RAIRO. RECHERCHE OPÉRATIONNELLE

BERNARD LIOUVILLE

Le problème du voyageur de commerce dans un produit cartésien de deux graphes

RAIRO. Recherche opérationnelle, tome 12, n° 3 (1978), p. 263-275

<http://www.numdam.org/item?id=RO_1978__12_3_263_0>

R.A.I.R.O. Recherche opérationnelle/Operations Research
(vol. 12, n° 3, août 1978, p. 263 à 275)

LE PROBLÈME DU VOYAGEUR DE COMMERCE DANS UN PRODUIT CARTÉSIEN DE DEUX GRAPHES (*)

par Bernard LIOUVILLE (1)

Résumé. — *On considère un produit cartésien de deux graphes, chacun d'eux ayant une chaîne pour arbre minimal. On donne une condition suffisante pour qu'il existe dans ce graphe-produit un arbre minimal inclus dans un cycle hamiltonien minimal, avec une démonstration constructive. Le temps de calcul nécessaire à la construction de ce cycle est borné par une fonction linéaire du nombre de sommets du produit.*

1. INTRODUCTION

1.1. Le problème posé et ses applications

Imaginons une machine-outil susceptible de subir des modifications qui lui permettent d'accomplir n_1 tâches différentes, et que cette machine soit demandée par n_2 utilisateurs, chacun d'eux ayant à effectuer les n_1 travaux. On connaît le coût des modifications possibles, et le coût des passages d'un utilisateur à l'autre. La minimisation du coût total des opérations permettant à tous les utilisateurs de faire tous leurs travaux, avec éventuellement une répétition cyclique du processus, est un problème du type « voyageur de commerce » dans un graphe que l'on peut définir comme le produit du « graphe des modifications » et du « graphe des utilisateurs ».

La solution générale de ce problème semble, malgré sa forme particulière, très complexe, et ne se déduit pas simplement d'une solution connue dans chacun des graphes facteurs. Nous restreindrons donc notre champ d'investigation, d'abord en supposant les coûts symétriques (nous parlerons de graphes « non orientés »),

(*) Reçu juin 1976, révisé mai 1977.
(1) Institut de Programmation, Université Paris VI.

ensuite en se limitant aux graphes ayant une structure particulière, que nous qualifierons de « linéaire ». Les problèmes suivants entrent dans ce cadre :

– impression d'un tableau par une imprimante, en respectant certains écarts entre les lignes, et entre les colonnes;

– balayage par un faisceau électronique d'un ensemble de points sur un écran;

– exploration de certaines parties d'un fichier dont l'accès est à double entrée;

– test de fonctionnement d'un engin dont l'état dépend de deux paramètres; etc.

Bien entendu, dans le cas où l'on connaît une solution dans les deux graphes facteurs sous la forme d'une chaîne hamiltonienne dont le coût de toutes les arêtes est le même (cas du quadrillage dont les faces sont des rectangles isométriques), la solution est simple et coïncide avec (ou contient) celle du problème de l'arbre minimal. Le but de cet article est de donner une condition suffisante à l'existence d'une solution de ce type, et un algorithme permettant de la construire. On trouvera quelques exemples simples dans lesquels cette condition est satisfaite à la fin de cette introduction.

1.2. Définitions

On dira que G est un *graphe linéaire* s'il est muni d'une valuation telle que l'un de ses arbres minimaux soit une chaîne.

Une solution au problème du voyageur de commerce (c'est-à-dire un cycle minimal ou une chaîne minimale) sera dite *absolument optimale* si elle contient un arbre minimal qui est une chaîne.

Les graphes considérés seront finis, non orientés, sans boucles ni arêtes multiples. Si G est un graphe, l'ensemble de ses sommets sera noté $X(G)$, et $E(G)$ l'ensemble de ses arêtes. On supposera de plus $X(G)$ et $E(G)$ non vides. On notera X_i pour $X(G_i)$ et E_i pour $E(G_i)$ $(i=1, 2)$.

