Méthodes d’optimisation

BUT Info 2e année

Florent Foucaud
Dipayan Chakraborty, Malika More, Adrien Wohrer

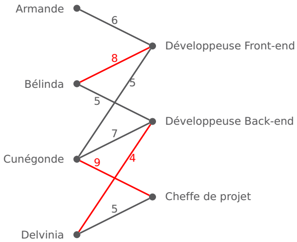
IUT CLERMONT AUVERGNE
Aurillac - Clermont-Ferrand - Le Puy-en-Velay
Montluçon - Moulins - Vichy

2022-2023

Couplage dans un graphe biparti

Problème : sélectionner un ensemble d'arêtes d'un graphe biparti, de telle façon que chaque sommet soit touché au plus une fois.

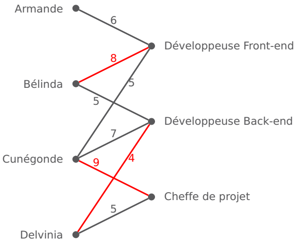
Objectif : maximiser le poids total du couplage



Couplage dans un graphe biparti

Problème : sélectionner un ensemble d'arêtes d'un graphe biparti, de telle façon que chaque sommet soit touché au plus une fois.

Objectif : maximiser le poids total du couplage



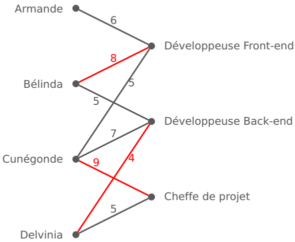
On écrit le PLNE suivant ($x_e = 1$ si l'arête e est sélectionnée, sinon $x_e = 0$) :

$$
\begin{array}{lllll}
\text{maximiser :} & \sum_{e \in E} p_e x_e & & & \\
\text{tel que :} & \displaystyle\sum_{e=uv \in E} x_e & \leq & 1 & \forall v \in V \\
 & x_e & \leq & 1 & \forall e \in E \\
 & x_e & \geq & 0 & \forall e \in E \\
 & x_e & \in & \mathbb{N} & \forall e \in E
\end{array}
$$

Couplage dans un graphe biparti

Problème : sélectionner un ensemble d'arêtes d'un graphe biparti, de telle façon que chaque sommet soit touché au plus une fois.

Objectif : maximiser le poids total du couplage



On écrit le PLNE suivant ($x_e = 1$ si l'arête e est sélectionnée, sinon $x_e = 0$) :

$$
\begin{array}{llll}
\text{maximiser :} & \sum_{e \in E} p_e x_e & & \\
\text{tel que :} & \displaystyle\sum_{e=uv \in E} x_e \leq 1 & \forall v \in V \\
& x_e \leq 1 & \forall e \in E \\
& x_e \geq 0 & \forall e \in E \\
& x_e \in \mathbb{N} & \forall e \in E
\end{array}
$$

Théorème

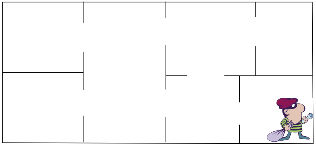
Pour un graphe biparti, les sommets du polytope associé à la relaxation linéaire du PLNE du couplage ont tous des coordonnées entières.

Donc, le simplexe va trouver "gratuitement" la solution optimale du PLNE !

Surveillance d’un bâtiment : couverture par sommets

Problème : couvrir les couloirs d’un bâtiment avec des détecteurs de mouvement.

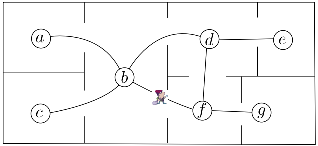
Objectif : minimiser le nombre de détecteurs



Surveillance d’un bâtiment : couverture par sommets

Problème : couvrir les couloirs d’un bâtiment avec des détecteurs de mouvement.

