une clique dans G est un stable dans $\overline{G}$ et vice-versa.

Séparateur minimal

Un séparateur d'un graphe $G = (V, E)$ est un sous-ensemble $S \subseteq V$ tel qu'il existe deux sommets a et b non adjacents dans G et non connectés dans $G[V \setminus S]$. On parle aussi de (a, b)-séparateur.

Un séparateur minimal est un séparateur minimal si aucune de ses parties n'est un séparateur (note : différents séparateurs minimaux peuvent avoir des tailles différentes).



$\{x, b\}$, $\{x, c\}$, et $\{x, d\}$ sont des (a, e)-séparateurs minimaux.
$\{x, b, c\}$ est un (a, e)-séparateur non minimal.
$\{b, c, d\}$ n'est pas un séparateur.

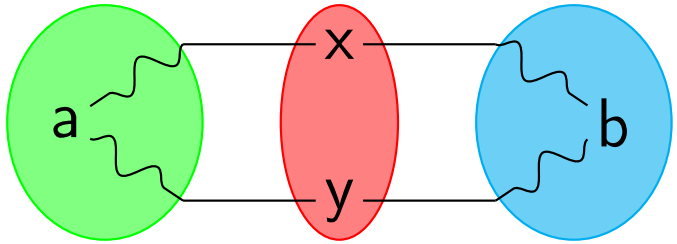
Séparateur minimal et graphe cordal (1/2)

Proposition Si G est cordal, tout séparateur minimal est une clique.

Démonstration Soit S un (a, b)-séparateur minimal de G.

Tout sommet de S est sur un chemin qui connecte a à b, sinon on pourrait le retirer de S et obtenir un séparateur plus petit.

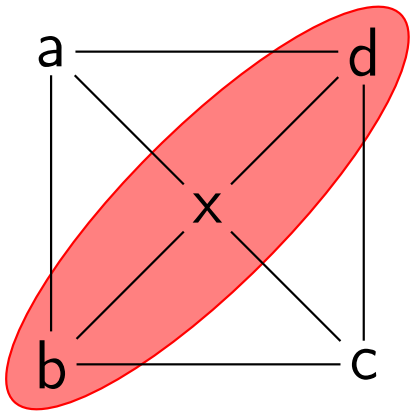
Supposons par l'absurde que S possède deux sommets x et y non adjacents. Prenons le plus petit cycle qui passe par a, x, b, y.



Ce cycle est au moins de longueur 4. Comme le graphe est cordal et que les sommets de part et d'autre du séparateur ne sont pas adjacents, le cycle admet nécessairement la corde (x, y) : cela contredit notre hypothèse.

Séparateur minimal et graphe cordal (2/2)

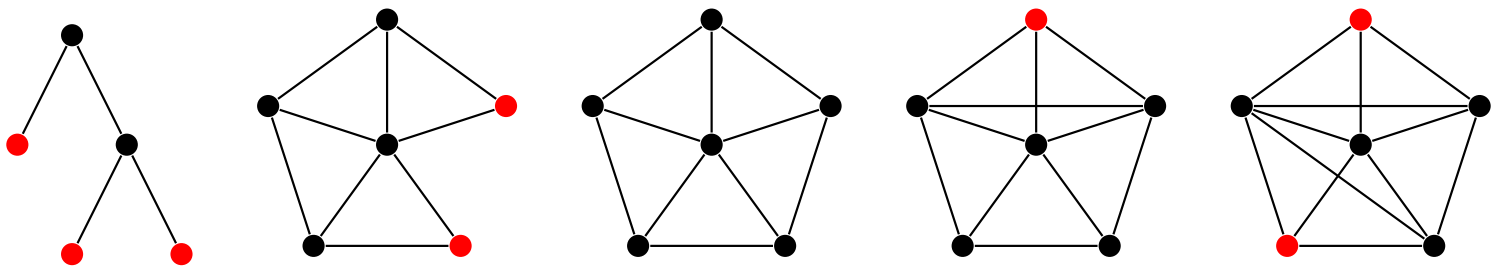
Application $\{b, x, d\}$ est un séparateur minimal du graphe suivant, mais il ne forme pas une clique : le graphe n'est donc pas cordal.



