Université de Nantes

—

UFR Sciences et Techniques

Master informatique parcours optimisation en recherche opérationnelle

Année académique 2018-2019

Travaux Encadrés de Recherche :

Planning serré

Auteur :
LATIF Mehdi

Référent
RUSU Irena

28 mai 2019

Table des matières

1 Introduction

1.1 Contexte général

Ce rapport est la synthèse du travail réalisé dans le cadre du module de Travaux Encadrés de Recherche effectué en première année du master en informatique, mention Optimisation en Recherche Opérationnelle, proposé par l'Université de Nantes.
Le sujet de cet écrit effectué sous la tutelle de Madame Irena RUSU porte sur l'application de la théorie des graphes pour la réalisation d'un problème d'affectation.

1.2 Problématique

Le sujet sur lequel nous devons réaliser cette étude correspond à une problématique réelle rencontrée chaque année par le département informatique de l'Université de Nantes pour l'affectation de créneaux horaires de travaux pratiques aux membres l'équipe enseignante. La complexité de cette tâche peut nécessiter la mise en place d'une solution automatisée que nous avons tenté de développer dans le cadre de ce projet de recherche.

Dans le cadre des enseignements proposés par le département informatique, certains créneaux de travaux pratiques peuvent avoir lieu toutes les deux semaines ; ces derniers peuvent être placés soit sur les semaines paires, soit sur les semaines impaires.
Chaque créneau est fixé à l'avance par les membres du service des emplois du temps, les jours et les heures ne pouvant être modifiés.
Un créneau horaire dans notre problème est un triplet défini comme suit :

$$c(g) = (j(g), h(g), p(g))$$

où g est un groupe d'étudiants affecté à ce créneau déjà fixé, $j(g)$ et $h(g)$ sont le jour et l'heure de début de ce créneau et $p(g)$ la parité de la semaine sur laquelle le créneau a lieu.
Deux groupes ont des créneaux complémentaires si le jour et l'heure sont les mêmes mais que leurs parités sont différentes.
L'université possède plusieurs salles de travaux pratiques permettant à plusieurs groupes d'être placés sur les mêmes créneaux horaires.
Un ensemble d'enseignants est prévu pour assurer ces travaux pratiques. Chaque professeur indique l'ensemble des créneaux qu'il peut prendre en charge ainsi que le nombre de créneaux qu'il souhaite se voir affecter. Par exemple, un professeur peut avoir quatre disponibilités différentes mais peut demander à n'être affecté que sur deux d'entre eux. De la même manière, un professeur peut être disponible sur trois créneaux et demander à être affecter à quatre cours correspondant à ces disponibilités. En général, un professeur peut demander à être affecté sur x de créneaux et être disponible sur y créneaux avec $x \neq y$.
Nous posons comme hypothèse qu'un enseignant est indifférent au groupe associé à un créneau donné et qu'il n'a pas de préférence pour un créneau ou un autre si ce dernier correspond à une de ses disponibilités. Cependant, l'enseignant possédant au moins deux disponibilités préféra se voir attribuer des créneaux complémentaires plutôt que des créneaux répartis sur des jours et des heures différents.
Notons de plus qu'un créneau non complémentaire attribué à un professeur et donc bloqué sur une semaine paire, respectivement impaire, est implicitement bloqué pour la semaine impaire, respectivement paire, car inutilisable par un autre enseignant.

	Lundi		Mardi	
Parité	**Pair**	**Impair**	**Pair**	**Impair**
8h	1	3	2	8
10h	9			4
12h	5		6	7

Table 1 – Exemple d'une instance pour notre problème.

Dans l'exemple présenté ci-dessus, 9 groupes de travaux pratiques sont répartis sur deux jours et trois plages horaires et des professeurs sont affectés à ces derniers.
On suppose que les quatre professeurs ont indiqué les disponibilités suivantes :

p_1 **:** Lundi 8h, Mardi 8h, Mardi 12h et souhaite obtenir 4 créneaux.

p_2 **:** Lundi 9h, Mardi 8h, Mardi 12h et souhaite obtenir 2 créneaux.

p_3 **:** Lundi 8h, Lundi 12h, Mardi 10h et souhaite obtenir 2 créneaux.

p_4 **:** Lundi 10h, Mardi 12h et souhaite obtenir 1 créneaux.

Les triplets définissant les groupes sont :

$$c(g_1) = (1, 8, 0) \quad c(g_2) = (2, 8, 1) \quad c(g_3) = (1, 8, 1) \quad c(g_4) = (2, 10, 1)$$
$$c(g_5) = (1, 12, 0) \quad c(g_6) = (2, 12, 0) \quad c(g_7) = (2, 12, 1) \quad c(g_8) = (2, 8, 1)$$
$$c(g_9) = (1, 10, 0)$$

Dans cet exemple, ont des créneaux complémentaires les groupes (g_1, g_3), (g_2, g_8) et (g_6, g_7).
En prenant en compte les différentes disponibilités des professeurs sur les créneaux indiqués dans l'instance, nous obtenons les affectations suivantes :

	Lundi		Mardi	
Parité	**Pair**	**Impair**	**Pair**	**Impair**
8h	1 - p_1	3 - p_1	2 - p_2	8 - p_1
10h	9 - p_4			4 - p_3
12h	5 - p_3		6 - p_1	7 - p_2

Table 2 – Exemple d'une affectation pour l'instance présentée.

Le professeur p_1 est affecté à deux créneaux complémentaires ainsi qu'à un non complémentaire et p_2 possède deux affectations non complémentaires. On peut remarquer qu'un échange des enseignants pour les créneaux du Mardi 8h et Mardi 12h permettrait à p_1 et p_2 d'obtenir pour chacun de nouvelles affectations complémentaires.
Les professeurs p_3 et p_4 sont affectés à des créneaux conformément à leurs disponibilités et ne pouvant être modifiés.

1.3 Apports de la théorie des graphes pour la résolution de ce problème

L'objectif de cette étude est de proposer un algorithme efficace permettant d'affecter aux professeurs les créneaux en fonction de leurs disponibilités tout en tentant de maximiser le nombre de créneaux complémentaires affectés à un même enseignant et ceci, en utilisant une modélisation à l'aide de la théorie des graphes.
Nous présenterons dans ce rapport deux approches afin de proposer une solution à ce problème d'optimisation, l'une sous forme d'un problème de couplage et la seconde à l'aide d'un problème de flot.

2 Eléments de théorie des graphes

Nous allons à présent rappeler une série de définitions et théorèmes utilisés dans ce projet de recherche. Ces éléments sont issus des supports de cours *Théorie des graphes* et *Graphes et Réseaux* dispensés en Master Informatique à l'université de Nantes par I.RUSU[1, 2]

2.1 Définitions générales

Définition. *Graphe Orienté*
Un graphe orienté $G = (V, E)$ *est un modèle abstrait représenté par* V *un ensemble fini de sommets et* $E = (u, v) \subseteq V \times V$ *un ensemble d'arcs représentant une relation* $u\mathcal{R}v$ *sur* S.

Définition. *Graphe non orienté*
Un graphe non orienté $G = (V, E)$ *est un modèle abstrait représenté par* V *un ensemble fini de sommets et* $E = (u, v) \subseteq V \times V$ *un ensemble d'arêtes représentant une relation symétrique* $u\mathcal{R}v$ *sur* S *i.e.* $(u, v) = (v, u)\ \forall u, v \in V$.

Définition. *Chemin dans un graphe*
Dans un graphe $G = (V, E)$*, un chemin de longueur* k*, d'origine* v_0 *et d'extrémité* v_k *est une suite d'arêtes/arc* $C = ((v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)))$ *telles que* $(v_{i-1}, v_i) \in E\ \forall i \in [\![1, k]\!]$

Définition. *Graphes bipartis*
Soit $G = (V, E)$ *un graphe.* G *est biparti s'il est possible de trouver une partition* (L, R) *des sommets de* G *vérifiant les propriétés suivantes :*

1. $V = L \cup R$ *et* $L \cap R = \emptyset$
2. *Toutes les arêtes/arcs* $e \in E$ *possèdent une extrémité dans chaque ensemble de la partition* (L, R).

