Rapport TP2 - Réseaux et Optimisation

Eldis YMERAJ
Maxence DE MONGELAS

21/03/2024

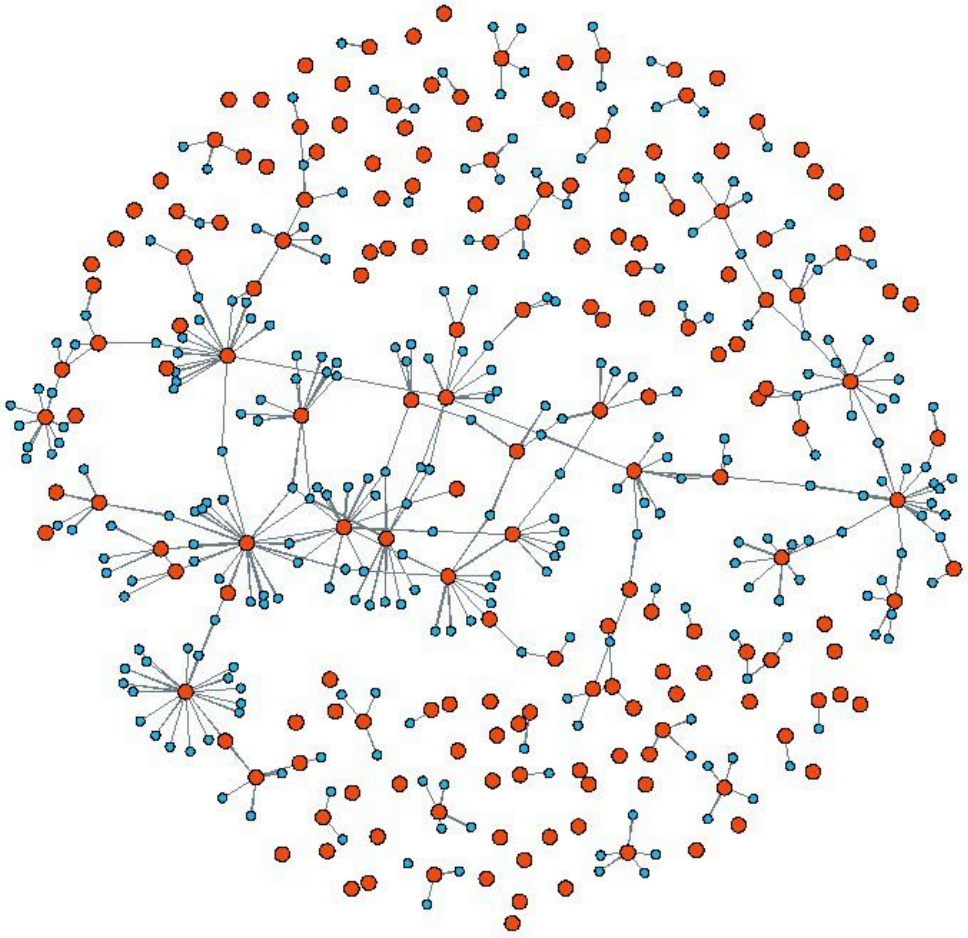

Table des matières

Introduction

Ce rapport explore l'optimisation avancée dans le cadre des graphes et réseaux en utilisant la programmation linéaire. Dans ce deuxième travail pratique, nous nous penchons sur des défis d'optimisation complexes, en mettant un accent particulier sur les problèmes de flots — à savoir la maximisation et la minimisation — ainsi que sur l'optimisation des structures de dominants dans les graphes. L'application d'outils d'optimisation sophistiqués, tels que IBM CPLEX, nous permet de concrétiser nos modèles théoriques en solutions pratiques.

Nous abordons une gamme d'exercices qui nous amènent à exploiter la théorie des graphes pour optimiser la distribution et la communication au sein des réseaux. Ces problématiques incluent la recherche de stratégies efficaces pour maximiser les flots de ressources, minimiser les coûts de transport et identifier des ensembles dominants, qui jouent un rôle clé dans la simplification et l'efficacité des réseaux. En résolvant ces problèmes, nous visons à affiner notre capacité à modéliser mathématiquement des situations complexes et à mettre en œuvre des stratégies d'optimisation avancées. Ce processus vise non seulement à découvrir des solutions optimales, mais aussi à comprendre en profondeur les principes sous-jacents à ces solutions, explorant ainsi les compromis nécessaires et les implications de nos modélisations pour des applications réelles et concrètes.

1 Organisation des Fichiers du TP

Ce deuxième travail pratique pour la matière Réseaux et Optimisation, est organisé en un dossier principal nommé `TP_RO`. Ce dossier comprend trois sous-dossiers correspondant à différentes parties du travail : `Dominants`, `Flomax`, et `Flomin`. Chacun de ces dossiers contient des fichiers spécifiques aux problèmes traités. Voici un détail de l'organisation des fichiers :

1.1 Dossier Flomax

Le dossier `Flomax` se concentre sur les problèmes de maximisation de flot. Il contient :

- **Modèle de Maximisation de Flot :** Le fichier `fmax.mod` est le modèle CPLEX utilisé pour le calcul du flot maximal.
- **Données :** Les données nécessaires pour exécuter le modèle se trouvent dans le fichier `donnees.dat`.
- **Exercice 2 de Flomax** qui contient le fichier Ex2.mod et Ex.dat.

1.2 Dossier Flomin

Le dossier `Flomin` traite des problèmes de minimisation de flot. Les fichiers inclus sont :

- **Modèle de Minimisation de Flot :** Le fichier `fmin.mod` représente le modèle CPLEX dédié à la minimisation des coûts de flot.
- **Données :** Le fichier `transport.dat` fournit les données nécessaires à la résolution du modèle.

1.3 Dossier Dominants

Enfin, le dossier `Dominants` contient les fichiers liés aux structures dominantes dans les graphes :

- **Sélection de Dominants :** Le fichier `DS.mod` est le modèle pour la sélection de dominants.
- **Données pour DS :** Les données pour le modèle de sélection de dominants sont situées dans `DS.dat`.
- **Ensemble Maximale Indépendant :** `PMIS.mod` définit le modèle pour trouver un ensemble maximal indépendant.
- **Clique Maximale :** Le fichier `WCIS.mod` est utilisé pour identifier la clique maximale dans un graphe.

Cette structure de fichiers a été conçue pour faciliter l'accès et l'organisation des différents modèles et leurs données associées, permettant ainsi une gestion efficace du projet d'optimisation.

2 Résolution du Problème du Flot Maximum

Formulation Linéaire

Soit le graphe orienté $G = (V, E)$ avec deux nœuds spéciaux s et p représentant respectivement la source et le puits, et une capacité u_{ij} pour chaque arc $(i, j) \in E$. La valeur du flot sur l'arc (i, j) est représentée par la variable x_{ij}, et nous cherchons à maximiser le flot sortant de la source v, tout en respectant les contraintes suivantes :

$$\begin{aligned}
&\text{Maximiser} \quad v \\
&\text{soumis aux contraintes} \\
&\sum_{j \in V} x_{ij} - \sum_{j \in V} x_{ji} = \begin{cases} +v & \text{si } i = s \\ -v & \text{si } i = p \\ 0 & \text{sinon} \end{cases} \quad \forall i \in V \\
&0 \leq x_{ij} \leq u_{ij} \quad \forall (i, j) \in E
\end{aligned}$$

2.1 Modélisation du Réseau Hydraulique

2.1.1 Exercice 1 : Maximisation de Flot entre Châteaux d'Eau et Villages

Dans cet exercice, l'objectif est d'acheminer l'eau des châteaux d'eau A, B, et C vers les villages D, E, F, et G. Les débits maximaux des canalisations et les demandes des villages sont spécifiés. Nous devons maximiser le flot d'eau tout en respectant les débits maximaux des canaux en entré et sortie des villages et châteaux d'eau.