Sommet simplicial

Un sommet simplicial est un sommet dont les voisins sont tous adjacents deux à deux. Autrement dit $N(x)$ est une clique, ou encore $G[N(x)]$ est complet.

- Si x est simplicial, $N(x)$ est une clique (par définition), et $\{x\} \cup N(x)$ est aussi une clique (puisque x est adjacent à tous ses voisins).
- Il s'agit d'une généralisation de la notion de feuille dans un arbre : une feuille est un sommet simplicial.



Lesquels de ces graphes sont cordaux ?

Sommet simplicial et graphe cordal

Proposition Si un graphe G (non vide) est cordal, il possède un sommet simplicial. De plus, si G n'est pas complet il possède deux sommets simpliciaux non adjacents.

Démonstration Par récurrence sur la taille du graphe. Si G est complet, tous les sommets sont simpliciaux. Éliminons ce cas.

Soient deux sommets a et b non adjacents, et soit S un (a, b)-séparateur minimal : notons $G[A]$ et $G[B]$ les graphes induits par les composantes de $G \setminus S$ qui contiennent $a \in A$ et $b \in B$.

Montrons que A possède un sommet simplicial :

- soit $G[A \cup S]$ est complet, a est simplicial dans $G[A \cup S]$
- sinon par récurrence il possède deux sommets simpliciaux non adjacents, et l'un d'entre eux doit être dans A.

Dans les deux cas, tous les voisins du sommet trouvé sont dans $A \cup S$ donc c'est aussi un sommet simplicial de G.

On montre de même que B possède un simplicial.

Ordre d'élimination simplicial

Un ordre d'élimination simplicial est un ordre $(x_1, x_2, \ldots, x_{|V|})$ sur les sommets de $G = (V, E)$ tel que x_i est un sommet simplicial de $G[\{x_i, x_{i+1}, \ldots, x_{|V|}\}]$. Autrement dit, à chaque fois qu'on retire un sommet de G son voisinage forme une clique.

Proposition Un graphe G est cordal si et seulement si il possède un ordre d'élimination simplicial.

Démonstration En appliquant récursivement la proposition précédente, on montre que si G est cordal, on trouve un sommet simplicial. Après suppression, le graphe reste cordal, et on recommence.

Dans l'autre sens, si un graphe n'est pas cordal, c'est-à-dire qu'il existe un cycle sans corde de plus de trois sommets, aucun de ces sommets ne pourra être éliminé.

LexBFS : Parcours en largeur lexicographique (1/2)

Algorithme LexBFS(G,s)

Entrée : $G = (V, E)$ et un sommet source $s \in V$.

Sortie : $\sigma = [x_1, x_2, \ldots, x_{|V|}]$ un ordre total des sommets accessibles de s.

$\forall x \in V$, $\mathrm{label}[x] \leftarrow []$

$\mathrm{label}[s] \leftarrow [|V|]$

Pour i de $|V|$ à 1 :