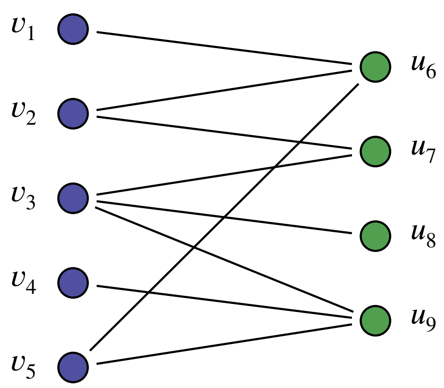

FIGURE 1 – Exemple de graphe biparti

2.2 Définitions relatives à notre problème

Comme nous l'avons évoqué dans l'introduction, nous travaillons sur le problème d'affectation de créneaux horaires en cherchant à maximiser le nombre de créneaux complémentaire et pour ce faire, nous tentons de le modéliser à l'aide de différents types de graphes. Nous allons à présent définir les objets et les propriétés dont nous aurons besoin pour résoudre notre problème.
Dans la suite des définitions, on considère un graphe biparti $G = (V, E)$ tel que $V = P \cup C$ avec C l'ensemble des créneaux considérés et P l'ensemble des professeurs auxquels nous souhaitons affecter les créneaux de C et E un ensemble d'arêtes.

Dans notre modélisation, un sommet $p \in P$ représente un professeur et un sommet $c \in C$ représente un créneau.
On rappelle succinctement les notations suivantes :

Définition.
On définit un créneau comme un triplet $c(g) = (j(g), h(g), p(g))$ avec g le groupe concerné, $j(g)$et $h(g)$ sont le jour et l'heure de début de ce créneau et $p(g)$ parité de la semaine sur laquelle le créneau a lieu.
On définit un professeur comme le triplet $p = (i, n, d)$ avec i le numéro du professeur, n le nombre de créneau(x) qu'il souhaite prendre en charge et d un ensemble de disponibilités i.e. des couples $(j(g), h(g))$ avec $j(g)$ le jour et $h(g)$ l'heure de la disponibilité d'un professeur, ceci indépendamment du groupe g par hypothèse.

Notation.
Dans notre modélisation, une semaine paire est représentée par $p(g) = 0$, $p(g) = 1$ sinon.

Définition. *Représentation de la disponibilité d'un professeur sur un créneau*
Un professeur $p = (i, n, d)$ peut être affecté à un créneau $c(g) = (j(g), h(g), p(g))$ si il existe une disponibilité de p dont le jour et l'heure coïncident avec ceux du créneaux $c(g)$. Une disponibilité est représentée dans le graphe G par une arête sortante du sommet $p \in P$ vers le sommet $c \in C$

Notation.
Nous introduisons une demande entière $d(v)$ sur chaque sommet de G telle que :

$$d(v) = \begin{cases} d < 0 & \textit{si } v \in P \textit{ où } d \textit{ est égal au nombre de cours souhaités par } v \\ d \geq 0 & \textit{si } v \in C \textit{ où } d \textit{ est égal au nombre de groupes présents sur le créneau } v \end{cases}$$

L'algorithme suivant présente la méthode de construction du graphe biparti pour notre problème d'affectation et ceci à partir d'un ensemble X de créneaux et Y un ensemble de professeurs.

Algorithme 1 : Construction du graphe biparti

Entrées : X un ensemble de professeurs, Y un ensemble de créneaux
Sorties : $G = (P \cup C, E)$ un graphe biparti
$P \leftarrow \emptyset$ `// L'ensemble des sommets professeurs`
$C \leftarrow \emptyset$ `// L'ensemble des sommets créneaux`
$E \leftarrow \emptyset$ `// L'ensemble des arêtes`
$\forall\ x \in X$ **faire**
    $C \leftarrow C \cup$ {**sommet**(x)}
$\forall\ y \in Y$ **faire**
    $P \leftarrow P \cup$ {**sommet**(y)}
    $\forall\ x' \in C$ **faire**
        Si *Le professeur y est disponible sur le créneau x'* **alors**
            $E \leftarrow E \cup$ {**arête**(y, x')}
retourner $G = (P \cup C, E)$

Dans le pseudo-code ci-dessus, la fonction **sommet(.)** prend en paramètre un professeur ou un créneau et retourne un sommet pour notre graphe ; Nous avons besoin de stocker les triplets dans chaque noeuds. la fonction **arête(.,.)** retourne quant à elle, l'arête entre deux sommets.

2.3 Construction d'une instance pour le problème d'affectation

Pour illustrer notre construction, nous allons nous intéresser à une instance du problème défini comme suit : On suppose que l'équipe enseignante est composée de 3 professeurs ayant les disponibilités suivantes :

	p_1	p_2	p_3
Disponibilités	Lu 14h, Ma 14h	Ma 14h, Me 14h	Lu 14h, Ma 14h, Me 14h
Nb cours souhaités	2	2	3

Table 3 – Instance illustrative pour nos modélisations - Ensemble de professeurs

On considère également un ensemble de créneaux défini de la manière suivante :

Semaine	Paire	Impaire
Lu 14	1,3	2
Ma 14	5	4,6
Me 14	7,8,9	

Table 4 – Instance illustrative pour nos modélisations - Ensemble de créneaux

Les chiffres à l'intersection des colonnes et des lignes représentent les numéros des groupes affectés à ces créneaux horaires. Rappelons qu'un créneau horaire peut être attribué à plusieurs groupes en même temps, il est donc de notre ressort de faire en sorte de trouver une affectation optimale pour chacun des professeurs.

Après construction, nous obtenons le graphe biparti suivant :

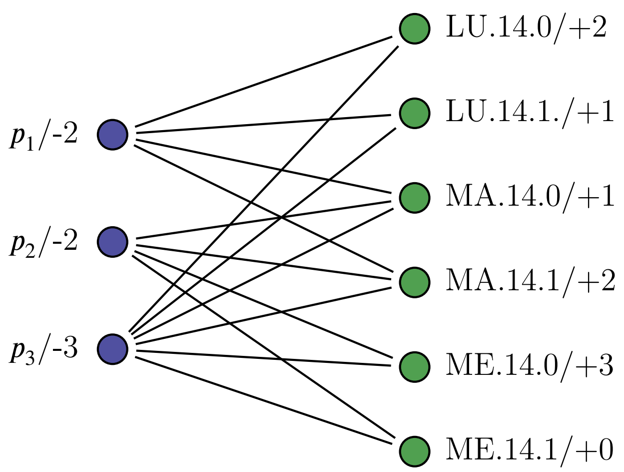

Figure 2 – Graphe biparti représentant l'instance illustrative

A partir de tel graphe, nous allons pouvoir commencer à appliquer nos modélisations sous forme de couplage et d'un flot pour pouvoir résoudre ce problème d'affectation.

3 Modélisations proposées pour la résolution du problème d'affectation

Nous allons à présent nous intéresser aux modélisations mises en oeuvre pour tenter de résoudre ce problème d'affectation en particulier. Comme nous l'avons évoqué précédemment, nous avons développé deux méthodes pour apporter une solution à ce problème, la première utilisant une formulation du problème sous forme d'un couplage et la seconde, utilisant une modélisation sous forme d'un réseau et la recherche d'un flot maximum de coût minimum.

3.1 Recherche d'une solution par modélisation sous forme d'un problème de couplage

A présent, nous allons étudier une première formulation du graphe obtenu par construction sous la forme d'un problème de couplage maximum dans un graphe biparti que nous allons modifier pour tenter d'obtenir des affectations complémentaires.

3.1.1 Définitions relatives au problème de couplage

Nous allons rappeler quelques définitions utiles pour la résolution du problème d'affectation optimale sous forme d'un couplage. Nous posons $G = (V, E)$ un graphe non orienté avec $n = |V|$ et $m = |E|$.

Définition. *Couplage*
Un couplage M est un sous ensemble d'arêtes de G ($M \subseteq E$) tel que tout sommet $v \in V$ est incident à au plus une arête de M.
Nous dirons alors que $e \in E$ est une arête du couplage si et seulement si $e \in M \subset E$

Définition. *Sommet saturé*
Un sommet $v \in V$ est dit saturé si il est incident à une arête de M. Dans le cas contraire, il est dit exposé.

Définition. *Cardinalité d'un couplage*
On appelle cardinalité d'un couplage, le nombre d'arêtes contenues dans M notée $|M|$

Définition. *Couplages maximal et maximum*
Un couplage M est dit maximal s'il est impossible d'ajouter une nouvelle arête au couplage.
Un couplage M est dit maximum s'il est de cardinalité maximale i.e. il est impossible de trouver un couplage de taille supérieure dans G

Définition. *Chemin alternant*
Un chemin alternant par rapport a un couplage M est un chemin dont les arêtes dans M et les arêtes dans $E \setminus M$ alternent.

