Étude de cas : k-couverture connexe minimum dans les réseaux de capteurs

BATY Léo, BRUNOD-INDRIGO Luca

3 novembre 2020

Table des matières

1 Calcul de bornes inférieures

Dans cette section, on pose le problème sous la forme d'un programme linéaire en nombres entiers (PLNE) et on en déduit des bornes inférieures par relaxation linéaire.

1.1 Notations

Voici tout d'abord les différentes notations qui seront utilisées dans la suite :

- $k \in \{1, 2, 3\}$
- $R^{capt} \leq R^{com}$
- T ensemble des cibles (targets), de cardinal n
  $\rightarrow$ $t \in T$, coordonnées (x_t, y_t)
  $\rightarrow$ $t, t' \in T, t \neq t', \Delta_{t,t'} = \sqrt{(x_t - x_{t'})^2 + (y_t - y_{t'})^2}$
- puit s de coordonnées (x_s, y_s)
- $E^{capt} = \{(t, t') \in T^2 \mid t \neq t', \Delta_{t,t'} \leq R^{capt}\}$
  $\rightarrow$ $E_t^{capt} = \{t' \in T \mid (t, t') \in E^{capt}\}$
  $\rightarrow$ **Graphe de captation** : $\boxed{G^{capt} = (T, E^{capt})}$
- $E^{com} = \{(t, t') \in (T \cup \{s\})^2 \mid t \neq t', \Delta_{t,t'} \leq R^{com}\}$
  $\rightarrow$ $E_t^{com} = \{t' \in T \mid (t, t') \in E^{com}\}$
  $\rightarrow$ **Graphe de communication** : $\boxed{G^{com} = (T \cup \{s\}, E^{com})}$
  $\rightarrow$ On note $\mathcal{D}(E^{com})$ l'ensemble des arrêtes orientées contenant les (t, t') et (t', t).
- $\forall t \in T$, $\delta_t = \mathbb{1}_{\{\text{capteur sur la cible } t\}}$
- $\forall e \in E^{com}$, $x_e = \mathbb{1}_{\{e \text{ dans l'arbre de communication}\}}$

1.2 Modèle PLNE

Variables de décision :

- $\forall t \in T$, $\delta_t = \mathbb{1}_{\text{capteur sur la cible } t}$
- $\forall e \in E^{com}$, $x_e = \mathbb{1}_{e \text{ dans l'arbre de communication}}$

On note $\delta_s = 1$.
|s|[2] $\delta, x, f \sum_{t \in T} \delta_t$

subject to $\sum_{t' \in E_t^{capt}} \delta_{t'} \geq k, \forall t \in T$

$\sum_{e \in E^{com}} x_e = n,$

$x_e \leq \delta_t, \forall e = (t, t') \in E^{com}$

$x_e \leq \delta_{t'}, \forall e = (t, t') \in E^{com}$

$\sum_{t' \in E_t^{com}} y_{(t',t)} - \sum_{t' \in E_t^{com}} y_{(t,t')} = \delta_t, \forall t \in T$

$\sum_{t \in E_s^{com}} y_{(s,t)} - \sum_{t \in E_s^{com}} y_{(t,s)} = \sum_{t \in T} \delta_t,$

$y_{(t,t')} \leq n \times x_e, \forall e = (t,t') \in E^{com}$
$y_{(t',t)} \leq n \times x_e, \forall e = (t,t') \in E^{com}$
$\delta_t \in \{0,1\}, \forall t \in T$
$x_e \in \{0,1\}, \forall e \in E^{com}$
$y_a \geq 0, \forall a \in \mathcal{D}(E^{com})$

- k-connexité : contrainte **??**

1.3 Relaxation linéaire

2 Heuristiques

2.1 heuristique 1

Algorithm 1 Heuristique par captations de coût minimal successives

Input :
$(T, G^{capt}, G^{com}, k)$: une instance du problème
L : une liste contenant k fois chaque cible de T dans un ordre quelconque

Output : $N \subset T$: le sous-ensemble de cibles sur lesquelles placer des capteurs

$N = \emptyset$
for all $t \in L$ **do**
  Choisir t' tel que $(t,t') \in E^{capt}$ et qui minimise le plus court chemin vers $s \cup N$ dans G^{com}
  $E^{capt} = E^{capt} \setminus (t,t')$
  Ajouter à N les cibles situées sur le plus court chemin de t' à $s \cup N$ dans G^{com}
end for
Renvoyer N

2.2 heuristique 2

Algorithm 2 Heuristique gloutonne avec relation d'ordre sur les cibles

Input :
$(T, G^{capt}, G^{com}, k)$: une instance du problème
$\prec$: une relation d'ordre sur les cibles

Output : $N \subset T$: le sous-ensemble de cibles sur lesquelles placer des capteurs

$N = \emptyset$
$Q = \{t \in T | (s, t) \in E^{com}\}$
$U = \emptyset$
while Toutes les cibles n'ont pas au moins k voisins appartenant à N dans G^{capt} **do**
  if $Q \setminus U \neq \emptyset$ **then**
    Choisir t dans $Q \setminus U$ maximal pour la relation d'ordre $\prec$
    if t a au moins un voisin dans G^{capt} ayant moins de k voisins dans N **then**
      $Q = Q \setminus \{t\}$
      $Q = Q \cup \{t' | (t, t') \in E^{com}\}$
      $N = N \cup \{t\}$
    else
      $U = U \cup \{t\}$
    end if
  else
    Choisir t dans $Q \cap U$ maximal pour la relation d'ordre $\prec$
    $Q = Q \setminus \{t\}$
    $Q = Q \cup \{t' | (t, t') \in E^{com}\}$
    $N = N \cup \{t\}$
  end if
end while
Renvoyer N

3 Métaheuristiques

3.1 Structure de voisinage

3.2 Recuit simulé

3.3 Recherche à voisinages multiples