    Choisir un sommet x d'étiquette lexicographique maximale

    $\sigma[i] \leftarrow x$

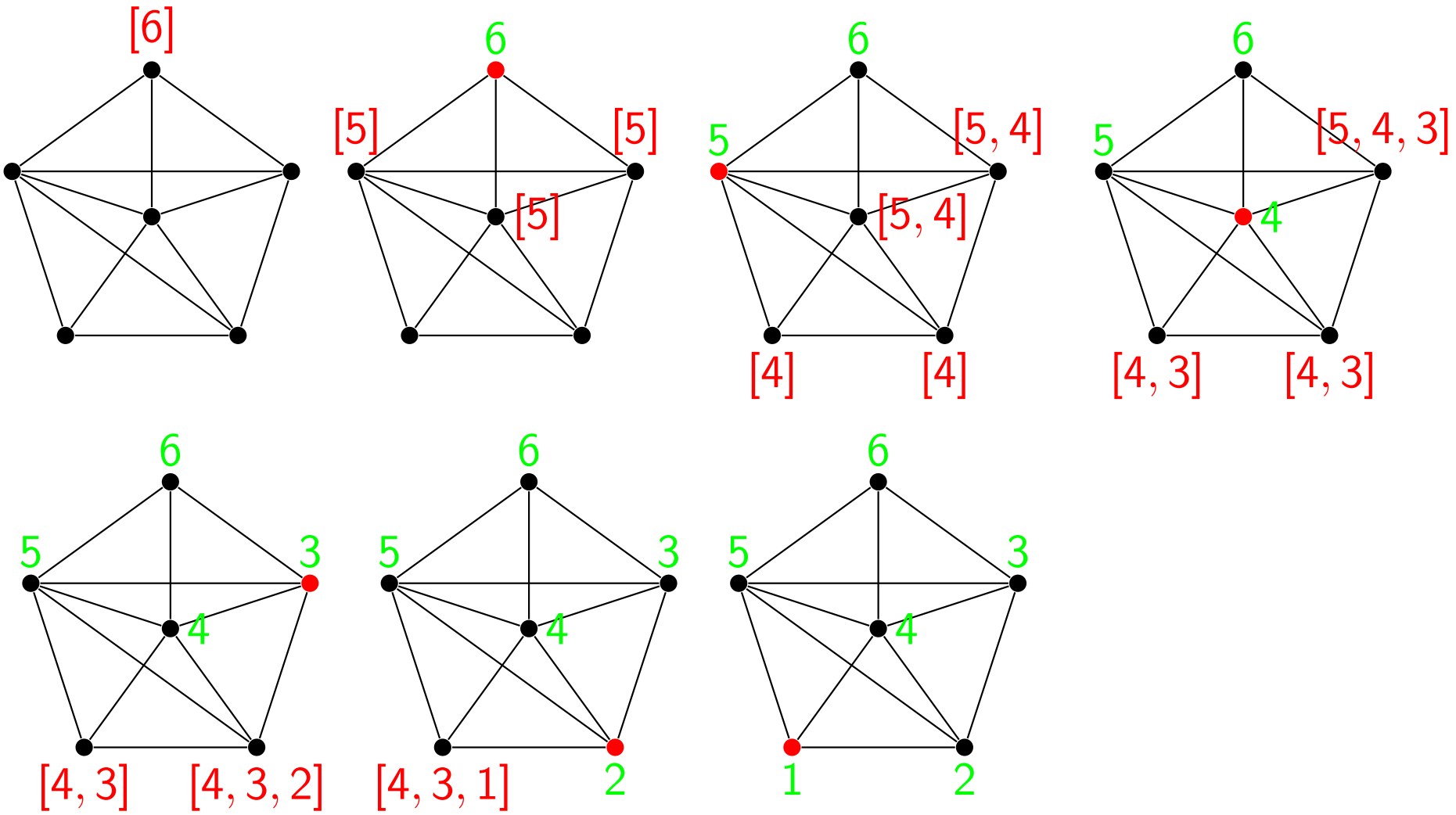
    Pour chaque voisin $y \in N(x)$:

        Si $y \notin \sigma$:

            Concaténer $i - 1$ à la fin de $\mathrm{label}[s]$

Propriété Si G est cordal, le dernier sommet visité par LexBFS est simplicial. (Quelle que soit la source choisie.) L'ordre σ est donc un ordre d'élimination simplicial.

LexBFS : Parcours en largeur lexicographique (2/2)



Sur ce graphe cordal on peut vérifier que l'ordre construit par LexBFS est bien un ordre d'élimination simplicial.

