Méthodes par séparation et évaluation

I.1. Introduction

Un problème de programmation linéaire en nombres entiers est un problème de programmation linéaire standard avec des *contraintes d'intégrité*, c'est-à-dire qu'on impose aux variables d'être entières. Autrement dit, si n désigne le nombre de variables, c'est un problème de la forme :

$$\text{Maximiser } c.x$$

avec des contraintes de la forme $\begin{cases} Ax \leq b \\ x \in \mathbf{N}^n \end{cases}$.

Alors qu'il existe des algorithmes polynomiaux pour résoudre les problèmes de programmation linéaire en variables réelles (l'algorithme du simplexe n'en est pas un), ce qui établit que ces problèmes sont polynomiaux, le problème de décision associé au problème de programmation linéaire en variables entières précédent est NP-complet.

Parmi les conséquences du fait qu'un problème est NP-complet figurent les propositions suivantes :

1. On ne connaît pas d'algorithme polynomial pour résoudre ce problème.

2. Si l'on connaissait un tel algorithme, alors on aurait automatiquement des algorithmes polynomiaux pour résoudre tous les problèmes de la classe NP, c'est-à-dire à l'heure actuelle un très grand nombre de problèmes répertoriés, venant d'horizons divers, et pour certains très étudiés ; de là à penser qu'il n'existe pas de tel algorithme...

3. Avoir prouvé qu'un problème est NP-complet a des conséquences sur la façon dont on cherche à le résoudre.

- Lorsque l'on doit traiter des problèmes de grande dimension, on peut s'attendre à ce que les algorithmes exacts ne réussissent pas à donner la solution optimale... faute de temps. On applique alors des méthodes appelées *heuristiques*, qui sont censées donner, en un temps raisonnable, une approximation de la solution (sans que l'on puisse parfois dire beaucoup de la façon dont elles approchent l'optimum).
- Pour les algorithmes exacts, on fera appel à des algorithmes comme la *programmation dynamique* (qui fera l'objet du chapitre suivant) ou comme les *méthodes par séparation et évaluation*, appelées aussi *méthodes arborescentes* ou encore *branch and bound*.

Dans ce chapitre, nous allons expliquer le fonctionnement des méthodes par séparation et évaluation à l'aide du problème du sac à dos ; puis nous indiquerons une façon, parmi d'autres, d'appliquer ces méthodes au problème du voyageur de commerce.

I.2. Problème du sac à dos : méthodes heuristiques

Supposons que nous désirions constituer le contenu d'un sac à dos et que nous ayons envie d'y mettre n objets de volumes respectifs v_1, ..., v_n. Ayant constaté que la somme des volumes des n objets était supérieure au volume V du sac à dos, nous affectons à chaque objet une utilité u_1, ..., u_n. Tous les coefficients u_i et v_i et V sont supposés strictement positifs. Nous modélisons alors le problème en introduisant n variables dites *variables de décision* x_1, ..., x_n, à valeurs dans {0, 1}, la valeur 0 attribuée à la variable x_i signifiant que nous n'emporterons pas l'objet i, la valeur 1 que nous le prenons. Si nous voulons avoir le sac le plus utile, nous voyons que nous sommes amenés à résoudre le problème suivant :

$$\text{Max} \sum_{j=1}^{n} u_j.x_j$$

avec les contraintes $\begin{cases} \sum_{j=1}^{n} v_j.x_j = V \\ x_j \in \{0,1\} \text{ pour } 1 \leq j \leq n \end{cases}$.

Il s'agit là d'un problème de programmation linéaire en variables bivalentes (on dit aussi « en 0-1 ») à une seule contrainte, connu sous le nom de *problème du sac à dos* (*knapsack problem* en anglais), parfois qualifié de *binaire*. On parlera encore de problème de sac à dos (ou parfois de *problème de sac à dos généralisé*) lorsque la contrainte reste unique mais que les variables peuvent être entières positives ou nulles et non plus nécessairement bivalentes. C'est ce problème (variables entières) que nous considérons dans ce qui suit.

I.2.1. Première heuristique

Pour résoudre le problème du sac à dos (généralisé), on peut avoir l'idée de « relâcher les contraintes d'intégrité », c'est-à-dire résoudre le problème en variables réelles, puis d'arrondir les solutions obtenues en prenant leurs parties entières par défaut. Nous allons voir que cette méthode ne peut être considérée que comme une heuristique sur l'exemple suivant :

$$\text{Maximiser } 10x_1 + 8x_2 + 5x_3$$

$$\text{avec les contraintes } \begin{cases} 6x_1 + 5x_2 + 4x_3 = 9 \\ x_j \in \mathbf{N} \text{ pour } 1 \leq j \leq 3 \end{cases}.$$

La relaxation des contraintes d'intégrité conduit à un problème de programmation linéaire dont la résolution, immédiate par la méthode du simplexe (*cf*. exercice 1), donne $x_1^* = 1{,}5$, $x_2^* = 0$, $x_3^* = 0$ et $z^* = 15$. La méthode des arrondis donne $x_1^* = 1$, $x_2^* = x_3^* = 0$ et $z^* = 10$, alors que le problème admet la solution $x_1^* = 0$, $x_2^* = 1$, $x_3^* = 1$ et $z^* = 13$, dont nous verrons qu'elle est optimale.

I.2.2. Seconde heuristique

On peut tout de suite imaginer une autre heuristique pour le problème du sac à dos (généralisé). Une approche de style qualité/prix, chère aux unions de consommateurs, incite à un classement des variables dans l'ordre décroissant des rapports utilité/volume. On peut alors examiner les variables dans cet ordre et choisir de donner, à la variable examinée, la plus grande valeur entière permettant de satisfaire à la contrainte compte tenu des choix déjà faits. Un tel algorithme est dit « glouton » parce qu'on traite les variables les unes après les autres en prenant des décisions que l'on ne remet pas en cause.

