IFT2125 - Séance de travaux pratiques 8 - # 5 à # 8

5 avril 2016

Université de Montréal

antakivi@iro.umontreal.ca

Numéro 5

Problème 9.42, p. 324, livre Brassard-Bratley : Adaptez l'algorithme de retour-arrière sur le problème du sac à dos pour qu'en plus de retourner la valeur optimale, il retourne aussi les objets inclus dans le sac à la fin de l'algorithme.

Solution

Nous ajoutons une pile S qui contient l'indice des objets actuellement dans le sac. L'algorithme préserve une pile S^* qui contient la meilleure solution et il modifie son contenu lorsqu'une meilleure solution est atteinte. L'usage d'une pile n'a pour but que de préserver l'ordre dans lequel les objets sont ajoutés. Une autre structure de données pourrait être utilisée.

```
Sac-a-dos(W):
    return Sac-a-dos'(1,W)

Sac-a-dos'(i,r):
    b <- 0
    S" <- {}

    for k <- i .. n:

        if w[k] <= r:
            b', S' <- Sac-a-dos'(k,r-w[k])

            if b' + v[k] > b :
                b <- b' + v[k]
                S" <- S'
                S".push(k)

    return (b,S")
```

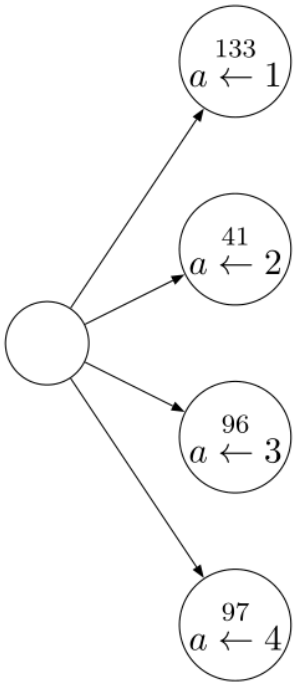
Numéro 6

Problème 9.53 a), p. 324, livre Brassard-Bratley : Utilisez l'algorithme branch-and-bound vu en classe pour résoudre le problème des assignations où la matrice des coûts est la matrice suivante :

	1	2	3	4
a	94	1	54	68
b	74	10	88	82
c	62	88	8	76
d	11	74	81	21

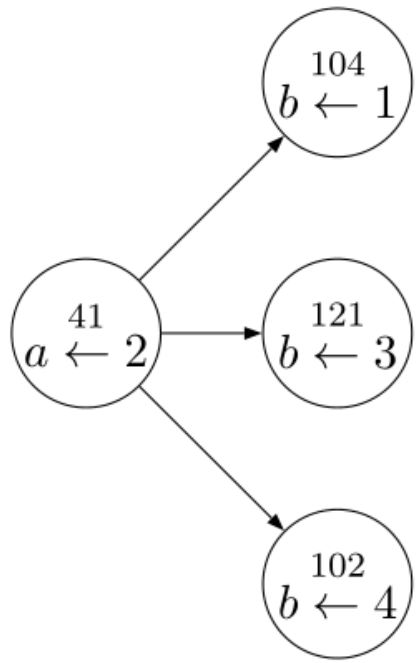
Solution

À partir de l'état initial, 4 options s'offrent à nous pour assigner une valeur à l'agent a.

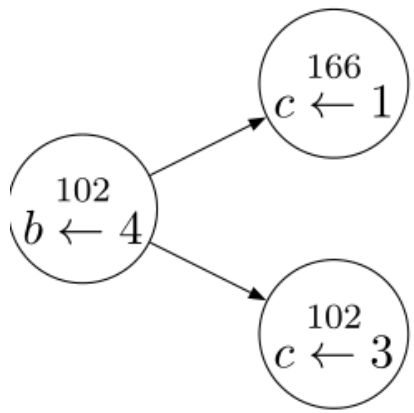


La valeur à côté du sommet correspond à une borne inférieure au coût si on passe par ce sommet. Celle-ci se calcul en rajoutant au coût actuel l'élément le plus petit sur chaque colonne correspondante à une valeur non-assignée (excluant les lignes qui ont étés déjà assignée). ex. sommet 1 : $h(a1) = \text{cout}_{a=1} + \sum_{x \in \{2,3,4\}} min_{y \in \{b,c,d\}} = 94 + 10 + 8 + 21 = 133$

