Algorithme exact de calcul du MDS

C’est quoi un MDS ?

MDS = Minimum Dominating Set

Définition : Dans un graphe G, le MDS est le plus petit sous-ensemble D de sommet tel que tout les sommets de G soient dominés par D.

Notion de “dominé”

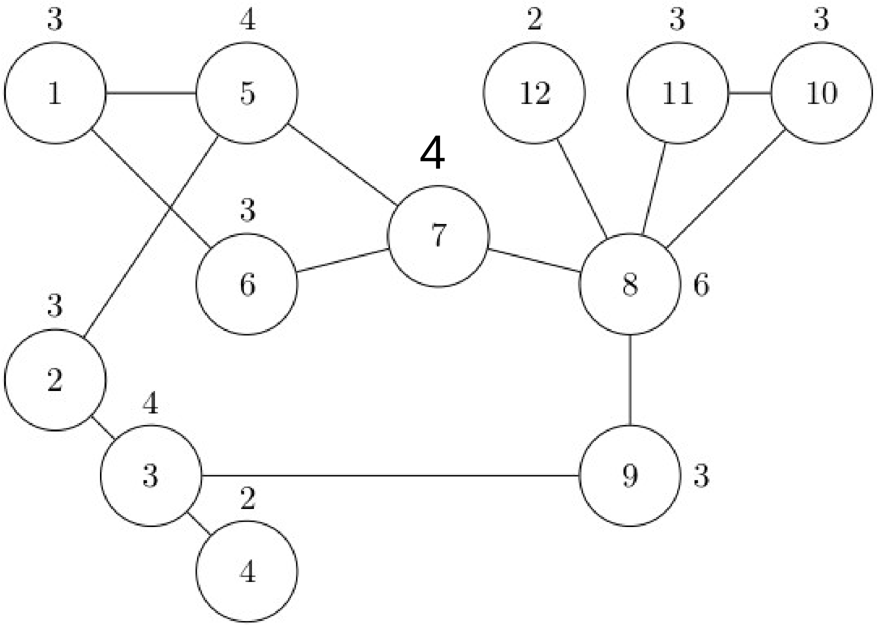
Un sommet v1 est dominé par un sommet v2 si :

- Soit v1=v2
- Soit v1 est voisin de v2 → L’arc u(v1, v2) existe.

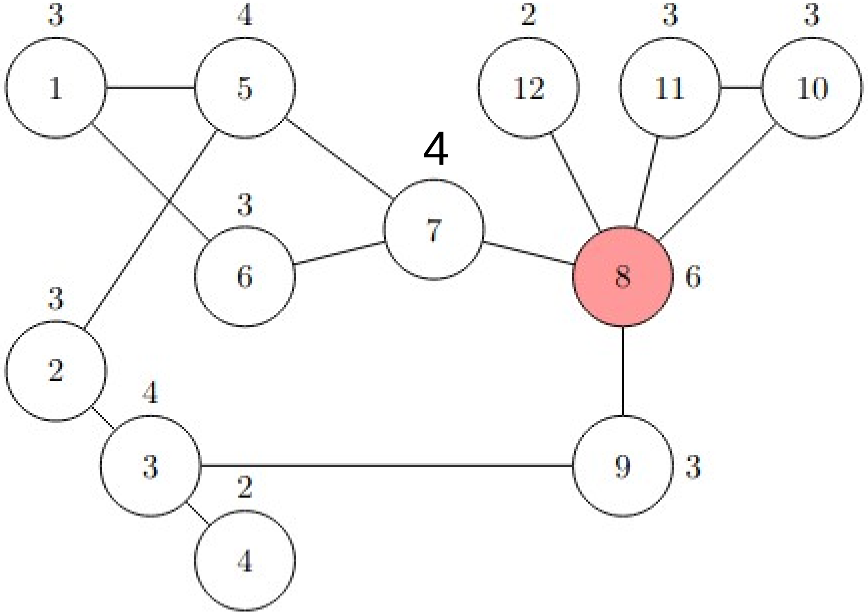
L’algorithme de calcul du MDS

- Un algorithme branch & bound avec :
  - G’, le graphe réduit de G
  - D_f, la solution finale partielle
  - U, l’ensemble des sommets de G’ non dominé par D_f
  - *D_0, la solution dite “imcumbent”*