Pour notre exemple, les rapports en question sont 5/3 pour x_1, 8/5 pour x_2 et 5/4 pour x_3. L'ordre induit par l'examen des rapports est donc x_1, x_2, x_3. Compte tenu de la contrainte, la plus grande valeur possible pour x_1 est 1. Ayant donné la valeur 1 à x_1, on voit qu'on doit donner la valeur 0 à x_2 et à x_3. On retrouve ici la même solution que celle fournie par la méthode précédente (mais c'est un hasard ; cette seconde heuristique est en fait toujours au moins aussi bonne que la première). Nous voyons aussi que pour l'une comme pour l'autre on ne peut parler que d'heuristique.

I.3. Méthode arborescente par séparation et évaluation pour le problème du sac à dos

Les méthodes arborescentes par séparation et évaluation s'appliquent à la résolution de problèmes d'optimisation combinatoire avec un grand nombre de solutions envisageables et, en particulier, à la résolution de problèmes de

programmation linéaire en nombres entiers où l'on peut définir la notion de solution réalisable, c'est-à-dire de solution satisfaisant les contraintes. Nous allons associer à cette méthode une arborescence dont les sommets correspondent à des ensembles de solutions réalisables, la racine de l'arborescence contenant l'ensemble de toutes ces solutions réalisables. L'expansion de cette arborescence est régie selon quatre ingrédients : un *principe de séparation*, un *principe d'évaluation*, l'utilisation d'une *borne* et une *stratégie de développement*.

I.3.1. Principe de séparation

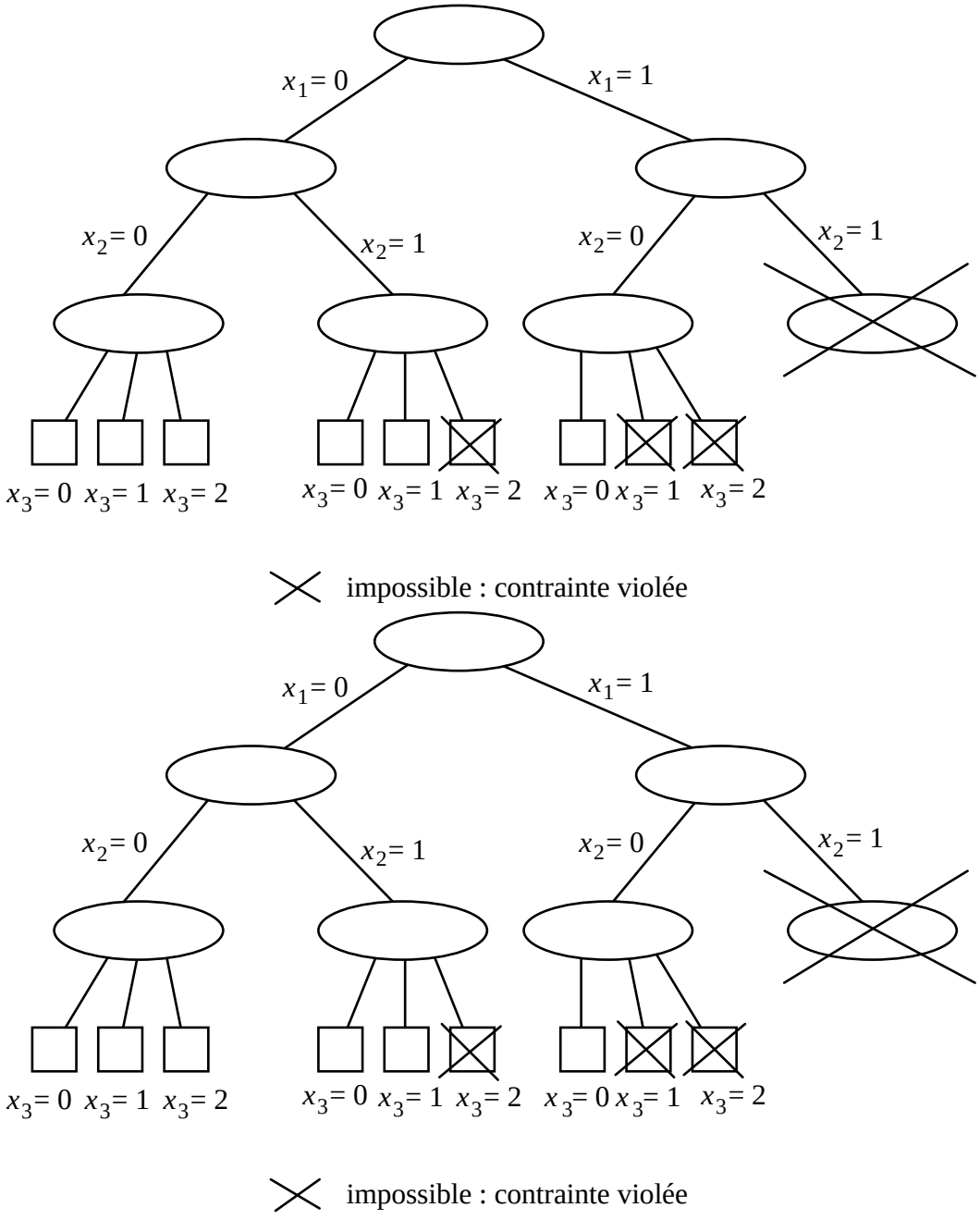
Le mot *séparation* représente le fait que nous partageons, en fonction d'un certain critère, l'ensemble des solutions réalisables contenues dans un sommet de cette arborescence (ensemble que nous ne connaissons pas explicitement) en sous-ensembles, ceux-ci devenant dans l'arborescence les fils du sommet considéré. La seule exigence dans ce partage est que nous ne perdions ni n'ajoutions de solution (autrement dit, l'union des sous-ensembles associés aux fils d'un sommet doit être égale à l'ensemble associé à ce sommet). Si on n'appliquait que le principe de séparation, celle-ci serait effectuée pour tout sommet contenant plus d'une solution.

