Coloration de graphes parfaits sans Balanced Skew Partition

Théophile Trunck

Séminaire graphes et structures discrètes

Avril 2014

Co-Auteurs

Travail avec :

- Maria Chudnovsky, Columbia University, New York
- Nicolas Trotignon, CNRS, LIP, ENS Lyon
- Kristina Vušković, Union University, Belgrade and Leeds University

Définitions

- Un graphe G est *de Berge* si dans G et dans $\overline{G}$, tous les trous (i.e. les cycles sans cordes de taille au moins 4) sont pairs.

Définitions

- Un graphe G est *de Berge* si dans G et dans $\overline{G}$, tous les trous (i.e. les cycles sans cordes de taille au moins 4) sont pairs.
- Un graphe G est *parfait* si tous les sous graphes induits H de G vérifient $\chi(H) = \omega(H)$.

Définitions

- Un graphe G est *de Berge* si dans G et dans $\overline{G}$, tous les trous (i.e. les cycles sans cordes de taille au moins 4) sont pairs.
- Un graphe G est *parfait* si tous les sous graphes induits H de G vérifient $\chi(H) = \omega(H)$.

Théorème (Chudnovsky, Robertson, Seymour and Thomas, 2002)

Les graphes de Berge sont parfaits.

Définitions

- Un graphe G est *de Berge* si dans G et dans $\overline{G}$, tous les trous (i.e. les cycles sans cordes de taille au moins 4) sont pairs.
- Un graphe G est *parfait* si tous les sous graphes induits H de G vérifient $\chi(H) = \omega(H)$.

Théorème (Chudnovsky, Robertson, Seymour and Thomas, 2002)

Les graphes de Berge sont parfaits.

Corollaire

Un graphe est de Berge si et seulement si il est parfait.

But de l'exposé

Trouver un algorithme permettant de colorier les graphes parfaits.

- Comme $\chi(G) = \omega(G)$, si on sait calculer $\omega(G)$ on sait le nombre de couleurs requis.

But de l'exposé

Trouver un algorithme permettant de colorier les graphes parfaits.

- Comme $\chi(G) = \omega(G)$, si on sait calculer $\omega(G)$ on sait le nombre de couleurs requis.
- La classe étant héréditaire, si on sait calculer $\omega(G)$ on sait trouver une clique maximum.

But de l'exposé

Trouver un algorithme permettant de colorier les graphes parfaits.

- Comme $\chi(G) = \omega(G)$, si on sait calculer $\omega(G)$ on sait le nombre de couleurs requis.
- La classe étant héréditaire, si on sait calculer $\omega(G)$ on sait trouver une clique maximum.
- Cependant les graphes parfaits ne sont pas clos par suppression ou contraction d'arête.

But de l'exposé

Trouver un algorithme permettant de colorier les graphes parfaits.

- Comme $\chi(G) = \omega(G)$, si on sait calculer $\omega(G)$ on sait le nombre de couleurs requis.
- La classe étant héréditaire, si on sait calculer $\omega(G)$ on sait trouver une clique maximum.
- Cependant les graphes parfaits ne sont pas clos par suppression ou contraction d'arête.
- Connaitre $\chi(G)$ ne donne pas une coloration.

Stable, clique et coloration

Théorème (Gröstchel, Lovász, and Schrijver)

Si on sait calculer la taille d'un stable maximum pondéré en temps $\mathcal{O}(T)$, on sait trouver une coloration en temps $\mathcal{O}(n^3.T)$.

- Si on sait calculer $\alpha(G)$, on sait trouver un stable maximum

Stable, clique et coloration

Théorème (Gröstchel, Lovász, and Schrijver)

Si on sait calculer la taille d'un stable maximum pondéré en temps $\mathcal{O}(T)$, on sait trouver une coloration en temps $\mathcal{O}(n^3.T)$.

- Si on sait calculer $\alpha(G)$, on sait trouver un stable maximum
- Comme G est parfait ssi $\overline{G}$ l'est on sait trouver une clique maximum.

Stable, clique et coloration

Théorème (Gröstchel, Lovász, and Schrijver)

Si on sait calculer la taille d'un stable maximum pondéré en temps $\mathcal{O}(T)$, on sait trouver une coloration en temps $\mathcal{O}(n^3.T)$.

- Si on sait calculer $\alpha(G)$, on sait trouver un stable maximum
- Comme G est parfait ssi $\overline{G}$ l'est on sait trouver une clique maximum.
- Soient $K_1, \ldots K_t$ des cliques maximums, on sait trouver un stable S qui les intersecte toutes.

Stable, clique et coloration

Théorème (Gröstchel, Lovász, and Schrijver)

Si on sait calculer la taille d'un stable maximum pondéré en temps $\mathcal{O}(T)$, on sait trouver une coloration en temps $\mathcal{O}(n^3.T)$.

- Si on sait calculer $\alpha(G)$, on sait trouver un stable maximum
- Comme G est parfait ssi $\overline{G}$ l'est on sait trouver une clique maximum.
- Soient $K_1, \ldots K_t$ des cliques maximums, on sait trouver un stable S qui les intersecte toutes.
  - On donne à chaque sommet v le poids $|\{K_i, v \in K_i\}|$.

Stable, clique et coloration

Théorème (Gröstchel, Lovász, and Schrijver)

Si on sait calculer la taille d'un stable maximum pondéré en temps $\mathcal{O}(T)$, on sait trouver une coloration en temps $\mathcal{O}(n^3.T)$.

- Si on sait calculer $\alpha(G)$, on sait trouver un stable maximum
- Comme G est parfait ssi $\overline{G}$ l'est on sait trouver une clique maximum.
- Soient $K_1, \ldots K_t$ des cliques maximums, on sait trouver un stable S qui les intersecte toutes.
  - On donne à chaque sommet v le poids $|\{K_i, v \in K_i\}|$.
  - $\alpha = \overline{\chi} = t$.

Stable, clique et coloration

Théorème (Gröstchel, Lovász, and Schrijver)

Si on sait calculer la taille d'un stable maximum pondéré en temps $\mathcal{O}(T)$, on sait trouver une coloration en temps $\mathcal{O}(n^3.T)$.

- Si on sait calculer $\alpha(G)$, on sait trouver un stable maximum
- Comme G est parfait ssi $\overline{G}$ l'est on sait trouver une clique maximum.
- Soient $K_1, \dots K_t$ des cliques maximums, on sait trouver un stable S qui les intersecte toutes.
  - On donne à chaque sommet v le poids $|\{K_i, v \in K_i\}|$.
  - $\alpha = \overline{\chi} = t$.
- Si $\omega(G - S) < \omega(G)$, on a trouvé une classe de couleur.

Stable, clique et coloration

Théorème (Gröstchel, Lovász, and Schrijver)

Si on sait calculer la taille d'un stable maximum pondéré en temps $\mathcal{O}(T)$, on sait trouver une coloration en temps $\mathcal{O}(n^3.T)$.

- Si on sait calculer $\alpha(G)$, on sait trouver un stable maximum
- Comme G est parfait ssi $\overline{G}$ l'est on sait trouver une clique maximum.
- Soient $K_1, \dots K_t$ des cliques maximums, on sait trouver un stable S qui les intersecte toutes.
  - On donne à chaque sommet v le poids $|\{K_i, v \in K_i\}|$.
  - $\alpha = \overline{\chi} = t$.
- Si $\omega(G - S) < \omega(G)$, on a trouvé une classe de couleur.
- Sinon on cherche une nouvelle clique maximum K_{t+1} de G qu'on ajoute à la liste (ceci n'arrive qu'au plus n fois).

Calculer α

Soit $\mathcal{M}_G$ l'ensemble des matrices $n \times n$ symétriques semi-définies positives de trace égale à 1 et telles que $M_{u,v} = 0$ si uv est une arête.

On note $\vartheta = \max\{\mathbf{1}^T M \mathbf{1}, M \in \mathcal{M}_G\}$

Calculer α

Soit $\mathcal{M}_G$ l'ensemble des matrices $n \times n$ symétriques semi-définies positives de trace égale à 1 et telles que $M_{u,v} = 0$ si uv est une arête.

On note $\vartheta = \max\{\mathbf{1}^T M \mathbf{1}, M \in \mathcal{M}_G\}$

Théorème

Pour tout graphe G, $\alpha(G) \leq \vartheta(G) \leq \overline{\chi}^*(G) \leq \overline{\chi}(G)$

Calculer α

Soit $\mathcal{M}_G$ l'ensemble des matrices $n \times n$ symétriques semi-définies positives de trace égale à 1 et telles que $M_{u,v} = 0$ si uv est une arête.

On note $\vartheta = \max\{\mathbf{1}^T M \mathbf{1}, M \in \mathcal{M}_G\}$

Théorème

Pour tout graphe G, $\alpha(G) \leq \vartheta(G) \leq \overline{\chi}^*(G) \leq \overline{\chi}(G)$

Théorème (Gröstchel, Lovász, and Schrijver, 1980's)

Il existe un algorithme qui pour tout graphe $G = (V, E)$ et réel $\varepsilon > 0$ calcule x, $|\vartheta(G) - x| < \varepsilon$ en temps polynomial en $|V|$ et en $\log(1/\varepsilon)$.

Calculer α

Soit $\mathcal{M}_G$ l'ensemble des matrices $n \times n$ symétriques semi-définies positives de trace égale à 1 et telles que $M_{u,v} = 0$ si uv est une arête.

On note $\vartheta = \max\{\mathbf{1}^T M \mathbf{1}, M \in \mathcal{M}_G\}$

Théorème

Pour tout graphe G, $\alpha(G) \leq \vartheta(G) \leq \overline{\chi}^*(G) \leq \overline{\chi}(G)$

Théorème (Gröstchel, Lovász, and Schrijver, 1980's)

Il existe un algorithme qui pour tout graphe $G = (V, E)$ et réel $\varepsilon > 0$ calcule x, $|\vartheta(G) - x| < \varepsilon$ en temps polynomial en $|V|$ et en $\log(1/\varepsilon)$.

Corollaire

Dans les graphes parfaits, $\alpha(G) = \overline{\chi}(G)$, on peut donc calculer $\alpha(G)$ en temps polynomial et donc trouver une coloration en temps polynomial.

Une coloration combinatoire

Théorème (Chudnovsky, Robertson, Seymour and Thomas, 2002)

Tout graphe de Berge est basique ou admet une décomposition

Peut-on calculer α, par induction ?

Graphes basiques

- Les graphes bipartis.

Graphes basiques

- Les graphes bipartis.
- Les complémentaires de graphes bipartis.