Définition. *Chemin augmentant*
un chemin alternant par rapport à M et dont les extrémités sont exposées est un chemin augmentant par rapport à M.

Définition. *Différence symétrique*
Soient A et B deux ensembles discrets, la différence symétrique de ces deux ensembles, notée $A \Delta B$, est définie par :

$$A \Delta B = (A \cup B) \setminus (A \cap B) = (A \setminus B) \cup (B \setminus A)$$

Algorithme. *Opération de transfert*
L'opération de transfert d'un couplage M par rapport à un chemin augmantant $\mathcal{P}$ correspond à l'application de la différence symétrique i.e. l'opération qui enlève de M toutes les arêtes du couplage qui sont dans $\mathcal{P}$ et ajoute à M les autres arrêtes de $\mathcal{P}$. Soient M', le couplage obtenu par application de l'opération de transfert :

$$M' = M \Delta \mathcal{P}$$

Par application de l'opération de transfert sur un couplage M le long d'un chemin augmentant $\mathcal{P}$, on obtient un nouveau couplage M' de cardinalité $|M'| = |M| + 1$

Théorème. *de Berge*
Un couplage $\boldsymbol{M}$ *d'un graphe* $\boldsymbol{G}$ *est maximum si et seulement si il n'existe pas de chemin augmentant* $\mathcal{P}$ *dans* $\boldsymbol{G}$.

3.1.2 Définitions spécifiques à notre problème formulé à l'aide d'un couplage

Nous finissons cette partie avec la définition de deux notions qui vont nous être utiles dans la suite du rapport pour pouvoir savoir s'il est possible d'améliorer notre couplage. Dans le graphe biparti $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ considéré, une arête du couplage est interprété comme une affectation d'un créneau à un professeur.

Définition. *Conflit horaire*
On dit qu'il existe un conflit horaire entre le professeur $\boldsymbol{p}$ *et le créneau* $\boldsymbol{c}(g)$ *si il existe un créneau* $c(g')$ *tel que* $(p, c(g')) \in \boldsymbol{M}$ *et*

$$j(g) = j(g') \wedge h(g) = h(g') \wedge p(g) = p(g')$$

Définition. *Complémentarité de deux créneaux*
Soient $\boldsymbol{c}(g) = (j(g), h(g), p(g))$ *et* $\boldsymbol{c}(g') = (j(g'), h(g'), p(g'))$ *;* $\boldsymbol{c}(g)$ *et* $\boldsymbol{c}(g')$ *sont complémentaires si et seulement si*

$$j(g) = j(g') \wedge h(g) = h(g') \wedge p(g) \neq p(g')$$

Pour une telle formulation de ce problème, nous allons maintenant chercher à obtenir un couplage maximum $\boldsymbol{M}$ où les arêtes $\boldsymbol{e} = (p, c(g)) \in \boldsymbol{M}$ représenteront des affectations de professeurs aux créneaux conformément à leurs disponibilités.

3.1.3 Résolution du problème d'affectation sur le graphe biparti

En l'état actuel de notre modélisation pour le problème d'affectation, il nous est impossible de rechercher un couplage sur notre graphe biparti ; En effet, cette configuration ne prend pas en considération les demandes des sommets et la possibilité qu'un professeur se voit affecter des créneaux sur des jours différents et ceci, par contrainte d'unicité de l'arête incidente à sommet dans $\boldsymbol{M}$.
Nous devons donc opérer une transformation de ce dernier pour pouvoir rechercher un couplage maximum.
La transformation que nous allons réaliser sur notre couplage consiste en la mutliplication des sommets en fonction des demandes $|d(v)|\ \forall v \in V = \boldsymbol{P} \cup \boldsymbol{C}$. Ainsi, le professeur p_1 ayant formulé une demande de deux créneaux se verra représenté par deux sommets dans le nouveau graphe biparti à savoir $p_{1,1}$ et $p_{1,2}$. La même transformation, opérée sur les sommets représentants des créneaux, engendrera pour le cours du lundi à 14h en semaine paire deux nouveaux sommets $(Lu, 14, 1, 0)$ et $(Lu, 14, 3, 0)$ où 1 et 3 sont les groupes fixés sur ce créneau.
Le nouveau graphe biparti obtenu pour l'instance illustrative est le suivant :

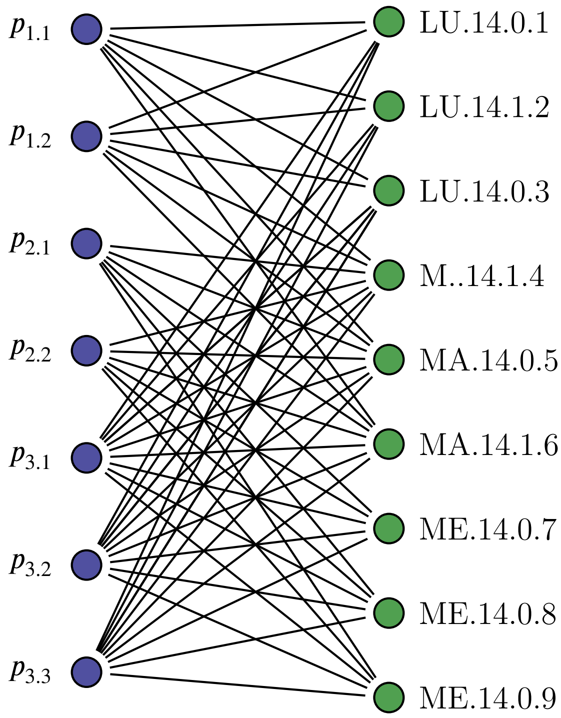

FIGURE 3 – Graphe biparti représentant l'instance illustrative - Duplication des sommets

Notons que dans cet exemple, le créneau du mercredi en semaine impaire ne contenait aucun groupe, il est donc logique qu'il n'apparaisse pas dans ce nouveau graphe.

Nous pouvons à présent appliquer la recherche d'un couplage maximum sur notre graphe biparti transformé. Le pseudo code présenté ci-dessous est issu du Wolsey[3] étudié à l'Université Libre de Bruxelles dans le cadre du cours d'optimisation combinatoire. Notons que cet algorithme a initialement été présenté dans le cours de Graphe donné par I.Rusu[1]. Cet algorithme s'exécute de la manière suivante :

Algorithme 2 : Algorithme de recherche d'un couplage maximum dans un graphe biparti

Entrées : $G = (V = P \cup C, E)$ un graphe biparti
Sorties : M un couplage maximum
$M \longleftarrow \emptyset$;
Etape 1 *Étiquetage*
  Etape 1.0 :
    Donner une étiquette $\star$ à tous les sommets exposés de P;
  Etape 1.1 :
    S'il n'existe pas d'étiquette non encore vérifée $\rightarrow$ Aller en étape 3;
    Choisir un sommet étiqueté $v \in V$ mais non encore examiné.;
      Si $v \in P \rightarrow$ Aller en étape 1.2;
      Si $v \in C \rightarrow$ Aller en étape 1.3;
  Etape 1.2 :
    Enregistrer le sommet $v \in P$;
    $\forall (u, v) \in E \setminus M$, donner à v une étiquette égale à u si v n'a pas encore d'étiquette.;
    Aller en étape 1.1;
  Etape 1.3 :
    Enregistrer le sommet $v \in C$. Si v est exposé, aller en étape 2.;
    Sinon, trouver l'arête $(u, v) \in M$ et donner à $u \in P$ une étiquette égale à v ;
    Aller en étape 1.1;
Etape 2 *Augmentation*
  Un chemin augmentant $\mathcal{P}$ a été trouvé. Utiliser les étiquettes pour construire le chemin à reculons à partir de $u \in C$;
  Appliquer l'opération de transfert $M \leftarrow M \Delta \mathcal{P}$;
  Effacer les étiquettes;
  Aller à l'étape 1.
Etape 3 *Terminaison*
  Le couplage M est maximum *i.e.* il n'existe plus de chemin augmentant dans G;
retourner *M le couplage maximum dans G*