Nous explorons d'abord le sommet qui semble le plus prometteur, c'est-à-dire le second sommet (41). Nous devons maintenant assigner une valeur à l'agent b

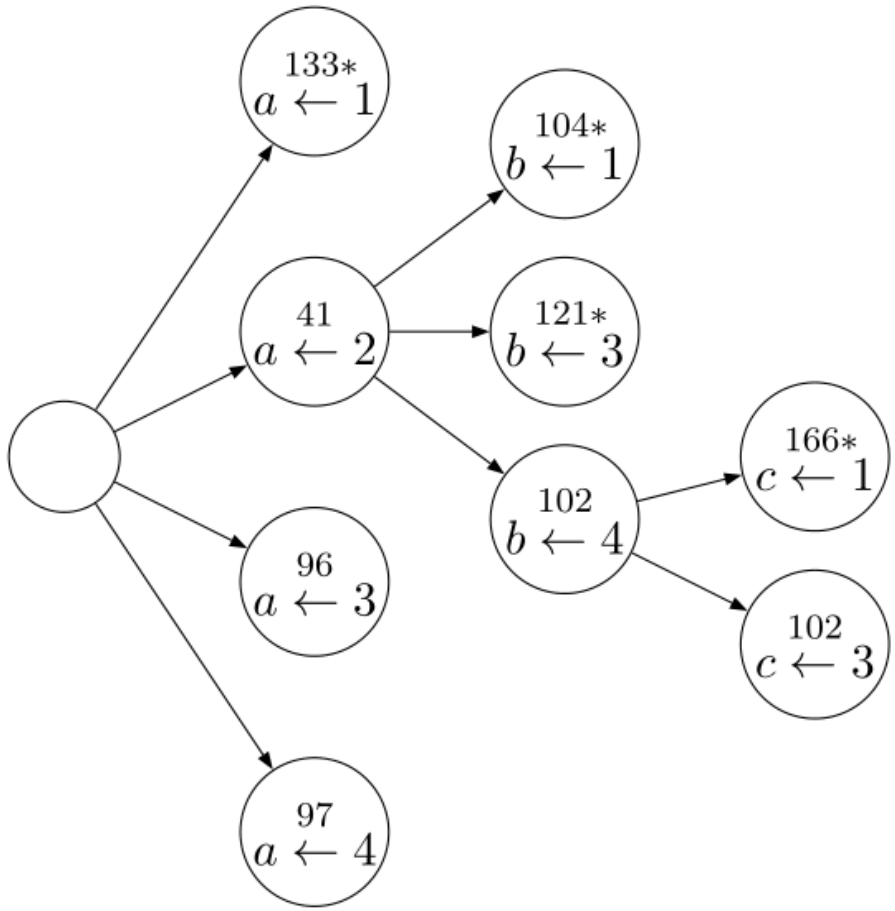


Explorons sommet qui possède la valeur la plus prometteuse, c'est-à-dire le troisième sommet (102). Nous devons maintenant assigner une valeur à l'agent c (et du coup l'agent d)

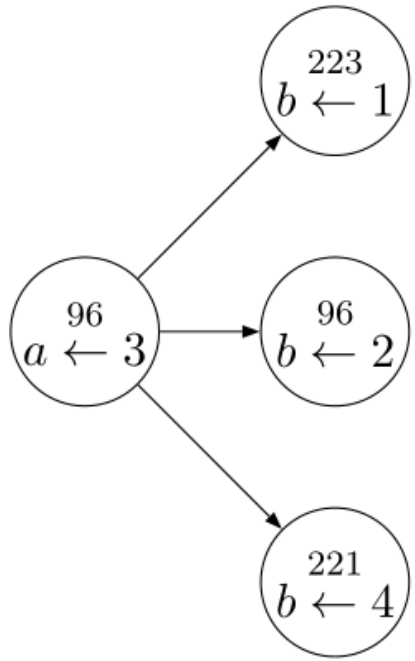


Notre fouille est complète dans cette section.