Graphes basiques

- Les graphes bipartis.
- Les complémentaires de graphes bipartis.
- Les lines graphes de bipartis.

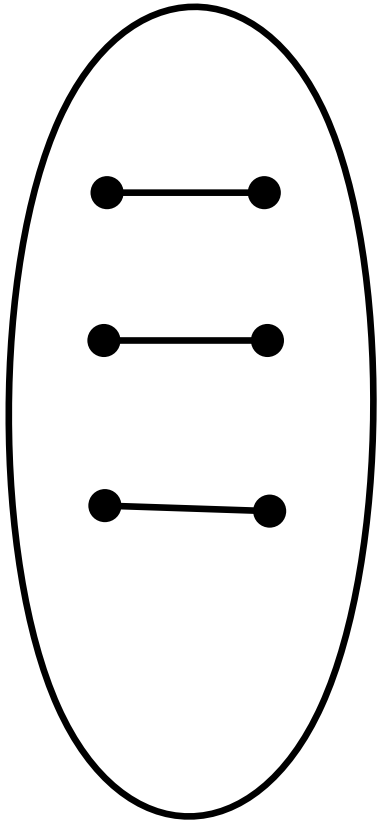
Graphes basiques

- Les graphes bipartis.
- Les complémentaires de graphes bipartis.
- Les lines graphes de bipartis.
- Les complémentaires de lines graphes de bipartis.

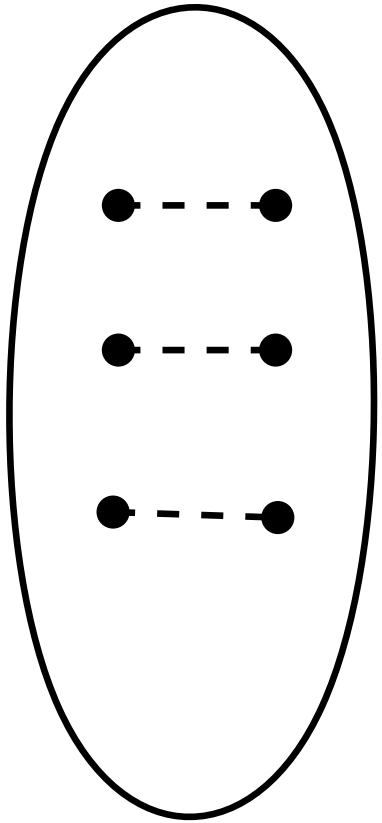
Graphes basiques

- Les graphes bipartis.
- Les complémentaires de graphes bipartis.
- Les lines graphes de bipartis.
- Les complémentaires de lines graphes de bipartis.
- Les doubles splits.