La complexité de cet algorithme est en $O(|V| \times |E|)$ car les opérations d'étiquetage, de recherche de chemin et d'augmentation peuvent se réaliser dans le pire des cas en $O(E)$ (par parcours de l'ensemble des arêtes de G) et l'ensemble des étapes est effectué en pire cas $|V|$ fois.
L'application de cet algorithme pour la recherche d'un couplage maximum dans l'instance illustrative nous permet d'obtenir le couplage suivant :

$$\begin{aligned} M = \quad & \{(p_{1,1}, Lu.14.0.3), (p_{1,2}, Ma.14.1.1), (p_{2,1}, Ma.14.0.5)\ (p_{2,2}, Me.14.0.8), \\ & (p_{3,1}, Lu.14.1.2), (p_{3,2}, Me.14.0.9), (p_{3,3}, Ma.14.1.6)\ \} \end{aligned}$$

L'affichage console obtenu avec notre implémentation en Java est présenté ci-dessous.

```
Prof n° 1 nbVoulu : 2 Dsp : [(1, 14h), (2, 14h)]
Nb créneaux cplt : 0
1/14/3/0 2/14/4/1
Prof n° 2 nbVoulu : 2 Dsp : [(2, 14h), (3, 14h)]
Nb créneaux cplt : 0
2/14/5/0 3/14/8/0
Prof n° 3 nbVoulu : 3 Dsp : [(1, 14h), (2, 14h), (3, 14h)]
Nb créneaux cplt : 0
1/14/2/1 3/14/9/0 2/14/6/1
```

TABLE 5 – Couplage obtenu - Affichage console avant échange

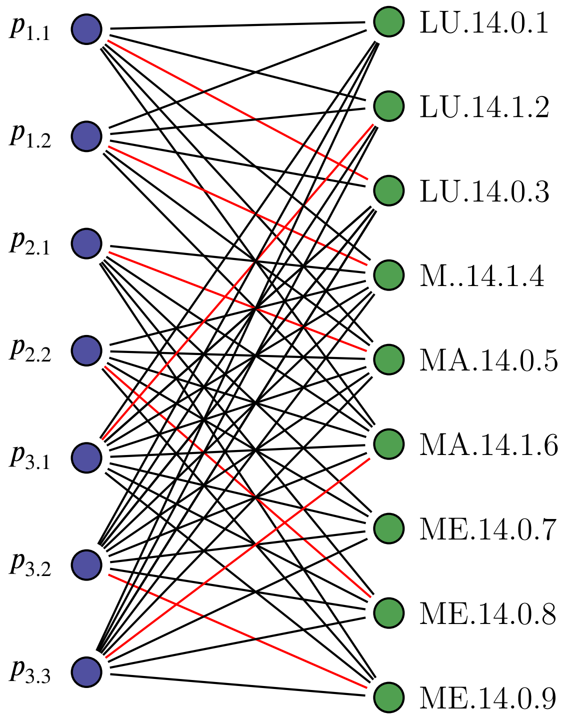

FIGURE 4 – Graphe biparti représentant l'instance illustrative - Couplage maximum

Comme nous pouvons l'observer sur la figure précédente et dans l'affichage console, les affectations obtenues par l'application de l'algorithme de recherche d'un couplage maximum sont correctes par rapport aux disponibilités de chaque professeurs et au nombre de créneaux souhaités par ces derniers. Cependant, ces affectations ne sont pas optimales dans la recherche de créneaux complémentaires. Étant donné que nous savons que ce couplage est maximum, nous allons devoir trouver une méthode basée sur des échanges d'affectations permettant de modifier les arêtes de ce couplage et ainsi maximiser le nombre de créneaux complémentaires pour chacun des professeurs.
Nous introduisons une nouvelle définition pour caractériser les sommets :

Définition. *Sommets fixés*
Nous appelons sommets fixés, des sommets pour lesquels nous avons réussi à générer deux affectation de créneaux complémentaires i.e. $(p, c(g))$ *et* $(p', c(g'))$ *sont des arêtes de* M *telles que* $c(g)$ *et* $c(g')$ *sont complémentaires,* p *et* p' *sont des sommets représentant le même professeur à un ordre de multiplicité près.*

Il serait tout à fait envisageable de remettre ce choix en question et de fixer un sommet dès lors qu'il possède au moins quatre ou tous ces créneaux complémentaires ; l'impact de cette hypothèse sur l'obtention de meilleures affectations pourra être étudié dans un second temps.
Nous émettons également l'hypothèse suivante :

Hypothèse. *Fixation irrévocable*
Dès lors qu'un sommet est fixé, celui-ci ne peut être échangé pour tenter d'obtenir une nouvelle affectation complémentaire.

Cette hypothèse sur notre problème, bien que simplificatrice et pouvant nous empêcher l'obtention de meilleures affectations, est un choix défini afin d'assurer d'assurer la terminaison de notre procédure d'échange.

Comme nous l'avons évoqué précédemment, l'opération visant à modifier le couplage pour obtenir des créneaux complémentaires consiste en l'échange d'affectations le long de chemins alternants. Supposons que $\boldsymbol{M}$ contiennent les affectations suivantes $(p_{1,1}, c(g_1))$ $(p_{1,2}, c(g_2))$, $(p_{2,1}, c(g_3))$ et $(p_{3,1}, c(g_4))$, ces dernières sont chacune non complémentaires. Supposons de plus qu'après un parcours de la partition $\boldsymbol{C}$ du graphe, nous observons que si l'on conserve l'affectation $(p_{1,1}, c(g_1)) \in \boldsymbol{M}$, p_1 peut obtenir un créneau complémentaire avec $c(g_4)$ s'il échange le créneau affecté à $p_{1,2}$, $c(g_2)$ avec $c(g_4)$ initialement affecté $p_{3,1}$. L'idée de l'échange va consister à forcer l'ajout de l'arête $(p_{1,2}, c(g_4))$ dans $\boldsymbol{M}$; cela ne sera possible que si nous arrivons à trouver un chemin alternant permettant de modifier le couplage tout en conservant sa propriété d'être maximum.

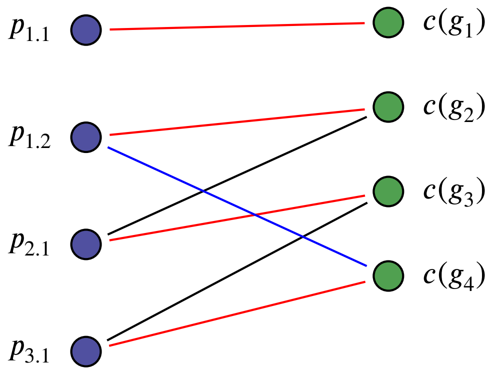

FIGURE 5 – Procédure d'échange - Graphe initial

Sur le graphe ci-dessus, sont en rouges les arêtes des $\boldsymbol{M}$ et l'arête dont nous souhaitons forcer l'ajout dans le couplage est colorée en bleu. Un chemin alternant au départ de $p_{1,2}$ vers le créneau $c(g_4)$ est détecté dans le graphe et est donné par la séquence d'arêtes suivantes :

$$\mathcal{P} = \left\{(p_{1,2}, c(g_2)), (c(g_2), p_{2,1})), (p_{2,1}, c(g_3)), (c(g_3), p_{3,1}), (p_{3,1}, c(g_4))\right\}$$

Dès lors que nous obtenons ce chemin alternant $\mathcal{P}$, nous devons à présent vérifier que les arêtes de $\mathcal{P}$ qui sont dans $\boldsymbol{E}$,qui vont être modifiées par application de la procédure d'échanges et ajoutées dans le couplage, n'entraînent pas conflit horaire pour les autres professeurs. Nous supposerons que dans cet exemple, aucun conflit n'est détecté.
Si nous avons pu vérifier que le chemin alternant trouvé est valide pour cet échange de créneaux (ne génère pas de conflit), alors nous pouvons appliquer la différence symétrique $\boldsymbol{M} \leftarrow \boldsymbol{M} \Delta \mathcal{P}$ et obtenir le nouveau couplage présenté ci-dessous :

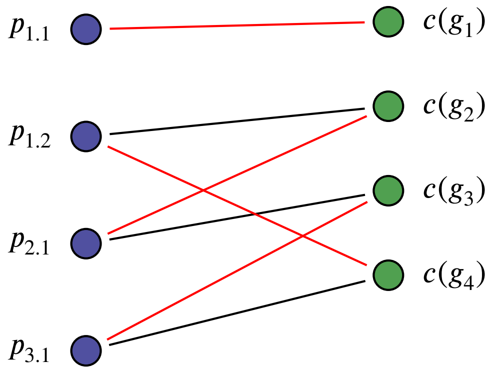

FIGURE 6 – Procédure d'échange - Graphe modifié

Nous pouvons à présent fixer les sommets des arêtes $(p_{1,1}, c(g_1))$ et $(p_{1,2}, c(g_4))$ car ces affectations sont complémentaires par hypothèse sur cet exemple.
Ainsi, en utilisant les chemin alternants dans le graphe, nous avons pu générer des affectations complémentaires et ceci sans modifier la cardinalité du couplage.