Nous allons illustrer le principe de séparation seul à partir du problème de sac à dos (généralisé) considéré au paragraphe I.2.1. Pour cet exemple, nous pouvons appliquer les critères de branchement suivants :

- une première séparation sera effectuée selon les valeurs de x_1 ; comme, d'après la contrainte, x_1 ne peut prendre que les valeurs 0 ou 1, nous séparons l'ensemble de toutes les solutions associé à la racine en deux sous-ensembles : l'un contient toutes les solutions pour lesquelles x_1 vaut 0 et l'autre contient celles pour lesquelles x_1 vaut 1 ;
- une deuxième séparation sera faite de la même façon suivant les valeurs possibles de x_2, à savoir 0 ou 1 ;
- enfin, une troisième séparation aura lieu selon les valeurs possibles de x_3 : 0, 1 ou 2, sauf pour le sommet vide correspondant aux choix $x_1 = x_2 = 1$.

Le dessin suivant montre ce qu'est l'arborescence obtenue pour notre exemple. Les carrés y représentent les cas pour lesquels les trois variables sont entièrement déterminées. Les sommets de l'arborescence barrés d'une croix en sont les sommets vides.

Il suffit, pour connaître la solution optimale, de calculer la valeur de la fonction objectif pour toutes les feuilles non vides de l'arborescence obtenue. Cette méthode peut être améliorée pour éviter l'examen de certaines branches. Il existe en effet deux raisons qui permettent de ne pas développer un sommet de l'arborescence : lorsque l'on peut montrer que ce sommet ne contient pas la solution optimale et lorsque que l'on sait résoudre directement le problème correspondant à ce sommet. Le principe d'évaluation et la borne peuvent fournir de telles indications.



I.3.2. Principe d'évaluation et utilisation de la borne

Dans un problème d'optimisation écrit sous forme de maximisation d'un certain objectif, la *borne* correspond à une valeur qu'on sait atteindre pour l'objectif, à l'aide d'une certaine solution réalisable, et qui est donc par définition un minorant du maximum. Ainsi, pour le problème de sac à dos (généralisé) abordé plus haut, les heuristiques précédentes ont montré que 10 est une borne

inférieure du maximum cherché.

Toujours dans le cas où on cherche un maximum, *évaluer un sommet*, c'est déterminer un majorant de l'ensemble des valeurs de l'objectif correspondant aux solutions contenues en ce sommet. Ainsi, ayant évalué un sommet *S*, on saura que pour l'ensemble des solutions correspondant à *S*, on ne peut pas faire mieux que la valeur de l'évaluation. Une évaluation d'un sommet est dite *exacte* si la valeur donnée par l'évaluation est atteinte par un élément de l'ensemble associé au sommet. Pour l'exemple précédent, la relaxation des contraintes d'intégrité avait permis d'obtenir la valeur 15. Ceci est donc un majorant du maximum cherché en valeurs entières (en effet, quand on relâche les contraintes d'intégrité, le domaine des solutions réalisables grandit et le maximum dans ce domaine ne peut qu'être supérieur ou égal à celui du domaine en valeurs entières), et on peut attribuer cette valeur comme évaluation de la racine. Cette évaluation n'est pas exacte, car la solution qui donne 15 n'est pas à composantes entières.

La borne et l'évaluation permettent de ne pas développer un sommet *S* dans les deux cas suivants (toujours pour une maximisation) qui viennent donc s'ajouter au cas précédent (les contraintes ne sont pas satisfaites) :

- l'évaluation de *S* est inférieure ou égale à la borne ;
- l'évaluation de *S* est exacte ; dans ce cas, si cette évaluation est strictement supérieure à la borne (sinon on est dans le cas précédent), on a exhibé une solution meilleure que celle associée à la borne : on modifie donc la borne et on se retrouve dans le cas précédent.

On voit qu'on a intérêt à calculer une bonne borne et à concevoir une fonction d'évaluation assez fine afin d'éliminer le plus tôt possible le plus de branches possible. Mais en même temps, ces calculs doivent pouvoir être effectués en un temps raisonnable...

Si on récapitule les cas qui permettent de ne pas séparer un sommet *S*, on obtient finalement :

1. On sait que toutes les solutions contenues dans *S* sont au plus aussi bonnes qu'une solution qu'on connaît déjà (par comparaison entre l'évaluation de *S* et la borne courante).
2. *S* est vide ou on connaît la meilleure des solutions contenues dans *S*, soit parce que l'évaluation est exacte, soit parce qu'une méthode directe a permis de conclure pour ce sommet.

I.3.3. Stratégie de développement

Le quatrième ingrédient consiste à déterminer la façon dont on va construire l'arborescence, c'est-à-dire dans quel ordre on va appliquer le critère de séparation aux sommets de l'arborescence.

Dans une première stratégie, appelée « en profondeur », on peut faire une construction en profondeur d'abord en descendant dans les branches jusqu'à ce qu'on trouve un sommet que l'on peut éliminer, auquel cas on remonte pour redescendre dans une autre direction (s'il en reste). Si l'arborescence était connue explicitement, cette construction correspondrait à l'application d'un parcours en profondeur à cette arborescence. Cette façon de procéder présente plusieurs

avantages, principalement au niveau de la place mémoire : l'encombrement en place mémoire est souvent relativement peu important, puisqu'il ne faut conserver que la description de la branche qu'on explore (il n'est pas nécessaire de conserver la description complète de l'arborescence) ; elle possède également des avantages au niveau des accès disques (en effet bien souvent on ne peut traiter tout le problème, vu sa taille, en mémoire centrale) ; enfin, elle permet plus facilement d'exploiter, quand il y en a, des informations disponibles au niveau du sommet qu'on examine et au moins partiellement utilisables pour les fils de celui-ci.

