

# Projet 2025 - 2026

## MPRO

Zacharie ALES, Daniel PORUMBEL

## 1 Objectifs du projet

Le but de ce projet est l'étude d'un problème de partitionnement robuste. Vous devrez résoudre ce problème :

1. par un algorithme de plans coupants ;
2. par un algorithme de *branch-and-cut* (via un LazyCallback) ;
3. par dualisation ;
4. par une heuristique.

## 2 Présentation du problème

### 2.1 Problème statique

Soient :

- un graphe non orienté  $G = (V, E)$  ;
  - une distance  $\ell_{ij}$  associée à chaque arête  $ij \in E$  ;
  - un poids  $w_i$  associé à chaque sommet  $i \in V$  ;
  - un nombre maximal de parties  $K \in \{1, \dots, n\}$  ;
  - un poids maximal d'une partie  $B \in \mathbb{N}$ .
- Remarque* : Le poids d'une partie correspond à la somme des poids des sommets qu'elle contient.

Le problème statique consiste à partitionner les sommets du graphe en au plus  $K$  parties de poids inférieur ou égal à  $B$ , de façon à minimiser le poids des arêtes à l'intérieur des parties.

Par exemple la partition  $\pi = \{\{1, 2, 3\}, \{4, 5\}, \{6\}\}$  sera une solution admissible si :

- $K \geq 3$  ;
- $w_1 + w_2 + w_3 \leq B$  ;
- $w_4 + w_5 \leq B$  ; et
- $w_6 \leq B$ .

La valeur de l'objectif associé à cette solution est  $\ell_{1,2} + \ell_{1,3} + \ell_{2,3} + \ell_{4,5}$ .

### 2.2 Problème robuste

Nous souhaitons résoudre une version robuste du problème statique car il est possible que les valeurs choisies pour les distances et les poids soient sous-évalués.

### 2.2.1 Incertitude sur les distances

Nous supposons que la distance entre deux sommets  $i$  et  $j$  peut au maximum être augmentée de  $3(\hat{\ell}_i + \hat{\ell}_j)$ . Le pourcentage total d'augmentation de la longueur des arcs est limité par le paramètre  $L$ .

En d'autres termes, on peut représenter les valeurs que peuvent prendre les distances du graphe par l'ensemble :

$$\mathcal{U}^1 = \left\{ \{ \ell_{ij}^1 = \ell_{ij} + \delta_{ij}^1(\hat{\ell}_i + \hat{\ell}_j) \}_{ij \in E} \text{ tel que } \sum_{ij \in E} \delta_{ij}^1 \leq L, \delta_{ij}^1 \in [0, 3] \forall ij \in E \right\} \quad (1)$$

*Remarque* : Seules les  $\delta_{ij}^1$  sont des variables.  $\ell_{ij}$ ,  $\hat{\ell}_i$  et  $L$  sont des données du problème.

### 2.2.2 Incertitude sur les poids

L'augmentation totale des poids des sommets du graphe est limitée par les paramètres  $W_v$  et  $W$ .

Les valeurs que peuvent prendre les poids du graphe peuvent être représentées par l'ensemble :

$$\mathcal{U}^2 = \left\{ \{ w_v^2 = w_v(1 + \delta_v^2) \}_{v \in V} \text{ tel que } \sum_{v \in V} \delta_v^2 \leq W, \delta_v^2 \in [0, W_v] \forall v \in V \right\} \quad (2)$$

*Remarque* : Seules les  $\delta_v^2$  sont des variables.  $w_v$ ,  $W$  et  $W_v$  sont des données du problème.

## 3 Travail demandé

### Exercice 1 Modélisation papier

1. Proposer une modélisation du problème statique sous la forme d'un programme linéaire en nombre entiers **compact** (i.e., comportant un nombre polynomial de contraintes et de variables par rapport à la taille du graphe).
2. Proposer une modélisation du problème robuste sous la forme d'un programme en nombres entiers.
3. **Résolution par plans coupants et LazyCallback**
  - a) Modifier le problème afin que la robustesse n'apparaisse plus dans l'expression de l'objectif mais dans les contraintes.
  - b) Définir les ensembles  $\mathcal{U}^{1*}$  et  $\mathcal{U}^{2*}$  que vous utiliserez initialement dans le problème maître.
  - c) Exhiber les sous-problèmes nécessaires à la résolution du problème robuste par plans coupants (les contraintes devront être exprimées de manière explicite).
  - d) Quelles sont les conditions à satisfaire pour qu'une solution du problème maître soit optimale ?
  - e) Quelle est l'expression des coupes ajoutées par chaque sous-problème ?
4. **Résolution par dualisation**
  - a) Reformuler l'objectif du problème robuste afin d'isoler le terme faisant intervenir les variables  $\delta_{ij}^1$ .
  - b) Exhiber le problème interne lié aux variables  $\delta_{ij}^1$ .
  - c) Dualiser ce problème.
  - d) Reformuler les contraintes robustes afin d'isoler le terme faisant intervenir les variables  $\delta_i^2$ .
  - e) Exhiber les problèmes internes liés aux variables  $\delta_i^2$ .
  - f) Dualiser ces problèmes.
  - g) Utiliser ces dualisations afin de présenter le problème robuste sous la forme d'un simple programme linéaire en nombres entiers.