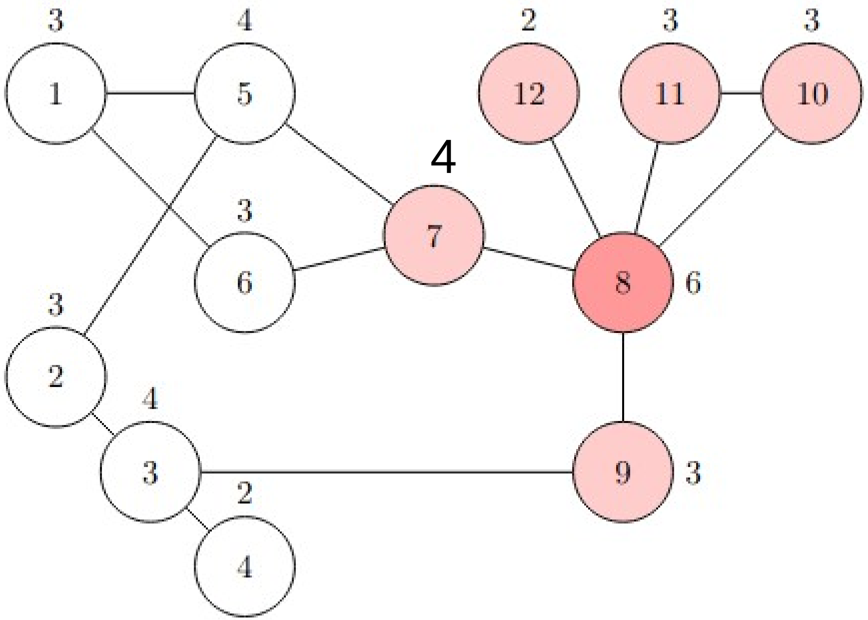
$D_0 = \varnothing$



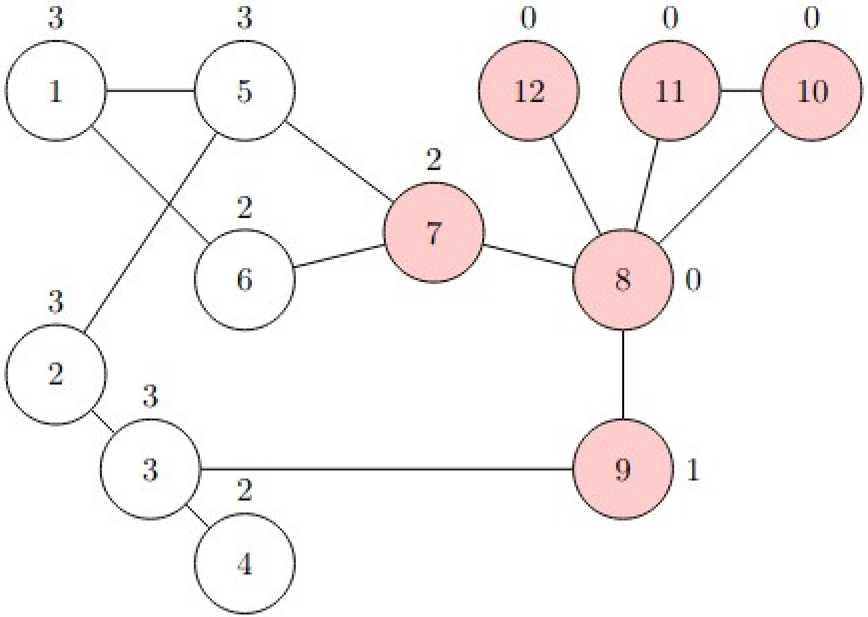
$D_0 = \varnothing$



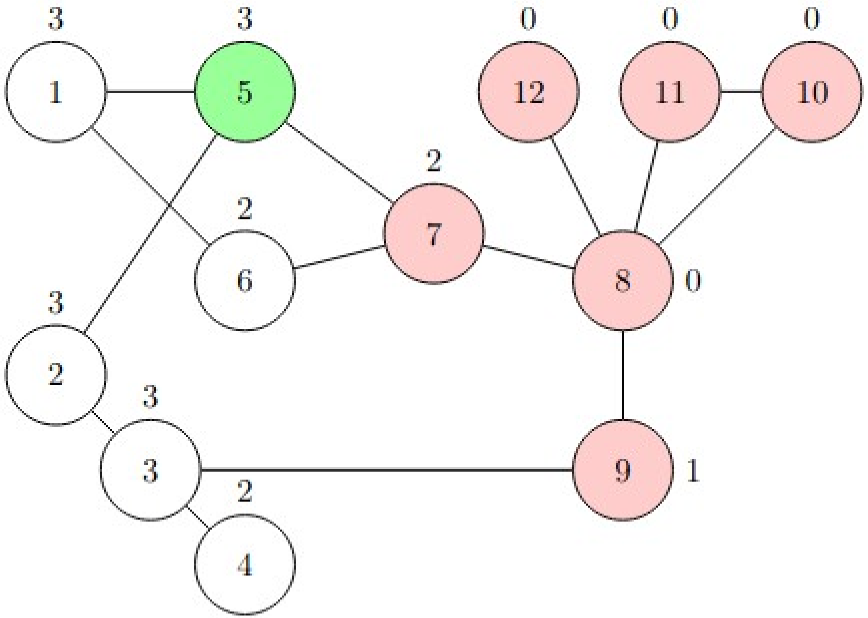
$D_0 = \{8\}$



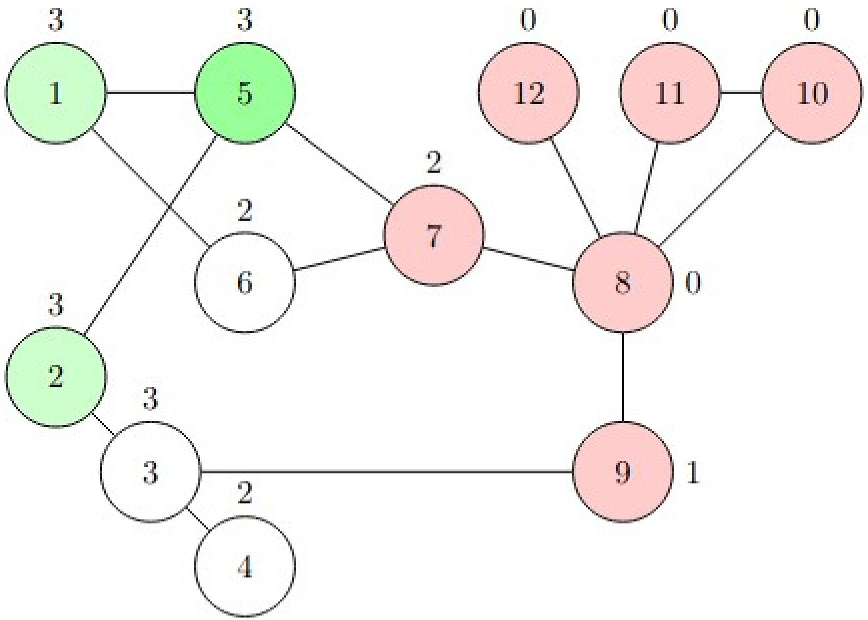
$D_0 = \{8\}$



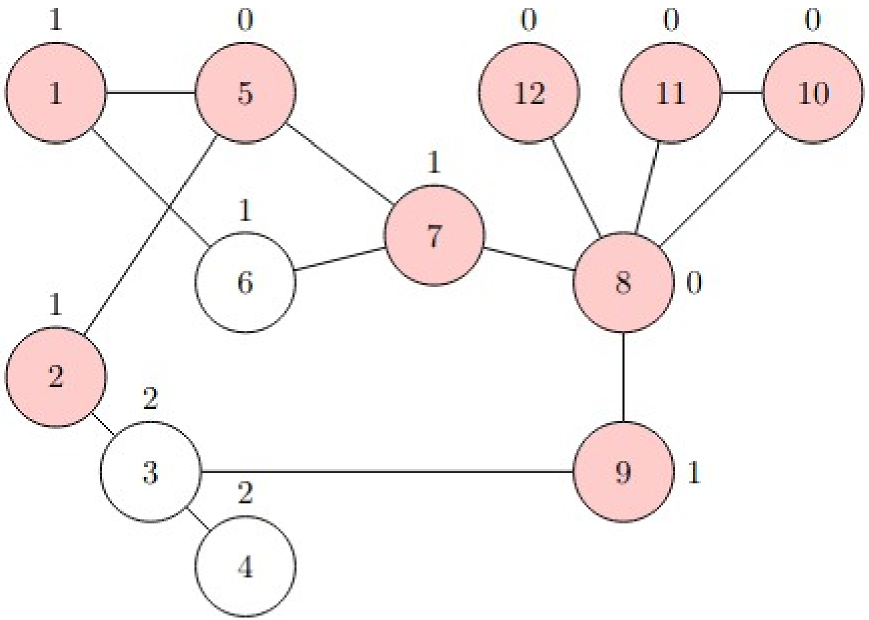