La procédure de recherche des chemins augmentant dans un graphe peut être implémentée sur la base d'un algorithme de parcours en profondeur dans lequel nous ajoutons une information nous permettant d'alterner le choix des arêtes dans M et $E \setminus M$. Pour ce faire, nous orientons les arêtes du graphe de la manière suivante : Nous définissons une variable booléenne $PtoC$ telle que

$$PtoC = \begin{cases} \text{Vrai} & \text{si } e \in M \\ \text{Faux} & \text{si } e \in E \setminus M \end{cases}$$

Nous déclarons également un ensemble Q d'arête du graphe permettant de stocker le chemin alternant s'il en existe un. Avant de débuter la procédure récursive, nous initialisons Q avec le sommet prof depuis lequel on souhaite démarrer la recherche.
Nous présentons alors l'algorithme de parcours en profondeur pour la recherche de chemin alternants :

Algorithme 3 : DFS_Alternant() - Algorithme de recherche des chemins alternants dans le graphe

Entrées : u le sommet de départ, d le sommet d'arrivée, Q le chemin alternant
Sorties : Q le chemin alternant contenant des arêtes de u vers d

```
visite[u] ← Vrai
PtoC ← Faux
Si u = d alors
  | retourner Q
fin
Si u est un sommet prof alors
  | PtoC ← Vrai
fin
∀ e ∈ N(u) faire
  | v ← sommetOpposé(e,u)
  | Si v n'est pas fixé ∧ v n'est pas visité ∧ e = PtoC alors
  |   | Q ← Q ∪ {v}
  |   | DFS_Alternant(v,d,Q)
  |   | Q ← Q \ {v}
  | fin
fin
visite[u] ← Faux
```

Dans cet algorithme, $\mathcal{N}(u)$ représente l'ensemble des arêtes incidentes au sommet u.
Cet algorithme nous permet de parcourir le graphe G en profondeur en alternant le chemin entre des arêtes de M et de $E \setminus M$. En effet, l'utilisation du booléen $PtoC$ nous permet d'orienté la recherche depuis un sommet telle que :
— si le sommet de départ u est dans la partition P, alors nous devons continuer le chemin vers C en suivant une arête de M
— si le sommet de départ u est dans la partition C, alors nous devons continuer le chemin vers P en suivant une arête de $E \setminus M$
Notons que dans ce parcours, nous réinitialisons la valeur du sommet $u \in V$ à la fin du parcours, cette opération nous permet de détecter l'ensemble des chemins alternants pouvant exister entre les sommets de départ et d'arrivée.

Nous effectuons alors les opérations de recherche de chemin alternants et de fixation de sommets tant que nous n'avons pas évalué la possibilité de réaliser des échanges dans les affectations professeurs/créneaux.

Algorithme 4 : Optimisation des affectations - Fonction d'appel

```
Q ← ∅                                    // Initialisation d'une liste de sommets "échangeables"
Pour tout p ∈ P faire
    Si p n'est pas fixé alors
        Q ← Q ∪ {p}
    fin
fin
Tant que Q ≠ ∅ faire
    p ← Tête (Q)
    Si (p n'est pas fixé) ∧ (p est affecté à un créneau) alors
        Echange (p)
    fin
fin
```

Pour expliciter l'algorithme de recherche d'améliorations dans les affectations, nous définitions les fonctions et notations suivantes :

autreSommetProf(.) : retourne l'ensemble des sommets associés à un même professeurs possédant un ordre de multiplicité différent.

gM(.) : acronyme de Get Mate, prend en paramètre un sommet retourne le sommet affecté à x par dans le couplage M.

$\mathbf{M_V}$ **:** l'ensemble des sommets V du couplage M pour un graphe biparti $G = (V, E)$.

cheminAlter(.,.) : retourne l'ensemble des chemins alternants entre deux sommets obtenus grâce à la procédure **DFS_Alternant(.,.,.)** présentée plus haut dans le rapport.

Algorithme 5 : Echange(p) - Procédure de recherche d'échanges d'affectations

```
Entrées : p un sommet de la partition P
stopR ← Faux                             // Booléen - arrêt de la recherche d'améliorations
Q ← autreSommetProf (p)
Tant que (|Q| ≠ ∅) ∧ (¬ stopR) faire
    oP ← Tête (Q)
    Si (oP non fixé) alors
        Si (p ∈ M_V) ∧ (oP ∈ M_V) ∧ (gM(p) et gM(oP) complémentaires) alors
            // Une complémentarité entre deux affectations initialement présentes à été détectée.
            On fixe les sommets p, gM(p), oP et gM(oP)
            stopR ← Vrai
        fin
        // Aucune fixation simple n'est possible, on cherche alors un chemin alternant pour tenter de trouver
           une nouvelle affectation
        H ← ∅                            // Recherche des créneaux possibles
        Pour tout c ∈ C faire
            Si (c non fixé) ∧ (p dispo sur c) ∧ ( gM(c) ≠ p) ∧ (gM(c) ≠ oP) alors
                H ← H ∪ {c}
            fin
        fin
        Tant que (|H| ≠ ∅) ∧ (¬ stopR) faire
            c ← Tête (H)
            Si (gM(oP) ∈ M_V) ∧ (c ∈ M_V)∧ (gM(p) et c complémentaires) alors
                // Une complémentarité entre deux affectations à été détectée. Recherche des chemins alternants.
                A ← cheminAlter(oP,c)
                Tant que |A| ≠ ∅ ∧ (¬stopR) faire
                    𝒫 ← Tête (A)
                    pivot ← Arc(p,gM(p))
                    Si (𝒫 ne génère pas de conflits dans les affectations courantes) ∧ pivot ∉ 𝒫 alors
                        // 𝒫 est un chemin alternant valide, on peut procéder à la modification de M et ajouter la nouvelle
                           affectation de oP
                        M ← M Δ 𝒫
                        M ← M ∪ {(oP, c)}
                        On fixe les sommets p, gM(p), oP et c
                        stopR ← Vrai
                    fin
                fin
            fin
        fin
    fin
fin
```

Pour finir, nous présenterons le couplage obtenu grâce à la méthode d'échange suivant des chemins alternants appliquée à l'instance illustrative. Le couplage obtenu est le suivant :

$$M = \{(p_{1,1}, Lu.14.0.3), (p_{1,2}, Lu.14.1.2), (p_{2,1}, Ma.14.1.4), (p_{2,2}, Me.14.0.8), (p_{3,1}, Ma.14.0.5), (p_{3,2}, Me.14.0.9), (p_{3,3}, Ma.14.1.6)\}$$

L'affichage console obtenu avec notre implémentation en Java est présenté ci-dessous.

```
Prof n° 1 nbVoulu : 2 Dsp : [(1, 14h), (2, 14h)]
Nb créneaux cplt : 2
1/14/3/0 1/14/2/1
Prof n° 2 nbVoulu : 2 Dsp : [(2, 14h), (3, 14h)]
Nb créneaux cplt : 0
2/14/4/1 3/14/8/0
Prof n° 3 nbVoulu : 3 Dsp : [(1, 14h), (2, 14h), (3, 14h)]
Nb créneaux cplt : 2
2/14/5/0 3/14/9/0 2/14/6/1
```

Table 6 – Couplage obtenu - Affichage console après échange

Nous pouvons donc observer que l'application de cette méthode d'échange à partir des chemins alternants sur le graphe biparti en explorant le couplage maximal nous permet de générer deux affectations complémentaires pour des professeurs alors que le premier résultat ne nous en proposait aucune.