Implémentation Naïve de LexBFS

On maintient une file de priorité des sommets (ordonnée suivant l'ordre lexicographique de leurs étiquettes) avec un tas :

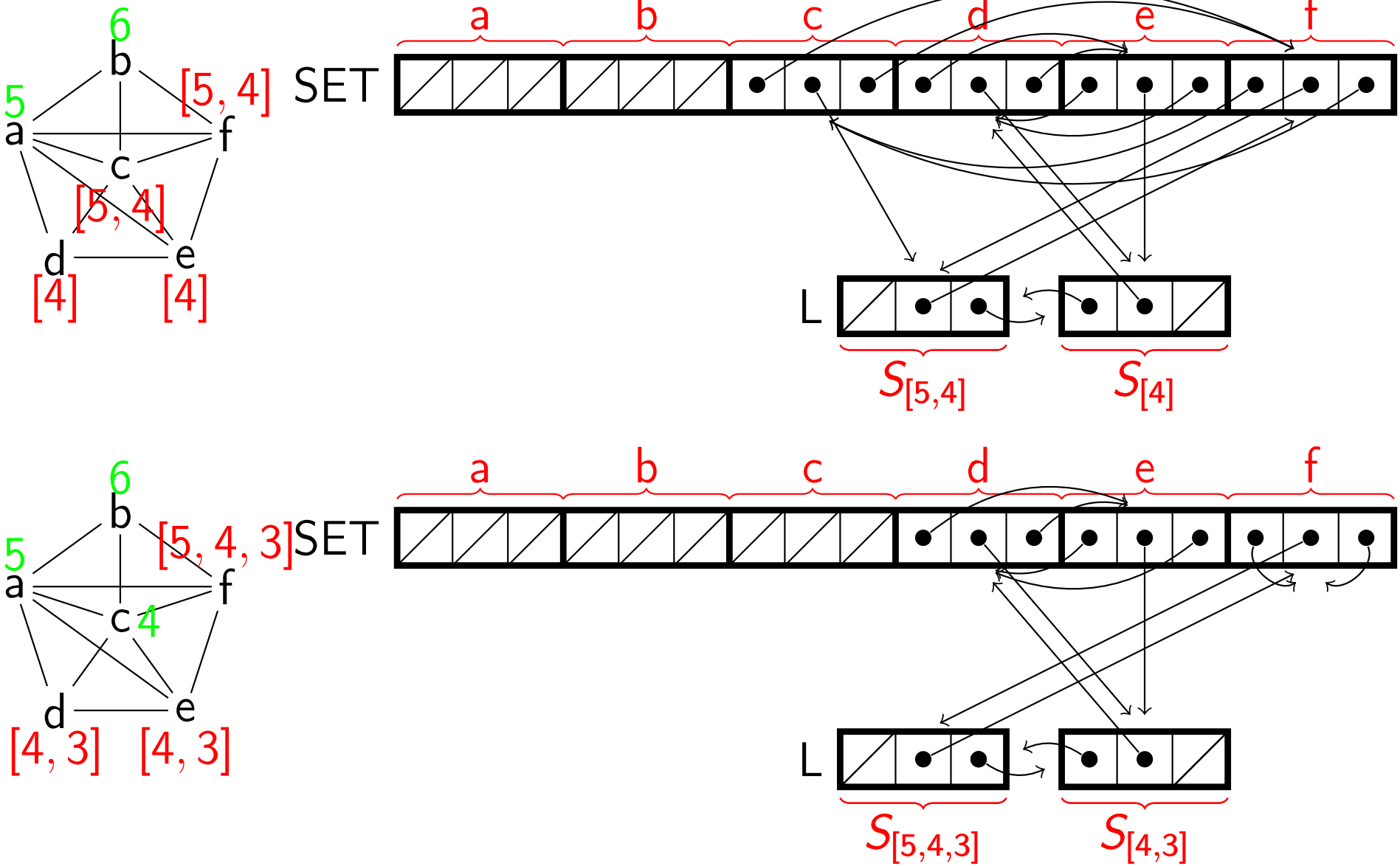
- Retirer le sommet maximal coûte $\mathrm{O}(\log n)$ où n est la taille du tas. On fait cette opération $|V|$ fois pour des tas de plus en plus petits. Pour tous ces retraits on fait donc $\mathrm{O}(\sum_{k=|V|}^{1} \log k) = \mathrm{O}(\log(|V|!)) = \mathrm{O}(|V| \log |V|)$ opérations.
- Changer l'étiquette d'un sommet dans le tas coûte $\mathrm{O}(\log n)$. On fait cet opérations au plus $|E|$ fois sur un tas de taille ou pire $|V|$, soit $\mathrm{O}(|E| \log |V|)$ opérations.