Les données pour cet exercice sont résumées dans le tableau suivant :

Canalisation	**AD**	**AE**	**AG**	**BD**	**BE**	**BF**	**CF**	**CG**
Débit (l/s)	10	15	20	20	5	15	10	10

Table 1 – Débits des canalisations entre les châteaux d'eau et les villages.

2.1.2 Méthodologie et Implémentation

Pour modéliser ce problème, nous avons utilisé la programmation linéaire en définissant des variables de décision x_{ij} pour le flot sur chaque arête (i, j). Les contraintes de conservation du flot pour chaque nœud non-source et non-puits sont appliquées, ainsi que les contraintes de capacité pour chaque arête. La fonction objectif est de maximiser le volume total v de flot de la source vers le puits.

2.1.3 Résultats

Le flot maximal trouvé est de 85 litres par seconde. La distribution des flots dans le réseau est indiquée ci-dessous, montrant la quantité d'eau acheminée dans chaque canaux. De la source (S) en passant par les

châteaux d'eau, puis les villages jusqu'au puis (T) :

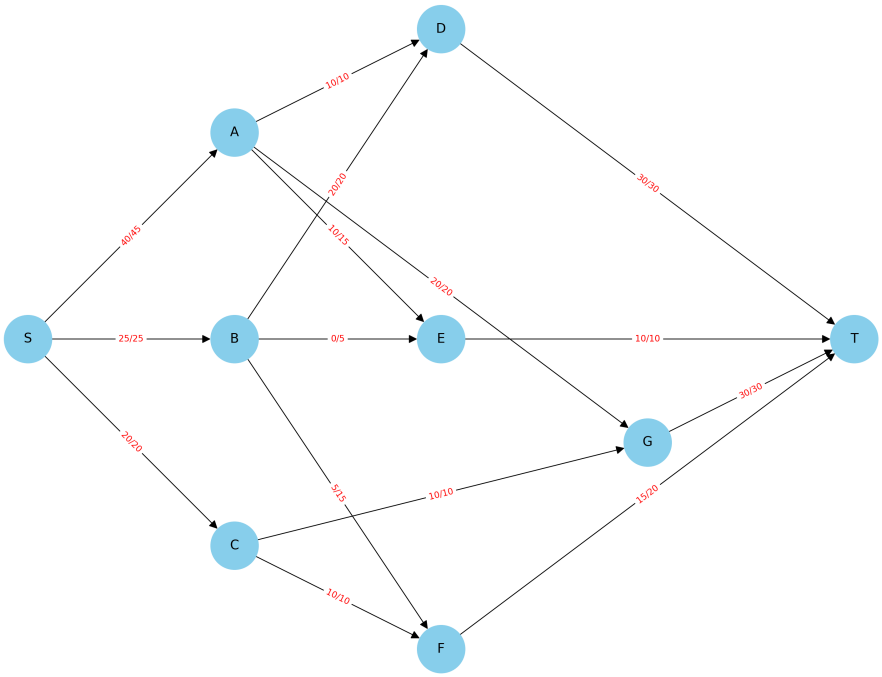

Figure 1 – Graphe flot maximum de S vers T

$$\begin{aligned} x_{\text{flux}} &= [SA, SB, SC, AD, AE, AG, BD, BE, BF, CF, CG, DT, ET, FT, GT]. \\ x_{\text{flux}} &= [40, 25, 20, 10, 10, 20, 20, 0, 5, 10, 10, 30, 10, 15, 30]. \end{aligned} \tag{1}$$

Les villages reçoivent les débits suivants à partir de chaque château d'eau :

- Village D reçoit 10 l/s de A et 20 l/s de B.
- Village E reçoit 10 l/s de A.
- Village F reçoit 5 l/s de B et 10 l/s de C.
- Village G reçoit 20 l/s de A et 10 l/s de C.

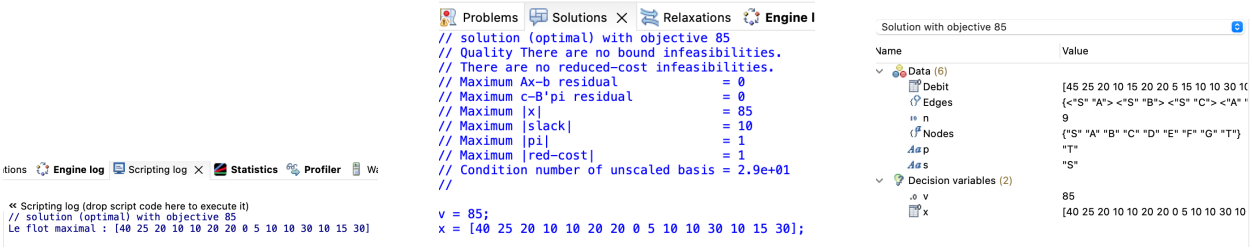

Figure 2 – Exercice 1.1 : Affichage des résultats sur la console.

On observe que le canal BE est vide, car le village E est déjà au maximum de ça capacité, soit 10/10. Par ailleurs, le village F pourrait recevoir plus d'eau. En effet, FT transmet 15/20. On a CF 10/10, mais BF 5/15 donc instinctivement, on voudrait augmenter le débit dans ce canal. Mais SB 25/25 et B redistribue déjà 25 de flux d'eau. Cette solution étant optimal dans cette configuration, pour remédier à ce problème,

il faudrait créer un canal en plus. Par exemple en AF, car SA n'est pas maximal 40/45 donc avec 5 en plus redistribués dans E le Flot serai encore plus maximal.

Ces types de stratégie comme l'ajustement de débits ou l'installation de nouvelles canalisations peut être envisagée en amont pour satisfaire pleinement les demandes.

Cette distribution des flux illustre à une échelle réduite la répartition des réseaux d'eau et souligne l'importance d'analyser le terrain en amont de projet de construction. Dans notre exemple, la ressource en question était l'eau. Elle devait être acheminée depuis des châteaux d'eau jusqu'aux villages. Notre objectif était de maximiser la quantité d'eau distribué tout en cherchant à répondre au mieux à la demande en eau de chaque village.Nous n'avons pas pu répondre pleinement à cette demande, certains villages n'ayant pas reçu leur capacité maximale en eau. En effet, le réseau de canaux utilisé n'était pas adéquat. Ce genre de problème peut être anticipé grâce à cette méthode d'analyse en amont de tout projet d'aménagement.

2.1.4 Exercice 2 : Affectation Optimale des Entiers aux Cases Admissibles

L'objectif de cet exercice est d'assigner un nombre entier à chaque case admissible d'un tableau comportant m lignes et n colonnes, de manière à maximiser la somme des nombres affectés tout en respectant des contraintes sur les lignes et les colonnes. Pour chaque ligne i, la somme des nombres dans les cases admissibles ne doit pas dépasser α_i, et pour chaque colonne j, cette somme ne doit pas dépasser β_j.

Les données pour cet exercice sont présentées dans le tableau suivant, où "A" représente une case admissible et "NA" une case non admissible :

	Colonne 1	**Colonne 2**	**Colonne 3**	**Colonne 4**
Ligne 1	A	A	A	A
Ligne 2	NA	A	NA	A
Ligne 3	NA	NA	NA	A

Table 2 – Répartition des cases admissibles (A) et non admissibles (NA).

Les contraintes pour les lignes et les colonnes sont données par les vecteurs $\alpha = (6, 4, 2)$ et $\beta = (1, 6, 3, 2)$ respectivement.

2.1.5 Méthodologie et Implémentation

Nous avons modélisé ce problème comme la recherche d'un flot maximum dans un réseau spécialement conçu. Les variables de décision x_{ij} représentent la quantité affectée à la case admissible (i, j). La fonction objectif vise à maximiser la somme de ces variables tout en respectant les contraintes de somme pour chaque ligne et chaque colonne.