Un *produit cartésien* de G_1 et G_2 est un graphe $G_1 . G_2$ dont l'ensemble de sommets $X(G_1 . G_2)$ est le produit cartésien de X_1 par X_2, et dont la relation d'adjacence est déterminée par celle de G_1 et celle de G_2. Comme nous nous occupons d'un problème du voyageur de commerce, G_1, G_2, *et* $G_1 . G_2$ *sont des graphes complets*, et la question de l'adjacence ne se pose pas. C'est la définition du coût sur les arêtes qui importe.

Soit $\{x_1 x_2, y_1 y_2\}$ une arête de $G_1 . G_2$ (le couple (x_1, x_2) est noté $x_1 x_2$). Nous distinguerons trois cas :

(i) $x_2 = y_2$. Dans ce cas nous dirons que l'arête est *de type* 1, et nous lui donnerons le coût de $\{x_1, y_1\}$;

(ii) $x_1 = y_1$. Nous dirons que l'arête est *de type* 2 et nous lui donnerons le coût de $\{x_2, y_2\}$;

(iii) $x_1 \neq y_1$ et $x_2 \neq y_2$. Nous dirons que l'arête est *de type* 3 et nous supposerons que son coût est au moins égal à celui de $\{x_1, y_1\}$ et à celui de $\{x_2, y_2\}$. Cette hypothèse revient à admettre que l'on ne peut effectuer « en même temps » deux opérations de types différents (déplacement horizontal et déplacement vertical par exemple, pour l'imprimante ou le faisceau d'électrons) sans qu'il en coûte au moins autant que d'effectuer la plus coûteuse de ces deux opérations.

1.3. Arbres minimaux dans $G_1 . G_2$

On sait que tout arbre minimal peut être obtenu au moyen de l'un des algorithmes dits de Kruskal [2], selon lequel on examine successivement les arêtes dans un ordre compatible avec le préordre des valeurs non décroissantes, en rejetant toute arête formant un cycle avec celles qui sont déjà choisies.

Soit $e = \{x_1 x_2, y_1 y_2\}$ une arête de type 3. En construisant un arbre minimal, il sera toujours possible d'examiner les deux arêtes : $\{x_1 x_2, y_1 x_2\}$ et $\{y_1 x_2, y_1 y_2\}$ avant e. Si aucune d'elles n'est rejetée, il est clair que e sera rejetée. Si l'une d'elles est rejetée, c'est que ses extrémités sont déjà liées par une chaîne dans l'arbre en construction. On voit que dans tous les cas il peut exister déjà, au moment de l'examen de l'arête e, une chaîne joignant $x_1 x_2$ à $y_1 y_2$ dans l'arbre en construction. Donc e peut toujours être rejetée, et l'on peut énoncer le résultat suivant :

Il existe un arbre minimal de $G_1 . G_2$ qui ne contient aucune arête de type 3.

Dans notre recherche d'une solution absolument optimale, nous nous limiterons à celles de ces solutions qui ne contiennent aucune arête de type 3. En d'autres termes nous substituerons au graphe complet $G_1 . G_2$ la *somme cartésienne* $G_1 + G_2$ [1 et 3], dont l'ensemble des arêtes est défini par : $\{x_1 x_2, y_1 y_2\} \in E(G_1 + G_2)$ si, et seulement si : $\{x_1, y_1\} \in E_1$ et $x_2 = y_2$, ou : $\{x_2, y_2\} \in E_2$ et $x_1 = y_1$; c'est-à-dire, dans le cas où G_1 et G_2 sont complets, que $E(G_1 + G_2)$ est l'ensemble des arêtes de type 1 ou 2. L'abandon des arêtes de type 3 est justifié par le résultat ci-dessus pour la recherche d'un arbre minimal, mais ne l'est pas en général pour la recherche d'un cycle hamiltonien minimal. Nous montrerons cependant à la fin du paragraphe 2 que tout cycle minimal dans $G_1 + G_2$ construit suivant notre algorithme est aussi minimal dans $G_1 . G_2$ (même pour G_1 et G_2 non complets).