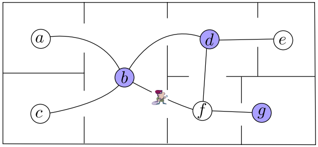
Objectif : minimiser le nombre de détecteurs



Surveillance d’un bâtiment : couverture par sommets

Problème : couvrir les couloirs d’un bâtiment avec des détecteurs de mouvement.

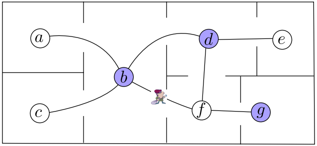
Objectif : minimiser le nombre de détecteurs



Surveillance d’un bâtiment : couverture par sommets

Problème : couvrir les couloirs d’un bâtiment avec des détecteurs de mouvement.

Objectif : minimiser le nombre de détecteurs



Le bâtiment est un graphe non-orienté $G = (V, E)$.

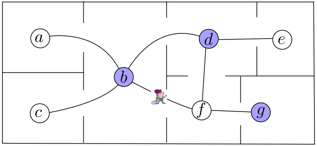
On écrit le PL en nombres entiers suivant :

Une variable x_v pour chaque sommet v : $x_v = 1$ si on a un détecteur sur v, 0 sinon.

Surveillance d'un bâtiment : couverture par sommets

Problème : couvrir les couloirs d'un bâtiment avec des détecteurs de mouvement.

Objectif : minimiser le nombre de détecteurs



Le bâtiment est un graphe non-orienté $G = (V, E)$.

On écrit le PL en nombres entiers suivant :

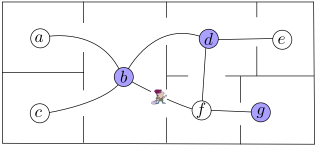
Une variable x_v pour chaque sommet v : $x_v = 1$ si on a un détecteur sur v, 0 sinon.

$$
\begin{array}{lrcl}
\text{minimiser :} & \multicolumn{3}{l}{x_a + x_b + x_c + x_d + x_e + x_f + x_g} \\
\text{tel que :} & x_a + x_b & \geq & 1 \\
 & x_b + x_c & \geq & 1 \\
 & x_b + x_d & \geq & 1 \\
 & x_b + x_f & \geq & 1 \\
 & x_d + x_f & \geq & 1 \\
 & x_d + x_e & \geq & 1 \\
 & x_f + x_g & \geq & 1 \\
 & x_a, \ldots, x_g & \leq & 1 \\
 & x_a, \ldots, x_g & \geq & 0 \\
 & x_a, \ldots, x_g & \in & \mathbb{N}
\end{array}
$$

Surveillance d'un bâtiment : couverture par sommets

Problème : couvrir les couloirs d'un bâtiment avec des détecteurs de mouvement.

Objectif : minimiser le nombre de détecteurs



Le bâtiment est un graphe non-orienté $G = (V, E)$.

On écrit le PL en nombres entiers suivant :

Une variable x_v pour chaque sommet v : $x_v = 1$ si on a un détecteur sur v, 0 sinon.

$$\begin{array}{lrcll} \text{minimiser :} & \sum_{v \in V} x_v & & & \\ \text{tel que :} & x_u + x_v & \geq & 1 & \forall uv \in E \\ & x_v & \leq & 1 & \forall v \in V \\ & x_v & \geq & 0 & \forall v \in V \\ & x_v & \in & \mathbb{N} & \forall v \in V \end{array}$$

Relaxation linéaire et algorithme d'approximation

Méthode 1 : résoudre le PLNE exactement par la méthode “brancher et borner”

Relaxation linéaire et algorithme d'approximation

Méthode 1 : résoudre le PLNE exactement par la méthode “brancher et borner”

Méthode 2 : résoudre la relaxation linéaire (PL normal, pas de nombres entiers)

$$
\begin{array}{lrcll}
\text{minimiser :} & \sum_{v \in V} x_v & & & \\
\text{tel que :} & x_u + x_v & \geq & 1 & \forall uv \in E \\
 & x_v & \leq & 1 & \forall v \in V \\
 & x_v & \geq & 0 & \forall v \in V
\end{array}
$$