2.1.6 Résultats

La solution optimale trouvée offre une somme totale de 12 pour les nombres affectés aux cases admissibles. Les détails de la solution sont les suivants :

$$x = [1, 2, 3, 0, 0, 4, 0, 2]. \tag{2}$$

Ce résultat montre comment les entiers doivent être répartis dans le tableau pour atteindre la somme maximale tout en respectant les contraintes spécifiées. La distribution précise des valeurs dans le tableau illustre

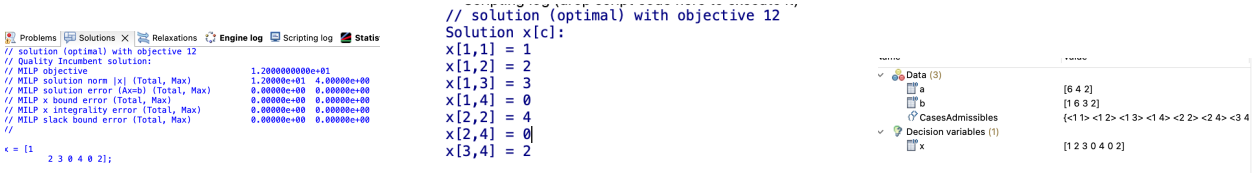

Figure 3 – Exercice 2 : Affichage des résultats sur la console.

l'efficacité de l'approche de modélisation et d'optimisation pour résoudre des problèmes d'affectation complexes.

Cette méthodologie démontre l'efficacité de la programmation linéaire et de l'optimisation dans la résolution de problèmes d'allocation de ressources avec contraintes. Elle offre une stratégie claire pour l'affectation optimale des ressources, même dans des cas où les contraintes rendent la tâche apparemment complexe.

3 Résolution du Problème du Flot de Coût Minimum

Considérons un graphe orienté $G = (V, E)$ où chaque arête $(i, j) \in E$ a une capacité inférieure l_{ij}, une capacité supérieure u_{ij}, et un coût unitaire c_{ij}. Les nœuds ont des valeurs associées b_i qui sont positives pour les sources, négatives pour les destinations, et nulles pour les nœuds intermédiaires. L'objectif est de trouver un flot qui minimise le coût total tout en respectant les contraintes de capacité.

La formulation linéaire du problème est la suivante :

- Les variables de décision : x_{ij} représentent la valeur du flot sur l'arc ij.
- Les contraintes de flot sont données par :

$$\sum_{j \in V} x_{ij} - \sum_{j \in V} x_{ji} = b_i, \quad \forall i \in E$$

  avec la condition que $l_{ij} \leq x_{ij} \leq u_{ij}, \quad \forall (i, j) \in E$.
- L'objectif est de minimiser le coût total du flot :

$$\min \sum_{(i,j) \in E} c_{ij} x_{ij}$$

3.1 Problème du transport

3.1.1 Exercice 1 : Problème de Transport

Dans cet exercice, nous avons des biens à transporter de trois ports vers quatre villes. Chaque ville a une demande spécifique, et chaque port a une quantité d'offre disponible. Le coût de transport de chaque port à chaque ville est donné et l'objectif est de minimiser le coût total du transport tout en répondant aux demandes des villes.

Les données pour cet exercice sont résumées dans le tableau suivant :

3.1.2 Méthodologie et Implémentation

Pour résoudre ce problème, nous avons employé un modèle de programmation linéaire. Les variables de décision x_{ij} indiquent le volume de biens transportés de la source i à la destination j. Nous avons établi des contraintes pour maintenir la conservation du flot à chaque nœud et pour respecter les capacités des arêtes.

Villes\Ports	P1	P2	P3	Demande
V1	8	8	9	15
V2	5	2	3	20
V3	6	7	4	30
V4	7	6	8	35
Offre	25	25	50	

Table 3 – Coûts de transport et demandes/offres pour l'exercice 1.

La fonction objectif est conçue pour minimiser la somme des produits des coûts unitaires c_{ij} par les flots x_{ij} sur toutes les arêtes.

3.1.3 Résultats exercice 2.1

La solution optimale obtenue avec un objectif de 520 a confirmé l'absence d'incohérences de capacité et de coût réduit. Le flot maximum dans le réseau était de 30, et avec un coût réduit maximal de 4.

Les flux optimisés sur chaque arête du réseau étaient comme suit :

$$x_{\text{flux}} = [15, 0, 0, 10, 0, 0, 0, 25, 0, 20, 30, 0]. \tag{3}$$

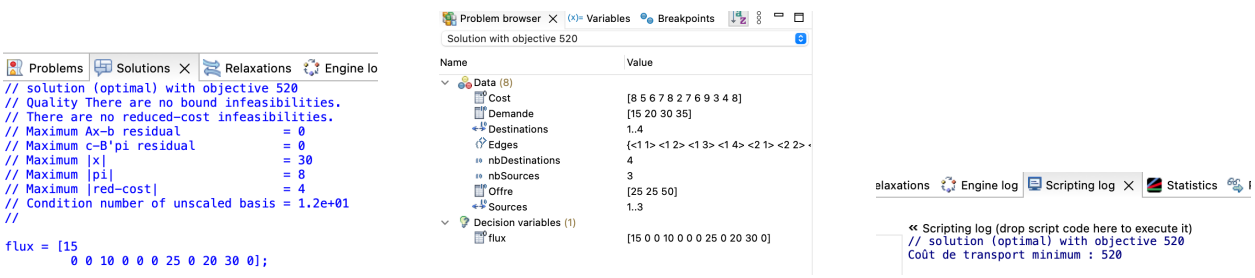

Figure 4 – Exercice 2.1 : Affichage des résultats sur la console.

Cette répartition des flux illustre la manière dont les biens doivent être acheminés depuis les ports vers les villes pour minimiser le coût global tout en satisfaisant la demande de chaque ville.

3.2 Problème d'Affectation de Coût Minimum

Le problème d'affectation de coût minimum est un cas particulier du problème de flot de coût minimum où l'on souhaite affecter des ressources à des tâches de manière à minimiser le coût total. Chaque ressource doit être affectée à une unique tâche et chaque tâche doit être effectuée par une unique ressource.

Dans notre cas, nous avons un ensemble d'employés $E1, E2, E3$ et un ensemble de tâches $T1, T2, T3$. Les coûts d'affectation des employés aux tâches sont donnés par le tableau suivant :

Employés/Tâches	E1	E2	E3
T1	210	175	315
T2	200	185	310
T3	225	190	320

Table 4 – Tableau des coûts d'affectation des employés aux tâches.

Pour formuler ce problème comme un problème de flot de coût minimum, nous avons créé un réseau biparti où chaque employé (source) est connecté à chaque tâche (destination) avec des arêtes dont les capacités sont fixées à 1. De même, la demande pour chaque tâche et l'offre de chaque employé sont fixées à 1. Cette configuration garantit qu'un seul employé est affecté à chaque tâche et vice-versa.

Le modèle de programmation linéaire utilisé pour résoudre ce problème est similaire à celui utilisé pour le problème de flot de coût minimum. Les variables de décision, les contraintes et la fonction objectif sont ajustées pour satisfaire les spécificités de l'affectation.

3.2.1 Résultats exercice 2.2

Les résultats obtenus à partir du solveur CPLEX sont les suivants :

```
Coût d'affectation minimum : 695
Affectation optimale : E1 à T2, E2 à T3, E3 à T1
```

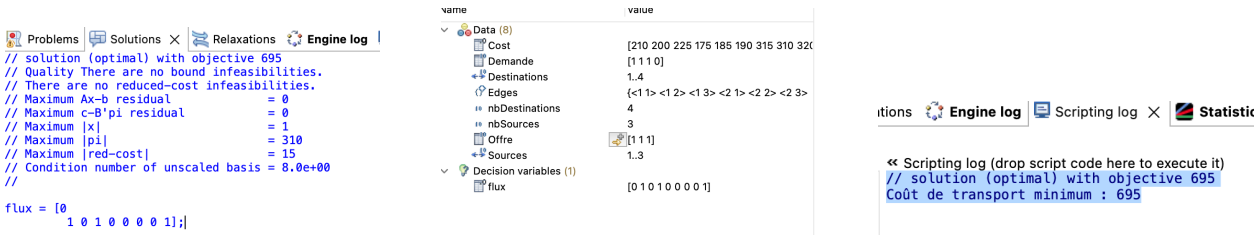

Figure 5 – Exercice 2.2 : Affichage des résultats sur la console.