La meilleure solution actuelle est $a \leftarrow 2, b \leftarrow 4, c \leftarrow 3, d \leftarrow 1$ et sa valeur est 102. Nous élaguons tous les sommets possédant une valeur inférieure (en les marquant d'un astérisque). L'arbre devient donc :

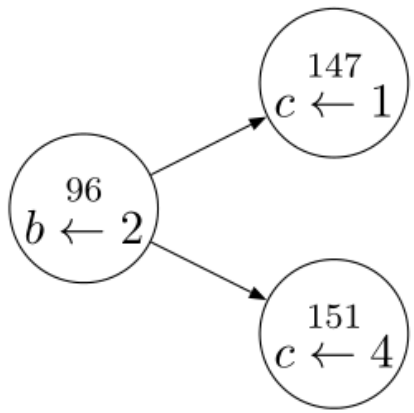


Retournons au premier niveau. Explorons le sommet (non élagué) qui possède la valeur la plus prometteuse, donc le troisième sommet (96). Fixons l'agent b, nous obtenons :



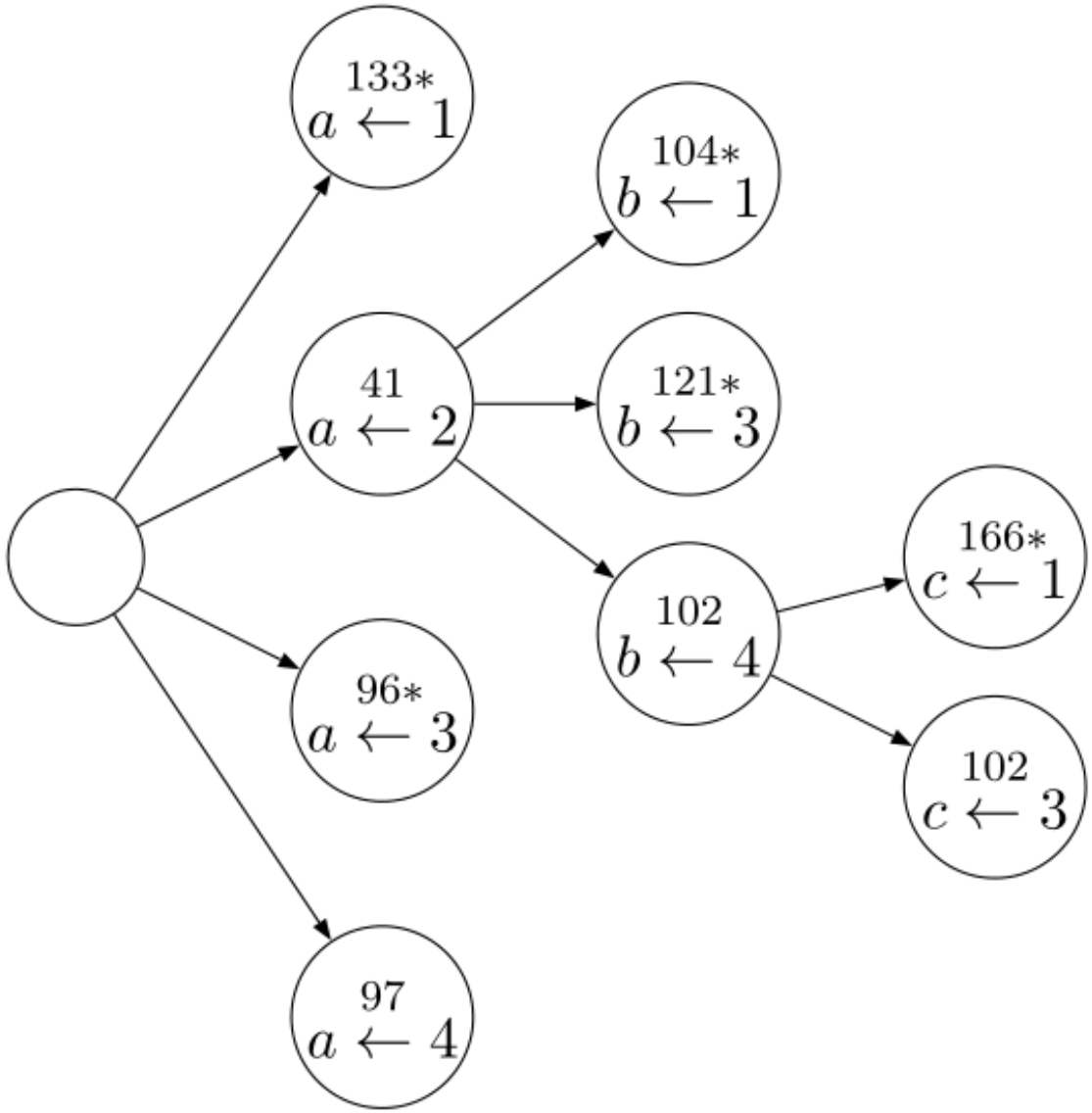
Le premier sommet (223) et le troisième sommet (221) sont élagués car leur valeur est supérieure à 102 (borne supérieure actuelle).