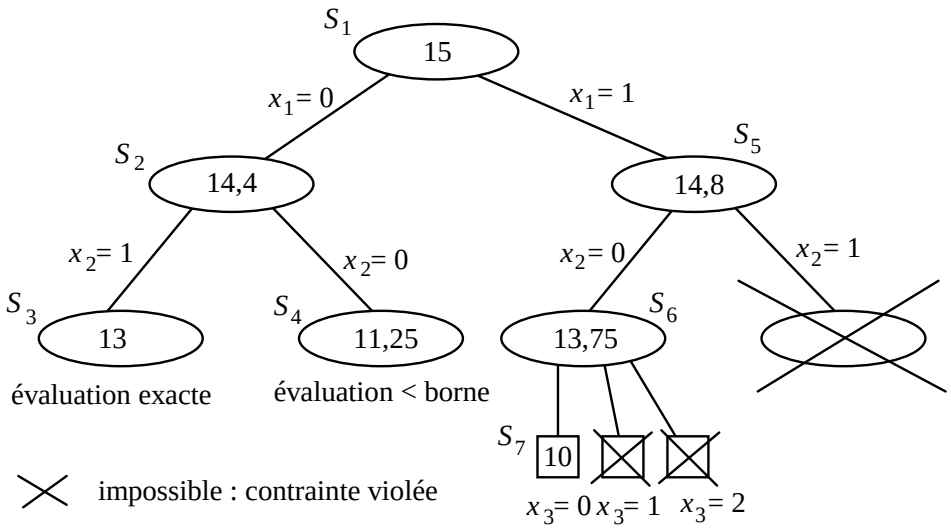
On peut aussi commencer par séparer les sommets qui ont l'évaluation la plus grande (pour une maximisation) en suspectant que ce sont eux qui contiennent la solution optimale : c'est dans le sommet de plus grande évaluation qu'on a l'espoir, justifié ou non, de trouver la meilleure solution. Cette stratégie, de type « meilleur d'abord », peut, pour certains problèmes, être plus rapide du fait qu'on essaie à chaque itération d'explorer la direction qui semble la plus prometteuse ; en revanche, la gestion de l'arborescence n'est pas triviale et consomme généralement beaucoup de place mémoire.

Enfin, on peut envisager une exploration en largeur d'abord, mais cette méthode est très rarement implémentée pour des questions d'efficacité et d'encombrement de la mémoire.

I.3.4. Application au problème de sac à dos

Nous pouvons reprendre l'étude de l'arborescence précédente en utilisant comme fonction d'évaluation la résolution du simplexe en variables réelles, qui est immédiate à effectuer dans le cas d'une seule contrainte : en effet, il est facile de se convaincre que, lorsque l'on applique l'algorithme du simplexe à un problème à une seule contrainte, si on choisit comme variable entrante la variable de choix de plus petit indice, les variables étant indicées dans l'ordre décroissant des rapports utilité/volume (*cf.* exercice 1), on obtient l'optimum en une itération. La stratégie retenue sera celle du parcours en profondeur d'abord. Enfin, on rappelle que la borne obtenue à l'aide des heuristiques vaut 10.

On obtient alors l'arborescence ci-dessous, en supposant que l'on a procédé à l'évaluation du sommet $x_1 = 0$, $x_2 = 1$ avant d'entamer la séparation du sommet $x_1 = x_2 = 0$, ce qui nous a permis d'améliorer la borne grâce à une évaluation de 13, qui était exacte. Les sommets de l'arborescence sont numérotés dans l'ordre selon lequel on les rencontre lors du développement en profondeur d'abord.



L'évaluation du sommet S_3 est obtenue par la solution $x_3 = 1$, en plus des choix $x_1 = 0$ et $x_2 = 1$. C'est donc une évaluation exacte. Par conséquent, il est inutile de développer ce sommet et on interrompt l'exploration de cette branche. De plus, la valeur obtenue, réalisable, est meilleure que la borne ; on change donc celle-ci, qui passe de 10 à 13.

Renonçant au développement de S_3, on remonte à S_2 pour examiner, s'il en existe, les autres choix relatifs à x_2, ce qui nous conduit à S_4 avec $x_2 = 0$. On obtient alors 11,25 comme évaluation de S_4. Celle-ci étant inférieure ou égale à la nouvelle borne, on ne développe pas S_4 et on remonte en S_2 puis en S_1, puisque aucun autre choix n'est possible pour x_2 au niveau de S_2.

Quand on arrive en S_5, la contrainte fait qu'il n'y a qu'une seule possibilité viable pour x_2 : 0, ce qui donne S_6. Les choix pour x_3, au niveau de S_6, conduisent soit à des solutions non réalisables, soit à un sommet S_7 d'évaluation exacte mais moins bonne que la borne actuelle. On interrompt donc l'exploration de S_6, on remonte en S_5 où il n'y a plus de prolongement réalisable, puis en S_1 où, de même, il n'y a plus de prolongement à envisager. On a donc terminé : le maximum cherché vaut 13 et est atteint par $x_1 = 0$, $x_2 = x_3 = 1$.

I.4. Application au problème du voyageur de commerce

Rappelons que le problème du voyageur de commerce est la recherche, dans un graphe complet valué, d'un cycle hamiltonien (c'est-à-dire passant une fois et une seule par tous les sommets) de valuation totale minimum. Ce problème est NP-difficile (plus précisément, le problème de reconnaissance associé est NP-complet). Nous allons montrer qu'il fait partie de la classe des problèmes de programmation linéaire en variables 0-1, dont il est d'ailleurs un représentant extrêmement célèbre. Nous décrirons ensuite une méthode par séparation et évaluation permettant de le résoudre.

I.4.1. Forme linéaire en 0-1 du problème du voyageur de commerce

Considérons le graphe complet $K_n = (X, E)$ à n sommets, c'est-à-dire le graphe d'ordre n où tous les sommets sont adjacents deux à deux. On suppose de plus qu'on a défini une valuation sur l'ensemble E des arêtes de ce graphe, autrement dit une application p de E dans R que l'on appelle « poids », le poids d'une arête $\{u, v\}$ étant noté p_{uv}. Associons à chaque arête $\{u, v\}$ de K_n une variable bivalente x_{uv} qui vaut 1 si on garde l'arête $\{u, v\}$ pour constituer le cycle hamiltonien, 0 sinon.