Relaxation linéaire et algorithme d'approximation

Méthode 1 : résoudre le PLNE exactement par la méthode "brancher et borner"

Méthode 2 : résoudre la relaxation linéaire (PL normal, pas de nombres entiers)

$$
\begin{array}{lccll}
\text{minimiser :} & \sum_{v \in V} x_v & & & \\
\text{tel que :} & x_u + x_v & \geq & 1 & \forall uv \in E \\
 & x_v & \leq & 1 & \forall v \in V \\
 & x_v & \geq & 0 & \forall v \in V
\end{array}
$$

On construit une solution S en "arrondissant" :

- si $x_v \geq 1/2$, on prend v dans la solution
- si $x_v < 1/2$, on ne prend pas v

Relaxation linéaire et algorithme d'approximation

Méthode 1 : résoudre le PLNE exactement par la méthode "brancher et borner"

Méthode 2 : résoudre la relaxation linéaire (PL normal, pas de nombres entiers)

$$
\begin{array}{llcll}
\text{minimiser :} & \sum_{v \in V} x_v & & & \\
\text{tel que :} & x_u + x_v & \geq & 1 & \forall uv \in E \\
 & x_v & \leq & 1 & \forall v \in V \\
 & x_v & \geq & 0 & \forall v \in V
\end{array}
$$

On construit une solution S en "arrondissant" :

- si $x_v \geq 1/2$, on prend v dans la solution
- si $x_v < 1/2$, on ne prend pas v

Théorème

L'ensemble S obtenu est une solution valide, de taille au plus 2 fois l'optimum.

(On dit qu'il y a un écart d'intégralité d'au plus 2.)

Relaxation linéaire et algorithme d'approximation

Méthode 1 : résoudre le PLNE exactement par la méthode "brancher et borner"

Méthode 2 : résoudre la relaxation linéaire (PL normal, pas de nombres entiers)

$$
\begin{array}{llcll}
\text{minimiser :} & \sum_{v \in V} x_v & & & \\
\text{tel que :} & x_u + x_v & \geq & 1 & \forall uv \in E \\
 & x_v & \leq & 1 & \forall v \in V \\
 & x_v & \geq & 0 & \forall v \in V
\end{array}
$$

On construit une solution S en "arrondissant" :

- si $x_v \geq 1/2$, on prend v dans la solution
- si $x_v < 1/2$, on ne prend pas v

Théorème

L'ensemble S obtenu est une solution valide, de taille au plus 2 fois l'optimum.

(On dit qu'il y a un écart d'intégralité d'au plus 2.)

Démonstration :
1) L'ensemble est bien valide, car pour toute arête uv, soit $x_u \geq 1/2$, soit $x_v \geq 1/2$.

Relaxation linéaire et algorithme d'approximation

Méthode 1 : résoudre le PLNE exactement par la méthode "brancher et borner"

Méthode 2 : résoudre la relaxation linéaire (PL normal, pas de nombres entiers)

$$
\begin{array}{lrcll}
\text{minimiser :} & \sum_{v \in V} x_v & & & \\
\text{tel que :} & x_u + x_v & \geq & 1 & \forall uv \in E \\
& x_v & \leq & 1 & \forall v \in V \\
& x_v & \geq & 0 & \forall v \in V
\end{array}
$$

On construit une solution S en "arrondissant" :

- si $x_v \geq 1/2$, on prend v dans la solution
- si $x_v < 1/2$, on ne prend pas v

Théorème

L'ensemble S obtenu est une solution valide, de taille au plus 2 fois l'optimum.

(On dit qu'il y a un écart d'intégralité d'au plus 2.)

Démonstration :

1) L'ensemble est bien valide, car pour toute arête uv, soit $x_u \geq 1/2$, soit $x_v \geq 1/2$.

2) Le coût total est au plus le nombre N de sommets v tel que $x_v \geq 1/2$.
La solution optimale $S*$ du PL relaxé vaut au moins $N/2$.
Donc la taille de S est au plus $2 \cdot S*$
C'est au plus 2 fois l'optimum du PLNE car on ne peut pas faire mieux que le PL.