La complexité totale est donc de l'ordre de $\mathrm{O}((|E| + |V|) \log |V|)$. Et encore : ceci néglige le fait que les comparaisons se font entre deux listes, ce qui ajoute un facteur $\mathrm{O}(|V|)$...
On peut obtenir $\mathrm{O}(|E| + |V|)$ avec une meilleure implémentation.

Implémentation $O(|E| + |V|)$ de LexBFS (1/2)

Au lieu d'une files de priorité de sommets (non numérotés), on utilise une liste ordonnée d'ensemble de sommets (non numérotés) de même étiquettes. Notons $S_\ell = \{x \in V \mid x \notin \sigma, \text{ et } \mathrm{label}[x] = \ell\}$
Initialement, la liste est $L = [S_{[|V|]}, S_{[]}]$, l'ensemble $S_{[|V|]}$ contenant la source, et l'ensemble $S_{[]}$ tous les autres états.
À chaque étape, on retire un élément x de l'ensemble en tête de L, puis on raffine les ensembles de L qui contiennent $N(x)$. Si un ensemble S_ℓ doit être raffiné, cela signifie qu'il faut créer un ensemble $S_{\ell i}$ immédiatement avant S_ℓ dans la liste L, et y déplacer certains éléments de S_ℓ. S_ℓ devra peut-être être effacé s'il est vide.
Chaque ensemble S_ℓ est représenté par une liste doublement chaînée.
Un tableau $SET[x]$ donne une cellule représentant x dans la liste doublement chaînée de l'ensemble S_ℓ contenant x, ainsi qu'un pointeur vers la position représentant S_ℓ dans la liste L.

Implémentation $O(|E| + |V|)$ de LexBFS (2/2)



Reconnaissance des graphes cordaux

Théorème Un graphe est cordal si et seulement si un ordre construit par LexBFS est un ordre d'élimination simplicial.

Complexité

- On sait construire un ordre avec LexBFS en $\mathrm{O}(|E| + |V|)$.
- La vérification que cet ordre est simplicial peut se faire avec la même complexité.

La complexité finale est donc $\mathrm{O}(|E| + |V|)$.

Plus de détails dans "Algorithmic graph theory and perfect graphs" (Martin Charles Golumbic).

Calcul d'un diamètre avec LexBFS

Théorème Deux parcours LexBFS permettent de calculer le diamètre d'un arbre, et d'approcher à un près le diamètre d'un graphe cordal.

Principe On lance un premier LexBFS qui termine sur un sommet u. À partir de ce sommet u, on lance un second LexBFS qui termine sur un sommet v. LexBFS étant un BFS, on peut lors de ce parcours en largeur calculer les distances de u à chaque sommet. On a $\Delta(G) = dist(u, v)$ sur les arbres. Sur les graphes cordaux $\Delta(G) = dist(u, v)$ ou $\Delta(G) = dist(u, v) + 1$.

Coloration d'un graphe cordal (1/2)

La coloration minimale (au sens du nombre de couleurs utilisées) d'un graphe quelconque est un problème NP-complet.

L'algorithme glouton de coloriage visite un graphe dans un ordre donné et affecte à chaque sommet la plus petite couleur utilisable. Le nombre de couleur utilisées peut-être différents selon l'ordre.

Un ordre tel que l'algorithme glouton donne la coloration minimale existe toujours ; mais il n'est pas forcément facile à trouver.

Un ordre parfait est un ordre des sommets tel que les coloriage effectué par l'algorithme glouton soit minimal pour G **et tous ses sous-graphes**.

Les graphes parfaitement ordonnençable sont les graphes pour les quels un ordre parfait existe.