On peut représenter la contrainte de constituer un cycle hamiltonien à l'aide des contraintes suivantes :

- pour tout sommet v, la somme des valeurs des variables associées aux arêtes ayant v comme extrémité est égale à deux, ceci traduisant le fait que dans un cycle hamiltonien tout sommet est de degré deux ;
- pour tout sous-ensemble Y de X autre que X, le nombre d'arêtes ayant ses deux extrémités dans Y est strictement plus petit que $|Y|$; cette condition implique que l'ensemble des arêtes gardées ne se décompose pas en plusieurs cycles.

Toutes ces contraintes s'expriment bien linéairement par rapport à l'ensemble des variables du problème, comme le montre la forme suivante :

$$\text{Min} \sum_{\{u,v\}\in E} p_{uv} . x_{uv}$$

$$\text{avec les contraintes} \quad \begin{cases} \forall u \in X, \ \sum_{v \in X} x_{uv} = 2 \\ \forall Y \subset X \text{ avec } Y \neq X, \ \sum_{\substack{\{u,v\}\in E \\ u\in Y,\ v\in Y}} x_{uv} < |Y| \\ \forall \{u, v\} \in E, \ x_{uv} \in \{0, 1\} \end{cases}$$

I.4.2. Définition d'une fonction d'évaluation

Soit G un graphe dans lequel on cherche un cycle hamiltonien de poids minimum. Une évaluation possible d'un sommet de l'arborescence d'une procédure par séparation et évaluation est justifiée par les remarques triviales ci-après :

- une chaîne constitue un arbre particulier ; la chaîne obtenue en supprimant d'un cycle hamiltonien de G un sommet quelconque x_0 et les deux arêtes qui lui sont adjacentes est un arbre couvrant de $G - x_0$: elle est donc de poids supérieur ou égal au poids d'un arbre de poids minimum couvrant $G - x_0$;
- dans un cycle hamiltonien, tout sommet a un degré égal à 2 ; la somme des poids des deux arêtes d'un cycle hamiltonien adjacentes à un sommet x_0 fixé est supérieure ou égale à la somme des poids des deux arêtes les

plus légères incidentes à x_0 dans G.

Choisissant de façon arbitraire un sommet x_0, le poids d'un cycle hamiltonien quelconque est donc supérieur ou égal à la somme des poids des deux arêtes les plus légères adjacentes à x_0 plus le poids d'un arbre de poids minimum couvrant $G - x_0$. Cette quantité constitue un minorant du poids d'un cycle hamiltonien optimal et pourra par conséquent constituer une fonction d'évaluation pour l'arborescence (l'évaluation doit en effet fournir un tel minorant puisque le problème de départ est un problème de minimisation).

I.4.3. Description d'une méthode par séparation et évaluation

Nous pouvons maintenant décrire, parmi d'autres, une méthode par séparation et évaluation utilisant l'évaluation définie ci-dessus (l'exercice 2 en donne une illustration).

On suppose qu'une borne a été calculée au préalable, par exemple à l'aide d'une des méthodes exposées dans les deux derniers chapitres de ce livre (faute de mieux, on peut toujours partir d'un cycle hamiltonien choisi au hasard, mais la borne qui en découlera risque d'être assez mauvaise).

Choisissons une fois pour toutes un sommet x_0, afin de procéder à l'évaluation décrite ci-dessus. Comme toujours, la racine de l'arborescence contient toutes les solutions réalisables possibles, c'est-à-dire ici l'ensemble de tous les cycles hamiltoniens du graphe. Arrivés à un sommet S de l'arborescence (y compris la racine), évaluons-le. Si l'évaluation de S est supérieure ou égale à la borne (attention, nous sommes en train de résoudre un problème de minimisation...), nous ne développons pas S et S est abandonné : nous savons déjà faire au moins aussi bien. Sinon, construisons le graphe partiel H correspondant à l'évaluation en ajoutant les deux arêtes adjacentes à x_0 de moindre poids dans le graphe courant G à celles d'un arbre de poids minimum couvrant $G - x_0$: H est un graphe connexe d'ordre n possédant n arêtes. Si H est un cycle hamiltonien, par définition l'évaluation de S est exacte ; on peut alors abandonner S : on ne fera pas mieux, en ce qui concerne les solutions contenues dans S, que ce qu'on vient d'obtenir ; en outre si cette évaluation exacte est inférieure à la valeur actuelle de la borne, on remet celle-ci à jour. Si H n'est pas hamiltonien, puisqu'il est connexe et comporte n arêtes, c'est qu'il contient au moins un sommet de degré strictement supérieur à deux. Considérons un tel sommet et les i ($i \geq 3$) arêtes e_1, e_2, ..., e_i auxquelles il est adjacent dans H. Pour définir le critère de séparation, remarquons, puisqu'un cycle hamiltonien est un graphe dans lequel tous les sommets ont un degré égal à 2, qu'on ne perd aucune solution en considérant successivement celles qui ne contiennent pas e_1, puis celles qui ne contiennent pas e_2, … et ainsi de suite jusqu'à l'arête e_i ; le sommet S de l'arborescence sera donc séparé selon i branches, chacune d'entre elles correspondant à l'interdiction d'une des i arêtes e_k ($1 \leq k \leq i$). On traduit la contrainte « ne pas contenir l'arête e_k » par le fait de mettre le poids de e_k à l'infini dans le graphe courant G.

