TP MLA : installer des liaisons dans un réseau

Vous pouvez travailler un binôme. Déposer vos fichiers à la fin de la séance dans une archive ZIP à l'aide de l'interface web : `cedric.cnam.fr/~porumbed/mla/soumettre.html`

1 Présentation théorique : 2 pages qui reprennent les slides du cours en détail

On s'intéresse à un problème qui demande d'installer des liaisons (câbles Ethernet, fibres optiques, etc.) sur les arêtes d'un graphe $G = (V, E)$ dans le but d'assurer la transmission de données depuis une source s vers un ensemble de terminaux T ($s \notin T$). L'objectif est de minimiser le coût des liaisons (câbles) installées, en considérant que toutes les liaisons ont le même coût (unitaire) d'installation/construction. Le modèle utilise des variables de décision y_{ij} pour indiquer le nombre de câbles installés entre les sommets i et j. Chaque liaison peut faire transmettre un débit maximal (bande passante) de b_{nd}. Pour chaque couple de sommets i et j, on définit une variable x_{ij} qui représente le flux de données de i à j. Ces dernières variables sont orientées, c-à-d x_{ij} n'est pas nécessairement égal à x_{ji}. Par contre, les variables $\mathbf{y}$ ne sont pas orientées, on peut considérer $y_{ij} = y_{ji}$. Notons d_i la demande d'un terminal $i \in T$. On obtient le modèle suivant :

$$\min \sum_{\{i,j\} \in E} y_{ij} \tag{1a}$$

$$\sum_{\{i,j\} \in E} x_{ji} - \sum_{\{i,j\} \in E} x_{ij} \geq 0, \ \forall i \notin T \cup \{s\} \tag{1b}$$

$$\sum_{\{i,j\} \in E} x_{ji} - \sum_{\{i,j\} \in E} x_{ij} \geq d_i, \ \forall i \in T \tag{1c}$$

$$b_{\text{nd}} y_{ij} - x_{ij} - x_{ji} \geq 0, \ \forall \{i,j\} \in E, \ i < j \tag{1d}$$

$$y_{ij} \in \mathbb{Z}_+, \ x_{ij}, x_{ji} \geq 0, \ \forall \{i,j\} \in E, \ i < j \tag{1e}$$

Les deux premières inégalités sont des contraintes de conservation de flot, légèrement adaptées pour simplifier un dual plus tard.[1] En particulier, (1c) indique qu'un terminal $i \in T$ peut consommer sa demande d_i. Il n'y a pas de contrainte (1b) ou (1c) pour $i = s$, car il n'y a pas de conservation de flot à la source. En effet, les flux x_{sj} avec $\{s, j\} \in E$ peuvent être aussi grands que nécessaire : c'est la bande passante installée qui permet de limiter la quantité de trafic. Ainsi, les contraintes (1d) indiquent que le trafic/flux total sur une arrête $\{i, j\}$ ne peut pas dépasser la quantité de bande passante installée, c-à-d le nombre y_{ij} de câbles installés multiplié par la bande passante b_{nd} de chaque câble. Ce modèle ignore les coûts du trafic/flux, c-à-d, il n'y a pas de coût associé au fait qu'on envoie des données sur un câble, ce qui est réaliste dans un réseau informatique.

La reformulation de Benders avec des coupes de faisabilité

Pour qu'une solution $\mathbf{y}$ soit réalisable dans le programme (1a)-(1e) plus haut, le LP ci-après doit avoir une solution réalisable (pour permettre la transmission du trafic $\mathbf{x}$). Attention, dans ce nouveau LP, on considère $\mathbf{y}$ comme des

1. En effet, on n'utilise pas les contraintes habituelles de conservation de flot (lois de Kirschoff) avec égalité. Tel que qu'il est écrit, les inégalités (1b)-(1c) permettent l'existence de pertes de flot. Cependant, une perte de flot n'apporte rien dans le modèle, car elle pourrait augmenter sans raison le trafic. Il y a toujours une solution optimale sans perte de flot.

paramètres et $\mathbf{x}$ comme des variables de décision.