4 Structure des dominants

Dans cette partie du tp, nous nous attelons à la tâche d'optimiser les structures dominantes au sein des réseaux sans fil. L'objectif principal est de minimiser la cardinalité des ensembles dominants, stables et presque connexes. Cette démarche vise à structurer le réseau de manière à assurer une couverture et une connectivité optimales tout en minimisant la consommation d'énergie. Pour atteindre cet objectif, nous adoptons des formulations en variables binaires qui nous permettent de modéliser précisément la sélection des sommets au sein du graphe représentant le réseau.

4.1 Formulations Mathématiques

Nous utilisons les variables binaires $x(i)$ pour indiquer si un sommet i est sélectionné ($x(i) = 1$) ou non ($x(i) = 0$). Les formulations suivantes ont été mises en œuvre :

— **Programme PDS (Dominant Set)** : L'objectif est de minimiser la somme des variables de sélection, $\min \sum_{i \in V} x(i)$, sous la contrainte que chaque sommet soit dominé, soit directement, soit par un voisin :

$$x(i) + \sum_{j \in N(i)} x(j) \geq 1, \quad \forall i \in V.$$

— **Programme PMIS (Maximal Independent Set)** : En plus de minimiser $\sum_{i \in V} x(i)$, ce programme impose que aucun couple de sommets adjacents ne soit simultanément inclus dans l'ensemble, traduit par la contrainte :

$$x(i) + x(j) \leq 1, \quad \forall \{i, j\} \in E.$$

— **Programme PWCIS (Weakly Connected Independent Set)** : L'objectif reste la minimisation de $\sum_{i \in V} x(i)$, avec la particularité que l'ensemble stable doit être presque connexe. Les contraintes sont ajustées pour refléter cette exigence, notamment en s'assurant qu'au moins un sommet sur le bord de tout sous-ensemble W soit sélectionné :

$$\sum_{i \in \text{Bord}(W)} x(i) \geq 1, \quad W \subseteq V.$$

4.2 Implémentation du Programme Dominant Set (PDS)

L'implémentation que nous avons fait vise à identifier un sous-ensemble de sommets dans un graphe qui soit dominant, c'est-à-dire, chaque sommet du graphe est soit dans l'ensemble dominant soit adjacent à au moins un sommet dans cet ensemble. L'objectif principal est de minimiser la cardinalité de cet ensemble, réduisant ainsi le coût associé à la couverture du réseau.

4.2.1 Structure du Code

Le fichier `DS.mod` contient le modèle OPL pour le PDS. Initialement, le nombre de sommets n et l'ensemble des arêtes `Edges` sont déclarés. Les sommets sont représentés par un ensemble `Nodes`, dérivé des arêtes, garantissant l'inclusion de tous les sommets présents dans le graphe. Un calcul préliminaire est effectué pour déterminer le voisinage de chaque sommet, stocké dans le tableau `voisin`.

Les variables de décision sont déclarées comme binaires (`dvar boolean x[Nodes]`), où $x[i] = 1$ indique que le sommet i est inclus dans l'ensemble dominant. La fonction objectif est de minimiser la somme de ces variables binaires, reflétant la cardinalité de l'ensemble dominant. La contrainte clé stipule que pour chaque sommet, soit le sommet lui-même soit au moins un de ses voisins doit être inclus dans l'ensemble dominant :

```
forall (node in Nodes) {
   x[node] + sum(v in voisin[node]) x[v] >= 1;
}
```

Après l'exécution du modèle, l'ensemble dominant optimal est extrait et affiché grâce à la collection `dominant`, qui filtre les sommets sélectionnés par la solution optimale.

4.2.2 Exemple 1 : PDS

Configuration du Graphe Pour cet exemple, le graphe est configuré avec $n = 7$ sommets et les arêtes suivantes :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,7\rangle, \langle 2,3\rangle, \langle 3,4\rangle, \langle 4,5\rangle, \langle 5,6\rangle, \langle 6,7\rangle\}.$$

Cette structure forme une chaîne linéaire reliant tous les sommets du graphe.

Résultats Après l'exécution du modèle PDS pour cette configuration, la solution optimale identifiée présente un objectif de minimisation avec une valeur de 3. L'ensemble dominant minimal trouvé est :

$$\{3, 6, 7\}.$$

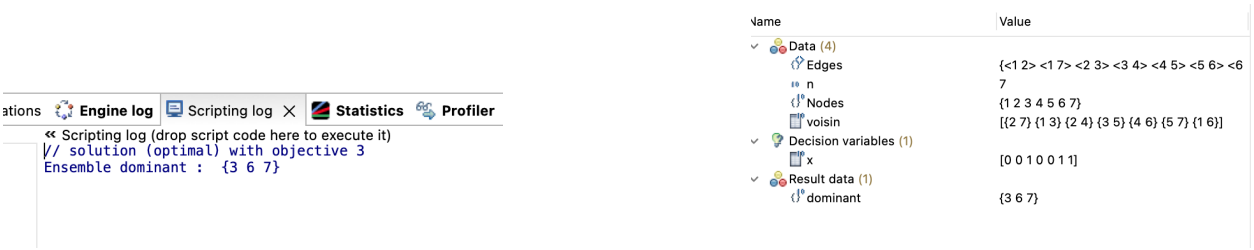

Figure 6 – Exemple 1(PDS) : Affichage des résultats sur la console.

4.2.3 Exemple 2 : PDS

Configuration du Graphe Dans cet exemple, le graphe est défini avec $n = 10$ sommets et un ensemble d'arêtes qui créent plusieurs branches à partir de certains sommets, illustrant une structure plus complexe :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,5\rangle, \langle 1,6\rangle, \langle 1,7\rangle, \langle 2,3\rangle, \langle 3,4\rangle, \langle 4,8\rangle, \langle 4,9\rangle, \langle 4,10\rangle\}.$$

Cette configuration reflète une certaine complexité dans la connectivité entre les sommets, défiant le modèle à identifier efficacement un ensemble dominant minimal.

Résultats L'exécution du modèle PDS pour cette configuration spécifique a produit une solution optimale avec un objectif minimal de 2, ce qui indique un ensemble dominant extrêmement réduit pour le graphe donné. Les sommets sélectionnés constituant l'ensemble dominant sont :

$$\{1, 4\}.$$

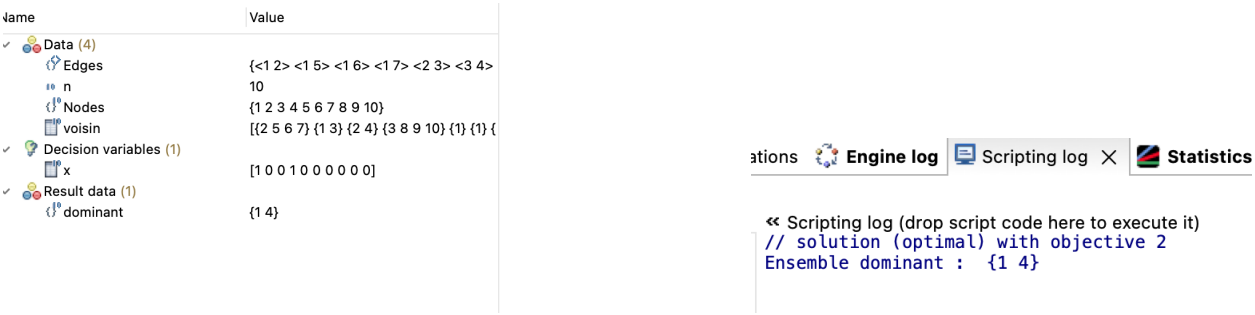

Figure 7 – Exemple 2(PDS) : Affichage des résultats sur la console.