On trouvera dans [4] une démonstration du :

THÉORÈME 1 : *Si A_1 est un arbre minimal de G_1, A_2 un arbre minimal de G_2, il existe un arbre minimal de $G_1 + G_2$ qui est inclus dans $A_1 + A_2$.*

Ce résultat permet de travailler à l'intérieur d'un graphe beaucoup plus réduit que le précédent, ayant $2n_1 n_2 - (n_1 + n_2)$ arêtes au lieu de : $(1/2) n_1 n_2 (n_1 + n_2 - 2)$, ou de : $(1/2) n_1 n_2 (n_1 n_2 - 1)$ pour le graphe complet, si n_i désigne le cardinal de X_i.

Dans tout ce qui suit, nous nous placerons donc dans le cas d'une somme de deux arbres, et puisqu'il s'agit de graphes linéaires, ces arbres seront des chaînes élémentaires.

1.4. Exemples de solutions

1) Dans le cas où tous les coûts de G_1 sont inférieurs à tous ceux de G_2, on obtient sans peine un arbre-chaîne minimal comme sur la figure 1 (les arêtes de type 1 sont horizontales, celles de type 2 verticales, les coûts sont les longueurs).

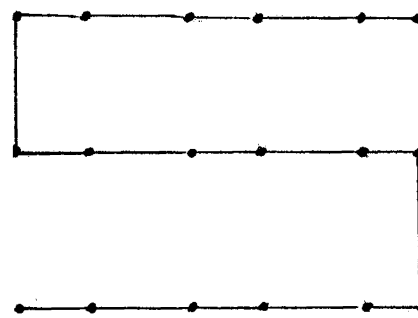


Figure 1

2) Sur la figure 2, on obtient encore une chaîne, bien que certains coûts de G_1 sont inférieurs, d'autres supérieurs à ceux de G_2. On remarquera que $|X_2|$ est impair.

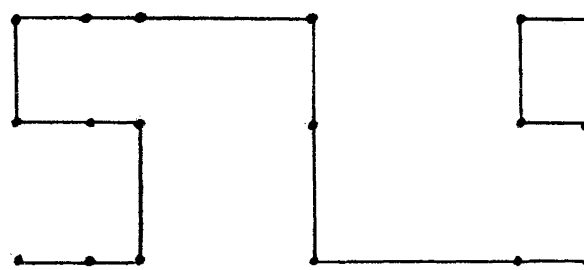


Figure 2

3) Sur la figure 3, G_1 a une seule arête de coût supérieur à ceux de G_2, et X_2 est pair. On obtient cette fois-ci une véritable solution (cycle) absolument optimale.

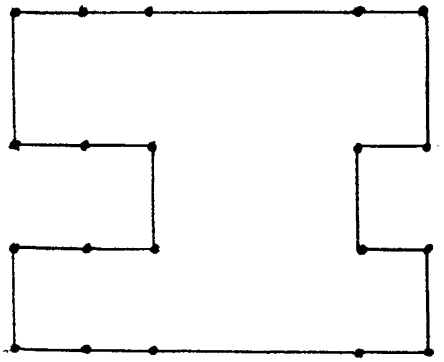


Figure 3

2. UNE CONDITION SUFFISANTE D'OPTIMALITÉ ABSOLUE

Nous utiliserons plus ou moins implicitement les propriétés suivantes [3] :

1) $G_1 + G_2$ *est isomorphe à* $G_2 + G_1$;

2) $G_1 + G_2$ *est connexe ssi* G_1 *et* G_2 *le sont.*

Nous poserons

$$p: \quad E(G_1 + G_2) \to E_1 \cup E_2$$

(G_1 et G_2 sont supposés *disjoints*) telle que, si $e = \{x_1 x_2, y_1 y_2\}$ ·

$$p(e) = \{x_i, y_i\},$$

où i est le type de l'arête e.

D'autre part :

$$f_1: \quad E_1 \to R_+; \qquad f_2: \quad E_2 \to R_+;$$

et

$$f: \quad E(G_1 + G_2) \to R_+$$

seront les fonctions coût, ou valuations des arêtes. On rappelle que f est définie par

$$f(e) = f_i(p(e))$$

pour toute arête e de type i. Par abus d'écriture, on confondra parfois f et f_i.

On sait que la construction d'un arbre minimal suppose le choix d'un ordre total sur les arêtes, compatible avec le préordre des valeurs non décroissantes. Soit v_1 et v_2 les ordres respectivement choisis sur E_1 et sur E_2. On a donc

$$e \underset{v_i}{\prec} e' \quad \Rightarrow \quad f_i(e) \leqq f_i(e')$$

pour tout e, e' dans E_i.