$D_0 = \{8\}$



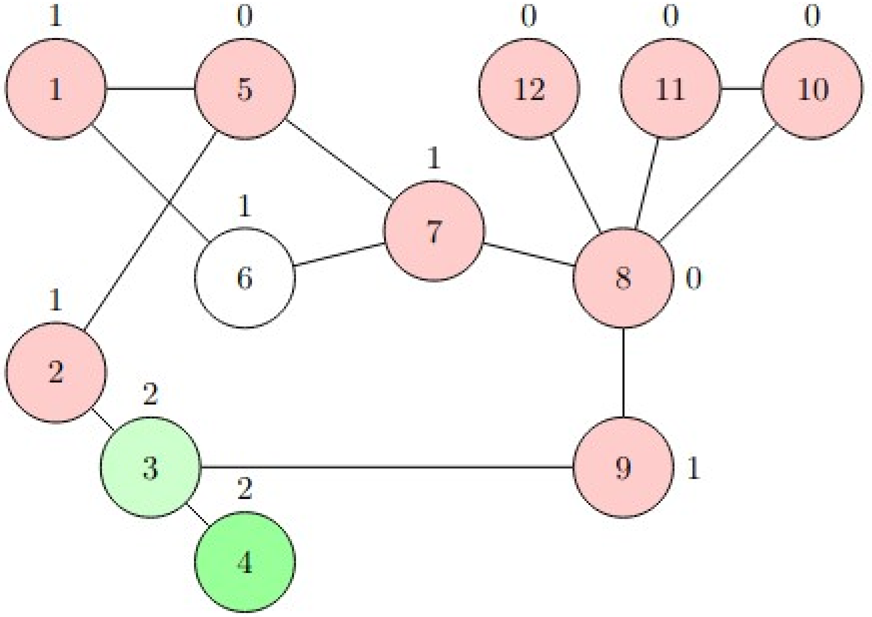
$D_0 = \{8;5\}$



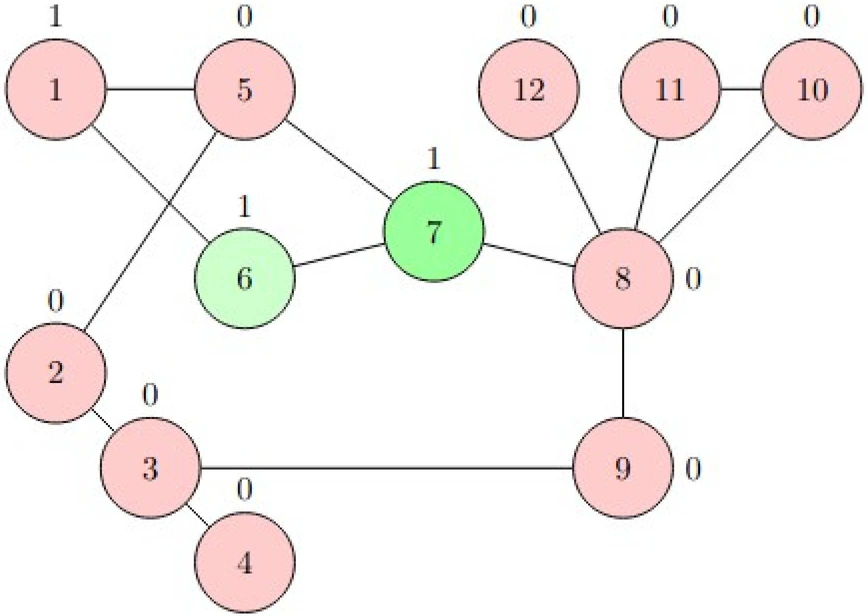
$D_0 = \{8;5\}$



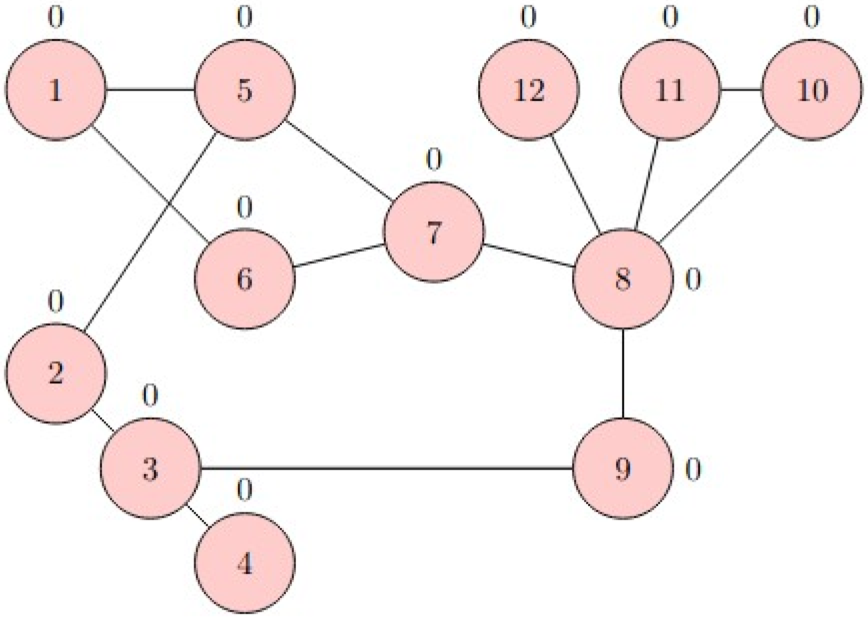
$D_0 = \{8;5;4\}$



$$D_0 = \{8;5;4;7\}$$



$D_0 = \{8;5;4;7\}$



L’algorithme de calcul du MDS