4.2.4 Exemple 3 : PDS

Configuration du Graphe Cet exemple se concentre sur un graphe composé de $n = 10$ sommets, présentant une structure qui s'apparente à un anneau avec des connexions transversales, comme décrit par l'ensemble des arêtes :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,8\rangle, \langle 1,9\rangle, \langle 2,3\rangle, \langle 2,6\rangle, \langle 3,4\rangle, \langle 4,5\rangle, \langle 5,6\rangle, \langle 5,10\rangle, \langle 6,7\rangle, \langle 7,8\rangle, \langle 9,10\rangle\}.$$

Cette configuration offre un mélange intéressant de séquentialité et de connectivité croisée entre les sommets, posant un défi unique pour l'identification d'un ensemble dominant minimal.

Résultats La solution optimale obtenue pour ce graphe, à la suite de l'exécution du modèle PDS, affiche un objectif de 4, indiquant que l'ensemble dominant minimal trouvé est constitué de quatre sommets. L'ensemble dominant optimal est le suivant :

$$\{4, 6, 8, 10\}.$$

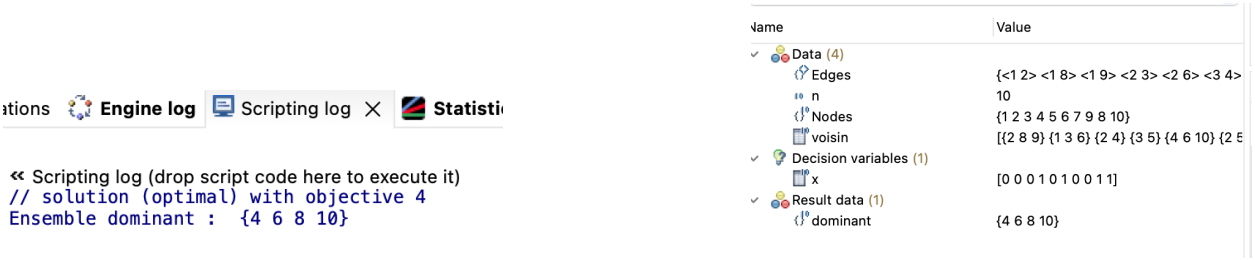

Figure 8 – Exemple 3(PDS) : Affichage des résultats sur la console.

4.2.5 Exemple 4 : PDS

Configuration du Graphe L'exemple 4 explore un graphe de $n = 11$ sommets avec des arêtes formant une configuration qui évoque une structure en étoile étendue. Les arêtes sont définies comme suit :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,3\rangle, \langle 1,4\rangle, \langle 2,3\rangle, \langle 3,5\rangle, \langle 3,6\rangle, \langle 4,5\rangle, \langle 4,7\rangle, \langle 5,6\rangle, \langle 5,8\rangle, \langle 7,8\rangle, \langle 7,9\rangle, \langle 8,10\rangle, \langle 9,10\rangle, \langle 10,11\rangle\}.$$

Cette disposition des arêtes présente des défis intéressants pour la détermination d'un ensemble dominant minimal, en raison de la présence de nœuds fortement connectés ainsi que de chaînes linéaires étendues.

Résultats La solution optimale trouvée pour cet arrangement particulier du graphe résulte en un objectif de minimisation de 3, mettant en évidence un ensemble dominant minimal constitué de trois sommets seulement. Ces sommets, formant l'ensemble dominant optimal, sont :

$$\{3, 7, 10\}.$$

Figure 9 – Exemple 4(PDS) : Affichage des résultats sur la console.

4.2.6 Exemple 5 : PDS

Configuration du Graphe Dans l'exemple 5, nous examinons un graphe densément connecté composé de $n = 12$ sommets. Les connexions entre les sommets sont caractérisées par une série d'arêtes formant un

réseau complexe :

$$\begin{aligned}\text{Edges} = \{&\langle 1,2\rangle, \langle 1,3\rangle, \langle 1,6\rangle, \langle 1,11\rangle,\\ &\langle 2,6\rangle, \langle 2,7\rangle, \langle 2,11\rangle, \langle 3,11\rangle,\\ &\langle 4,10\rangle, \langle 5,7\rangle, \langle 5,9\rangle, \langle 5,8\rangle,\\ &\langle 5,11\rangle, \langle 5,12\rangle, \langle 7,11\rangle, \langle 8,10\rangle,\\ &\langle 8,11\rangle, \langle 9,10\rangle, \langle 9,12\rangle, \langle 10,12\rangle\}.\end{aligned}$$

Résultats La solution optimale obtenue pour ce graphe révèle un objectif de minimisation de 3, signifiant que l'ensemble dominant minimal est constitué de seulement trois sommets. Ces sommets, formant l'ensemble dominant optimal, sont identifiés comme suit :

$$\{1, 10, 11\}.$$

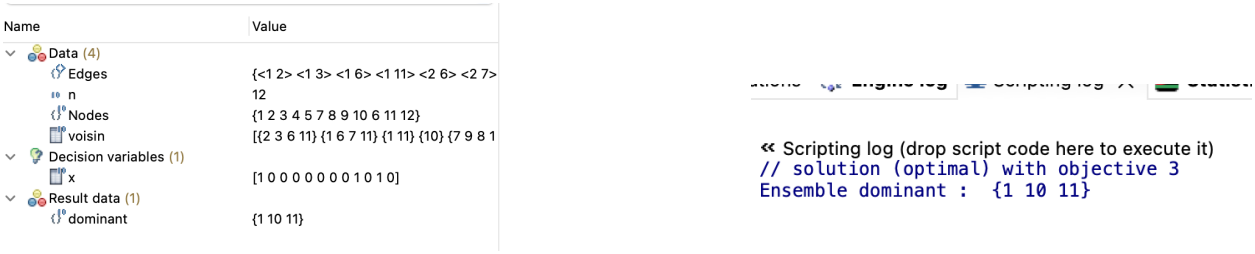

Figure 10 – Exemple 5(PDS) : Affichage des résultats sur la console.

4.2.7 Exemple 6 : PDS

Configuration du Graphe L'exemple 6 met en scène un graphe de $n = 13$ sommets et les arcs suivantes :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,8\rangle, \langle 1,12\rangle, \langle 2,3\rangle, \langle 2,6\rangle, \langle 3,4\rangle, \langle 3,13\rangle, \langle 4,5\rangle, \langle 4,11\rangle, \langle 5,6\rangle, \langle 5,9\rangle, \langle 6,7\rangle, \langle 7,8\rangle, \langle 7,10\rangle, \langle 8,11\rangle, \langle 9,10\rangle, \langle 12,13\rangle\}.$$

Résultats Nous avons une solution optimale avec un objectif de 5, indiquant que cinq sommets sont nécessaires pour former un ensemble dominant qui couvre efficacement tout le graphe. Ces sommets sélectionnés sont :

$$\{6, 12, 13, 11, 10\}.$$

Figure 11 – Exemple 6(PDS) : Affichage des résultats sur la console.

4.2.8 Exemple 7 : PDS

Configuration du Graphe Pour l'exemple 7, nous explorons un graphe plus complexe de $n = 14$ sommets. Ce graphe se distingue par sa structure riche en connexions, avec plusieurs nœuds servant de centres de distribution reliant différentes parties du réseau :

Edges $= \{\langle 1,2\rangle, \langle 1,7\rangle, \langle 1,8\rangle, \langle 2,3\rangle, \langle 2,9\rangle, \langle 3,4\rangle, \langle 3,14\rangle, \langle 4,5\rangle, \langle 4,13\rangle, \langle 5,6\rangle, \langle 5,12\rangle, \langle 6,7\rangle, \langle 6,11\rangle, \langle 7,10\rangle, \langle 8,11\rangle, \langle 8,14\rangle, \langle 9,10\rangle, \langle 9,13\rangle, \langle 10,12\rangle, \langle 11,13\rangle, \langle 12,14\rangle\}$.