Les graphes cordaux sont parfaitement ordonnençables ! Il suffit de choisir l'ordre inverse d'un ordre d'élimination simplicial : c'est-à-dire en fait l'ordre dans lequel LexBFS découvre les états.

Coloration d'un graphe cordal (2/2)

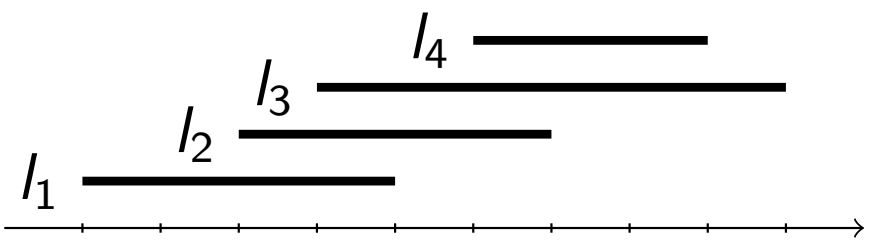
GreedyColor($G = (V, E)$, σ)
Entrée : un graphe G, un ordre sur les sommets σ
Sortie : un tableau de couleurs C indicé par les nœuds

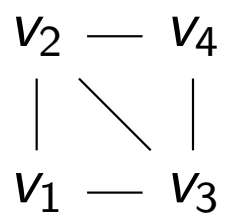
pour toute couleur n : $Avail(n) \leftarrow 1$
pour tout sommet x pris dans l'ordre σ :
    pour tout voisin $y \in N(x)$ déjà colorié :
        $Avail(C(y)) \leftarrow 0$
    $i \leftarrow 0$
    // La boucle suivante fait moins de $1 + |N(x)|$ itérations
    répéter $i \leftarrow i + 1$ jusqu'à ce que $Avail(i) = 1$
    $C(x) \leftarrow i$
    pour tout voisin $y \in N(x)$ déjà colorié :
        $Avail(C(y)) \leftarrow 1$
retourner C

$O(|V| + |E|)$ opérations.

Graphes d'intervalles

Soit $I_1, I_2, \ldots, I_n$ un ensemble d'intervalles de $\mathbb{R}$. Notons $G = (V, E)$ le graphe $V = \{v_1, v_2, \ldots, v_n\}$ et $(v_i, v_j) \in E \iff I_i \cap I_j \neq \emptyset$ représentant les intersections de ces intervalles.





Un graphe est un graphe d'intervalles si et seulement si il existe un ensemble d'intervalles de $\mathbb{R}$ qui le réalise (au sens ci-dessus).

Application à la planification

Dans un lycée, chaque classe (d'élèves) possède son emploi du temps qu'on considère comme un ensemble d'intervalles deux à deux disjoints (un intervalle par cours).
Si l'on réunit l'ensemble des intervalles correspondant au cours de toutes les classes, il y aura beaucoup d'intersections possibles.
Notons G le graphe d'intervalles de tous les cours du lycée. Pour faire des économies de ménage on veut savoir le nombre minimal de salles que l'on doit utiliser pour que tous les cours puisse avoir lieu (dans des pièces différentes, cela va de soit).
Cela revient à colorier de graphe de façon que deux sommets adjacents ne portent pas la même couleur. Le nombre de salles minimal est donc le nombre chromatique du graphe d'intervalles.

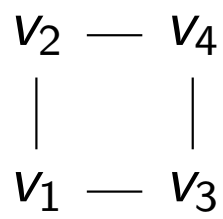
Les graphes d'intervalles sont cordaux

Théorème

Tout graphe d'intervalles est cordal.

En effet

Il n'est pas possible de trouver un ensemble d'intervalles qui réalise un cycle de longueur ≥ 4 sans corde :

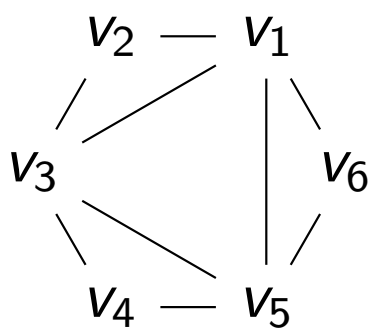


La réciproque est fausse

Tout graphe cordal n'est pas forcément un graphe d'intervalles.