Choisissons sur $E_1 \cup E_2$ un ordre total v compatible avec l'ordre partiel $v_1 \cup v_2$, et inclus dans le préordre des valeurs non décroissantes. Ainsi, si $e \in E_i$ et $e' \in E_j$ on aura

$$e \underset{v}{\prec} e' \quad \Rightarrow \quad ((i=j \text{ et } e \underset{v_i}{\prec} e') \text{ ou } (i \neq j \text{ et } f(e) \leqq f(e'))).$$

G_1 et G_2 étant des chaînes élémentaires ayant respectivement n_1 et n_2 sommets, on associe à chacune de leurs arêtes un entier compris entre 1 et $n_i - 1$ qui représente son rang dans la chaîne (parcourue dans un sens arbitraire), et l'on identifie cette arête au couple (i, x), $i \in \{1, 2\}$. Dans $G_1 + G_2$ on adoptera pour les arêtes les notations : $(1, x, y)$ pour désigner l'arête : $\{s_{1,x}s_{2,y}, s_{1,x+1}s_{2,y}\}$, et : $(2, x, y)$ pour désigner l'arête ; $\{s_{1,y}s_{2,x}, s_{1,y}s_{2,x+1}\}$, où $s_{i,z}$ représente le sommet de rang z dans G_i.

On notera u l'ordre partiel défini sur $E_1 \cup E_2$ par le rang des arêtes

$$(i, x) \underset{u}{\prec} (j, y) \quad \Leftrightarrow \quad (i=j \text{ et } x \leqq y).$$

Définissons deux équivalences α et β sur $E_1 \cup E_2$:

$$e \underset{\alpha}{\sim} e' \text{ ssi : } \exists i,\ e \in E_i,\ e' \in E_i \quad \text{et } \nexists e'' \in E_j,\ j \neq i \text{ et } (e \underset{v}{\prec} e'' \underset{v}{\prec} e' \text{ ou } e' \underset{v}{\prec} e'' \underset{v}{\prec} e).$$

On notera : $C_1, C_2, \ldots, C_k$, les classes définies par cette équivalence, indexées dans l'ordre quotient v/α.

$$e \underset{\beta}{\sim} e' \text{ ssi : } e \in C_i,\ e' \in C_i \text{ et } \forall j > i,\ \nexists e'' \in C_j,\ (e \underset{u}{\prec} e'' \underset{u}{\prec} e' \text{ ou } e' \underset{u}{\prec} e'' \underset{u}{\prec} e).$$

On notera : $C_{i,j}$ $(1 \leqq i \leqq k;\ 1 \leqq j \leqq l_i)$ les classes définies par cette équivalence, avec

$$\bigcup_{1 \leqq j \leqq l_i} C_{i,j} = C_i$$

(il est clair en effet que $\beta \subset \alpha$).

Nous pouvons maintenant énoncer la condition suffisante :

THÉORÈME 2 : a) *si pour tout* i, $2 \leqq i \leqq k-2$, *et pour tout* j, *le nombre* $|C_{i,j}|$ *est pair, et si de plus* $|C_{k-1}|$ *est impair et* $|C_k| = 1$, *alors* $G_1 + G_2$ *admet un cycle absolument minimal;*

b) *si pour tout* i, $2 \leqq i \leqq k-1$, *et pour tout* j, $|C_{i,j}|$ *est pair, alors* $G_1 + G_2$ *admet un arbre minimal qui est une chaîne.*

REMARQUE : Cette condition n'est pas nécessaire, comme on peut s'en rendre compte sur l'exemple suivant : G_1 *ayant trois sommets,* G_2 quatre sommets, on pose

$$f_1(1, 1) = 1, \qquad f_1(1, 2) = 3,$$
$$f_2(2, 1) = 2, \qquad f_2(2, 2) = 4, \qquad f_2(2, 3) = 2.$$

Les arêtes se répartissent en quatre classes :

$$C_1 = \{(1, 1)\}; \qquad C_2 = \{(2, 1), (2, 3)\}; \qquad C_3 = \{(1, 2)\}; \qquad C_4 = \{(2, 2)\}.$$