Double split

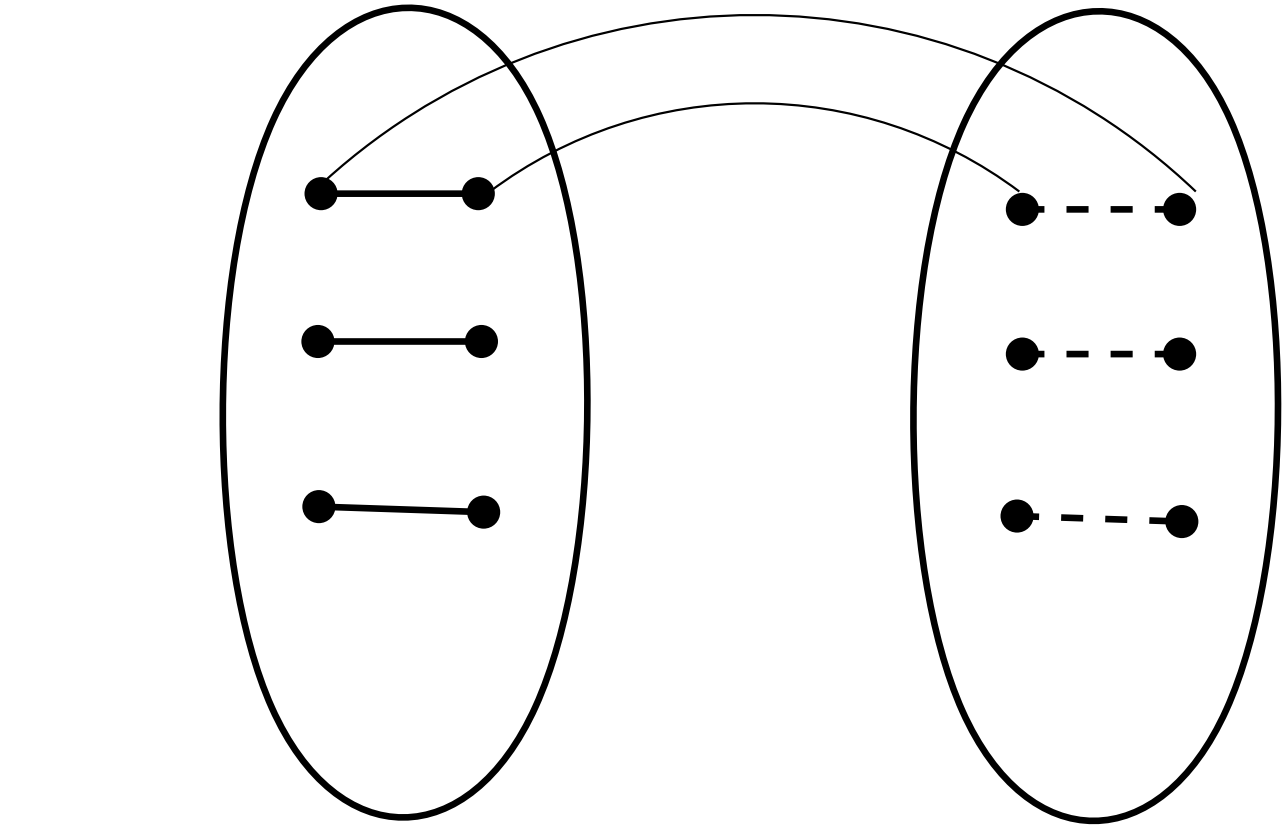


+ all no edges

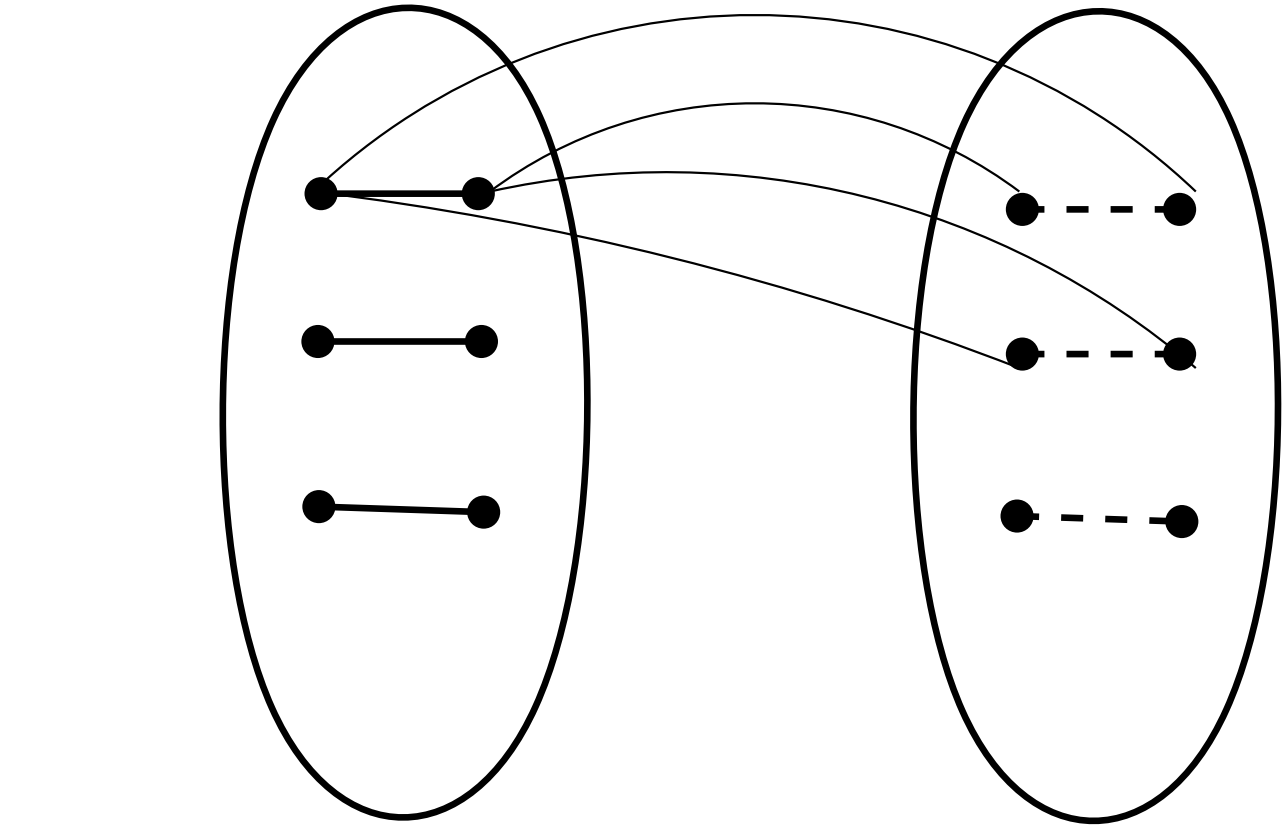


+ all edges

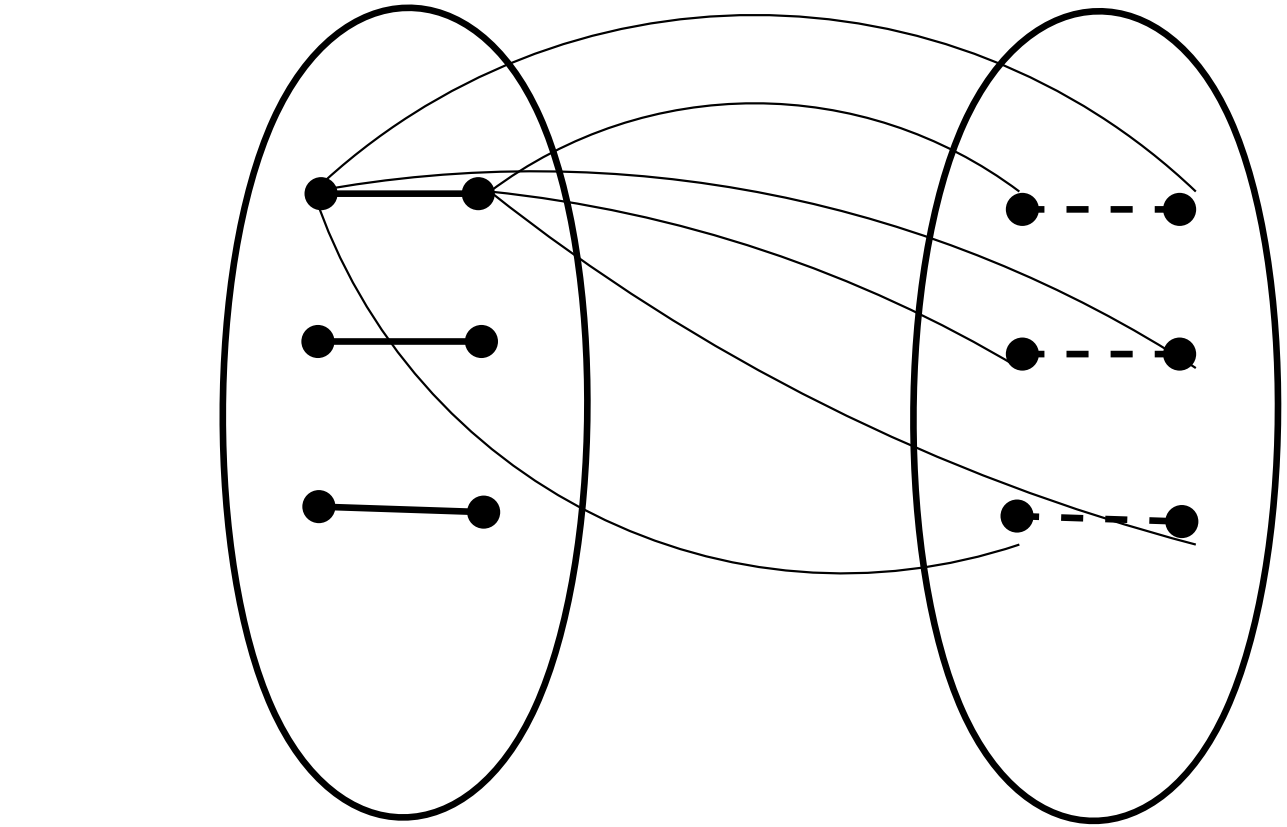
Double split



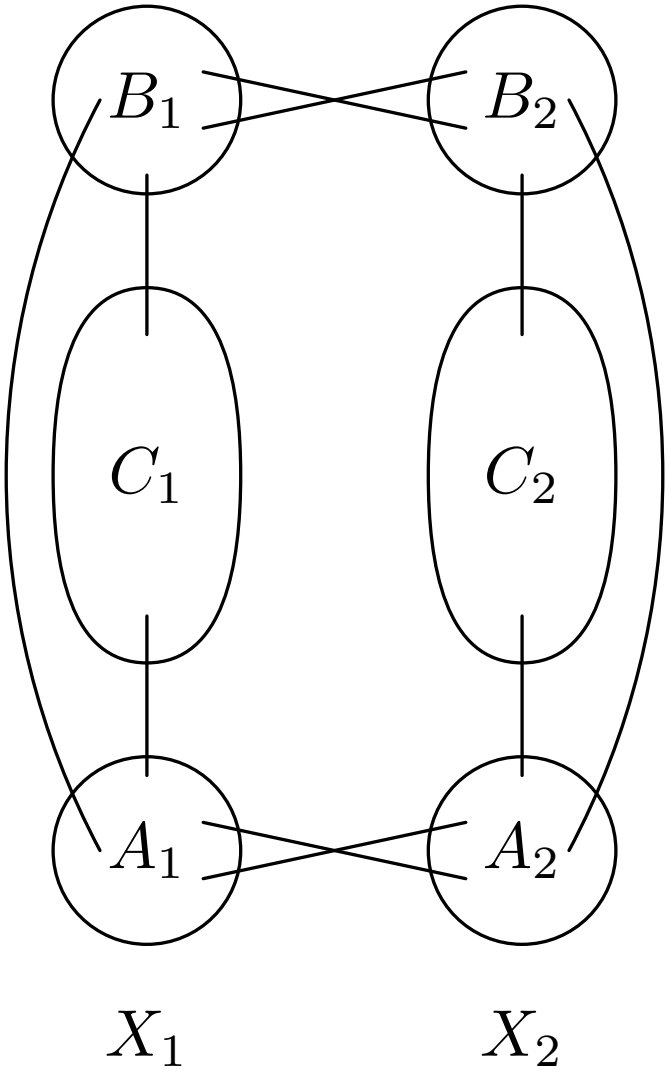
Double split



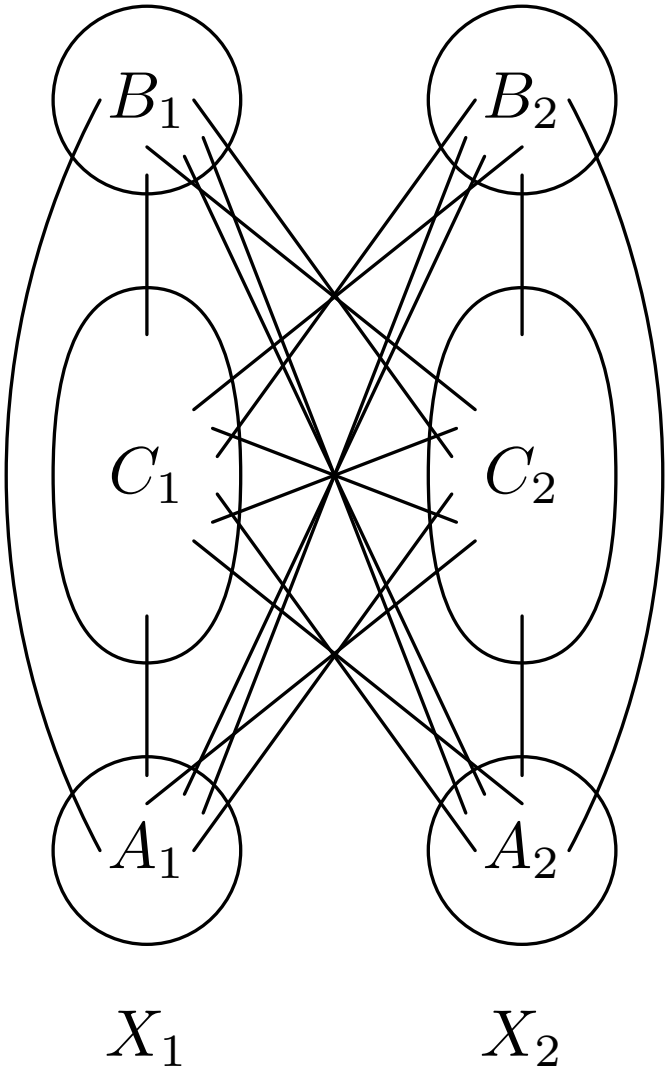
Double split



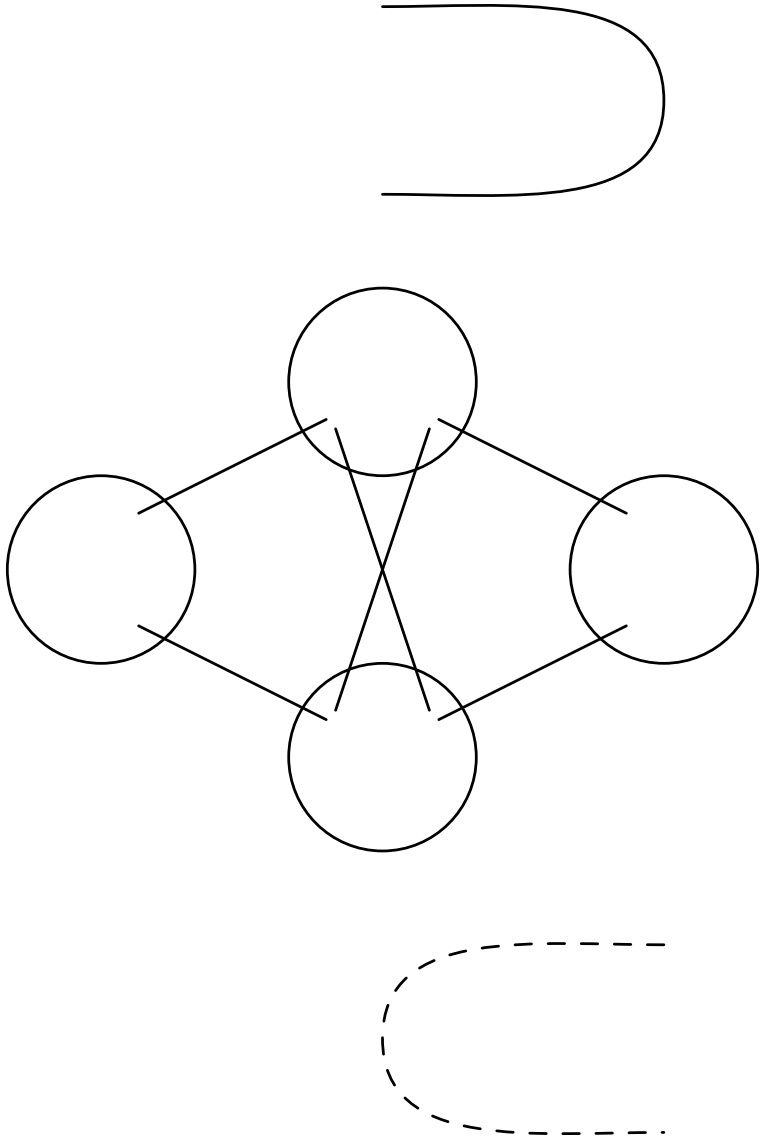
2-join



2-join



Balanced Skew Partition



Résultat principal

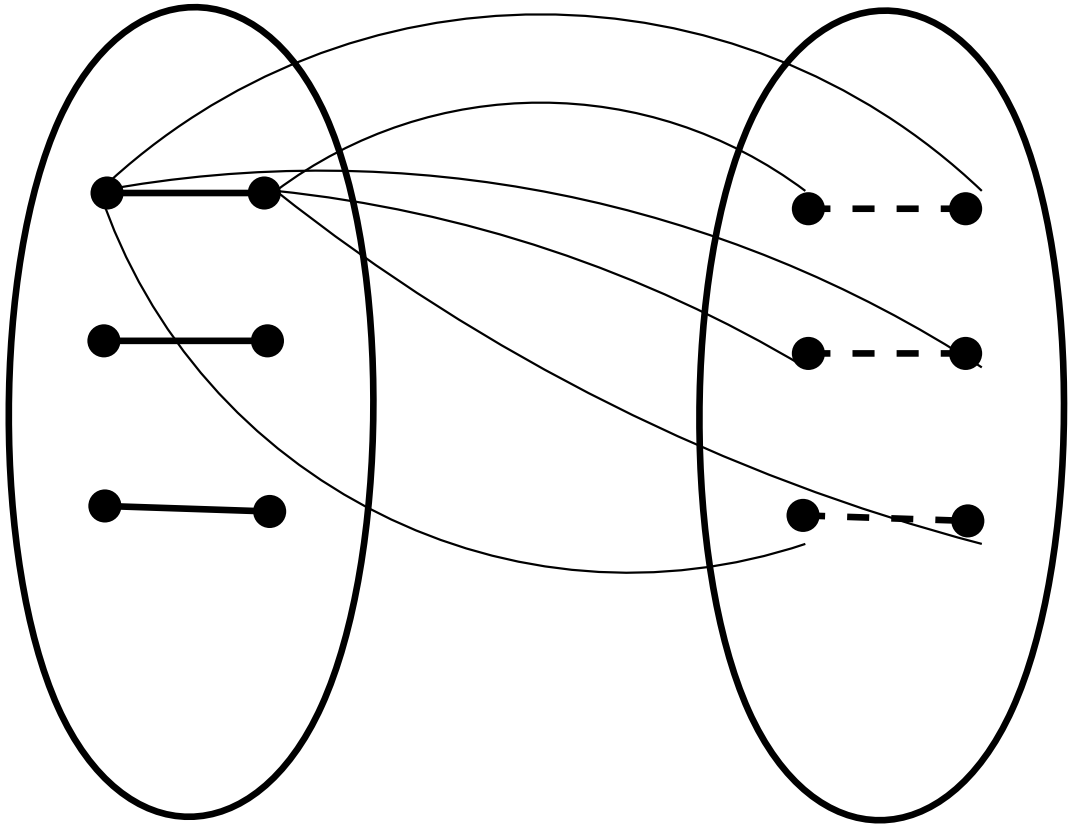
Théorème (Chudnovsky, Trotignon, T., Vušković)

Il existe un algorithme en $\mathcal{O}(n^5)$ pour le stable maximum pondéré dans les graphes de Berge sans BSP. (On peut également trouver le stable).

Résultats sur les classes de bases

Classe	Reconnaissance	Stable max
Bipartite	$\mathcal{O}(n+m)$	$\mathcal{O}(n^3)$
$\overline{\text{Bipartite}}$	$\mathcal{O}(n+m)$	$\mathcal{O}(n^2)$
Line BIP	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$
$\overline{\text{Line BIP}}$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$
Double Split	$\mathcal{O}(n+m)$	$\mathcal{O}(n^4)$

Cas des double split



On note X l'ensemble du matching et Y celui de l'antimatching.

Cas des double split

Si on connait X, Y, un stable maximum est :

- Dans $G[X]$ qui est biparti

Cas des double split

Si on connait X, Y, un stable maximum est :

- Dans $G[X]$ qui est biparti
- Dans $G[Y]$ qui est un complément de biparti

Cas des double split

Si on connait X, Y, un stable maximum est :

- Dans $G[X]$ qui est biparti
- Dans $G[Y]$ qui est un complément de biparti
- Composé d'un sommet de Y et de ses non-voisins dans X.

Cas des double split

Si on connait X, Y, un stable maximum est :

- Dans $G[X]$ qui est biparti
- Dans $G[Y]$ qui est un complément de biparti
- Composé d'un sommet de Y et de ses non-voisins dans X.

Cas des double split

Si on connait X, Y, un stable maximum est :

- Dans $G[X]$ qui est biparti
- Dans $G[Y]$ qui est un complément de biparti
- Composé d'un sommet de Y et de ses non-voisins dans X.

On peut reconnaitre un double split et trouver (X, Y)

- On devine une arête ab de X, tous les non-voisins de a et de b vont dans X, ceux voisins d'uniquement un des deux dans Y.

Cas des double split

Si on connait X, Y, un stable maximum est :

- Dans $G[X]$ qui est biparti