Explorons le second sommet. Nous fixons l'agent c (et du coup l'agent d) et nous obtenons :

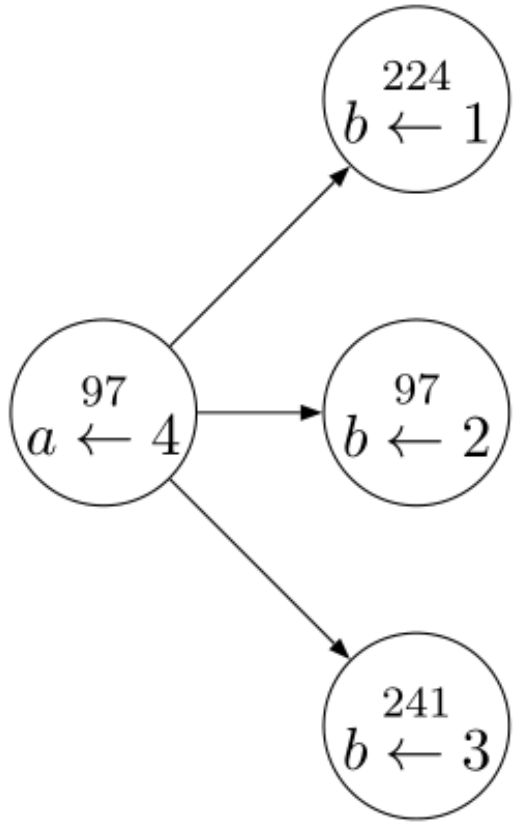


Les deux sommets doivent donc être élagués puisque leur valeur est inférieure à 102. Notre fouille est complète dans cette section.

L'arbre devient donc :

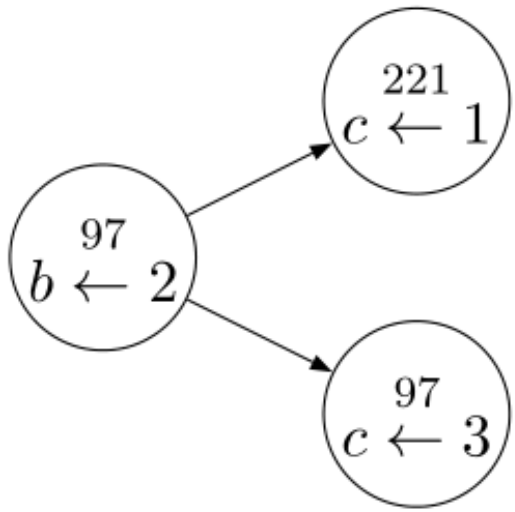


Retournons au premier niveau. Explorons le dernier sommet (97). Fixons l'agent b, nous obtenons :