Exemple

Il n'est pas possible de trouver un ensemble d'intervalles qui réalise le graphe cordal :



Allocation de registres : durée de vie des variables

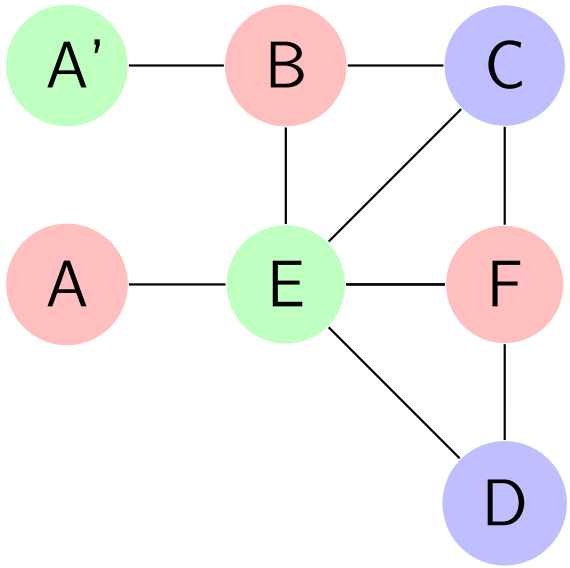
Considérons une séquence d'opérations entre plusieurs variables.

		a	b	c	d	e	f
1	$a \leftarrow 1$						
2	$e \leftarrow a + 1$	✓					
3	$d \leftarrow e \times a$	✓				✓	
4	$f \leftarrow d \times 2$				✓	✓	✓
5	$c \leftarrow d + f \times e$				✓	✓	✓
6	$b \leftarrow f + 2$			✓		✓	✓
7	$a \leftarrow b + c + e$		✓	✓		✓	
8	return $a \times b$	✓	✓				

On étudie la durée de vie de chaque variable, c'est-à-dire les moments où la valeur doit être stockée : dès l'affectation, jusqu'à sa dernière utilisation. Entre la dernière utilisation est l'affectation suivante, il est inutile de conserver la valeur : on pourrait aussi bien considérer que le a du bas est une seconde variable a'.

Allocation de registres : graphe d'interférence

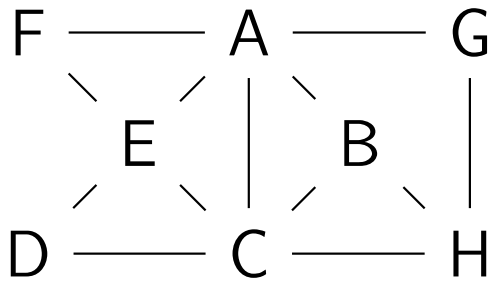
Le graphe d'interférence relie toutes les variables qui peuvent être en vie au même moment. Si l'on considère la durée de vie de la variable comme un intervalle (il y a deux intervalles pour *a*), ce graphe n'est rien d'autre qu'un graphe d'intervalles :



Un LexBFS à partir de A donnera (par exemple) l'ordre d'élimination A',B,C,D,F,E,A. Une coloration gloutonne dans l'ordre inverse permet d'allouer les registres (un par couleur) en temps linéaire.

Qui a tué le Duc de Densmore ?

- Chaque femme ne s'est rendue qu'une fois au château : le graphe représentant la relation "a rencontré" doit donc un graphe d'intervalle (les intervalles des présences de chaque femme au château).
- Dessinons-le :



- Ce graphe n'est pas cordal car les cycles (A, B, H, G) et (A, C, H, G) sont sans corde. Donc ce ne peut pas être un graphe d'intervalle : quelqu'un a menti.
- D'autre part le cycle (A, B, C, D, E, F) bien que possédant des cordes (le sous-graphe induit est effectivement cordal) ne peut pas non-plus apparaître dans un graphe d'intervalles.
- La personne commune aux 3 cycles posant problème est A...