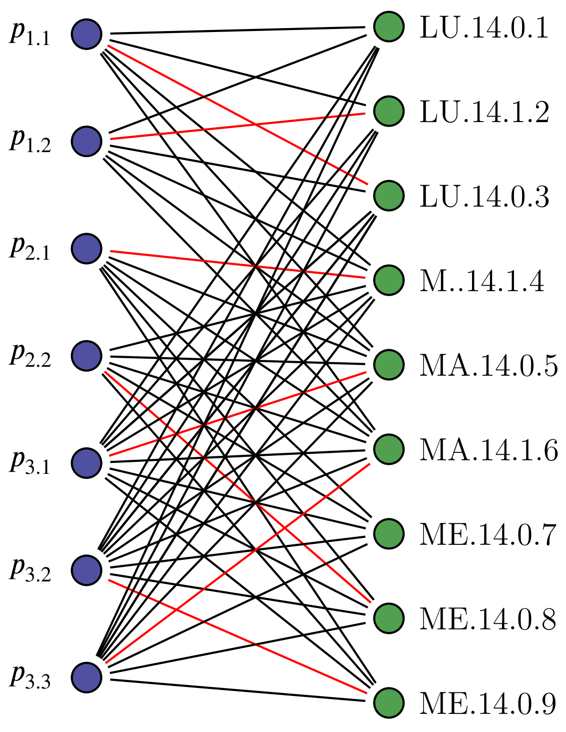

Figure 7 – Graphe biparti représentant l'instance illustrative - Couplage maximum amélioré

3.1.4 Conclusion de l'approche

L'utilisation d'un algorithme de recherche de couplage dans le graphe biparti semble donc être une voie prometteuse pour tenter d'obtenir une solution admissible la plus proche de l'optimalité possible.
Il serait cependant intéressant de voir l'impact de l'aléatoire pour la génération du couplage initial et du tri de la liste des sommets à examiner lors de la phase d'amélioration des affectations. Notons également qu'il serait possible de définir une gestion plus *intelligente* basée sur le nombre de créneaux souhaité par chacun des professeurs, les disponibilités ou encore l'étude a priori de l'impact d'une affectation sur la solution finale obtenue afin d'obtenir de meilleurs résultats.

3.2 Recherche d'une solution par modélisation sous forme d'un problème de flot

Après avoir détaillé une première formulation pour le problème d'affectation optimale, nous allons étudier une seconde formulation basée sur un problème de recherche du flot maximum de coût minimum. L'utilisation d'une méthode de flot pour la résolution est motivée par le fait que bon nombre de problèmes d'optimisation combinatoire peuvent être réduits en problème de recherche d'un flot maximum. De plus, nous avons étudié durant les travaux dirigés du cours de Graphes et Réseaux, la réduction du problème de couplage maximum dans un graphe biparti [1]Partant du constat que les algorithmes sur les flots peuvent être des outils beaucoup plus généraux que des algorithmes dédiés et ceci avec des méthodes possédant une complexité beaucoup plus intéressante, nous allons présenter et développer les transformations que nous avons mises en place pour résoudre notre problème.

3.2.1 Définitions relatives au problème de flot

Nous allons à présent rappeler quelques définitions utiles pour la résolution du problème d'affectation optimale sous forme d'un problème de flot maximum de coût minimum. Dans cette partie, nous allons considérer un graphe orienté valué $G = (V, E, c_{\min}, c_{\max}, w)$.

Définition. *Réseau*
Un Réseau est un graphe $G = (V \cup \{s,t\}, E, c_{\min}, c_{\max}, w)$ *avec* s *la source et* t *le puit tel que tout sommet de* V *se trouve sur un chemin entre* s *et* t*. Chaque arc* $e = (u, v) \in E$ *possède une capacité minimale* $c_{\min}(u, v) \in \mathbb{Z}$ *telle que* $c_{\min}(u, v) \geq 0$ *si* $(u, v) \in E$ *et* $c_{\min}(u, v) = 0$ *si* $(u, v) \notin E$ *, une capacité maximale* $c_{\max}(u, v) \in \mathbb{Z}$ *telle que* $c_{\max}(u, v) \geq 0$ *si* $(u, v) \in E$ *et* $c_{\max}(u, v) = 0$ *si* $(u, v) \notin E$ *et un coût* $w(u, v) \in \mathbb{Z}$ *tel que* $w(u, v) = 0$ *si* $(u, v) \notin E$*,* $w(u, v)$ *est fixé à une valeur arbitraire sinon. Notons également que le coût suit une relation d'antisymétrie telle que* $w(v, u) = -w(u, v)\ \forall u, v \in V$*.*

Définition. *Flot*
Un flot dans un réseau G *est une fonction* $f : V \times V \to \mathbb{Z}$ *qui doit vérifier la contrainte de capacité définie par* $c_{\min} \leq f(u, v) \leq c_{\max}(u, v)\ \forall u, v \in V$*, le principe de conservation du flot tel que* $\forall v \in V \setminus \{s, t\}$*,* $\sum(f(v, u) | u \in S) = 0$ *et la relation d'antisymétrie telle que* $\forall u, v \in V, f(v, u) = -f(u, v)$*.*

Définition. *Valeur et coût d'un flot*
Soit $G = (V \cup \{s,t\}, E, c_{\min}, c_{\max}, w)$ *un réseau.*
La valeur d'un flot est donnée par la relation suivante :

$$|f| = \sum (f(s, v) | v \in V)$$

et vérifie les propriétés suivantes :

1. Dans ce problème, nos avions n étudiants experts en programmation et m étudiants experts en algorithmique ; L'objectif était de créer un nombre maximum de binômes composés d'étudiants ayant chacun une expertise différente.

— $f(u,u) = 0 \ \forall u \in V$ - *On ne permet pas à un sommet de conserver une unité de flot.*
— $\forall u \in V \setminus \{s,t\} \ \sum (f(u,v) | v \in V) = 0$ - *Conservation du flot.*
— $|f| = \sum (f(u,t) | v \in V)$ - *La valeur du flot arrivé au puit est égale à celle du flot envoyé depuis la source.*

Le coût d'un flot est donné par la relation suivante :

$$w(f) = \sum (f(u,v) \times w(u,v) | f(u,v) > 0)$$

Définition. *Flot maximum de coût minimum*
Le flot maximum de coût minimum est un flot f dont la valeur $|f|$ est maximum et qui a, en plus, le coût minimum $w(f)$ parmi les flots de valeur maximum

Reprenons alors le graphe biparti obtenu représentant l'instance illustrative que nous utilisons depuis le début de ce rapport.

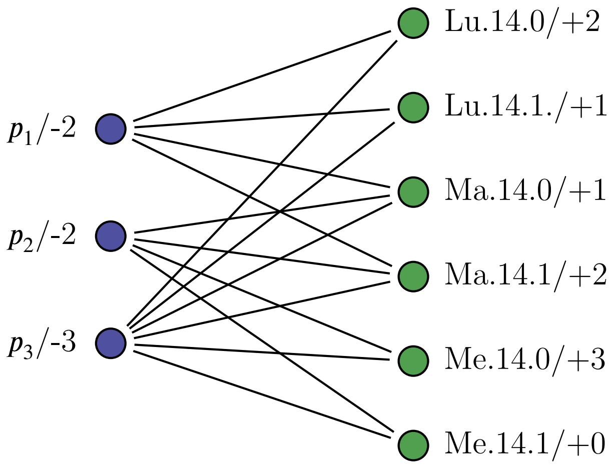

FIGURE 8 – Graphe biparti représentant l'instance illustrative

3.2.2 Construction du réseau à partir du graphe biparti

La transformation d'un tel graphe biparti en problème de flot à partir d'un graphe biparti $G = (V = (P \cup C), E)$ s'obtient en construisant le réseau correspondant $G' = (V', E')$ en suivant les étapes suivantes :

Étape 1 : Nous ajoutons deux nouveaux sommets s la source et t le puit tels que $V' = V \cup \{s,t\}$.

Étape 2 : Nous orientons les arêtes initialement présentes dans E de la partition P vers C et nous ajoutons de nouveaux arcs depuis et vers les sommets $s,t \in V'$ nouvellement créés tels que $E' = \{(s,u) : u \in P\} \cup \{(u,v) : (u,v) \in E\} \cup \{(v,t) : v \in C\}$.

Étape 3 : Pour les arêtes initialement présentes dans l'ensemble E, nous limitons la capacité maximale $c_{\max}(e)$ à une unité de flot et une capacité minimale $c_{\min}(e)$ égale à 0.

Étape 4 : Pour les sommets $v \in V$ ayant une demande $d(v) \neq 0$, si

- **$d(v) < 0$:** Nous fixons la valeur de la capacité maximale de l'arc $(s,v) \in E'$ à la valeur de la demande $|d(v)|$ et nous fixons la capacité minimale à 0
- **$d(v) > 0$:** Nous fixons la valeur de la capacité maximale de l'arc $(v,t) \in E'$ à la valeur de la demande $d(v)$ et nous fixons la capacité minimale à 0

Étape 5 : Nous considérons dans un premier temps que le coût de l'arc $e \in E$ est fixé à une valeur arbitraire c.