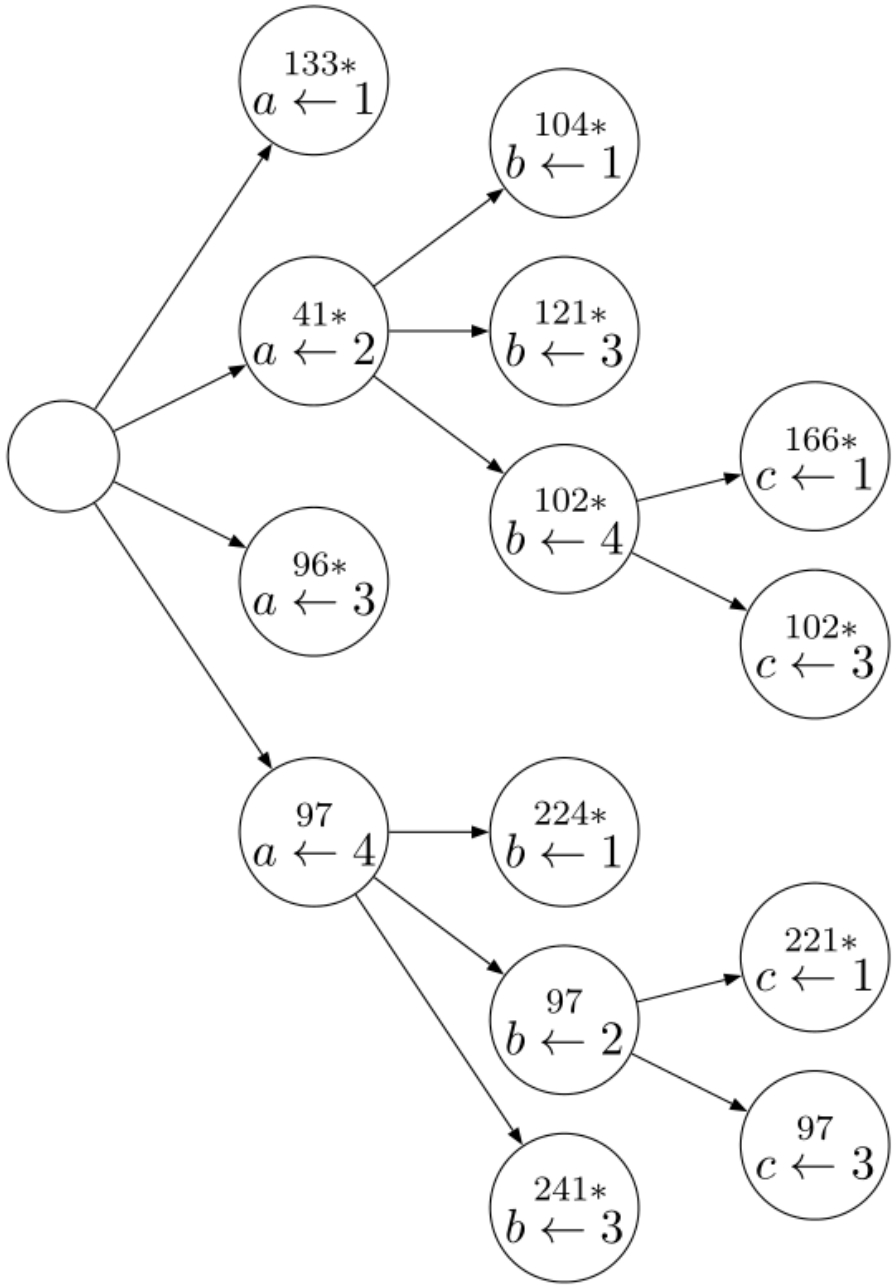
Le premier sommet (224) et le troisième sommet (241) sont élagués car leur valeur est supérieure à 102 (borne supérieure actuelle). Explorons le second sommet.

Nous fixons l'agent c (et du coup l'agent d) et nous obtenons :



Le premier sommet (221) est élagué. La solution optimale devient $a \leftarrow 4, b \leftarrow 2, c \leftarrow 3, d \leftarrow 1$ avec une valeur de 97.

L'arbre final est donc (sans l'expansion du second sommet par souci de lisibilité) :



Numéro 7

Montrez comment résoudre le problème du sac à dos pour lequel on a n types d'objets de valeurs v_i, $1 \leq i \leq n$ et de poids w_i, $1 \leq i \leq n$, et une capacité maximale pour notre sac de W avec une méthode branch-and-bound.