Résultats L'analyse de ce graphe a révélé une solution optimale nécessitant un ensemble dominant de cinq sommets pour assurer une couverture complète. Les sommets formant cet ensemble optimal sont :

$$\{1, 3, 4, 6, 10\}.$$

Figure 12 – Exemple 7(PDS) : Affichage des résultats sur la console.

4.2.9 Exemple 8 : PDS

Configuration du Graphe Pour l'exemple 7, nous explorons un graphe encore plus complexe de $n = 17$ sommets et les arcs suivants :

Edges $=\{\langle 1,2\rangle, \langle 1,3\rangle, \langle 1,4\rangle, \langle 1,5\rangle, \langle 2,3\rangle, \langle 2,6\rangle, \langle 3,4\rangle, \langle 3,7\rangle, \langle 4,5\rangle, \langle 4,8\rangle, \langle 5,9\rangle, \langle 6,7\rangle, \langle 6,10\rangle, \langle 7,8\rangle, \langle 7,11\rangle, \langle 8,9\rangle, \langle 8,12\rangle, \langle 9,13\rangle, \langle 10,11\rangle, \langle 10,14\rangle, \langle 11,12\rangle, \langle 11,15\rangle, \langle 12,13\rangle, \langle 12,16\rangle, \langle 13,17\rangle, \langle 14,15\rangle, \langle 15,16\rangle, \langle 16,17\rangle\}$.

Résultats L'analyse de ce graphe a révélé une solution optimale nécessitant un ensemble dominant de 4 sommets pour assurer une couverture complète. Les sommets formant cet ensemble optimal sont :

$$\{3, 9, 10, 16\}.$$

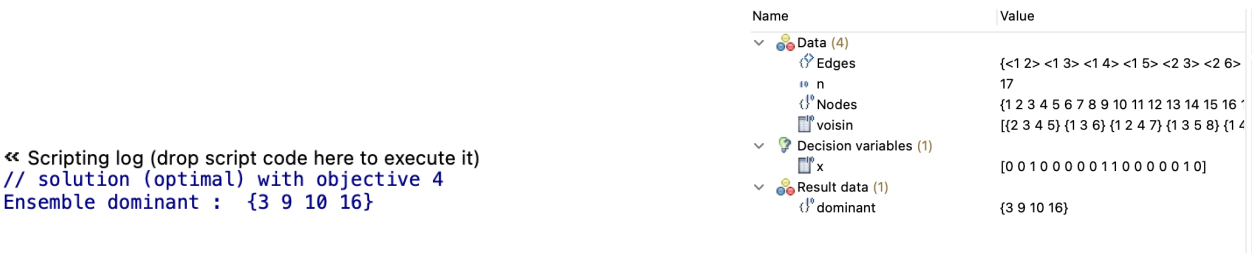

Figure 13 – Exemple 8(PDS) : Affichage des résultats sur la console.

4.3 Implémentation du Programme Stable Indépendant Maximal (PMIS)

L'objectif de l'implémentation du Programme Stable Indépendant Maximal (PMIS) est de trouver un sous-ensemble de sommets dans un graphe tel que cet ensemble soit stable, c'est-à-dire qu'aucune arête ne relie deux sommets quelconques de cet ensemble, tout en étant dominant, c'est-à-dire que chaque sommet du graphe soit soit inclus dans cet ensemble soit adjacent à au moins un sommet de cet ensemble. Le but ultime est de minimiser la cardinalité de cet ensemble, ce qui permet de réduire le coût associé à la couverture du réseau tout en garantissant la connectivité.

4.3.1 Structure du Code

Le fichier `MIS.mod` contient le modèle OPL pour le PMIS. Initialement, le nombre de sommets n et l'ensemble des arêtes `Edges` sont déclarés. Les sommets sont représentés par un ensemble `Nodes`, dérivé des arêtes, garantissant l'inclusion de tous les sommets présents dans le graphe. Un calcul préliminaire est effectué pour déterminer le voisinage de chaque sommet, stocké dans le tableau `voisin`.

Les variables de décision sont déclarées comme binaires (`dvar boolean x[Nodes]`), où $x[i] = 1$ indique que le sommet i est inclus dans l'ensemble stable. La fonction objectif est de minimiser la somme de ces variables binaires, reflétant la cardinalité de l'ensemble stable.

4.3.2 Exemple 1 : PMIS

Configuration du Graphe Pour cet exemple, le graphe est configuré avec $n = 7$ sommets et les arêtes suivantes :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,7\rangle, \langle 2,3\rangle, \langle 3,4\rangle, \langle 4,5\rangle, \langle 5,6\rangle, \langle 6,7\rangle\}.$$

Cette structure forme une chaîne linéaire reliant tous les sommets du graphe.

Résultats On a trouvé une valeur de la fonction objectif $= 3$ et l'ensemble dominant MIS est :

$$\{1, 3, 5\}.$$

Figure 14 – Exemple 1(MIS) : Affichage des résultats sur la console.

4.3.3 Exemple 2 : PMIS

Configuration du Graphe Dans cet exemple, le graphe est défini avec $n = 10$ sommets et un ensemble d'arêtes qui créent plusieurs branches à partir de certains sommets, illustrant une structure plus complexe :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,5\rangle, \langle 1,6\rangle, \langle 1,7\rangle, \langle 2,3\rangle, \langle 3,4\rangle, \langle 4,8\rangle, \langle 4,9\rangle, \langle 4,10\rangle\}.$$

Résultats On remarque qu'on a le mem résultat que pour l'ensemble DS :

$$\{1, 4\}.$$

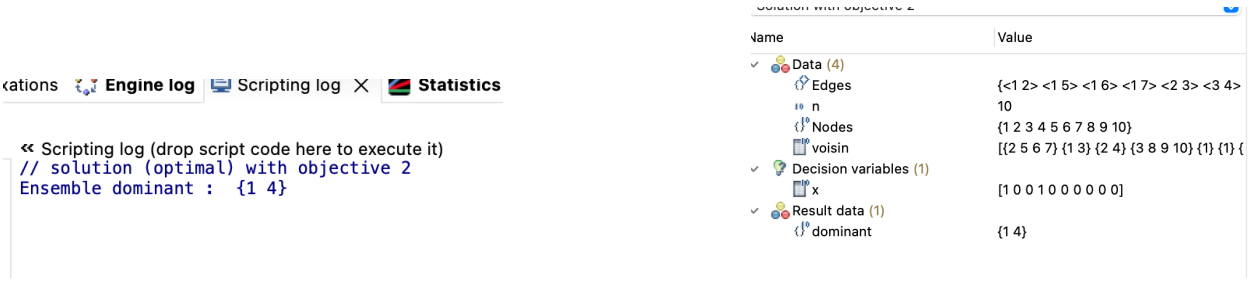

Figure 15 – Exemple 2(MIS) : Affichage des résultats sur la console.