$$\sum_{\{i,j\}\in E} x_{ji} - \sum_{\{i,j\}\in E} x_{ij} \geq 0,\ \forall i \notin T \cup \{s\} \tag{2a}$$

$$\sum_{\{i,j\}\in E} x_{ji} - \sum_{\{i,j\}\in E} x_{ij} \geq d_i,\ \forall i \in T \tag{2b}$$

$$-x_{ij} - x_{ji} \geq -b_{\text{nd}} y_{ij},\ \forall \{i,j\} \in E,\ i < j \tag{2c}$$

$$x_{ij} \geq 0,\ \forall \{i,j\} \in E \tag{2d}$$

La fonction objectif n'a pas été écrite de manière explicite, parce que tous ses coefficients valent 0 (pas de coût pour la transmission de données). Un tel PL est soit réalisable avec une valeur objectif de 0 soit non réalisable. Et il faut que ce PL soit réalisable pour que $\mathbf{y}$ soit faisable dans (1a)-(1e). Ainsi, son **PL dual doit être borné !** En effet, si le dual d'un PL primal est non-borné, alors la valeur objectif de la solution optimale du PL primal ne peut pas être finie.

Pour écrire le PL dual, on introduit : (i) une variable duale $v_i \geq 0$ pour les sommets $i \notin T \cup \{s\}$ associée à la contrainte (2a), (ii) une variable duale $v_i \geq 0$ pour les sommets $i \in T$ associée à (2b), et (iii) une variable duale $v_{ij} \geq 0$ pour (2c) et toutes les arêtes $\{i,j\}$. Le nombre total de variables duales est : $\Big|V \setminus \big(T \cup \{s\}\big)\Big| + |T| + |E| = |V| + |E| - 1$. On obtient le PL dual suivant :

$$\max \quad -\sum_{\{i,j\}\in E} b_{\text{nd}} y_{ij} v_{ij} + \sum_{i\in T} d_i v_i \tag{3a}$$

$$x_{ij}: \quad -v_{ij} - v_i + v_j \leq 0 \quad \forall \{i,j\} \in E,\ i < j \tag{3b}$$

$$x_{ji}: \quad -v_{ij} + v_i - v_j \leq 0 \quad \forall \{i,j\} \in E,\ i < j \tag{3c}$$

$$\mathbf{v} \geq \mathbf{0}, \tag{3d}$$

où on utilise la convention que si i (resp. j) vaut s alors le terme v_i (resp. v_j) disparait dans (3b)-(3c). Notons que si $\mathbf{v}$ est une solution réalisable de ce PL dual, alors $\alpha\mathbf{v}$ est aussi réalisable, pour $\alpha \geq 0$. C'est pour cette raison que la fonction objectif de ce PL dual **ne doit pas dépasser** 0 pour que ce PL dual soit borné.

Ce PL dual peut être intégrée (voir aussi les diapos du cours) dans le programme Benders ci-dessus, [2], où la contrainte (4b) ne fait que répéter (3a)-(3d) :

$$\min \mathbf{1}^\top \mathbf{y} \tag{4a}$$

$$0 \geq -\sum_{\{i,j\}\in E} b_{\text{nd}} y_{ij} v_{ij} + \sum_{i\in T} d_i v_i, \forall \mathbf{v} \in \underbrace{\Big\{\mathbf{v} \geq \mathbf{0} : -v_{ij} - v_i + v_j \leq 0, -v_{ij} + v_i - v_j \leq 0\Big\}}_{\mathscr{P}} \tag{4b}$$

$$\mathbf{y} \in \mathbb{Z}^n_+ \tag{4c}$$

On n'envisage pas d'écrire toutes les contraintes (4b), car il y a trop de solutions réalisables $\mathbf{v}$. Cependant, il est possible de démarrer avec une version du PL (4a)-(4c) avec un très petit ensemble (même vide) de contraintes (4b). On obtient ainsi un programme maître relâché (avec beaucoup de contraintes enlevées). Pour toute solution $\mathbf{y}$ associé au maître relaxé, l'algorithme de Plans Coupants essaye de trouver une contrainte (4b) violée par $\mathbf{y}$.