Solution

Nous ordonnons d'abord les éléments de telle sorte que $v_i/w_i \geq v_{i+1}/w_{i+1}$ pour tout i. Nous considérons ensuite des solutions partielles. Si $x_1, \ldots, x_k$ sont fixées de telle sorte que

$$\sum_{i=1}^{k} x_i w_i \leq W,$$

alors nous remarquons que la valeur pouvant être obtenue en ajoutant des éléments de type $k+1, \ldots, n$ ne peut dépasser

$$\sum_{i=1}^{k} x_i v_i + \left(W - \sum_{i=1}^{k} x_i w_i \right) v_{k+1}/w_{k+1}. \quad (*)$$

Nous résolvons donc le problème de la façon suivante. Nous considérons un arbre dont la racine représente la solution vide.

- À chaque niveau i de l'arbre, nous fixons une valeur pour x_i.
- Chaque sommet est identifié par une solution partielle $(x_1, \ldots, x_k)$ et la valeur maximale associée (*).

Nous résolvons donc le problème de la façon suivante. Nous considérons un arbre dont la racine représente la solution vide.

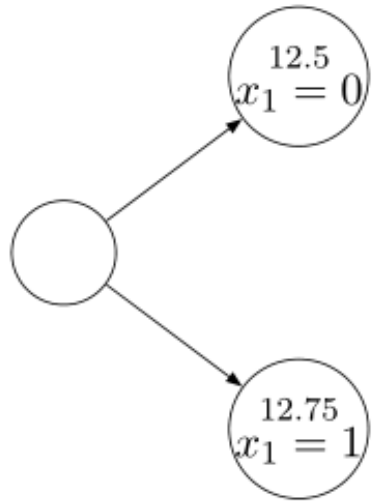
- À chaque niveau i de l'arbre, nous fixons une valeur pour x_i.
- Chaque sommet est identifié par une solution partielle $(x_1, \ldots, x_k)$ et la valeur maximale associée (*).
- Lors de l'exploration d'un sommet, nous considérons seulement les successeurs qui satisfont la contrainte de poids. Lorsqu'un choix s'impose, nous considérons d'abord le sommet ayant la valeur la plus prometteuse (c.-à-d. la plus grande).

Lorsqu'une solution (complète) est obtenue, tous les sommets, ayant une valeur maximale inférieure à celle de cette solution, sont élagués. Le processus est répété jusqu'à ce qu'il n'y ait plus de sommet à explorer. La solution optimale est celle du sommet de niveau n n'ayant pas été élagué.

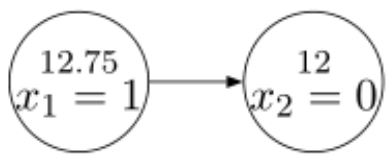
Numéro 8

Problème 9.54, p. 324, livre Brassard-Bratley : Utilisez l'algorithme branch-and-bound développé au numéro précédent pour trouver la solution au problème du sac à dos suivant : on a 4 types d'objets de poids et valeurs respectifs $w_1 = 7$, $w_2 = 4$, $w_3 = 3$, $w_4 = 2$ et $v_1 = 9$, $v_2 = 5$, $v_3 = 3$, $v_4 = 1$ et la capacité maximale de notre sac à dos est $W = 10$.

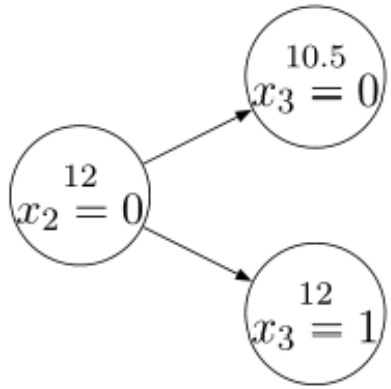
En fixant d'abord x_1, nous obtenons le graphe suivant :