4.3.4 Exemple 3 : PMIS

Configuration du Graphe Cet exemple se concentre sur un graphe composé de $n = 10$ sommets, présentant une structure qui s'apparente à un anneau avec des connexions transversales, comme décrit par l'ensemble des arêtes :

$$\text{Edges} = \{\langle 1, 2\rangle, \langle 1, 8\rangle, \langle 1, 9\rangle, \langle 2, 3\rangle, \langle 2, 6\rangle, \langle 3, 4\rangle, \langle 4, 5\rangle, \langle 5, 6\rangle, \langle 5, 10\rangle, \langle 6, 7\rangle, \langle 7, 8\rangle, \langle 9, 10\rangle\}.$$

Résultats La solution optimale obtenue pour ce graphe affiche un objectif de 4, avec l'ensemble MIS :

$$\{2, 4, 7, 10\}.$$

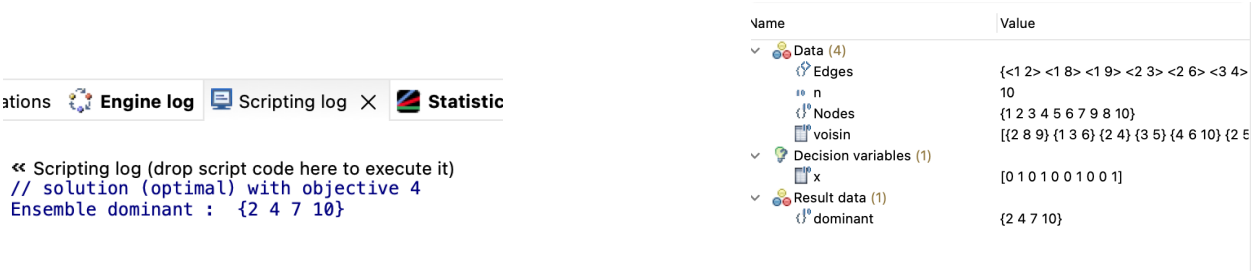

Figure 16 – Exemple 3(MIS) : Affichage des résultats sur la console.

4.3.5 Exemple 4 : PMIS

Configuration du Graphe L'exemple 4 explore un graphe de $n = 11$ sommets avec des arêtes suivantes :

$$\text{Edges} = \{\langle 1, 2\rangle, \langle 1, 3\rangle, \langle 1, 4\rangle, \langle 2, 3\rangle, \langle 3, 5\rangle, \langle 3, 6\rangle, \langle 4, 5\rangle, \langle 4, 7\rangle, \langle 5, 6\rangle, \langle 5, 8\rangle, \langle 7, 8\rangle, \langle 7, 9\rangle, \langle 8, 10\rangle, \langle 9, 10\rangle, \langle 10, 11\rangle\}.$$

Cette disposition des arêtes présente des défis intéressants pour la détermination d'un ensemble dominant minimal, en raison de la présence de nœuds fortement connectés ainsi que de chaînes linéaires étendues.

Résultats Ici on remarque qu'on a la meme soluion que pour l'ensemble DS :

$$\{3, 7, 10\}.$$

Figure 17 – Exemple 4(MIS) : Affichage des résultats sur la console.

4.3.6 Exemple 5 : PMIS

Configuration du Graphe Dans l'exemple 5, $n = 12$ et les arcs suivants :

$$\begin{aligned}\text{Edges} = \{&\langle 1,2\rangle, \langle 1,3\rangle, \langle 1,6\rangle, \langle 1,11\rangle,\\ &\langle 2,6\rangle, \langle 2,7\rangle, \langle 2,11\rangle, \langle 3,11\rangle,\\ &\langle 4,10\rangle, \langle 5,7\rangle, \langle 5,9\rangle, \langle 5,8\rangle,\\ &\langle 5,11\rangle, \langle 5,12\rangle, \langle 7,11\rangle, \langle 8,10\rangle,\\ &\langle 8,11\rangle, \langle 9,10\rangle, \langle 9,12\rangle, \langle 10,12\rangle\}.\end{aligned}$$

Résultats La solution optimale = 3, avec l'ensemble MIS : :

$$\{1, 4, 5\}.$$

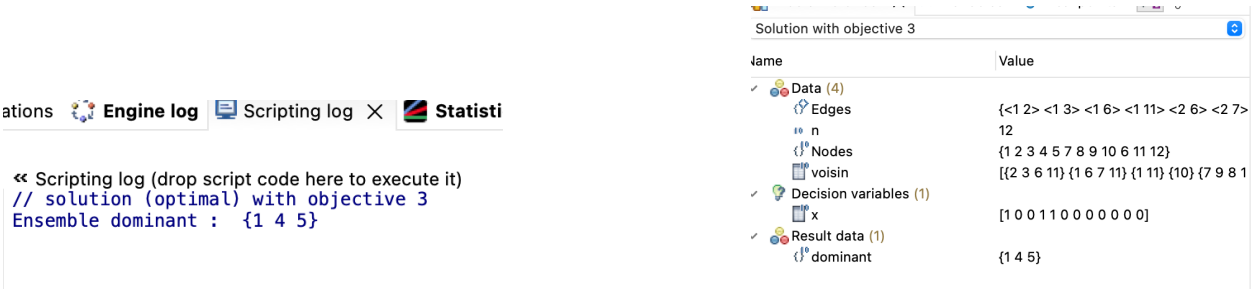

Figure 18 – Exemple 5(MIS) : Affichage des résultats sur la console.

4.3.7 Exemple 6 : PMIS

Configuration du Graphe L'exemple 6 met en scène un graphe de $n = 13$ sommets et les arcs suivantes :

$$\text{Edges} = \{\langle 1,2\rangle, \langle 1,8\rangle, \langle 1,12\rangle, \langle 2,3\rangle, \langle 2,6\rangle, \langle 3,4\rangle, \langle 3,13\rangle, \langle 4,5\rangle, \langle 4,11\rangle, \langle 5,6\rangle, \langle 5,9\rangle, \langle 6,7\rangle, \langle 7,8\rangle, \langle 7,10\rangle, \langle 8,11\rangle, \langle 9,10\rangle, \langle 12,13\rangle\}.$$

Résultats Nous avons une solution optimale avec un objectif de 5, l'ensemble dominantest le suivant :

$$\{1, 3, 5, 11, 10\}.$$

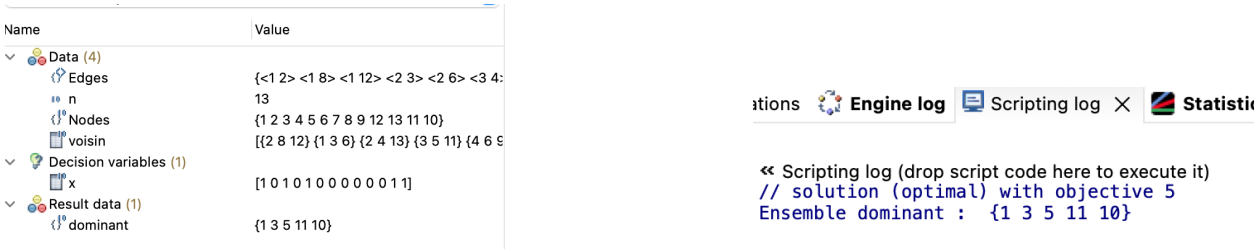

Figure 19 – Exemple 6(MIS) : Affichage des résultats sur la console.

4.3.8 Exemple 7 : PMIS

Configuration du Graphe Pour l'exemple 7, $n = 14$ sommets et les arcs suivants :

Edges $= \{\langle 1,2\rangle, \langle 1,7\rangle, \langle 1,8\rangle, \langle 2,3\rangle, \langle 2,9\rangle, \langle 3,4\rangle, \langle 3,14\rangle, \langle 4,5\rangle, \langle 4,13\rangle, \langle 5,6\rangle, \langle 5,12\rangle, \langle 6,7\rangle, \langle 6,11\rangle, \langle 7,10\rangle, \langle 8,11\rangle, \langle 8,14\rangle, \langle 9,10\rangle, \langle 9,13\rangle, \langle 10,12\rangle, \langle 11,13\rangle, \langle 12,14\rangle\}$.

Résultats L'analyse de ce graphe a révélé une solution optimale nécessitant un ensemble dominant de cinq sommets pour assurer une couverture complète. Les sommets formant cet ensemble optimal sont :

$$\{2, 5, 8, 10, 13\}.$$

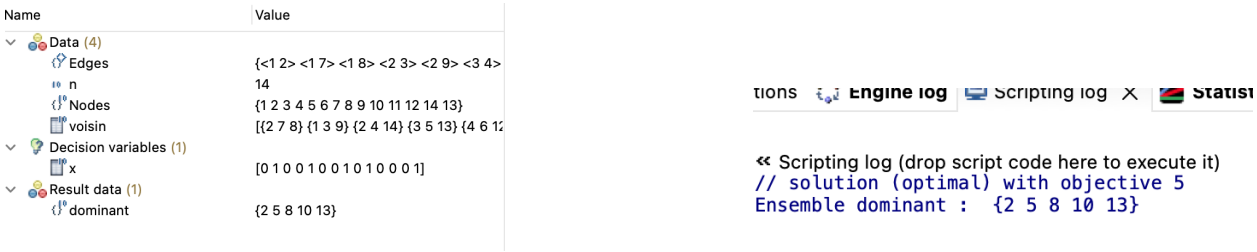

Figure 20 – Exemple 7(MIS) : Affichage des résultats sur la console.