### Exercice 2 Résolution numérique

1. Implémenter la résolution du problème robuste par dualisation, par plans coupants et par *branch-and-cut*. Vous pourrez tenter d'améliorer les performances de ces méthodes en les adaptant. Vous pouvez par exemple, essayer de résoudre heuristiquement les sous-problèmes, d'ajouter des callbacks, de fournir des solutions initiales à CPLEX, de retirer de la symétrie, de simplifier les données en entrée, de renforcer les coupes robustes ajoutées, ...
2. Choisir une des deux options :

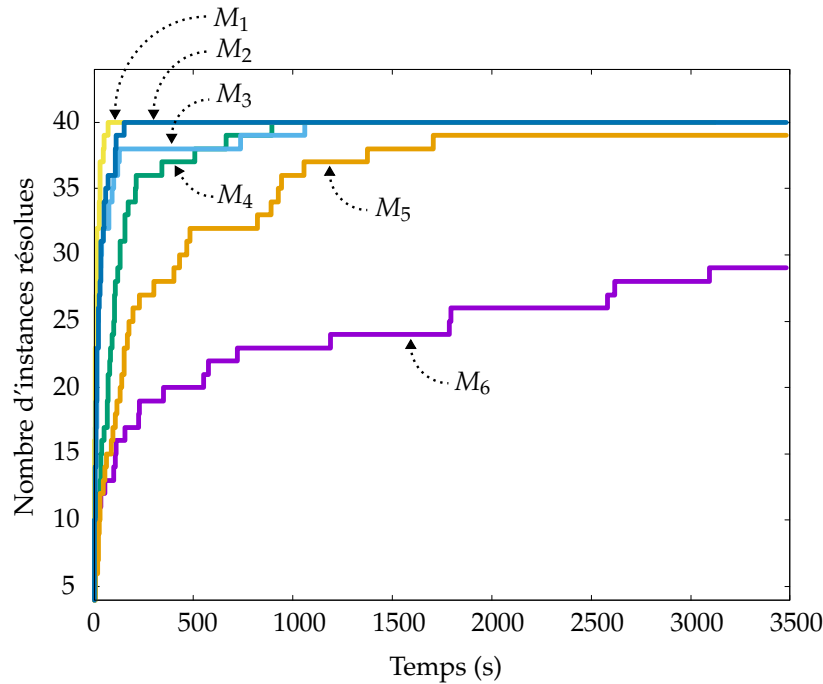


Figure 1: Exemple de diagramme de performances.

- (Choix 1) - Implémenter une heuristique permettant de résoudre ce problème. Cette heuristique devra fournir une garantie de performances a posteriori (en utilisant par exemple la relaxation du problème dualisé).
  - (Choix 2) - Résoudre le problème robuste en utilisant une formulation non compacte du problème.
3. a) Comparer les performances de ces quatre méthodes sur les données fournies. Vous ne pourrez pas résoudre optimalement toutes les instances. Vous pourrez donc fixer un temps maximal et calculer le gap entre la meilleure solution réalisable obtenue (si vous en avez obtenu une) et la meilleure relaxation du problème.
- b) Présenter vos résultats sous la forme d'un tableau tel que le suivant (la colonne *PR* correspond au prix de la robustesse qui est le gap entre la valeur de l'objectif d'une solution optimale robuste et d'une solution optimale statique) :

Instance	PR	Plans coupants		Branch-and-cut		Dualisation		Heuristique	
		Time	Gap	Time	Gap	Time	Gap	Time	Gap
20_USA-road-d.BAY	8%	10s	0%	10s	0%	10s	0%	10s	0%
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

- c) La lecture de tableaux de ce type pouvant parfois être fastidieuses, nous allons également représenter les résultats sous la forme de diagrammes de performances. Un diagramme de performances est une représentation graphique des performances de différentes méthodes sur les mêmes instances d'un même problème. Il représente pour chaque méthode le nombre d'instance résolues en fonction du temps. Par exemple, la figure 3 permet de comparer aisément les performances de 6 méthodes.

## 4 Echancier et modalités

Ce projet s'effectue en binôme.

- **Création d'un dépôt Git dédié au projet (à faire avant le 15 décembre)** (par exemple sur le site github) contenant un fichier README.md à la racine où vous indiquerez vos noms et une rapide description

(pas plus de quelques phrases). Ce fichier devra avoir été édité par chaque étudiant du binôme (*i.e.*, chaque étudiant devra y avoir effectué un commit). Vous recevrez un sondage à remplir pour nous informer de l'adresse de ce dépôt. S'il est privé, il faudrait nous en donner l'accès (mon compte github est associé à l'adresse zacharie.ales@gmail.com). Vous mettrez votre code et vos rapports sur ce dépôt.

- **Modélisation papier (à déposer sur git au plus tard le 21 décembre)**

Une modélisation vous sera fournie par mail quelques jours après la date de rendu (afin de vous permettre de corriger d'éventuelles erreurs de modélisation). A noter que plusieurs modélisations peuvent être valides et que si la vôtre diffère, cela peut être une piste intéressante à explorer.

- **Rapport (à déposer sur git au plus tard le 6 février)**

Ce rapport devra présenter les algorithmes proposés ainsi que les résultats expérimentaux obtenus. Vous mentionnerez en annexe une solution optimale obtenue pour chaque instance ainsi que la valeur de son objectif.

- **Soutenances (10 février 2023)**

L'évaluation se fera d'une part sur la justification et l'analyse des méthodes de résolution proposées pour le problème étudié, et d'autre part sur la qualité effective de ces algorithmes de résolution.