- Dans $G[Y]$ qui est un complément de biparti
- Composé d'un sommet de Y et de ses non-voisins dans X.

On peut reconnaitre un double split et trouver (X, Y)

- On devine une arête ab de X, tous les non-voisins de a et de b vont dans X, ceux voisins d'uniquement un des deux dans Y.
- On devine une non-arête ab de Y, tous les voisins de a et de b vont dans Y, ceux voisins d'uniquement un des deux dans X.

Cas des double split

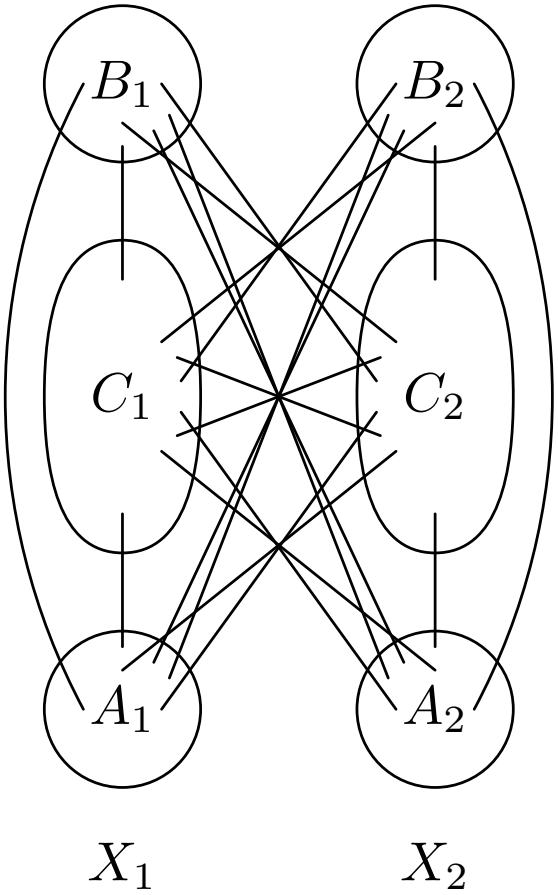
Si on connait X, Y, un stable maximum est :

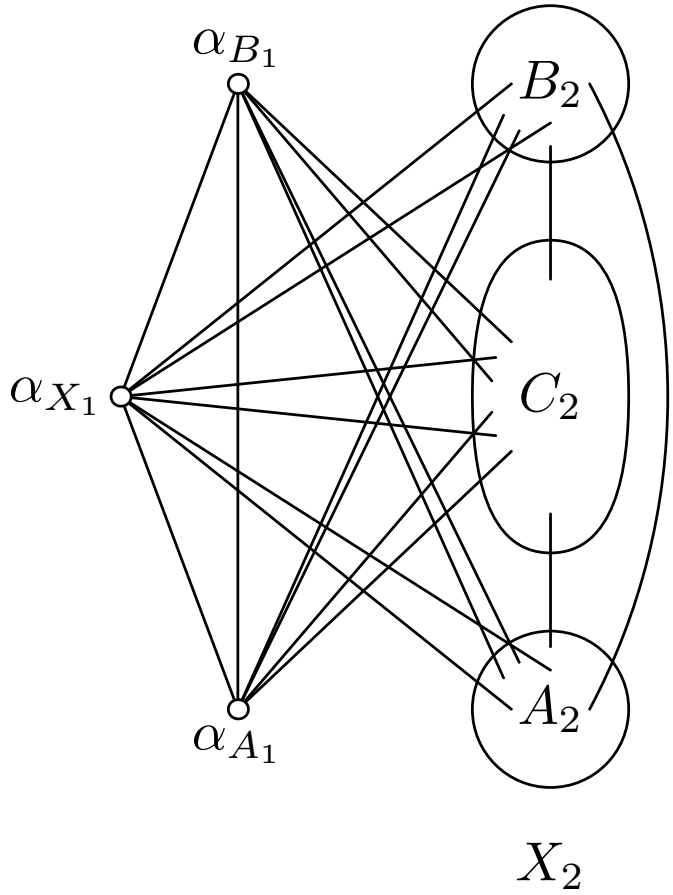
- Dans $G[X]$ qui est biparti
- Dans $G[Y]$ qui est un complément de biparti
- Composé d'un sommet de Y et de ses non-voisins dans X.

On peut reconnaitre un double split et trouver (X, Y)

- On devine une arête ab de X, tous les non-voisins de a et de b vont dans X, ceux voisins d'uniquement un des deux dans Y.
- On devine une non-arête ab de Y, tous les voisins de a et de b vont dans Y, ceux voisins d'uniquement un des deux dans X.
- Il n'y a pas d'arête dans X ni de non-arête dans Y, c'est un split graphe que l'on peut reconnaitre en temps linéaire (en regardant les degrés)

Utiliser les décompositions





Récurrence

On trouve une décomposition, on pose alors plusieurs questions du petit coté, ce qui nous permet d'encoder l'information du grand coté, sur lequel on ne pose qu'une seule question.

Nous avons : $T(n) \leq 4T(|V(T_X)|) + T(|V(T_Y)|) + dn^4$

Avec $|V(T_X)| \leq |V(T_Y)|$ et $6 \leq |V(T_X)|, |V(T_Y)| \leq n-1$.

On peut alors montrer que $T(n) = \mathcal{O}(n^5)$.

Complications techniques

Voici l'idée générale, mais il y a des complications :

- Les blocs ne préservent pas la classe (introduisent des skew-partitions)

Complications techniques

Voici l'idée générale, mais il y a des complications :

- Les blocs ne préservent pas la classe (introduisent des skew-partitions)
- Les blocs ne préservent pas être de Berge (introduisent des cyles impairs)

Complications techniques

Voici l'idée générale, mais il y a des complications :

- Les blocs ne préservent pas la classe (introduisent des skew-partitions)
- Les blocs ne préservent pas être de Berge (introduisent des cyles impairs)
- Les blocs ne sont pas tous compatibles avec les classes de bases.