4.3.9 Exemple 8 : PMIS

Configuration du Graphe Pour l'exemple 8, nous explorons un graphe encore plus complexe de $n = 17$ sommets et les arcs suivants :

Edges $= \{\langle 1,2\rangle, \langle 1,3\rangle, \langle 1,4\rangle, \langle 1,5\rangle, \langle 2,3\rangle, \langle 2,6\rangle, \langle 3,4\rangle, \langle 3,7\rangle, \langle 4,5\rangle, \langle 4,8\rangle, \langle 5,9\rangle, \langle 6,7\rangle, \langle 6,10\rangle, \langle 7,8\rangle, \langle 7,11\rangle, \langle 8,9\rangle, \langle 8,12\rangle, \langle 9,13\rangle, \langle 10,11\rangle, \langle 10,14\rangle, \langle 11,12\rangle, \langle 11,15\rangle, \langle 12,13\rangle, \langle 12,16\rangle, \langle 13,17\rangle, \langle 14,15\rangle, \langle 15,16\rangle, \langle 16,17\rangle\}$.

Résultats Nous remarquons qu'on a le meme resultat que pour PDS :

$$\{3, 9, 10, 16\}.$$

4.4 Implémentation Ensemble Indépendant Faiblement Connecté (PWCIS)

L'objectif du Programme pour la Couverture Presque Connexe (PWCIS) est de trouver un sous-ensemble de sommets dans un graphe qui soit presque connexe, c'est-à-dire que entre deux sommets quelconques de cet

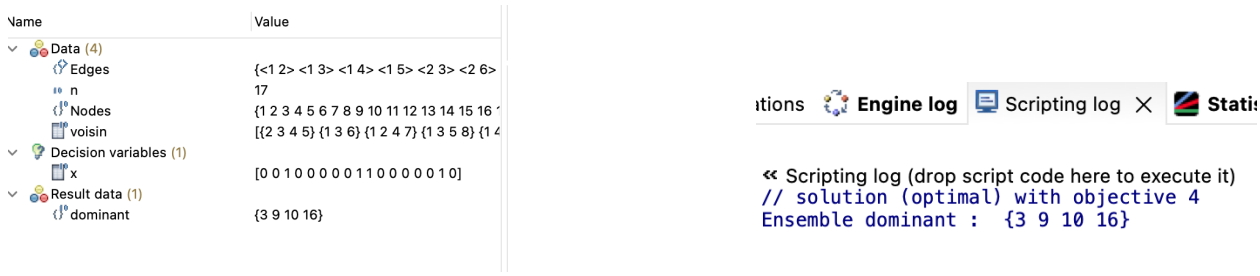

Figure 21 – Exemple 8(MIS) : Affichage des résultats sur la console.

ensemble existe une chaîne qui passe uniquement par des arêtes entre cet ensemble et son complémentaire dans le graphe, tout en étant dominant, c'est-à-dire que chaque sommet du graphe est soit inclus dans cet ensemble soit adjacent à au moins un sommet de cet ensemble. Le but ultime est de minimiser la cardinalité de cet ensemble, ce qui permet de réduire le coût associé à la couverture du réseau tout en maintenant une certaine connectivité.

Pour l'implémentation de cette partie, nous avons rencontré des difficultés car notre programme s'exécutait sans afficher de résultats. Malgré plusieurs tentatives de débogage, nous n'avons pas réussi à identifier le problème. Nous avons exploré différentes approches, mais sans succès.

Voici le fichier WCIS.mod :

```
int n = ...;
tuple Edge {int u; int v;}
{Edge} Edges = ...;
{int} Nodes = {i.u | i in Edges} union {i.v | i in Edges};

// Calcul des sous-ensembles de sommets et leur complémentaires
// 'S' est l'ensemble de tous les sous-ensembles possibles des sommets
// 'Sub' contient les sommets de chaque sous-ensemble
// 'Compl' contient les sommets qui ne sont pas dans le sous-ensemble correspondant
// 'bordW' est initialisé comme vide et contiendra les sommets frontières des sous-ensembles
range S=1..ftoi(round(2^(n)));
{int} Sub[s in S]={i|i in Nodes: (s div ftoi(round(2^(i-1))))mod 2==1};
{int} bordW[s in S]={};
{int} Compl[s in S]=Nodes diff Sub[s];
{int} voisin [Nodes] ;

// Calcul des voisins de chaque sommet et mise à jour des sommets frontières pour chaque
    sous-ensemble
execute {
   for (var e in Edges) {
      voisin[e.u].add(e.v);
      voisin[e.v].add(e.u);
   }

   for (var s in S) {
      for (var e in Edges) {
         if(Sub[s].contains(e.u) && Compl[s].contains(e.v)) {
            bordW[s].add(e.u);
         }
         if(Sub[s].contains(e.v) && Compl[s].contains(e.u)) {
            bordW[s].add(e.v);
         }
      }
   }
}

dvar boolean x[Nodes];
```

```
37 minimize sum(n in Nodes) x[n];
38
39 subject to {
40    // Chaque sous-ensemble de sommets doit avoir au moins un sommet frontière dans l'
       ensemble dominant
41    forall(s in S) {
42       sum(i in bordW[s]) x[i] >= 1;
43    }
44
45    // Aucune arête ne doit avoir ses deux extrémités dans l'ensemble dominant
46    // Ceci assure qu'on ne choisit pas plus de sommets que nécessaire
47    forall(edge in Edges) {
48       x[edge.u] + x[edge.v] <= 1;
49    }
50 }
51
52 // Construction de l'ensemble dominant à partir de la solution optimale
53 {int} dominant = {n | n in Nodes : x[n] == 1};
54 execute {
55    writeln("Ensemble dominant : ", bordW);
56 }
```

Difficultés rencontrées

Lors de ce deuxième TP, nous avons été confrontés à des obstacles plus importantes par rapport au premier TP. La résolution, l'implémentation et la modélisation des problèmes se sont révélées plus ardues, nécessitant une compréhension plus approfondie des concepts abordés.

En particulier, nous avons rencontré des obstacles significatifs dans la troisième partie, concernant le programme PWCIS. Bien que nous ayons travaillé avec attention pour transcrire correctement le programme dans notre code, nous avons été confrontés à un problème persistant : le programme ne parvient pas à afficher les résultats et semble charger indéfiniment. Malgré nos efforts, cette problématique demeure non résolue.

Conclusion

En conclusion, ce deuxième TP consacré au solveur IBM-Cplex a été une expérience enrichissante qui nous a permis de consolider notre maîtrise d'OPL à travers une diversité de problèmes. De la gestion des Flots aux problèmes d'optimisation mathématique, nous avons été confrontés à des défis variés qui nous ont aidés à mieux comprendre les implications pratiques de la modélisation linéaire.

En explorant ces différents cas d'utilisation, nous avons réalisé l'importance cruciale d'une analyse approfondie dans le cadre de projets impliquant de grandes infrastructures.

De plus, cette expérience nous a également révélé la puissance de l'implémentation de formulations mathématiques spécifiques pour résoudre des problèmes d'optimisation linéaire.

En somme, ce TP a été une étape importante dans notre parcours d'apprentissage, nous fournissant des outils et des perspectives précieuses pour aborder des défis similaires dans le domaine de l'informatique et de l'optimisation.

Table des figures