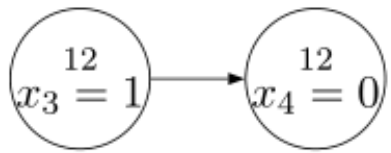
Explorons le sommet qui possède la valeur la plus prometteuse, donc le second sommet (12.75). Nous fixons maintenant x_2 et nous obtenons :



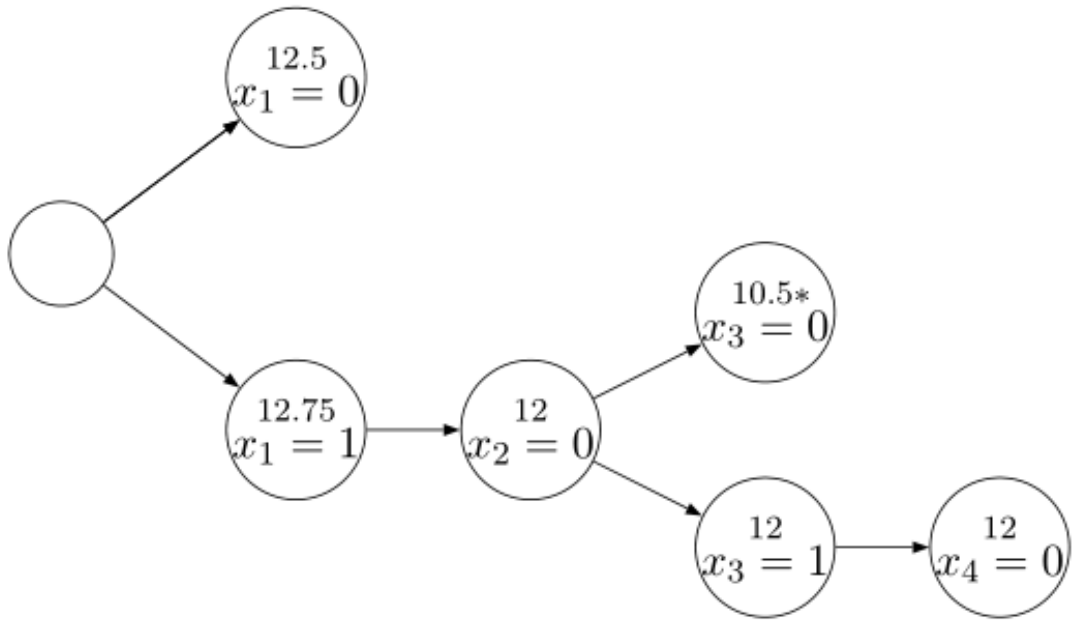
Il n'y a qu'un sommet, explorons-le. Nous fixons maintenant x_3. Nous obtenons :



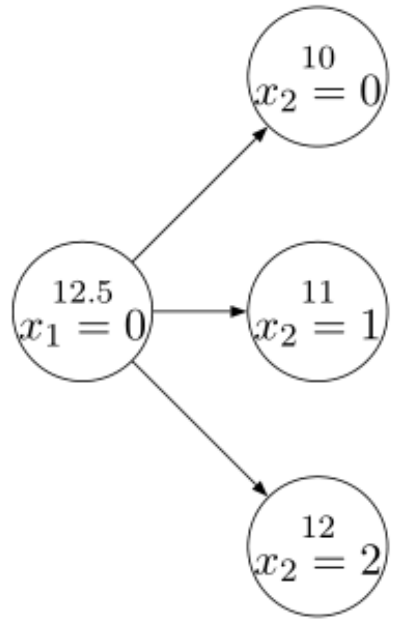
Explorons le sommet qui possède la valeur la plus prometteuse, donc le second sommet (12). Nous fixons maintenant x_4 et nous obtenons :



La solution actuelle est donc $\{x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0\}$ et sa valeur est 12. Nous élaguons le sommet dont la valeur maximale est 10.5. L'arbre actuel est donc :