Complications techniques

Voici l'idée générale, mais il y a des complications :

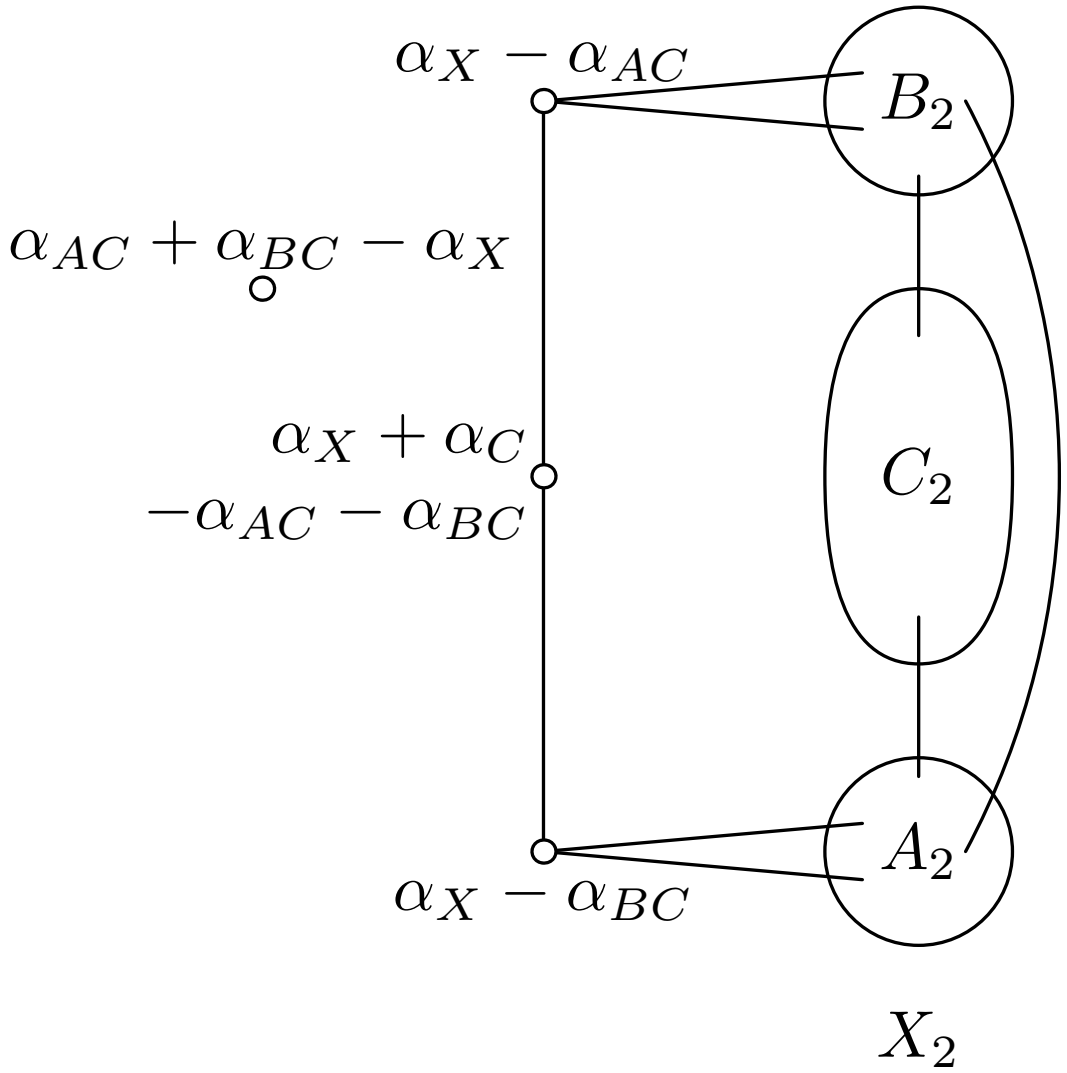
- Les blocs ne préservent pas la classe (introduisent des skew-partitions)
- Les blocs ne préservent pas être de Berge (introduisent des cyles impairs)
- Les blocs ne sont pas tous compatibles avec les classes de bases.
- Rien ne garanti que le bloc ne va pas disparaitre lors d'une décomposition future.

Complications techniques

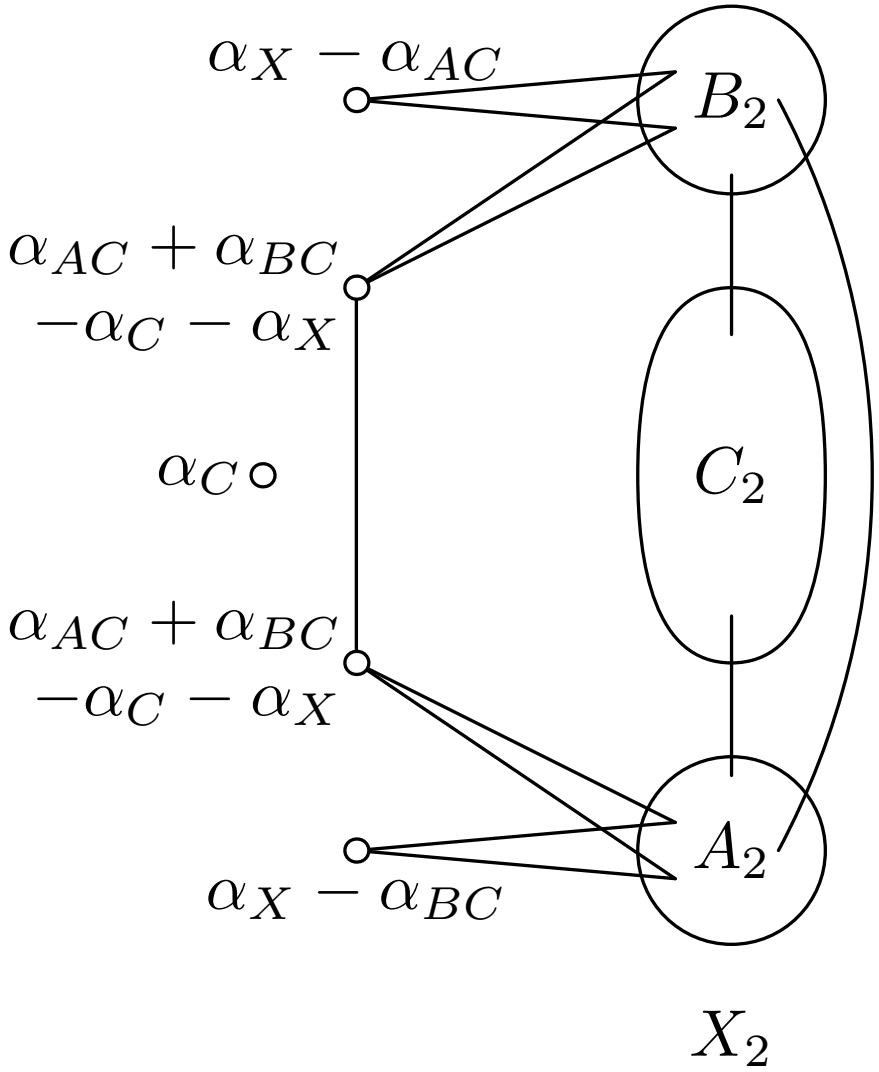
Voici l'idée générale, mais il y a des complications :

- Les blocs ne préservent pas la classe (introduisent des skew-partitions)
- Les blocs ne préservent pas être de Berge (introduisent des cyles impairs)
- Les blocs ne sont pas tous compatibles avec les classes de bases.
- Rien ne garanti que le bloc ne va pas disparaitre lors d'une décomposition future.
- Les poids sont incohérents suivant la parité du 2-joint.

Calculer α avec les 2-joints pairs



Calculer α avec les 2-joints impairs



Magic Lemma

Lemme

Soit (X_1, X_2) *un 2-joint.*

- *si* (X_1, X_2) *est pair alors* $\alpha_X + \alpha_C \geq \alpha_{AC} + \alpha_{BC}$
- *si* (X_1, X_2) *est impair alors* $\alpha_X + \alpha_C \leq \alpha_{AC} + \alpha_{BC}$

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti
- Y_A (Y_B) les sommets connexes à A (B) dans $C \cup D$

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti
- Y_A (Y_B) les sommets connexes à A (B) dans $C \cup D$
- P plus court chemin de A à B dans $C \cup D$

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti
- Y_A (Y_B) les sommets connexes à A (B) dans $C \cup D$
- P plus court chemin de A à B dans $C \cup D$
- P est pair absurde

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti
- Y_A (Y_B) les sommets connexes à A (B) dans $C \cup D$
- P plus court chemin de A à B dans $C \cup D$
- P est pair absurde
- $Y_A \cap Y_B = \emptyset$ et Y_A anticomplet à Y_B

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti
- Y_A (Y_B) les sommets connexes à A (B) dans $C \cup D$
- P plus court chemin de A à B dans $C \cup D$
- P est pair absurde
- $Y_A \cap Y_B = \emptyset$ et Y_A anticomplet à Y_B
- $Z_A = (D \cap Y_A) \cup (C \cap Y_B) \cup (C \setminus (Y_A \cup Y_B))$

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

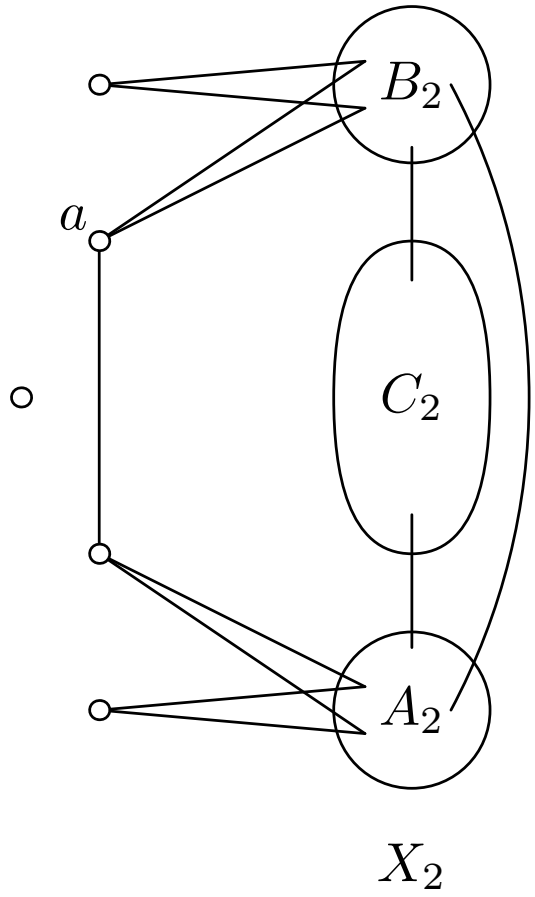
- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti
- Y_A (Y_B) les sommets connexes à A (B) dans $C \cup D$
- P plus court chemin de A à B dans $C \cup D$
- P est pair absurde
- $Y_A \cap Y_B = \emptyset$ et Y_A anticomplet à Y_B
- $Z_A = (D \cap Y_A) \cup (C \cap Y_B) \cup (C \setminus (Y_A \cup Y_B))$
- $Z_B = (D \cap Y_B) \cup (C \cap Y_A) \cup (D \setminus (Y_A \cup Y_B))$

2-joint impair, $\alpha_C + \alpha_X \leq \alpha_{AC} + \alpha_{BC}$

Démonstration.