- Un algorithme branch & bound avec :
  - *G’, le graphe réduit de G*
  - *D_f, la solution finale partielle*
  - U, l’ensemble des sommets de G’ non dominé par D_f
  - $D_0 = \{8;5;4;7\}$

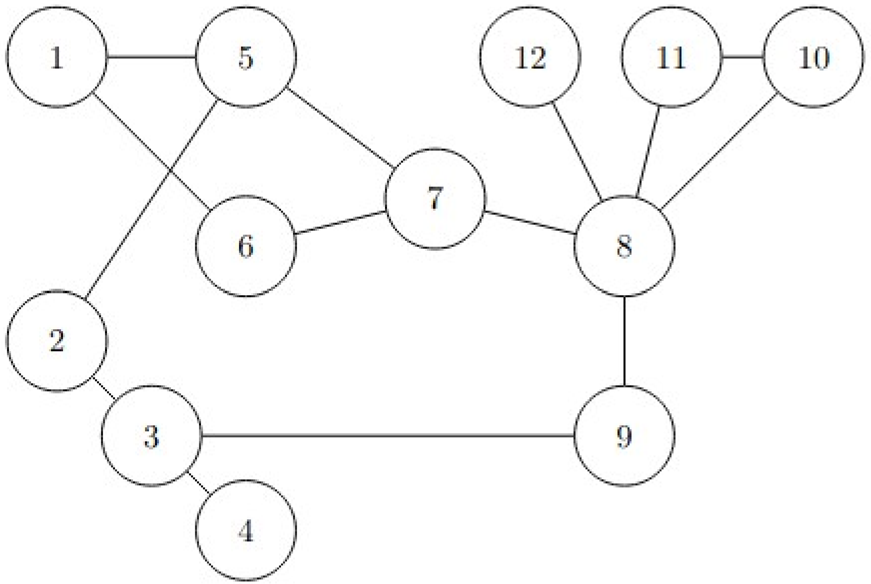
“Calcul” de D_f & Création de G’