I.5. Exercices

Exercice 1

On considère un problème de programmation linéaire standard, à une seule contrainte, défini par

$$\text{Max} \sum_{j=1}^{n} u_j . x_j \text{ avec } \sum_{j=1}^{n} v_j . x_j \leq V \text{ et } x_j \geq 0 \text{ pour } 1 \leq j \leq n.$$

Tous les coefficients u_j et v_j sont supposés strictement positifs et l'on suppose les variables classées par rapports utilité/volume décroissants ou, pour rester dans un formalisme plus mathématique, suivant les valeurs décroissantes des rapports u_j/v_j . Montrer que la variable x_1 est entrante et que, en la faisant entrer en base, on atteint l'optimum de l'objectif en une seule étape. Exprimer la valeur de l'objectif en fonction des différents coefficients.

Corrigé de l'exercice 1

Le coefficient de la variable x_1 étant, par hypothèse, positif, la variable x_1 est entrante et on l'échange donc avec l'unique variable en base, qui correspond à l'unique contrainte. On avait

$$x_{n+1} = V - \sum_{j=1}^{n} v_j . x_j$$

et, après l'échange, on obtient :

$$x_1 = \frac{1}{v_1}\left(V - \sum_{j=2}^{n} v_j . x_j - x_{n+1} \right).$$

En reportant cette valeur dans la fonction objectif, il vient :

$$z = \sum_{j=1}^{n} u_j . x_j = \frac{u_1}{v_1}\left(V - \sum_{j=2}^{n} v_j . x_j - x_{n+1} \right) + \sum_{j=2}^{n} u_j . x_j$$

ou encore

$$z = \frac{u_1 . V}{v_1} + \sum_{j=2}^{n}\left(u_j - \frac{u_1 . v_j}{v_1} \right). x_j - \frac{u_1}{v_1} x_{n+1}$$

Compte tenu de la numérotation adoptée, les coefficients de toutes les variables qui interviennent dans l'écriture de z sont négatifs ou nuls. On a donc trouvé la valeur maximum de z en une itération, et celle-ci est égale à $u_1.V/v_1$.

Exercice 2

Appliquer la méthode par séparation et évaluation décrite ci-dessus pour résoudre la problème du voyageur de commerce dans le graphe K_6 (c'est-à-dire le graphe complet à six sommets) dont la matrice des poids est la suivante :

	x	y	z	t	u	v
x	$+\infty$	4	7	2	5	4
y	4	$+\infty$	3	2	1	2
z	7	3	$+\infty$	2	6	3
t	2	2	2	$+\infty$	5	3
u	5	1	6	5	$+\infty$	2
v	4	2	3	3	2	$+\infty$

Que faudrait-il faire si on voulait connaître tous les cycles hamiltoniens de poids minimum ?

Corrigé de l'exercice 2

DETERMINATION D'UNE BORNE B

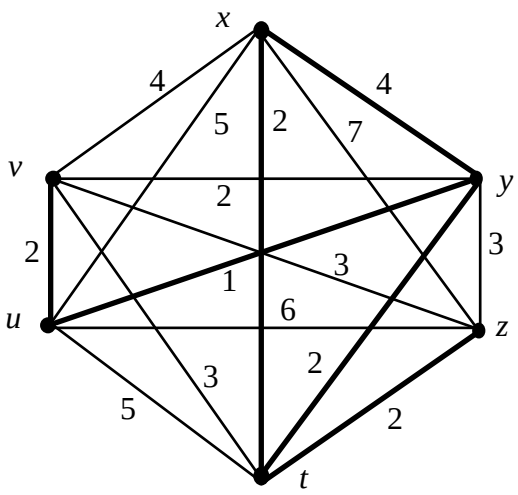
On applique une heuristique de prolongement d'une chaîne en choisissant à chaque étape l'arête la moins lourde permettant ce prolongement (bien sûr en ne créant un cycle qu'à la fin). En partant de x, on peut par exemple choisir successivement $\{x, t\}$, $\{t, y\}$, $\{y, u\}$, $\{u, v\}$, choix que l'on complète nécessairement par l'adjonction des arêtes $\{x, z\}$ et $\{z, v\}$, ce qui donne le cycle hamiltonien $x\ t\ y\ u\ v\ z\ x$, de poids 17. Nous commençons la recherche avec cette borne : $B = 17$.

STRATEGIE DE DEVELOPPEMENT

Bien qu'elle soit plus consommatrice de place mémoire et qu'elle pose parfois des problèmes pour gérer l'arborescence et déterminer le sommet auquel appliquer le principe de séparation, nous adopterons ici une stratégie de type « meilleur d'abord », assez facile à mettre en œuvre à la main. Par conséquent, quand on effectuera une séparation, tous les sommets obtenus seront évalués, et on repartira du sommet non encore séparé de moindre évaluation.

ÉVALUATION DE LA RACINE DE L'ARBORESCENCE

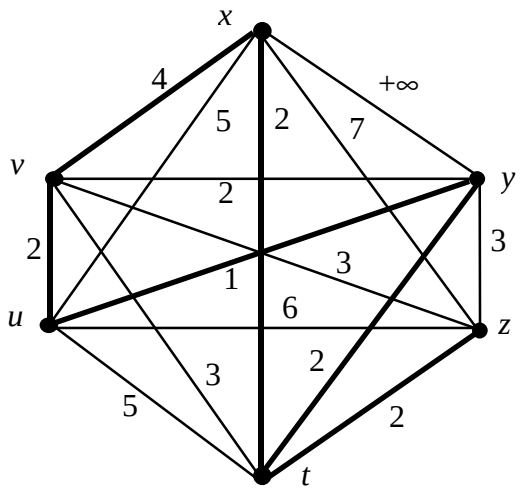
Prenons x pour jouer le rôle du x_0 du texte ; les deux arêtes de poids minimum incidentes à x sont les arêtes $\{x, t\}$ et par exemple $\{x, y\}$. Un arbre de poids minimum de $K_6 - x$ nous sera fourni par l'algorithme de Kruskal ou par celui de Prim. Avec les deux arêtes incidentes à x, on obtient par exemple le graphe H en gras, de poids 13, poids qui donne l'évaluation de la racine.