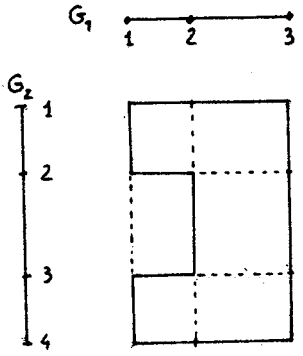
La classe C_2 se subdivise en deux sous-classes :

$$C_{2,1} = \{(2, 1)\} \qquad \text{et} \qquad C_{2,2} = \{(2, 3)\}$$

ayant chacune un seul élément. La condition n'est donc pas satisfaite [ni a) ni b)]. On vérifiera cependant que la suite :

$$(11, 21, 31, 32, 33, 34, 24, 14, 13, 23, 22, 12, 11)$$

constitue un cycle absolument minimal.



COROLLAIRES : 1) *si toute arête de* E_1 *a une valeur inférieure ou égale à toutes les valeurs des arêtes de* E_2, *alors il existe un arbre minimal qui est une chaîne dans* $G_1 + G_2$. *Si de plus* $|X_1|$ *est pair et* $|X_2| = 2$, *alors il existe un cycle absolument minimal.*

2) *si* $|X_2|$ *est impair et si aucune arête de* E_1 *n'a une valeur comprise entre celles de deux arêtes de* E_2, *alors il existe une chaîne absolument minimale.*

3) *si* E_1 *a une seule arête de valeur supérieure à toutes celles de* E_2, *et si toutes les autres arêtes de* E_1 *ont une valeur inférieure à toutes celles de* E_2, *et si de plus* $|X_2|$ *est pair, alors il existe un cycle absolument minimal.*

Démonstration immédiate à partir du théorème : Avant de démontrer le théorème 2, il nous faut donner l'algorithme de construction du cycle (ou de la chaîne) minimal(e), dans le cas où son existence est assurée; mais nous commencerons par donner un algorithme permettant de reconnaître si la condition suffisante est satisfaite ou non.

Algorithme de reconnaissance

On associe d'abord à chaque arête de $E_1 \cup E_2$ sa classe C_i. On peut en profiter pour faire en même temps la vérification de la parité des $|C_i|$ et des conditions désirées sur $|C_{k-1}|$ et $|C_k|$, selon que l'on désire un cycle, ou seulement une chaîne.

Ce travail demande $2(n_1+n_2)$ opérations, si les arêtes sont déjà rangées dans un ordre de valeurs non décroissantes dans E_1, et dans E_2.

On passe ensuite à la reconnaissance de la condition sur les classes $|C_{i,j}|$. On utilise à cet effet une pile P, et l'on pose

$$C(i, x)=j \quad \text{ssi} \quad (i, x)\in C_j.$$

La procédure REJET termine l'exécution en donnant une réponse négative.

Pour $i := 1, 2$ *faire*:
  $P := \emptyset$;
  Pour $x := 1$ *pas* 1 *jusqu'à* n_i *faire*:
    (soit l le sommet de la pile)
    si $1<j<l$ *ou* $P=\emptyset$ *alors*: Poser j sur la pile;
    si $j=l$ *alors*: effacer l;
    si $j>l$ *alors*: REJET;
  si P contient autre chose que k ou $k-1$ *alors*: REJET;

Fin La condition est satisfaite; on peut passer à la construction.

COMPLEXITÉ : Cet algorithme demande n_1+n_2 opérations, ce qui donne un total de $3(n_1+n_2)$ avec la détermination des classes C_i et les vérifications préliminaires.

Algorithme de construction

On commence par associer à chaque arête de $(E_1 \cup E_2) \setminus C_1$ un booléen $b[i, x]$ représentant en un certain sens sa « parité ». Si $(i, x) \in C_1$, $b[i, x]$ reste indéterminé. ε est une variable booléenne auxiliaire.