Le graphe réduit G’ de G est le graphe dans lequel on identifie les sommets v_1 devant forcément appartenir au MDS (et qui appartiendront à D_f), et on supprime tout les sommets v_2 qui n’ont pas de voisin que v_1 n’a pas

Divisions des voisins

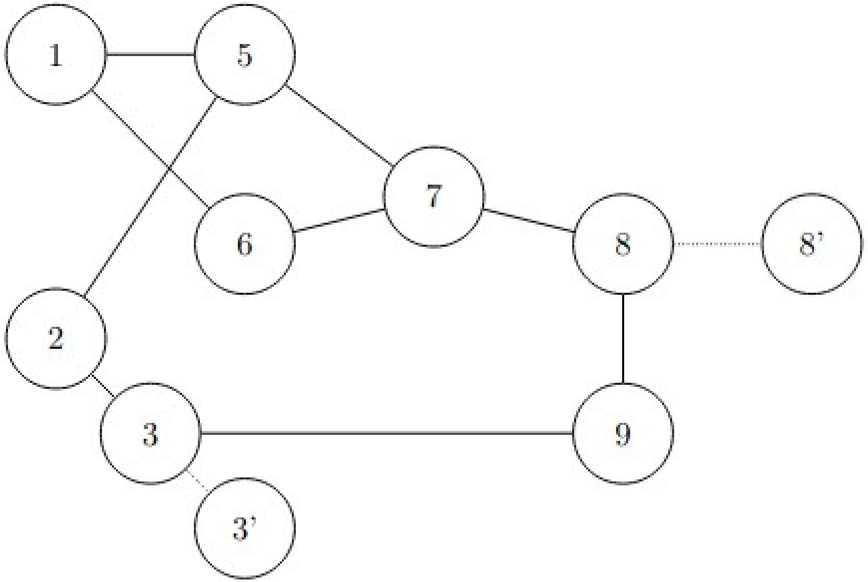
- $N_1(v)$: Ensemble des voisins qui ont un voisin n’appartenant pas à $N(v)$
- $N_2(v)$: Ensemble des voisins qui ont un voisin en commun avec $N_1(v)$
- $N_3(v)$: Ensemble des voisins qui ne sont pas dans $N_1(v)$ U $N_2(v)$

→ Autrement dit, on va mettre tout les sommets qui ont $N_3(v) \neq Ø$ dans D_f et transformer $N_3(v)$ et $N_2(v)$ quand $N_3(v) \neq Ø$ en v’



v	$N_1(v)$	$N_2(v)$	$N_3(v)$
1	{5;6}		
2	{5;3}		
3	{2;9}		{4}
4	{3}		
5	{1;2;7}		
6	{1;7}		
7	{5;6;8}		
8	{7;9}	{11;10}	{12}
9	{3;8}		
10	{8}	{11}	
11	{8}	{10}	
12	{8}		

$D_f = \{3;8\}$



Graphe G' réduit de G

L’algorithme de calcul du MDS

- Un algorithme branch & bound avec :
  - G’, le graphe réduit de G
  - $D_f = \{3;8\}$
  - $U = \{1;5;6\}$
  - $D_0 = \{8;5;4;7\}$

Problème ?

Trop d’embranchements possibles !!

-> N'importe quel sommet non présent dans D_f pourrait intégrer la solution finale

-> On veut réduire le nombre de branches

Fonction ReduceBranches

- Elle prend les mêmes paramètres que la fonction branch & bound : G’, D_f, U, D_0

- L’idée c’est de réduire le nombre de sommet sur lesquels la suite de la solution D_f pourrait se baser → Réduire le nombre de branche à explorer.

Pour commencer...

- On va partitionner le sous-ensemble V \ D en 4 sous-ensembles.

	S1	S2	S3	S4
Dominé	NON	OUI	NON	OUI
“branched”	NON	NON	OUI	OUI

Tout les sommets de S1 U S2 sont candidats à intégrer Df et à faire parti du sous-ensemble de branches

Les données…

- Un ensemble P de sommets (initié à Ø), avec P ⊆ U
- Un ensemble C de sommets, avec C = S1 U S2

  → Autrement dit, C est l'ensemble de sommets "unbranched"

- Π l'ensemble de k IS de $G[P]^2$

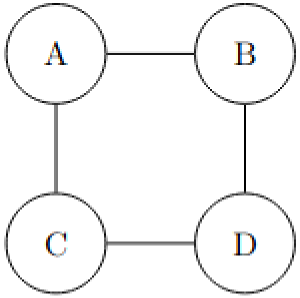
NB : U = S1 U S3

G[P]² c'est quoi ?

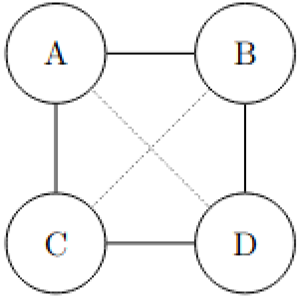
C'est le hop-2 graph du sous-graphe G[P] de G induit par P

→ Un hop-2 graph de G, c'est un graphe dont les sommets qui ont une "distance" de 2 entre eux possèdent un arc.