Ce graphe n’est pas un cycle hamiltonien parce que certains sommets ont un degré strictement supérieur à 2. C’est le cas par exemple du sommet *y*. Nous allons utiliser le fait que dans *H* ce sommet est de degré 3 pour effectuer les trois branchements décrits ci-dessous.

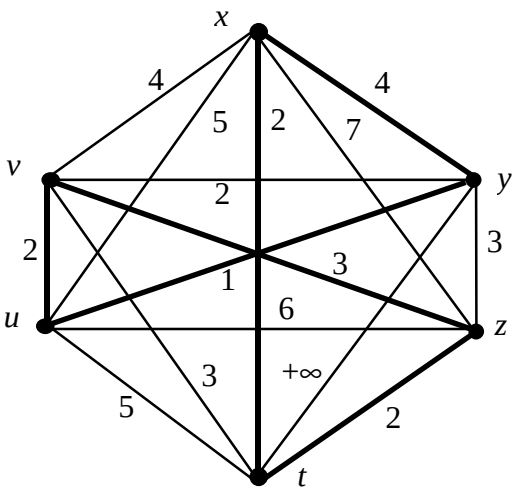
1ER BRANCHEMENT

L’arête {*x*, *y*} est interdite, c’est-à-dire que son poids devient infini, ce qui ne modifie pas l’arbre couvrant de $K_6 - x$. Les deux arêtes de poids minimum incidentes à *x* sont maintenant {*x*, *t*} et {*x*, *v*}. L’évaluation de ce sommet est toujours 13, mais elle n’est pas exacte, le graphe correspondant (ci-contre) n’étant pas un cycle hamiltonien.



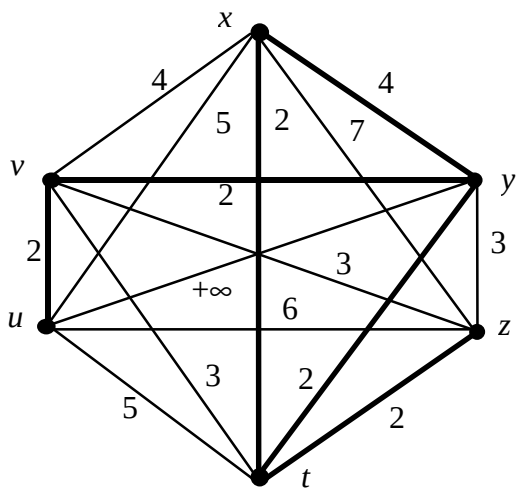
2E BRANCHEMENT

L’arête {*y*, *t*} est interdite (son poids devient infini), ce qui modifie l’arbre couvrant de $K_6 - x$. Parmi les nouveaux arbres de poids minimum, considérons celui qui, ainsi que les deux arêtes de poids minimum incidentes à *x* sont en gras dans le graphe ci-dessous. On obtient un cycle hamiltonien de poids 14. On peut donc mettre à jour la borne *B* qui passe de 17 à 14, et d’autre part ne pas développer ultérieurement le sommet auquel on vient d’aboutir dans l’arborescence, son évaluation étant égale à la nouvelle valeur de *B*.

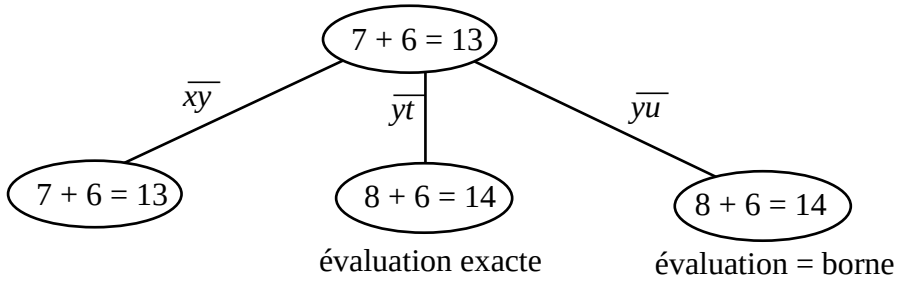


3E BRANCHEMENT

L'arête $\{y, u\}$ est interdite (son poids devient infini), ce qui modifie l'arbre couvrant de $K_6 - x$. Les nouveaux arbres de poids minimum ont un poids égal à 8 (voir graphe ci-contre). Avec les deux arêtes de poids minimum incidentes à x, on obtient ainsi une évaluation de 14, donc égale à B. Par conséquent, il est inutile de développer l'arborescence à partir de ce sommet : il ne conduira pas à un cycle hamiltonien plus intéressant que celui qu'on connaît déjà.



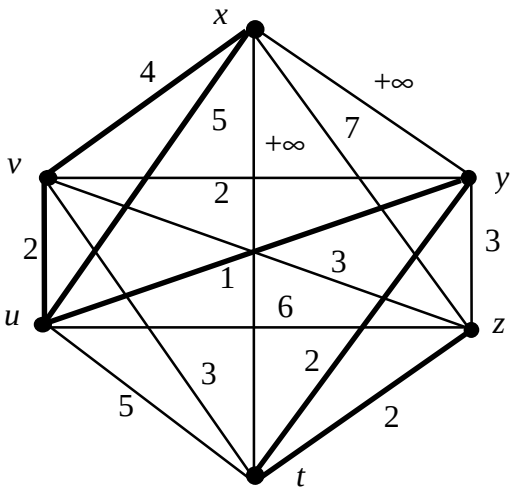
Les opérations que l'on vient d'effectuer peuvent être résumées par le dessin suivant, qui donne l'état actuel de l'arborescence. Les arêtes $\{\alpha, \beta\}$ interdites y sont représentées par $\overline{\alpha\beta}$.



Il ne reste plus à développer que le sommet situé à gauche dans le dessin précédent. Dans le graphe correspondant à cette évaluation (voir plus haut le graphe obtenu par le premier branchement), le sommet t est de degré 3. On effectue donc trois nouveaux branchements à partir de ce sommet de l'arborescence, en interdisant successivement, en plus de l'arête $\{x, y\}$, les arêtes $\{t, x\}$, $\{t, y\}$ puis $\{t, z\}$.