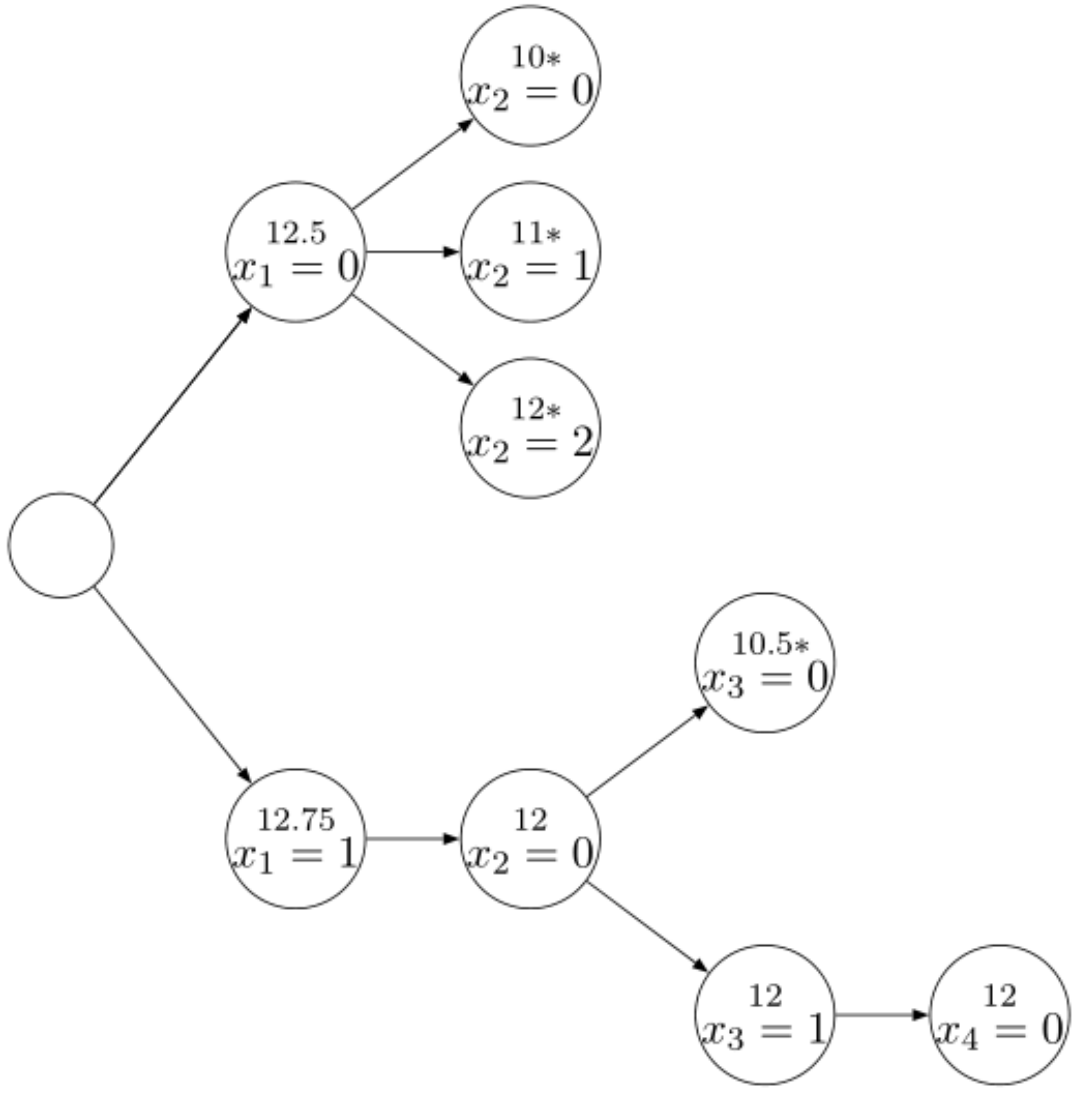


Retournons au premier niveau et explorons le sommet $x_1 = 0$. Explorons le sommet qui possède la valeur la plus prometteuse, donc le second sommet (12.75). Nous fixons maintenant x_2 et nous obtenons :



Les deux premiers sommets doivent être élagués car leur valeur maximale (10 et 11) est inférieure à 12. Il est inutile d'explorer le troisième sommet puisqu'il ne peut améliorer la valeur optimale.

L'arbre final est donc :



La solution optimale est donc $\{x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0\}$ et sa valeur est 12.