Notation. *Valuation des arcs*
L'écriture de l'information sur les arcs est définie comme suit :

Capacité minimale | *Valeur du flot* | *Capacité maximale* | *Coût*

Par application des différentes étapes de construction, nous obtenons pour l'instance illustrative le réseau $G' = (V', E')$ suivant :

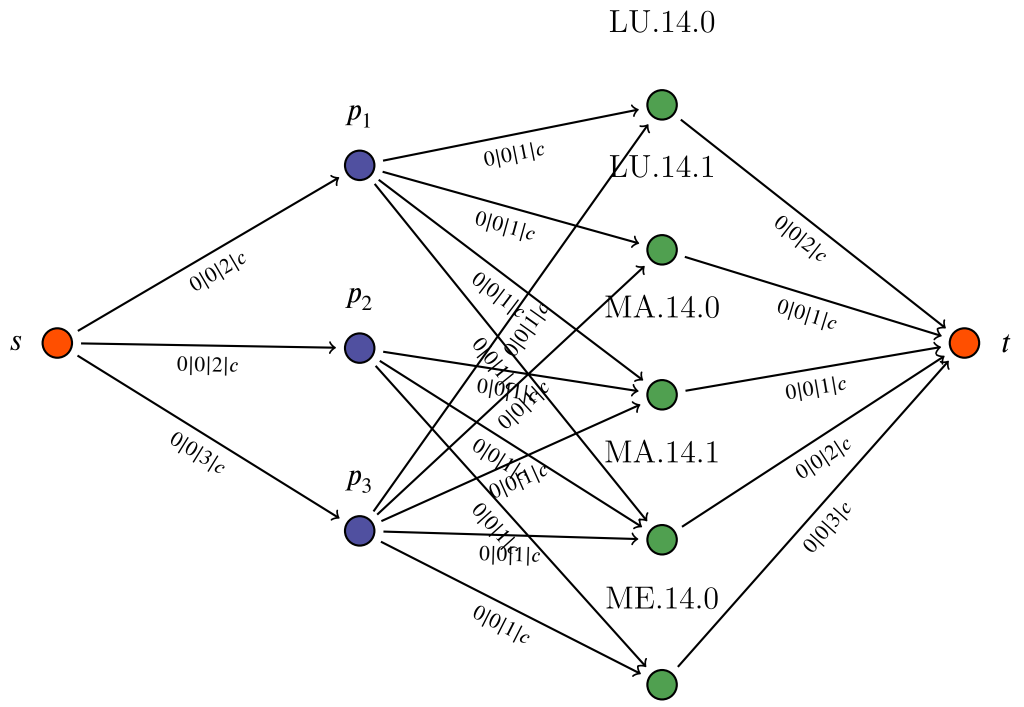

FIGURE 9 – Instance illustrative - Réseau correspondant

Théorème. *Intégrité du flot*
Si les fonctions de capacités sont à valeurs entières, alors il existe un flot maximum de coût minimum f qui sera à valeur entière.

Par application de ce théorème, nous pouvons en déduire que les arêtes initialement présentes dans G qui possèdent des capacités $c_{\text{min}}(e), c_{\text{max}}(e) \in \mathbb{Z}^+ \ \forall e \in E$ seront des arcs représentant des affectations non optimales pour notre problèmes lorsque nous allons chercher à résoudre le problème du flot maximum de coût minimum.

A partir du réseau obtenu, il nous faut maintenant déterminer l'affectation optimale *i.e.* un flot maximum de coût minimum. En l'état actuel, un flot traversant le réseau nous retournera une affectation possible mais non optimale ne prenant pas en compte la maximisation de la complémentarité des créneaux. Il nous faut donc trouver une méthode pour *contrôler* l'écoulement du flot afin que nous puissions favoriser des solutions complémentaires.
Deux paramètres dans ce réseau semblent permettre de contraindre le flot à générer des affectations complémentaires, il s'agit des bornes inférieures (ou capacité minimale) des arcs et les coûts. Nous devons donc jouer sur les valeurs de ces deux paramètres pour pouvoir trouver une solution optimale et ceci sur les arcs d'affectations (les arcs initialements dans E).

Remarque. *sur l'importance des bornes inférieures*
La borne inférieure sur un arc peut être un outil très utile permettant de limiter le passage du flot dès lors que la quantité arrivant à une extrémité de l'arc n'est pas suffisante.

Notons cependant que la présence de ces bornes inférieures n'est pas un frein pour la recherche du flot maximum de coût minimum. En effet, il est toujours possible, à l'aide de la construction que nous allons présenter ci dessous, de se ramener à la recherche d'un flot maximum sur un réseau où les arcs ne sont certes plus contraints par des bornes inférieures mais qui conserve implicitement ces dernières.

Construction. *Modification du réseau pour réduire les bornes inférieures.*
Soient $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E}, c_{\min}, c_{\max}, w)$ *un graphe orienté valué et* $\boldsymbol{u}, v \in \boldsymbol{V}$ *deux sommets formant l'arc* $\boldsymbol{e}$ *et ayant pour demande respective* $\boldsymbol{d}_e(\boldsymbol{u})$ *et* $\boldsymbol{d}_e(v)$*. Supposons que l'arc* $\boldsymbol{e}$ *possède une capacité minimale* $c_{\min}(e)$ *et une capacité maximale* $c_{\max}(e)$*. On peut construire un nouveaux réseau* $\boldsymbol{G}' = (\boldsymbol{V}, \boldsymbol{E}, c'_{\min}, c'_{\max}, w)$ *tel que :*

$$c_{\max}(e)' = c_{\max}(e) - c_{\min}(e) \text{ et } c_{min}(e)' = 0 \ \forall e \in E'$$

Les demandes de chaques sommets de $\boldsymbol{G}'$ *sont recalculées de la manière suivante :*

$$d'(s) = d(s) + \sum_{i \in \mathcal{N}^-(s)} c_{\min}(i) - \sum_{i \in \mathcal{N}^+(s)} c_{min}(i) \ \forall s \in V$$

où $\mathcal{N}^-(s)$ *et* $\mathcal{N}^+(s)$ *sont respectivement les arcs entrants et sortant du sommet* $s \ \forall s \in V$

Après avoir construit ce nouveau réseau, nous devrons répéter l'étape 4 de notre procédure de construction et il nous sera possible de résoudre le problème de recherche d'un flot maximum de coût minimum.

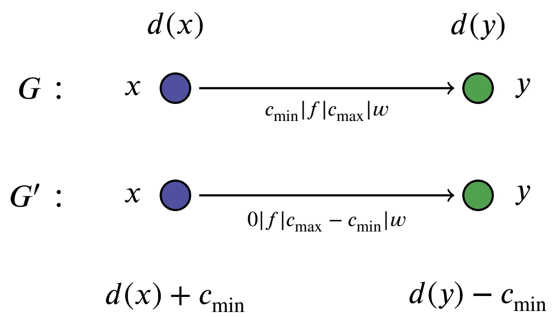

FIGURE 10 – Opération de réduction des bornes inférieures sur un réseau.

3.2.3 Première tentative de résolution

Une première tentative pour contrôler le flot en utilisant les paramètres mentionnés précédemment revient à ajouter des noeuds dits *de contrôles* entres les sommets des deux partitions. Sur les arcs créés entre les dits noeuds, nous affectons aux arcs des coûts négatifs permettant de réduire la valeur du flot si ce dernier passe à travers ces arcs. Une modélisation partielle des noeuds de contrôles mis en place est présentée ci-dessous :