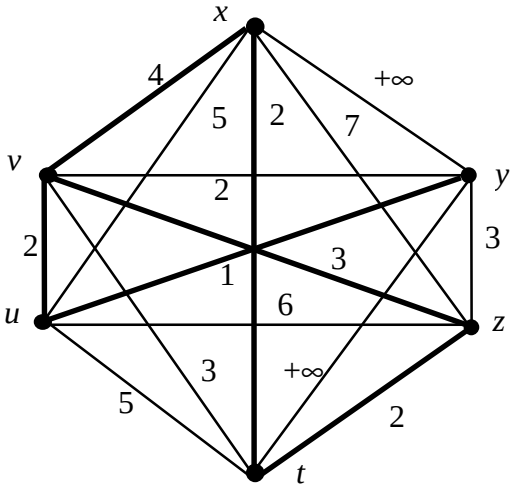
4E BRANCHEMENT

L'arête $\{t, x\}$ est interdite (son poids devient infini). Les deux arêtes de poids minimum incidentes à x sont maintenant $\{x, v\}$ et $\{x, u\}$. Le poids du graphe en gras dans le dessin ci-dessous, égal à 16 (= 7 + 9), donne l'évaluation du sommet de l'arborescence auquel on aboutit. Cette évaluation étant supérieure à B, on ne développera pas ce sommet.



5E BRANCHEMENT

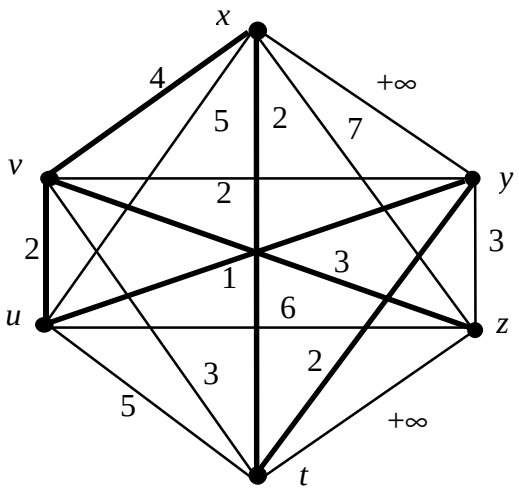
L'arête $\{t, y\}$ est interdite (son poids devient infini). Un arbre de poids minimum de $K_6 - x$ avec l'infini pour poids de $\{t, y\}$ a déjà été calculé pour le 2e branchement, de poids 8. Comme de plus les deux arêtes de poids minimum incidentes à x sont $\{x, t\}$ et $\{x, v\}$, on obtient 8 + 6 = 14 comme évaluation de ce nouveau sommet de l'arborescence. Cette évaluation étant égale à B, on ne développera pas ce sommet.



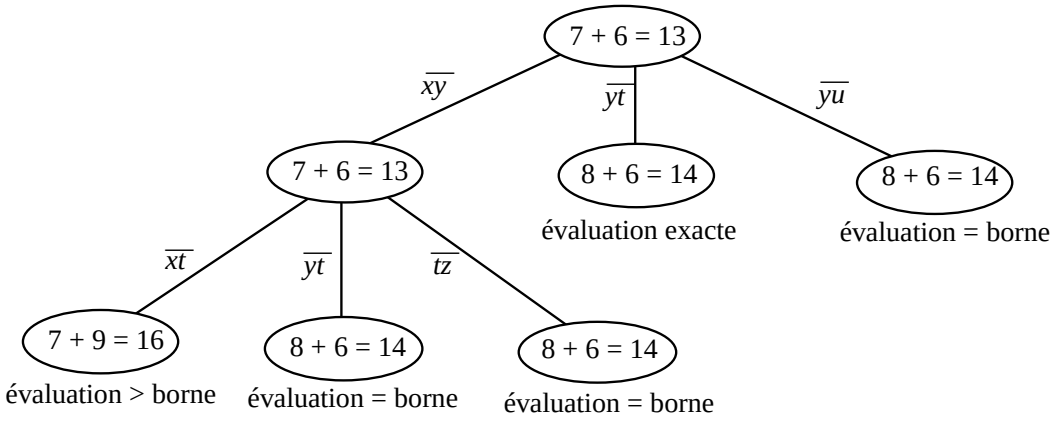
On pouvait d'ailleurs prévoir qu'il ne serait pas intéressant de développer cette branche, puisque, par rapport au 2e branchement, on a interdit une arête supplémentaire ; on ne pouvait donc pas obtenir mieux que par le 2e branchement.

6E BRANCHEMENT

L'arête $\{t, z\}$ est interdite (son poids devient infini). L'évaluation de ce nouveau sommet de l'arborescence est donnée par le poids du graphe en gras dans le dessin ci-contre, et vaut donc 14, ce qui n'est pas mieux que la borne : on ne développera pas ce sommet.



Il ne reste plus de sommet à développer : on a terminé. L'arborescence totale est donc réduite aux sommets du dessin suivant :



Le poids minimum d'un cycle hamiltonien vaut 14 et le cycle hamiltonien obtenu au 2e branchement est une solution optimale du problème.

Si on voulait déterminer tous les cycles hamiltoniens de poids minimum, il faudrait encore développer tous les sommets dont l'évaluation vaut 14, puisqu'ils sont susceptibles de contenir de telles solutions. Plus généralement, il conviendrait de modifier la description des méthodes par séparation et évaluation de façon à n'éliminer que les sommets de l'arborescence qui ne contiennent aucune solution réalisable (les contraintes du problème ne sont pas toutes respectées) ou dont l'évaluation est strictement moins bonne que la borne (c'est-à-dire strictement supérieure dans le cas d'une minimisation et strictement inférieure dans le cas d'une maximisation) ; les sommets d'évaluation égale à la borne seraient donc développés, y compris lorsque l'évaluation est exacte.