Algorithme glouton

Définition (algorithme glouton / greedy algorithm)

C'est un type d'algorithme qui construit une solution pas à pas, en prenant la meilleure décision à chaque étape

→ (optimum local mais pas forcémeent global).

Le Glouton ou Carcajou, un animal du grand nord

("Wolverine" en anglais)

Exemple : arbre couvrant de poids minimum

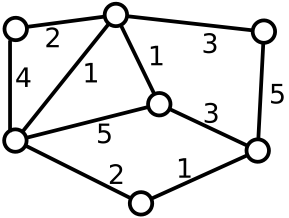
Problème : Construire un arbre couvrant T d'un graphe G (T touche tous les sommets)

Objectif : minimiser le poids total des arêtes de l'arbre

Exemple : arbre couvrant de poids minimum

Problème : Construire un arbre couvrant T d'un graphe G (T touche tous les sommets)

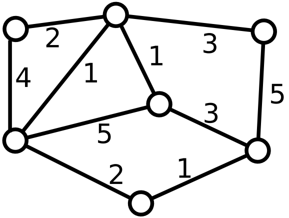
Objectif : minimiser le poids total des arêtes de l'arbre



Exemple : arbre couvrant de poids minimum

Problème : Construire un arbre couvrant T d'un graphe G (T touche tous les sommets)

Objectif : minimiser le poids total des arêtes de l'arbre



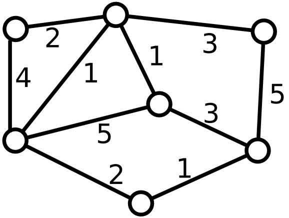
Algorithme de Jarník-Prim :

- Choisir un sommet v_0 de départ ; $T = (\{v_0\}, \{\})$
- Tant que tous les sommets de G ne sont pas dans l'arbre T
  - Choisir un sommet v dans $V(G) \setminus V(T)$ qui a le plus petit coût de rattachement à T
  - $T = T \cup \{v\}$
- Renvoyer T

Exemple : arbre couvrant de poids minimum

Problème : Construire un arbre couvrant T d'un graphe G (T touche tous les sommets)

Objectif : minimiser le poids total des arêtes de l'arbre



Algorithme de Jarník-Prim :

- Choisir un sommet v_0 de départ ; $T = (\{v_0\}, \{\})$
- Tant que tous les sommets de G ne sont pas dans l'arbre T
  - Choisir un sommet v dans $V(G) \setminus V(T)$ qui a le plus petit coût de rattachement à T
  - $T = T \cup \{v\}$
- Renvoyer T

Vojtěch Jarník
(1897-1970)

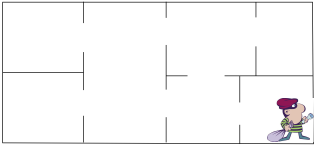
Robert C. Prim
(1921-)

Edsger W. Dijkstra
(1930-2002)

Surveillance d’un bâtiment : couverture par sommets

Problème : couvrir les couloirs d’un bâtiment avec des détecteurs de mouvement. Pour un sommet v, poser un détecteur coûte $c(v)$.

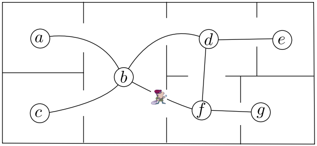
Objectif : minimiser le coût total de la pose des détecteurs



Surveillance d'un bâtiment : couverture par sommets

Problème : couvrir les couloirs d'un bâtiment avec des détecteurs de mouvement.
Pour un sommet v, poser un détecteur coûte $c(v)$.

Objectif : minimiser le coût total de la pose des détecteurs

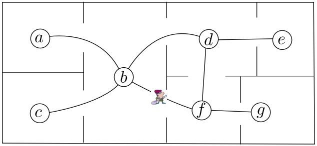


Le bâtiment est un graphe non-orienté $G = (V, E)$.