Exemple de hop-2 graph



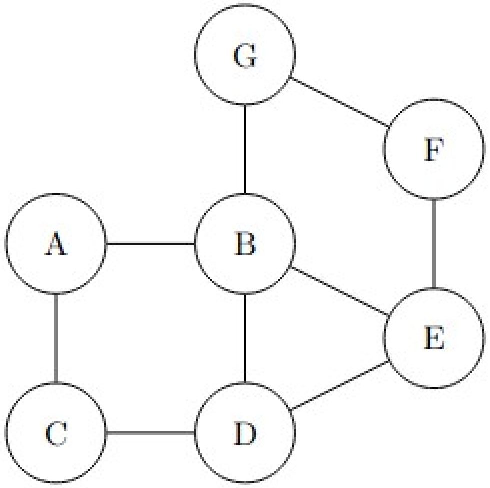
Graph G d’exemple



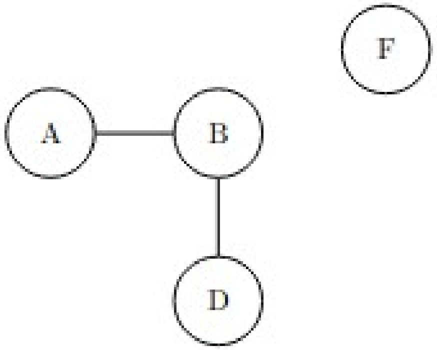
2-hop graph de G

$G[P]^2$ c'est quoi ?

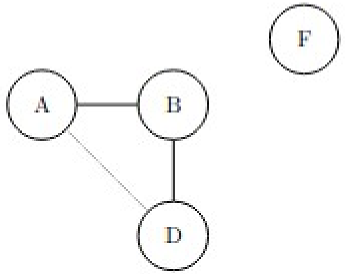
Admettons P = {A;B;D;F}



Graph G



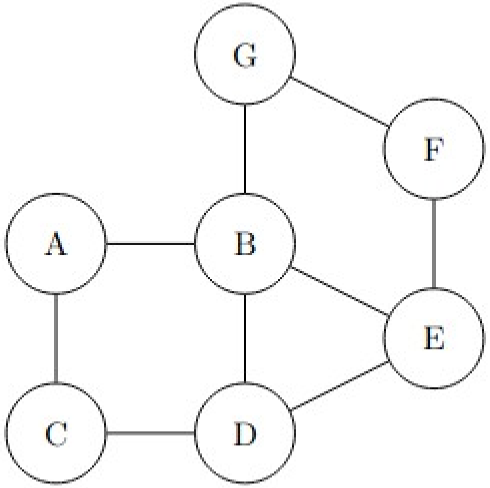
Graph G[P]



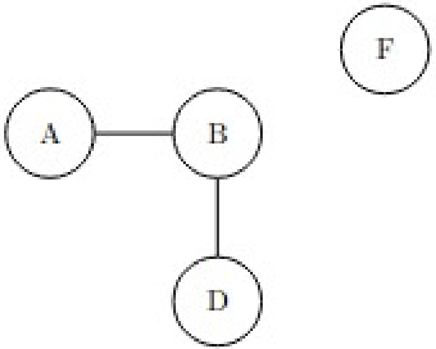
Graph $G[P]^2$

$G[P]^2$ c’est quoi ?

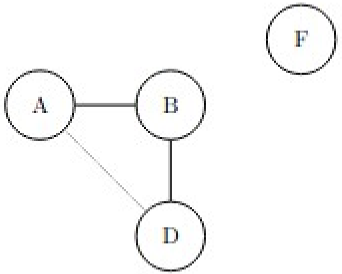
Admettons $P = \{A;B;C;F\}$



Graph G

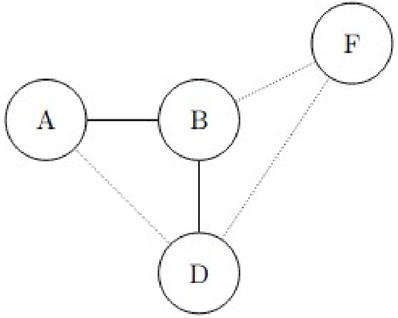


Graph G[P]



Graph $G[P]^2$

$G[P]^2 \neq G^2[P]$



Graph $G^2[P]$

Identifier l'ensemble P

On note I_j un IS de $G[P]^2$.

→ On cherche à obtenir un $|I_j| \geq |D_0| - |D_f|$

Propriété :

Soit $G = (V,E)$ un graphe

Soit G^2 le 2-hop graphe correspondant

Si I est IS (“Independent Set”) de G^2, alors

$| I | \leq | MDS(G) |$

Identifier l'ensemble P

On note I_j un IS de $G[P]^2$.

→ On cherche à obtenir un $|I_j| \geq |D_0| - |D_f|$

→ Si $|I_j| \geq |D_0| - |D_f|$ alors $|MDS(G[P])| \geq |D_0| - |D_f|$

→ $|MDS(G[P])| + |D_f| \geq |D_0|$

Note : On peut supprimer toutes les branches qui incluent P pour compléter Df car ça ne sera pas mieux que D_0.

Comment “remplir” P ?

On va, à l’aide d’une heuristique, calculer la somme de l’impact de l’insertion d’un sommet u présent dans U sur la cardinalité de tout les IS non vide. Si elle est positive, alors on insère u dans P

$$\sum_{j=0}^{k} \delta(I_j ; u) \, where \, I_k \, first \, empty \, set$$

δ(I_j ; u) c'est quoi ?