1. *Pour* $i := 1, 2$ *faire* :

   $\varepsilon := vrai$;

   pour $x := 1$ *pas* 1 *jusqu'à* n_i *faire* :

   si $C(i, x) \neq 1$ *alors* : $b[i, x] := \varepsilon$;

   $\varepsilon := \neg \varepsilon$;

On peut alors passer à l'étape suivante, où les arêtes sont examinées suivant l'ordre v.

La procédure POSER (i, x, y) a pour effet d'ajouter à l'ensemble des arêtes, initialement vide, l'arête (i, x, y).

Le signe : $\oplus$ désigne le « ou » exclusif. La lettre j désigne l'un des deux nombres : 1 ou 2, tel que $j \neq i$.

Soit donc (i, x) la plus petite arête dans l'ordre v.

2. Tant que $C(i, x) = 1$:

   POSER (i, x, y) pour tout y, $1 \leqq y \leqq n_j$, et passer à l'arête suivante en retournant à 2.

3. Soit (i, x) l'arête à examiner.

   Si $b[i, x]$ *alors* POSER $(i, x, 1)$;

   Pour $y := 1$ *pas* 1 *jusqu'à* $n_j - 1$ *faire* :

   si (j, y) non encore examinée *et* $(b[i, x] \oplus b[j, y])$

   alors :

   ⌈ POSER (i, x, y);
   ⌊ POSER $(i, x, y+1)$;

   Si le dernier test pour lequel (j, y) n'était pas encore examinée était négatif (à cause de b) *alors* : POSER (i, x, n_j);

   Si il y a encore des arêtes à examiner, passer à l'arête suivante (dans l'ordre v) et retourner à 3;

Fin : On a construit une chaîne absolument minimale. Si la condition requise (th. 2) est satisfaite on achève la construction du cycle absolument minimal par une ultime opération : POSER (i, x, n_j).

COMPLEXITÉ : La première étape demande $n_1 + n_2$ opérations. Pour les étapes 2 et 3 on dénombre un maximum de $2 n_1 n_2$ examens d'arêtes, et la pose de $n_1 n_2$ arêtes.

Si $N = n_1\, n_2$ est le nombre de sommets de $G_1 + G_2$, on peut donc affirmer que la complexité croît linéairement en N.

Démonstration du théorème 2. Montrons d'abord que l'algorithme précédent construit toujours un arbre minimal.

A l'étape 2, la construction commence comme celle de l'arbre minimal (toutes les composantes connexes de G_j sont triviales, aucune arête de type j n'ayant été posées).

A l'étape 3, si $Y = [y, \ldots, y']$ est une composante connexe dans le graphe partiel de G_j engendré par les arêtes déjà examinées, alors il existe un et un seul élément z de Y tel que (i, x, z) soit POSéE. En effet, l'un des quatre cas suivants peut se présenter :

a) $y = 1$ et $y' \neq n_j$: alors l'arête (i, x, y) sera posée si $b[i, x]$ est *vrai*. La définition du tableau b (étape 1) et la condition du théorème (le nombre d'arêtes de type j déjà examinées et n'appartenant pas à C_1 est pair dans Y) impliquent que $b[j, y']$ est *vrai*. Donc l'une des deux arêtes : (i, x, y) et (i, x, y'), et l'une seulement sera POSéE (à moins que $y = y'$ auquel cas on a une seule arête, qui sera POSéE).

b) $y \neq 1$ et $y' = n_j$: le dernier test concernait $(j, y-1)$; donc l'une des deux arêtes : (i, x, y) et (i, x, y'), et l'une seulement sera POSéE (à moins que $y = y'$ auquel cas on a une seule arête, qui sera posée).

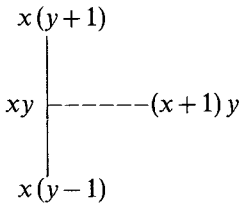
c) $y = 1$ et $y' = n_j$: on vérifie aisément le même résultat dans ce cas.

d) $y \neq 1$ et $y' \neq n_j$: la condition du théorème implique : $b[j, y-1] \oplus b[j, y']$. Encore une fois, l'une des deux arêtes (i, x, y) et (i, x, y') et l'une seulement, sera posée (à moins que $y = y' \ldots$).

Enfin il est clair qu'aucune des arêtes : (i, x, z), $z \in Y$, $z \neq y$ et $z \neq y'$, ne peut être posée.