Surveillance d'un bâtiment : couverture par sommets

Problème : couvrir les couloirs d'un bâtiment avec des détecteurs de mouvement.
Pour un sommet v, poser un détecteur coûte $c(v)$.

Objectif : minimiser le coût total de la pose des détecteurs



Le bâtiment est un graphe non-orienté $G = (V, E)$.

On écrit le PL en nombres entiers suivant :

Une variable x_v pour chaque sommet v : $x_v = 1$ si on a un détecteur sur v, 0 sinon.
Chaque sommet v a un poids $c(v)$ (le coût de la pose d'un détecteur sur v).

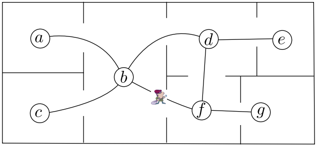
$$
\begin{array}{lrcll}
\text{minimiser :} & \sum_{v \in V} x_v c(v) & & & \\
\text{tel que :} & x_u + x_v & \geq & 1 & \forall uv \in E \\
 & x_v & \leq & 1 & \forall v \in V \\
 & x_v & \geq & 0 & \forall v \in V \\
 & x_v & \in & \mathbb{N} & \forall v \in V
\end{array}
$$

On obtient une solution approchée (au pire deux fois moins bonne que l'optimum) en passant par la relaxation linéaire de ce PL. (Voir le cours précédent.)

Surveillance d’un bâtiment : couverture par sommets

Problème : couvrir les couloirs d’un bâtiment avec des détecteurs de mouvement.
Pour un sommet v, poser un détecteur coûte $c(v)$.

Objectif : minimiser le coût total de la pose des détecteurs



Le bâtiment est un graphe non-orienté $G = (V, E)$.

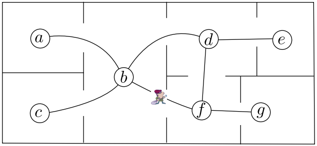
Algorithme glouton n°1 :

- $S = \{\}$
- Tant que toutes les arêtes ne sont pas couvertes :
  - Choisir un sommet v qui touche le plus grand nombre d’arêtes non couvertes par S
  - $S = S \cup \{v\}$
- Renvoyer S

Surveillance d'un bâtiment : couverture par sommets

Problème : couvrir les couloirs d'un bâtiment avec des détecteurs de mouvement.
Pour un sommet v, poser un détecteur coûte $c(v)$.

Objectif : minimiser le coût total de la pose des détecteurs



Le bâtiment est un graphe non-orienté $G = (V, E)$.

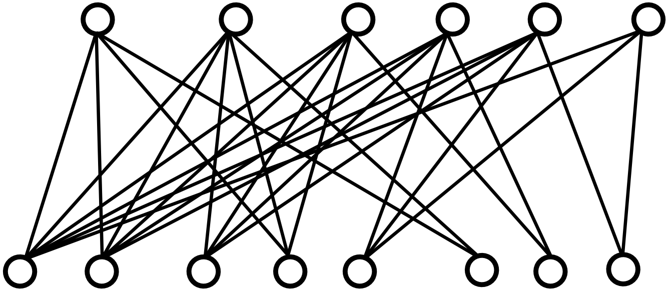
Algorithme glouton n°1 :

- $S = \{\}$
- Tant que toutes les arêtes ne sont pas couvertes :
  - Choisir un sommet v qui touche le plus grand nombre d'arêtes non couvertes par S
  - $S = S \cup \{v\}$
- Renvoyer S

Algorithme glouton n°2 :

- $S = \{\}$
- Tant que toutes les arêtes ne sont pas couvertes :
  - Choisir un sommet v qui maximise le ratio "nombre d'arêtes non couvertes" / $c(v)$
  - $S = S \cup \{v\}$
- Renvoyer S

Aucun des deux algos gloutons n'est bon



(tous les sommets ont le même poids)

Les algorithmes gloutons du transparent précédent n'ont

aucune garantie de performances...