- δ(I_j ; u) = $\begin{cases} 1 & u \in S_1 \wedge N_1 = \emptyset \wedge N_2 = \emptyset \\ 0 & u \in S_1 \wedge N_1 = \emptyset \wedge N_2 \neq \emptyset \\ 1 - |N_1| & u \in S_1 \cup S_2 N_1 \neq \emptyset \\ 0 & u \in S_2 \wedge N_1 = \emptyset \\ 0 & u \in S_3 \wedge N_2 \neq \emptyset \\ 1 & u \in S_3 \wedge N_2 = \emptyset \end{cases}$

$$N_1 = N_G(u) \wedge I_j$$

Voisins de u dans I_j

$$N_2 = N_G[u] \wedge N_G[I_j] \wedge P \wedge C$$

Voisins communs à u et I_j dans P
(et non branché)

Conséquence du remplissage de P

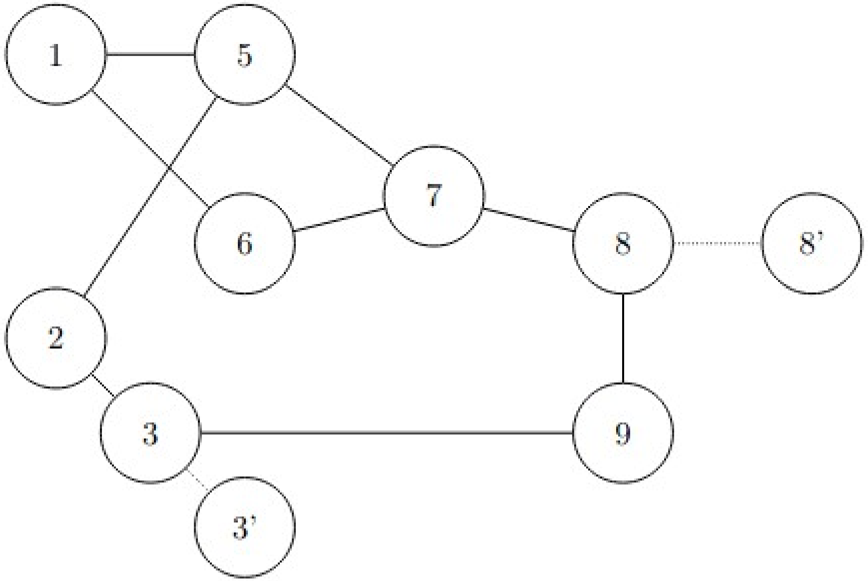
Lorsque l'on ajoute un sommet u à P, $\forall$ I_j non vide

- $\delta(I_j ; u) > 0 \rightarrow I_j \leftarrow I_j \cup \{u\}$
- $\delta(I_j ; u) = 0 \rightarrow$ On ne fait rien
- $\delta(I_j ; u) < 0 \rightarrow I_j \leftarrow$ retire $|$Voisins de u dans $I_j| - 1$ sommets conflictuels de I_j

Exemple avec notre graphe

Pour rappel :

- $D_f = \{3;8\}$
- $U = \{1;5;6\}$
- $D_0 = \{8;5;4;7\}$



Graphe G' réduit de G

u=6

$P = \emptyset$; $\forall j, I_j = \emptyset$

Donc : $P \leftarrow P \cup \{6\}$; $I_0 \leftarrow I_0 \cup \{6\}$

$u = 5$

$P = \{6\}$; $I_0 = \{6\}$ $\forall$ $j>0$, $I_j = \emptyset$

$\sum_{j=0}^{k} \delta(I_j ; u)\, where\, I_k\, first\, empty\, set$: $\delta(I_0 ; u)$

$N_1 = \{1 ; 7\} \cap \{6\} = \emptyset$

$N_2 = \{1;5;7\} \cap \{1;6;7\} \cap \{6\} = \emptyset$

$u \in S_1$

$N_1 = \varnothing$; $N_2 = \varnothing$; $u \in S_1$

$$\delta(I_j ; u) = \begin{cases} 1 & u \in S_1 \wedge N_1 = \varnothing \wedge N_2 = \varnothing \\ 0 & u \in S_1 \wedge N_1 = \varnothing \wedge N_2 \neq \varnothing \\ 1 - |N_1| & u \in S_1 \cup S_2 N_1 \neq \varnothing \\ 0 & u \in S_2 \wedge N_1 = \varnothing \\ 0 & u \in S_3 \wedge N_2 \neq \varnothing \\ 1 & u \in S_3 \wedge N_2 = \varnothing \end{cases}$$

$P \leftarrow P \cup \{u\}$; $P = \{6;5\}$

Insère t-on u dans I_0 ?