Pour trouver une contrainte violée, il faut maximiser

$$\max_{\mathbf{v} \in \mathscr{P}} \quad -\sum_{\{i,j\}\in E} b_{\text{nd}} y_{ij} v_{ij} + \sum_{i\in T} d_i v_i \tag{5}$$

Ce sous-problème trouve une coupe de faisabilité Benders, c-à-d, si la valeur de (5) est supérieure à 0, alors on coupe $\mathbf{y}$. Après avoir ajouté une nouvelle contrainte violée, il faut re-optimiser le maître relaxé pour trouver une nouvelle solution $\mathbf{y}$ et répéter jusqu'à ce que $\mathbf{y}$ devient faisable (le sous-problème ne trouve plus de contrainte violée).

2. Toutes les sommes indexé par les arêtes $\{i,j\} \in E$ utilise la convention $i < j$. À chaque fois qu'on écrit y_{ij} ou v_{ij}, on suppose que $i < j$.

2 TP : Implémenter une décomposition de Benders

Vous pouvez utiliser un langage de votre choix avec `guroby` ou `cplex`. Ce sujet continue en se basant sur le langage `C++`. Si vous utilisez un autre langage, c'est à vous de s'assurer que vous pouvez obtenir les même fonctionnalités.

Exercice 1 Écrire un programme pour résoudre le modèle Benders (4a)-(4c) à l'aide d'un algorithme à base de plan coupants. Étant donné (4c), les variables $\mathbf{y}$ doivent rester entières à chaque itération. Après avoir résolu chaque sous-problème de Benders, on ajoute une nouvelle contrainte (4b) si la valeur de (5) dépasse 0.
Chaque sous-problème demande de résoudre un PL avec $|V| + |E| - 1 = n + m - 1$ variables, l'objectif (5) et les contraintes (3b)-(3d). N'hésitez pas à vous inspirer du code à droite pour démarrer

Des instances sont disponibles à l'adresse ci-dessous. Une instance est représentée par une liste d'arêtes $\oplus$ une liste de demandes ; la source est toujours le sommet `0`.
`http://cedric.cnam.fr/~porumbed/mla/`

```
IloEnv env;
IloModel model(env);
IloNumVarArray v(env, m+n-1, 0, INT_MAX);


IloExpr expr
for(i=0;i<m;i++)
    expr-=bnd*y[i]*v[i]; //m arêtes {i,j}
for(i=m;i<m+n-1;i++)
    expr+=b[i-m+1]*v[i]; //n-1 sommets i,
                //on saute b[0] (la source)
IloObjective obj(env, expr,
                    IloObjective::Maximize);
model.add(obj);
```

Ajout fonction objectif

```
IloRangeArray constr(env);
IloExpr e(env);
for(i=0;i<m+n-1;i++)
     e += v[i];
constr.add(e==1);
```

Contrainte artificielle $\mathbf{1}^\top \mathbf{v} = 1$ pour faire Cplex trouver un sommet et non pas un rayon!

Exercice 2 Modifier le programme précédent pour réaliser l'optimisation en deux étapes :

étape 1 On résout à l'optimalité le programme Benders avec des variables $\mathbf{y}$ fractionnaires, c-à-d, on relâche $y_i \in \mathbb{Z}$ et on résout(4a)-(4b) au lieu de (4a)-(4c)

étape 2 Sans penser à enlever les contraintes trouvées à l'étape 1, on remet la contraintes $y_i \in Z$ (il suffit d'appeler `cutPlanes.turnAllVarsInteger()`) et on fait tourner l'algorithme à base de plans coupants comme à l'exercice précédent.

Question 1 : Quelle méthode nécessite moins d'itérations en nombres entiers pour une bande passante de 1 ?

Question 2 : Quelle méthode nécessite moins d'itérations en nombres entiers pour une bande passante de 3 ?

Exercice 3 Il n'est pas obligatoire de résoudre le problème de départ (1.a)-(1.e) avec la décomposition de Benders. Pour la version avec une bande passante de 1, il suffit de trouver le plus court chemin de la source vers chaque terminal (ex, avec l'algorithme de Dijkstra ou avec un solveur). Toutes les arêtes d'un tel chemin sont utilisées pour construire la solution optimale. Si une arête $\{i, j\}$ fait partie de 7 plus courts chemins, alors on met $y_{ij} = 7$.

question 1 Montrer théoriquement qu'on obtient la solution optimale !

question 2 Comparer la solution trouvée à l'aide de la décomposition de Benders avec la solution trouvée simplement en calculant les chemins les plus courts.