- D (C) stable max de X_1 (C_1)
- $C \cup D$ est biparti
- Y_A (Y_B) les sommets connexes à A (B) dans $C \cup D$
- P plus court chemin de A à B dans $C \cup D$
- P est pair absurde
- $Y_A \cap Y_B = \emptyset$ et Y_A anticomplet à Y_B
- $Z_A = (D \cap Y_A) \cup (C \cap Y_B) \cup (C \setminus (Y_A \cup Y_B))$
- $Z_B = (D \cap Y_B) \cup (C \cap Y_A) \cup (D \setminus (Y_A \cup Y_B))$
- Z_A stable de $A_1 \cup C_1$, Z_B stable de $B_1 \cup C_1$

Rester dans la classe



- Si un graphe de Berge admet un star cutset, il admet une BSP
- a est le centre d'un star cutset
- On ne reste pas dans la classe

Arêtes optionnelles

On procède en deux temps

- On utilise des blocs de décompositions simples qui préservent la classe.

Arêtes optionnelles

On procède en deux temps

- On utilise des blocs de décompositions simples qui préservent la classe.
- Une fois qu'on a complètement décomposé on étend les blocs avec les informations de construction pour faire le calcul.

Arêtes optionnelles

On procède en deux temps

- On utilise des blocs de décompositions simples qui préservent la classe.
- Une fois qu'on a complètement décomposé on étend les blocs avec les informations de construction pour faire le calcul.
- Lors de l'expansion on connait le type du graphe de base on peut alors choisir une expansion qui préserve la classe de base.

Arêtes optionnelles

On procède en deux temps

- On utilise des blocs de décompositions simples qui préservent la classe.
- Une fois qu'on a complètement décomposé on étend les blocs avec les informations de construction pour faire le calcul.
- Lors de l'expansion on connait le type du graphe de base on peut alors choisir une expansion qui préserve la classe de base.

Arêtes optionnelles

On procède en deux temps

- On utilise des blocs de décompositions simples qui préservent la classe.
- Une fois qu'on a complètement décomposé on étend les blocs avec les informations de construction pour faire le calcul.
- Lors de l'expansion on connait le type du graphe de base on peut alors choisir une expansion qui préserve la classe de base.

Attention à ne pas casser les blocs de décomposition. Pour cela on utilise des *arêtes optionnelles* qui ont deux gros avantages :

- On encode bien les informations d'adjacences floues

Arêtes optionnelles

On procède en deux temps

- On utilise des blocs de décompositions simples qui préservent la classe.
- Une fois qu'on a complètement décomposé on étend les blocs avec les informations de construction pour faire le calcul.
- Lors de l'expansion on connait le type du graphe de base on peut alors choisir une expansion qui préserve la classe de base.

Attention à ne pas casser les blocs de décomposition. Pour cela on utilise des *arêtes optionnelles* qui ont deux gros avantages :

- On encode bien les informations d'adjacences floues
- Les décompositions utilisent des vraies arêtes et ne risquent pas de décomposer les blocs.

Arêtes optionnelles

On procède en deux temps

- On utilise des blocs de décompositions simples qui préservent la classe.
- Une fois qu'on a complètement décomposé on étend les blocs avec les informations de construction pour faire le calcul.
- Lors de l'expansion on connait le type du graphe de base on peut alors choisir une expansion qui préserve la classe de base.

Attention à ne pas casser les blocs de décomposition. Pour cela on utilise des *arêtes optionnelles* qui ont deux gros avantages :

- On encode bien les informations d'adjacences floues
- Les décompositions utilisent des vraies arêtes et ne risquent pas de décomposer les blocs.

Arêtes optionnelles

On procède en deux temps

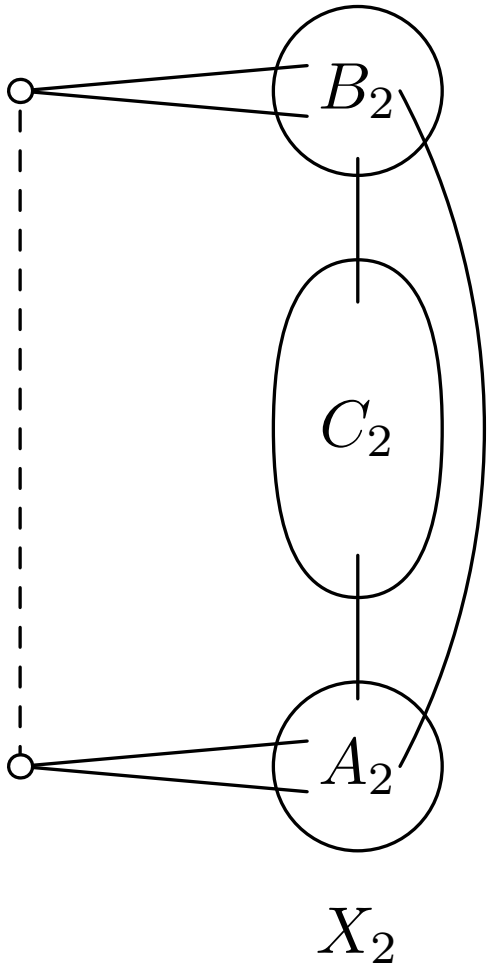
- On utilise des blocs de décompositions simples qui préservent la classe.
- Une fois qu'on a complètement décomposé on étend les blocs avec les informations de construction pour faire le calcul.
- Lors de l'expansion on connait le type du graphe de base on peut alors choisir une expansion qui préserve la classe de base.

Attention à ne pas casser les blocs de décomposition. Pour cela on utilise des *arêtes optionnelles* qui ont deux gros avantages :

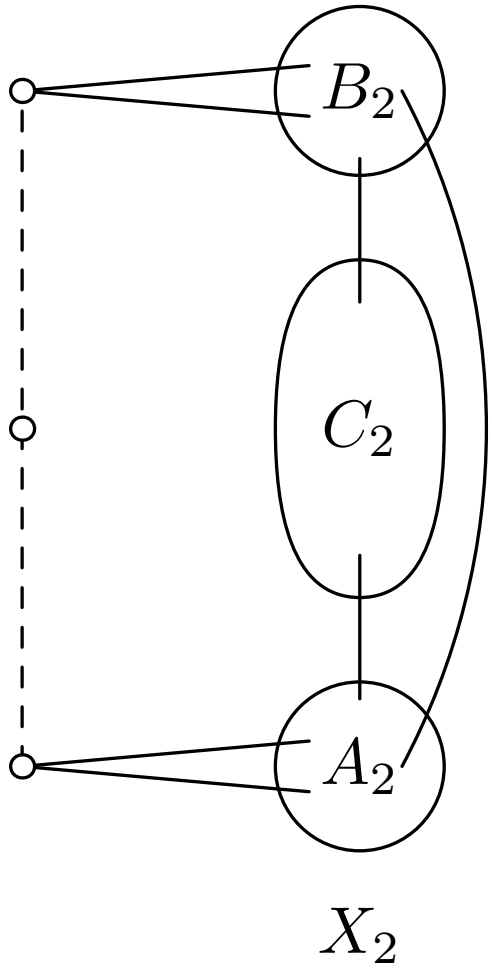
- On encode bien les informations d'adjacences floues
- Les décompositions utilisent des vraies arêtes et ne risquent pas de décomposer les blocs.

La sémantique des arêtes optionnelles est la suivante : une propriété $\mathcal{P}$ est vraie sur un trigraphe T, si pour toute transformation des arêtes optionnelles en arête ou non-arête (le choix est local) la propriété est vraie sur la réalisation G et T.