On a déjà calculé $\delta(I_0 ; u) = 1$ donc on insère

$I_0 \leftarrow I_0 \cup \{u\}$; $I_0 = \{6;5\}$

$u = 1$

$P = \{6;5\}$; $I_0 = \{6;5\}$; $\forall\ j>0,\ I_j = \varnothing$

$\sum_{j=0}^{k} \delta(I_j ; u)\, where\, I_k\, first\ empty\ set$: $\delta(I_0 ; u)$

$N_1 = \{6 ; 5\} \cap \{6 ; 5\} = \{6 ; 5\}$

$N_2 = \{1 ; 5 ; 6\} \cap \{1 ; 5 ; 6 ; 7\} \cap \{6 ; 5\} = \{6 ; 5\}$

$u \in S_1$

- $N_1 = \{6 ; 5\}$; $N_2 = \{6 ; 5\}$; $u \in S_1$

$$\delta(I_j ; u) = \begin{cases} 1 & u \in S_1 \wedge N_1 = \varnothing \wedge N_2 = \varnothing \\ 0 & u \in S_1 \wedge N_1 = \varnothing \wedge N_2 \neq \varnothing \\ 1 - |N_1| & u \in S_1 \cup S_2 N_1 \neq \varnothing \\ 0 & u \in S_2 \wedge N_1 = \varnothing \\ 0 & u \in S_3 \wedge N_2 \neq \varnothing \\ 1 & u \in S_3 \wedge N_2 = \varnothing \end{cases}$$

$\delta(I_0 ; u) = 1 - |\{6 ; 5\}| = -1 : P \leftarrow P$

Insère t-on u dans un I_j ?

Comme on a pas inséré u dans P, on ne l’insère dans aucun I_j.

On récupère le plus grand cardinal de tout les I_j, independent set de $G[P]^2$ qu'on note Ib

Si Ib < $|D_0| - |D_f|$ $\rightarrow$ On retourne C (S1 U S2)

Si un tel set P a été identifié premièrement, il n’existera jamais de set P qui respectera cette condition, on ne peut donc pas réduire le nombre de branches (c’est pourquoi on return C)

Dans notre exemple : $\max\{|I_j| \mid I_j \in \Pi\}$ est 2, et $|D_0|$ et $|D_f|$ est 2, donc elle n’est pas inférieur. On doit donc continuer...

P encore plus grand ?

On va chercher à élargir P

→On va insérer “virtuellement” dans P un à un les sommets de S2 et regarder l’effet que ça a sur le cardinal de chacun des I_j.

- Si $\max\{|I_j| \mid I_j \in \Pi\} < |D_0| - |D_f|$ alors on annule le changement sur tout les I_j et sur P

On a bientôt finit...

Une fois que nous sommes passés sur tout les sommets de S2, on return C \ P qui constituera les branches sur lesquels on travaillera.

$P = \{6;5\}$; $I_0 = \{6;5\}$

$S2=\{2;7;9\}$

$\delta(I_0 ; 9) = 0$ car $N_1 = Ø$ et $9 \in S2$

- La cardinalité de I_0 ne change pas donc $P \leftarrow P \cup \{9\}$

$P = \{6 ; 5 ; 9\}$; $I_0 = \{6;5\}$

$S2=\{2;7;9\}$

- $\delta(I_0 ; 7) = 1-|I_0| = -1$ car $N_1 = \{5;6\}$ et $9 \in S1 \cup S2$

  $\rightarrow$ On retire $|N(7) \cap I_0|-1=1$ sommet « conflictuel » à I_0 (5 ou 6)

- La cardinalité de I_0 devient $1 < |D_0| - |D_f|$ donc on annule le changement sur I_0 et $P \leftarrow P$

$P = \{6 ; 5 ; 9\}$; $I_0 = \{6;5\}$

$S2=\{2;7;9\}$

- $\delta(I_0 ; 2) = 1-|I_j| = -1$ car $N_1 = \{5\}$ et $7 \in S1 \cup S2$
    - $\rightarrow$ On retire $|N(2) \cap I_0|-1=0$ sommet « conflictuel » à I_0 (5 ou 6)

- La cardinalité de I_0 est inchangé donc $P \leftarrow P \cup \{2\}$

On fait quoi avec nos branches ?

- On ordonne B comme ci-contre $|N(b_1) \cup U| \geq |N(b_2) \cup U| \geq \ldots$

  $\forall\, b_i \in B$, on rappelle la fonction de branch & bound

  $\rightarrow$ $BnB(G' \,;\, D_f \cup \{b_i\} \,;\, U \cup \{N[b_i]\} \,;\, D_0)$ et on insèrera son résultat final dans un ensemble D’

Si $|D'| < |D_0|$, alors $D_0 \leftarrow D'$ et on renvoie D_0 à l'appel récursif précédent…

Si c'était le dernier appel récursif alors D_0 est le MDS du graphe G

Dans notre exemple...

$B = C \setminus P = \{1;2;5;6;7;9\} \setminus \{2;5;6;9\} = \{1;7\}$

$b_1 = 7$ car 7 a 2 voisins non dominé et un indice plus élevé

$b_2 = 1$ car 1 a 2 voisins non dominé mais un indice moins élevé

On relance BnB avec $D_f=\{3;8;7\}$

b_1

P={1} ; I_0 = {1}

Après élargissement de P : on retourne P = {1;5;6}

Puis on les ordonne comme ci contre : {6;5;1}

Puis on lance BnB avec D_f = {3;8;7;6}

U = Ø → On retourne D_f qui est de même taille que D_0 donc on abandonne cette branche et on backtrack

b_2

BnB avec D_f={3;8;1}

U = Ø → On retourne D_f qui plus petit que D_0 donc $D_0 \leftarrow D_f$

Il n’y a plus de branche à explorer, on renvoie donc comme résultat D_0 qui est le MDS de G.