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)
- $E^{capt} = \{(t, t') \in T^2 \mid t \neq t', \Delta_{t,t'} \leq R^{capt}\}$
  - $\rightarrow$ $E^{capt}_t = \{t' \in T | \ (t, t') \in E^{capt}\}$
  - $\rightarrow$ **Graphe de captation** : $\boxed{G^{capt} = (T, E^{capt})}$
- $E^{com} = \{(t, t') \in (T \cup \{s\})^2 | \ t \neq t', \Delta_{t,t'} \leq R^{com}\}$
  - $\rightarrow$ $E^{com}_t = \{t' \in T | \ (t, t') \in E^{com}\}$
  - $\rightarrow$ **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

On décrit une solution à l'aide des variables de décision suivantes :

- $\forall t \in T$, $\delta_t = \mathbb{1}_{\{\text{capteur sur la cible } t\}}$
- $\forall e \in E^{com}$, $x_e = \mathbb{1}_{\{e \text{ dans l'arbre de communication}\}}$ (L'arbre de communication est un arbre couvrant toutes les cibles possédant un capteur dans le graphe de communication G^{capt})
- $\forall a \in \mathcal{D}(E^{com})$, $y_a \in \mathbb{R}_+$: on définit un flot $\mathcal{D}(E^{com})$, qui part de s, de valeur égale au nombre de capteurs placés, et dont chaque cibles possédant un capteur réceptionne une unité de ce flot.

Afin que les contraintes définies ci-après soient valables aussi pour la source s on fixe $\delta_s = 1$:

- On minimise le nombre de capteurs placés : (1a)
- k-connexité : (1b)
- nombre d'arêtes de l'arbre fixé à n :(1c)
- Une arête n'appartient pas à l'arbre couvrant si au moins une de ses deux extrémités ne possède pas de capteur : (1d), et (1e)

- Pour chaque cible, la différence entre le flot entrant et le flot sortant vaut 1 si elle possède un capteur, et 0 sinon : (1f)
- Le différence entre le flot sortant et le flot entrant dans la source est égal au nombre de cibles placées : (1g)
- On interdit le flot sur les arcs dont l'arête associées n'est pas dans l'arbre de communication entre les capteurs : (1h) et (1i)

Voici le PLNE complet obtenu :

$$\min_{\delta, x, f} \quad \sum_{t \in T} \delta_t \tag{1a}$$

$$\text{s.t.} \quad \sum_{t' \in E_t^{capt}} \delta_{t'} \geq k, \qquad \forall t \in T \tag{1b}$$

$$\sum_{e \in E^{com}} x_e = n, \tag{1c}$$

$$x_e \leq \delta_t, \qquad \forall e = (t, t') \in E^{com} \tag{1d}$$

$$x_e \leq \delta_{t'}, \qquad \forall e = (t, t') \in E^{com} \tag{1e}$$

$$\sum_{t' \in E_t^{com}} y_{(t',t)} - \sum_{t' \in E_t^{com}} y_{(t,t')} = \delta_t, \qquad \forall t \in T \tag{1f}$$

$$\sum_{t \in E_s^{com}} y_{(s,t)} - \sum_{t \in E_s^{com}} y_{(t,s)} = \sum_{t \in T} \delta_t, \tag{1g}$$

$$y_{(t,t')} \leq n \times x_e, \qquad \forall e = (t, t') \in E^{com} \tag{1h}$$

$$y_{(t',t)} \leq n \times x_e, \qquad \forall e = (t, t') \in E^{com} \tag{1i}$$

$$\delta_t \in \{0, 1\}, \qquad \forall t \in T \tag{1j}$$

$$x_e \in \{0, 1\}, \qquad \forall e \in E^{com} \tag{1k}$$

$$y_a \geq 0, \qquad \forall a \in \mathcal{D}(E^{com}) \tag{1l}$$

1.3 Relaxation linéaire et obtention des bornes inférieures

Nous avons implémenté le PLNE en Julia en utilisant la librairie JuMP (librairie de programmation mathématique) couplé avec le solveur Gurobi (licence académique), afin de résoudre la relaxation linéaire de ce problème, voire la résolution exacte par Branch and Bound pour les plus petites instance. Nous obtenons exactement les mêmes bornes inférieures que celles fournies avec les instances.

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

Afin d'améliorer les solutions obtenues à l'aide de l'heuristique , nous avons mis en oeuvre une métaheuristique de type recuit simulé comme vu en cours, en utilisant le voisinage défini par la seconde heuristique . Nous avons utilisé les paramètres suivant :

nom de l'heuristique

nom de la 2eme heuristique

- Paramètre multiplicateur de **décroissance de la température** : $\phi = 0.95$
- **Nombre d'itérations** par palier de température : number_iterations = 10000

TABLE 1 – Comparaison entre : résultats du recuit simulé | bornes inférieures

$(\mathbf{k}, \mathbf{R_{capt}}, \mathbf{R_{com}})$	**150-7-4**	**225-8-10**	**625-12-100**	**900-15-20**	**1500-15-100**	**1500-18-100**
(1, 1, 1)	31 | 21	45 | 26	89 | 54	137 | 82	138 | 80	198 | 113
(1, 1, 2)	19 | 19	25 | 25	55 | 53	83 | 80	88 | 80	121 | 112
(1, 2, 2)	8 | 6	11 | 8	24 | 16	37 | 23	37 | 23	55 | 32
(1, 3, 3)	6 | 6	8 | 8	18 | 16	27 | 22	28 | 23	39 | 31
(2, 1, 1)	44 | 40	56 | 50	119 | 106	185 | 162	189 | 159	261 | 225
(2, 1, 2)	39 | 39	50 |50	109 | 106	166 | 161	174 159	239 | 225
(2, 2, 2)	12 | 11	16 | 15	34 | 31	52 | 44	53 | 45	73 | 62
(2, 3, 3)	11 | 11	16 | 15	32 | 31	49 | 44	52 | 45	69 | 62
(3, 1, 1)	62 | 60	79 | 78	166 | 162	253 | 246	260 | 239	355 | 341
(3, 1, 2)	60 | 60	78 | 78	166 | 162	251 | 245	257 | 239	355 | 341
(3, 2, 2)	17 | 17	23 | 23	49 | 46	74 | 66	75 | 67	103 | 93
(3, 3, 3)	17 | 17	23 | 23	48 | 46	72 | 66	75 | 67	105 | 93

- **Température initiale** : nous avons fixé un paramètre correspond à la probabilité initiale de garder une modification qui dégrade l'objectif à initial_keep_probability = 0.8. Puis en calculant une moyenne empirique de la valeur de dégradation d'une solution, on en déduit une valeur de la température initiale : $\boxed{T_0 = \frac{-\Delta_{mean}}{\log(\text{initial\_keep\_probability})}}$
- **Température finale** : de manière similaire au calcul la température initiale on a choisi final_keep_probability $= 1e-10$ qui correspond à la probabilité finale d'accepter une solution qui augmente l'objectif de 1 catpeur. On obtient donc $\boxed{T_{final} = \frac{-1}{\log(\text{final\_keep\_probability})}}$

local search

3.3 Recherche à voisinages multiples

explain + results