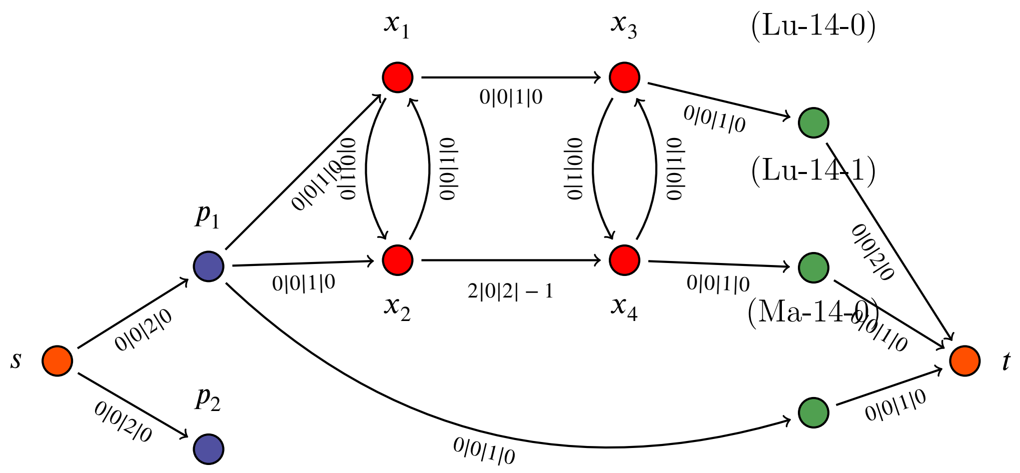

FIGURE 11 – Instance illustrative - Modélisation 1

L'ajout de ces 4 noeuds de contrôle, répété pour chaque noeud professeur dès lors qu'il est disponible sur des créneaux complémentaires, permet d'obtenir une réduction de la valeur du flot lorsque f réussi à faire passer deux unités par l'arc (x_2, x_4).
Cependant, après plusieurs essais, il nous est apparu que cette modélisation n'était pas valide pour la résolution de notre problème d'affectation, la cause étant que l'arc transverse (l'arc (x_2, x_4)) s'avérait être une contrainte bien trop forte. En effet, la recherche d'un flot de coût minimum dans ce réseau impose à un professeur disponible sur un créneau donné de prendre obligatoirement le créneau complémentaire à celui que l'on tente de lui affecter. Nous préférons faire passer des unités de flots depuis p_1 vers les créneaux $Lu14$I et $Lu14$P si et seulement si c'est possible *i.e.* il existe une capacité résiduelle sur des arcs le permettant. Dans le cas contraire et s'il est impossible d'affecter les deux créneaux complémentaires, il serait toujours préférable d'affecter un des deux créneaux plutôt que de ne rien affecter.

3.2.4 Seconde tentative de résolution

Nous réalisons alors que cette modélisation ne nous permet pas de considérer les différentes combinaisons d'affectations possibles car elle est trop restrictive.
En considérant les résultats obtenus avec la première tentative de modélisation, nous allons tenter d'améliorer cette dernière en considérant la possibilité d'attribuer à un professeur un créneau non complémentaire.
Dans cette modélisation, nous ajoutons à chaque professeurs deux unités dites de *remplissage* qui viennent s'ajouter à la capacité maximale des arcs (s, v) $\forall v \in \boldsymbol{P}$ ainsi que 8 nouveaux noeuds de contrôle. Tout en conservant l'arc transverse permettant de réduire le coût total du flot pour l'affectation de créneaux complémentaires, nous ajoutons également un chemin alternatif permettant d'autoriser l'affectation de créneaux même s'ils ne sont pas complémentaires.

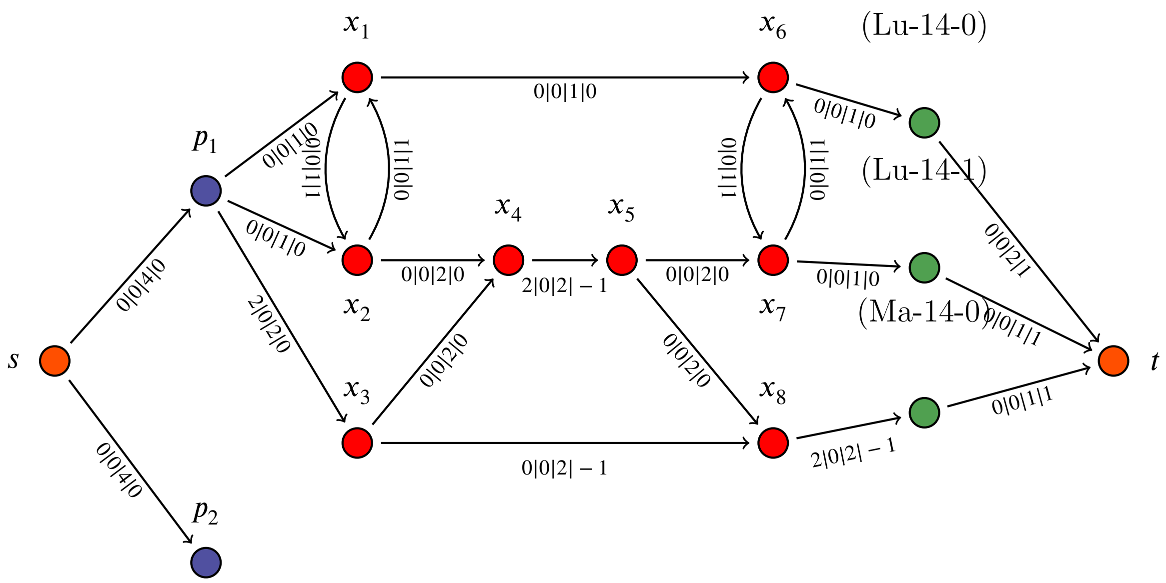

Figure 12 – Instance illustrative - Modélisation 2

La motivation pour l'ajout de ces unités de remplissage consiste en le fait qu'elles peuvent venir saturer l'arc transverse (l'arc (x_4, x_5)) lorsque ce dernier n'est pas emprunté par des les unités initiales ,dites *utiles*, associées à la demande du professeur en termes de créneaux souhaités.
Cependant, après avoir effectué plusieurs essais, nous nous rendons compte que l'ajout de ces noeuds de contrôle et de ces deux unités supplémentaires ne permet pas de caractériser la complémentarité des affectations. En effet, nous constatons qu'il existe dans certains cas des combinaisons d'unités utiles et de remplissage qui ne permettent pas de préférer une solution contenant des créneaux complémentaires à une autre composée de créneaux non complémentaires.

3.2.5 Conclusion de l'approche

Devant la complexité de construction d'un réseau sous forme d'un problème de flot, nous préférons nous arrêter sur cette modélisation qui, sur les deux solutions apportées, ne semble pas être source d'une amélioration au problème d'affectation de créneaux complémentaires.

4 Conclusion

L'objectif de cette étude était de proposer une méthode de résolution pour un problème d'affectation d'enseignants sous la contrainte de maximisation des complémentarités entre les créneaux. Ce projet a permis de mettre en lumière l'approche d'une solution issue de la théorie des graphes pour résoudre une problématique réelle d'optimisation combinatoire.
Malgré la généricité de l'algorithmique des flots, il s'est avéré qu'une telle formulation n'était pas à même de fournir des résultats pour le problème donné. La méthode de résolution par application d'un algorithme de recherche de couplage maximal est donc la seule à ce jour permettant de fournir une solution pouvant s'approcher de l'optimalité. Pour continuer l'étude en utilisant cette approche, il serait intéressant de tester les résultats obtenus sur de plus grandes instances ; En effet, quelques cas d'études ont été développés pour tester les capacités de l'algorithme à générer des affectations complémentaires mais ce dernier n'a pas été éprouvé avec plus de professeurs et de créneaux. Un générateur d'instances aléatoires a été développé à l'aide du langage R mais le temps imparti n'a pas permis de mener de réelles expérimentations numériques. Notons de plus que l'expérimentation

sur de plus grandes instances nous permettrait de détecter des cas spécifiques pouvant se présenter et potentiellement modifier notre stratégie et les hypothèses développées pour l'obtention d'une solution proche de l'optimalité. Une modification de l'implémentation sera également à réaliser pour envisager une poursuite du projet et notamment en utilisant un algorithme de flots pour l'obtention du couplage maximum initial et non une méthode dédiée comme développée dans ce projet.
En plus d'être un approfondissement des notions étudiées en cours, ces travaux encadrés de recherche ont permis de mettre évidence l'application de la théorie des graphes comme un outil à part entière dans la résolution de problèmes d'optimisation ainsi que son adaptabilité face aux contraintes rencontrées. De part sa complexité, ce projet a nécessité une implication personnelle s'inscrivant dans la continuité et une constante remise en question. Je sors grandi de cette expérience et remercie Madame RUSU de m'avoir permis de la vivre.

Références

[1] Irena RUSU. Graphes - Cours Master 1, 2018.

[2] Irena RUSU. Graphes et Réseaux - Cours Master 1, 2018.

[3] L.A. Wolsey. Integer Programming. Wiley Series in Discrete Mathematics and Optimization. Wiley, 1998.

Table des figures

Liste des tableaux

Liste des Algorithmes