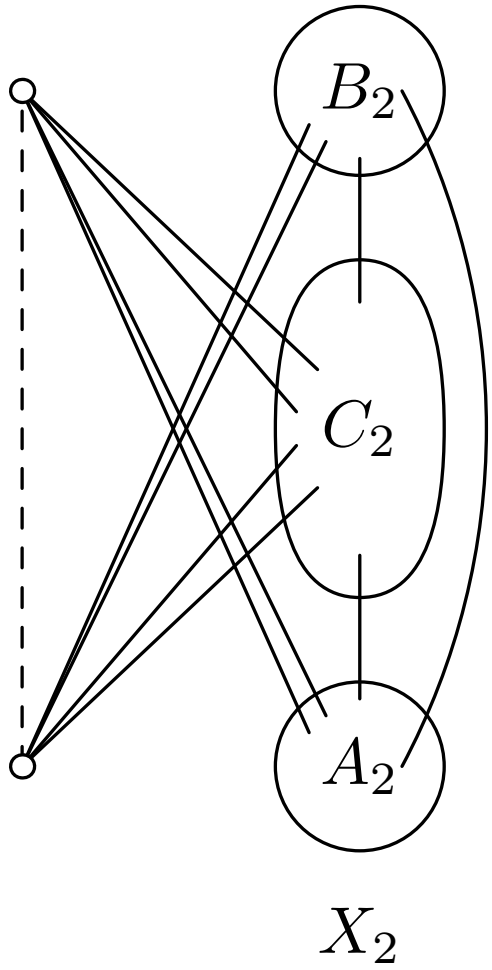
Blocs de décomposition, 2-join impair



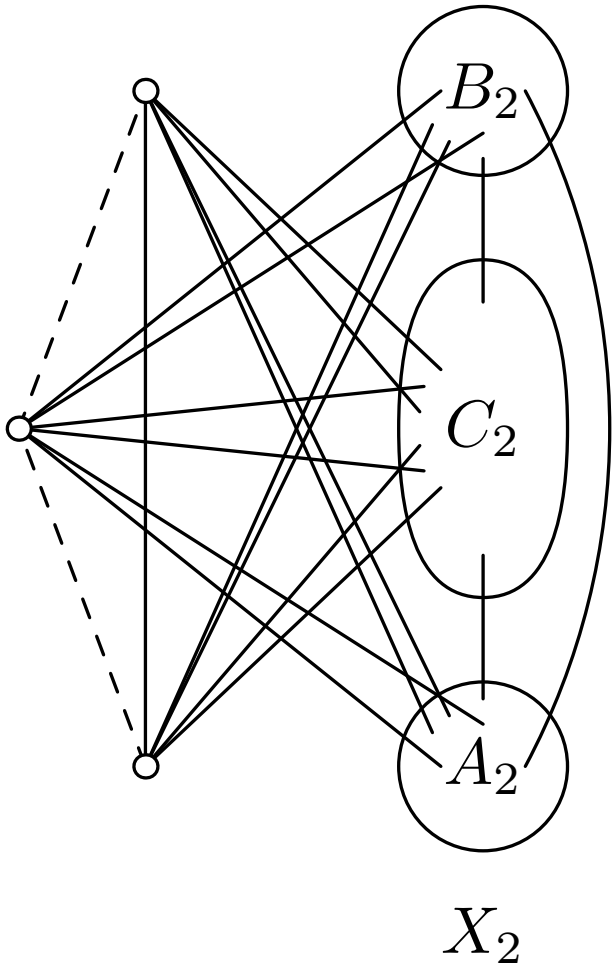
Blocs de décomposition, 2-join pair



Blocs de décomposition, $\overline{\text{2-join}}$ impair

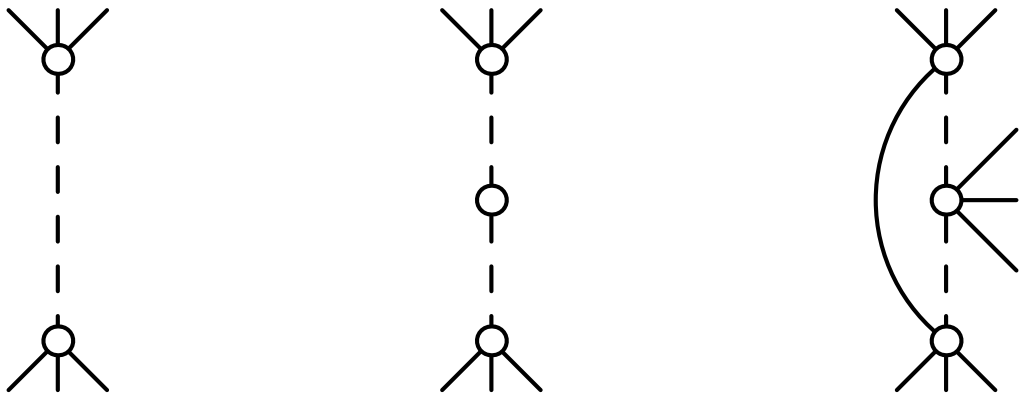


Blocs de décomposition, $\overline{\text{2-join}}$ pair



Trigraphe bigame

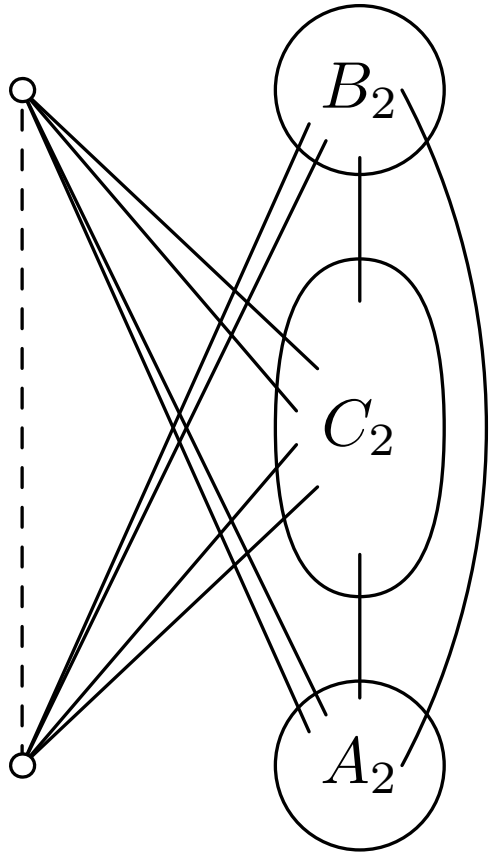
On travaille sur des trigraphes bigames.



Le théorème de décomposition a en fait été prouvé sur les trigraphe monogame, mais on peut l'adapter (induction sur le nombre de sommet de degré optionnel 2).

Rester dans la classe

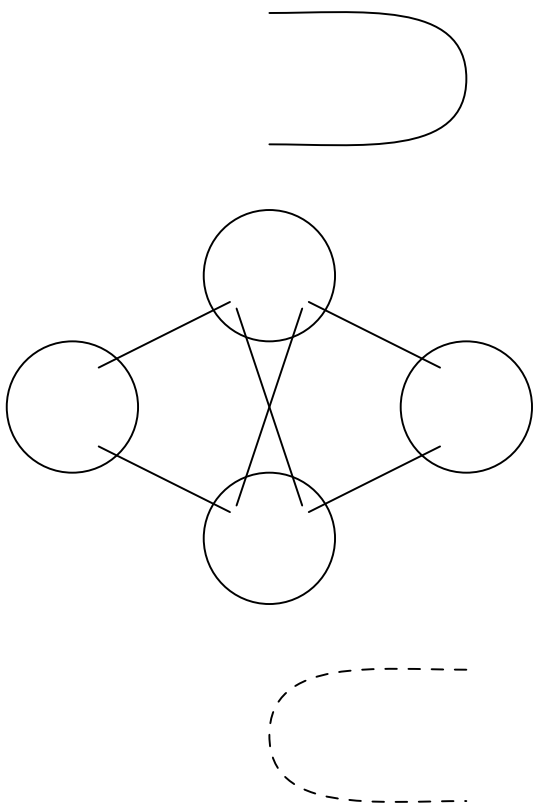
Il est clair que nos blocs préservent être bigame.

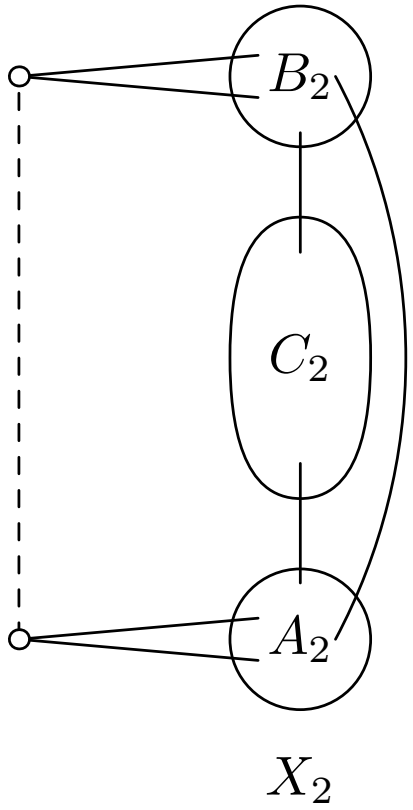


X_2

Ils restent également de Berge (on peut remplacer a et b par des sommets de A_1 et B_1).

Pas de skew-partition





Si on a une skew-partition avec deux ensembles équilibrés, alors il existe une skew-partition équilibrée.

Résumé

Algorithme

- **Tant-que** le trigraphe n'est pas basique, trouver une décomposition.

Résumé

Algorithme

- **Tant-que** le trigraphe n'est pas basique, trouver une décomposition.
- Calculer toutes les informations du petit coté.

Résumé

Algorithme

- **Tant-que** le trigraphe n'est pas basique, trouver une décomposition.
- Calculer toutes les informations du petit coté.
- Construire le bloc de décomposition étiqueté avec les informations de la décomposition

Résumé

Algorithme

- **Tant-que** le trigraphe n'est pas basique, trouver une décomposition.
- Calculer toutes les informations du petit coté.
- Construire le bloc de décomposition étiqueté avec les informations de la décomposition
- **Fin tant-que**

Résumé

Algorithme

- **Tant-que** le trigraphe n'est pas basique, trouver une décomposition.
- Calculer toutes les informations du petit coté.
- Construire le bloc de décomposition étiqueté avec les informations de la décomposition
- **Fin tant-que**
- On a maintenant un trigraphe basique étiqueté.

Résumé

Algorithme

- **Tant-que** le trigraphe n'est pas basique, trouver une décomposition.
- Calculer toutes les informations du petit coté.
- Construire le bloc de décomposition étiqueté avec les informations de la décomposition
- **Fin tant-que**
- On a maintenant un trigraphe basique étiqueté.
- Étendre chaque arête optionnelle en fonction de son étiquette et du type de trigraphe basique.

Résumé

Algorithme

- **Tant-que** le trigraphe n'est pas basique, trouver une décomposition.
- Calculer toutes les informations du petit coté.
- Construire le bloc de décomposition étiqueté avec les informations de la décomposition
- **Fin tant-que**
- On a maintenant un trigraphe basique étiqueté.
- Étendre chaque arête optionnelle en fonction de son étiquette et du type de trigraphe basique.
- Calculer le stable maximum pondéré sur ce trigraphe.

Décompositions extrêmes

Une décomposition est extrême si un bloc de décomposition est toujours basique.

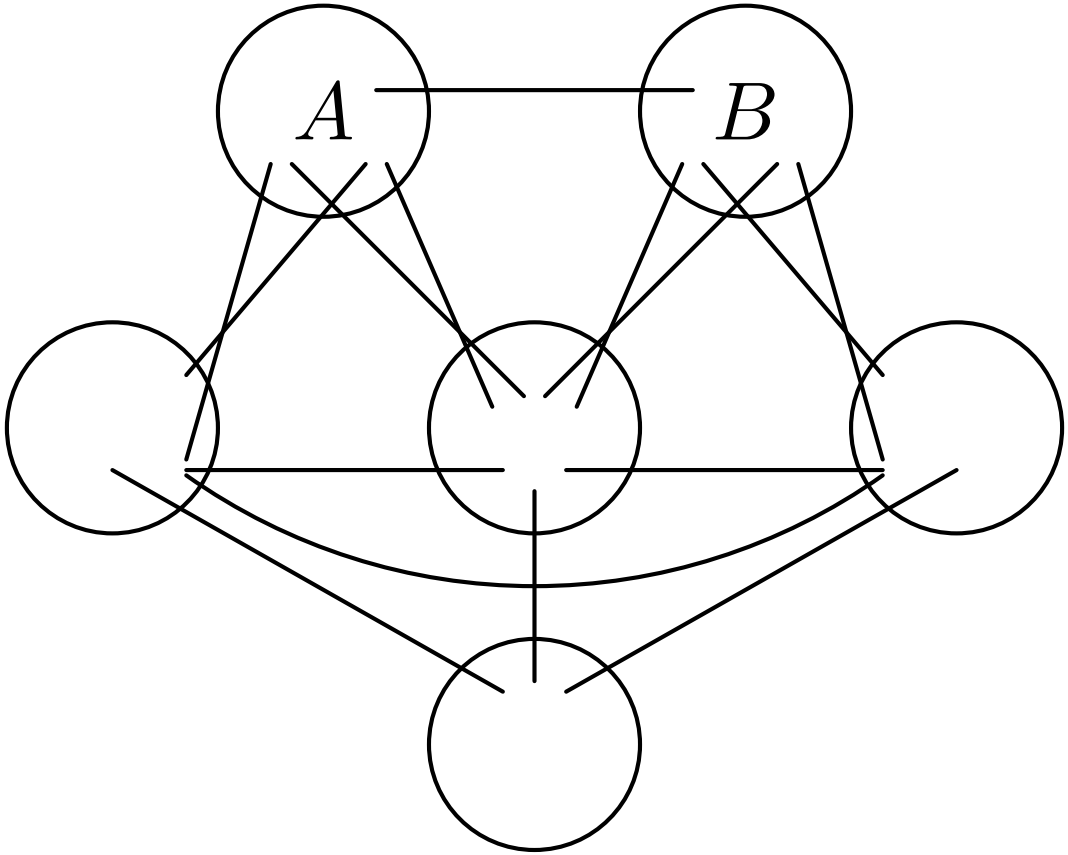
Théorème

Tout trigraphe de Berge bigame sans BSP est ou bien basique, ou admet un 2-joint extrême, un $\overline{2\text{-join}}$ extrême, ou une paire homogène extrême.

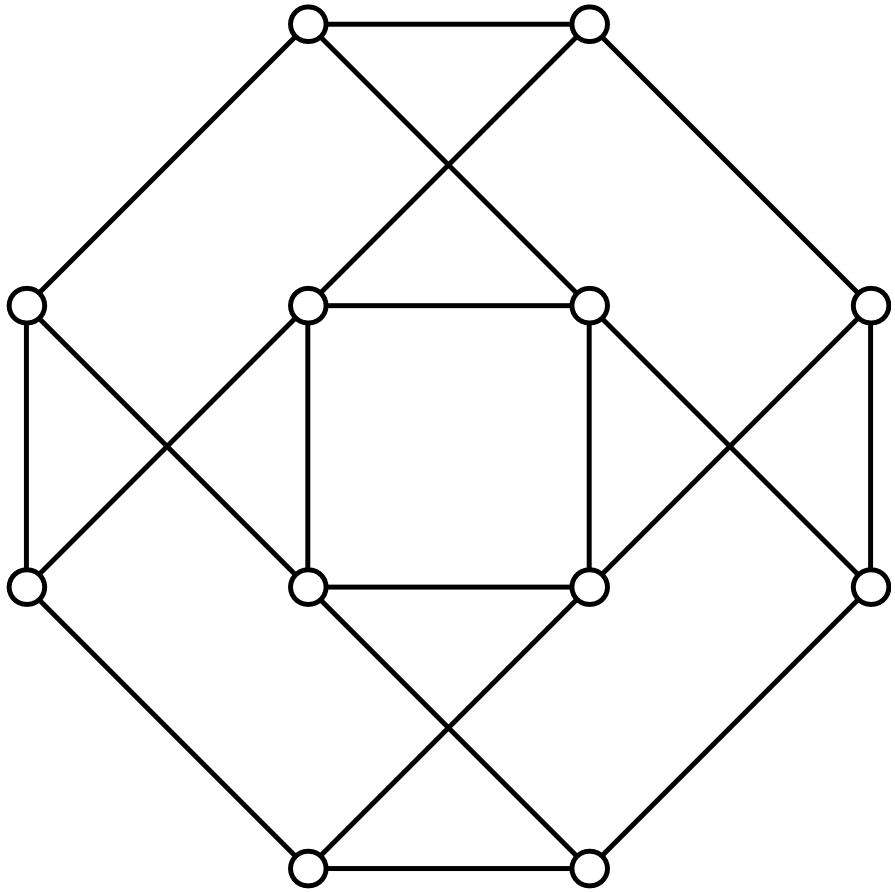
Théorème

Nous avons un algorithme en temps $\mathcal{O}(n^5)$ pour trouver une décomposition extrême.

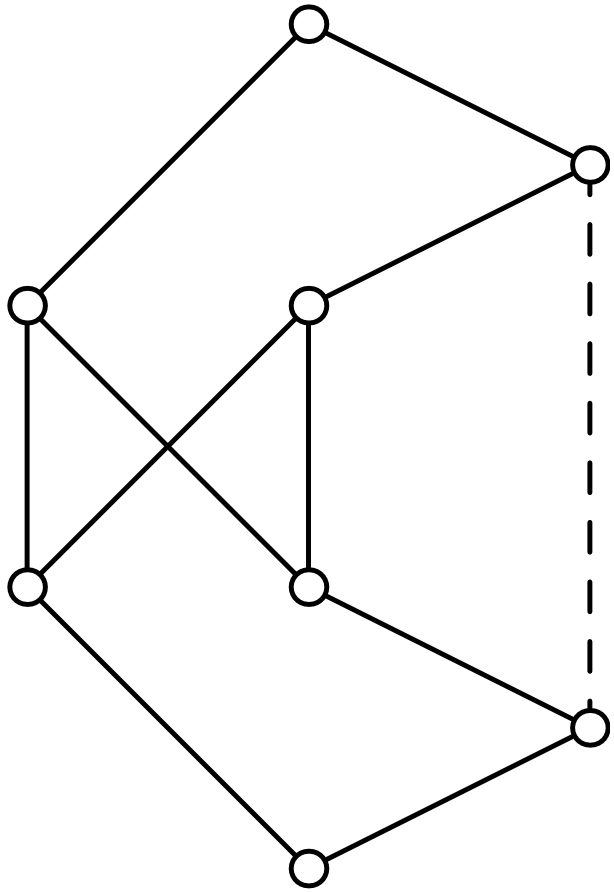
Paire Homogène



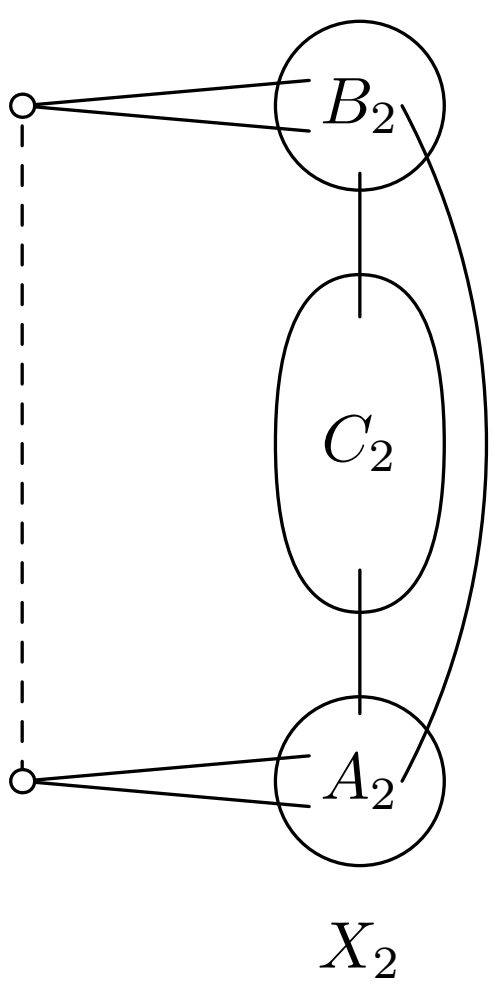
Intersection de 2-joint

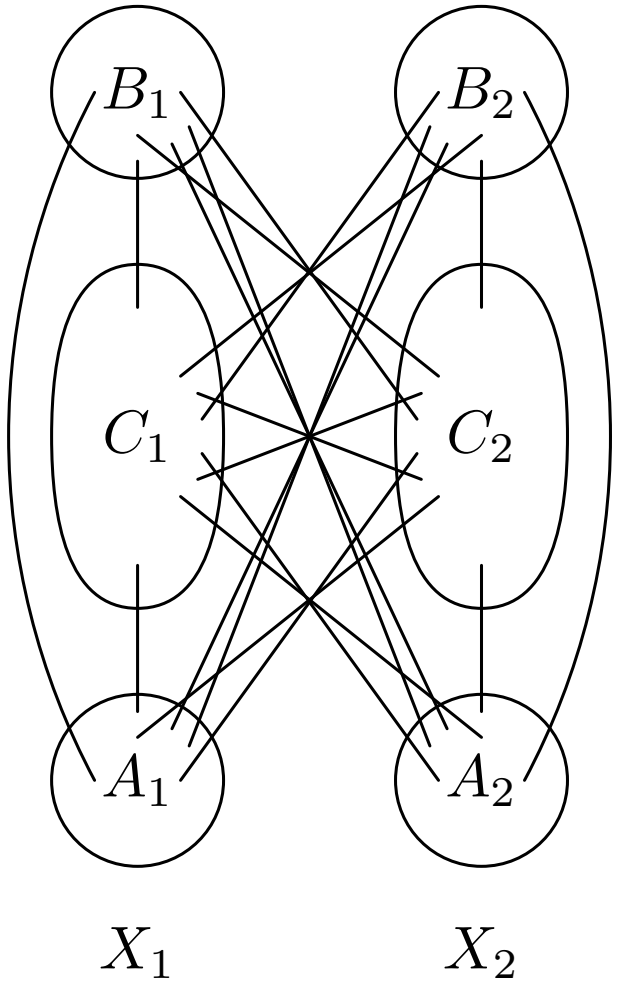


Intersection de 2-joint



Utilité de la paire homogène





C_1 et C_2' sont non vides.

Conclusion

Notre méthode utilise des choses spécifiques aux graphes de Berge, mais :

- L'idée des arêtes optionnelles simplifie énormément l'encodage des adjacences floues et préserve naturellement les blocs.
- Notre décomposition à d'autres applications que calculer le stable (clique-stable séparation, Erdös-Hajnal fort, 2-clique coloration)
- Dans certains cas ces applications peuvent s'étendre aux classes closes par k-joint généralisé.
- Ajouter des décompositions peut donner des décompositions extrêmes

Merci !