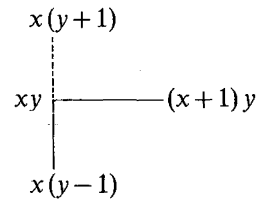
On a donc bien construit un arbre minimal, conformément à l'algorithme de Kruskal. Montrons maintenant que cet arbre est une chaîne, et pour cela il nous suffit de montrer qu'à aucun moment l'algorithme de construction ne peut produire de sommet de degré 3.

Si une telle situation apparaissait au cours de la construction, elle pourrait apparaître de deux façons :



a) Les deux arêtes $(j, y-1, x)$ et (j, y, x) étant posées, on poserait (i, x, y). On vient de voir que c'est impossible, les sommets de rangs $y-1$, y, $y+1$ appartenant à une même composante connexe.

Même raisonnement si l'on veut poser $(i, x-1, y)$.



b) deux arêtes de type différent, ayant un sommet commun, étant posées, soit par exemple : (i, x, y) et $(j, y-1, x)$, on en poserait une troisième telle que : (j, y, x) par exemple.

Nous allons montrer que : $b[i, x-1] \oplus b[j, y]$ est *faux*, ce qui suffira à prouver que (j, y, x) ne peut être posée. Pour cela, on examine les deux cas suivants :

(i) l'arête (i, x) précède $(j, y-1)$ dans l'ordre v. Dans ce cas, lorsqu'on examine l'arête $(j, y-1)$, on ne s'arrête pas à la valeur x pour poser $(j, y-1, x)$ et $(j, y-1, x+1)$, car la réponse au test est négative. Et si cependant on pose $(j, y-1, x)$, c'est que la réponse au test précédent (valeur $x-1$) est positive. On a donc

$$b[i, x-1] \oplus b[j, y-1] = vrai.$$

D'autre part, il est clair que $(j, y) \notin C_1$ [sinon (j, y, x) serait déjà posé]. D'après la construction du tableau b, on a donc

$$b[j, y-1] \oplus b[j, y]$$

et l'on en déduit

$$b[i, x-1] \quad \Leftrightarrow \quad b[j, y];$$

(ii) on fait le même raisonnement en examinant l'arête (i, x), et en supposant : $(j, y-1) \underset{v}{\prec} (i, x)$.

La réponse au test est négative pour la valeur $y-1$. Comme cependant on pose (i, x, y), la réponse au test suivant (valeur y) est positive, et : $b[i, x] \oplus b[j, y]$ est *vrai*; d'autre part, en raisonnant comme précédemment, on voit que : $b[i, x-1] \oplus b[i, x]$ est *vrai*, ce qui implique encore : $b[i, x-1] \Leftrightarrow b[j, y]$.

Le soin de vérifier que le même raisonnement s'applique dans les sept autres combinaisons possibles des deux arêtes posées (de types différents) et de la troisième à poser, ainsi que dans tous les cas particuliers tels que : $x \in \{1, 2, n_i - 1, n_1\}$ ou : $y \in \{1, 2, n_j - 1, n_j\}$, est laissé au lecteur.

Si la condition *a*) du théorème est satisfaite, l'algorithme conduit à poser l'arête (i, x, n_j). La longueur du cycle ainsi formé est celle de l'arbre minimal, soit : λ, plus $f(i, x)$. Cette valeur est évidemment minimale pour un cycle contenant un arbre minimal. Montrons que le cycle obtenu est minimal parmi tous les cycles hamiltoniens de $G_1 . G_2$.

Soit C un tel cycle, et e l'une de ses arêtes de valeur maximale. On sait [2] que tout arbre recouvrant d'un graphe contient au moins une arête de valeur supérieure ou égale à la valeur de toute arête d'un arbre minimal. La chaîne obtenue à partir de C en supprimant l'arête e contient donc une arête de valeur supérieure ou égale à $f(i, x)$; d'où : $f(e) \geqq f(i, x)$, et la longueur de C est au moins égale à $\lambda + f(i, x)$.

Q.E.D.

Exemple d'application du théorème 2 et des algorithmes

$$n_1 = 6;\ n_2 = 7;$$

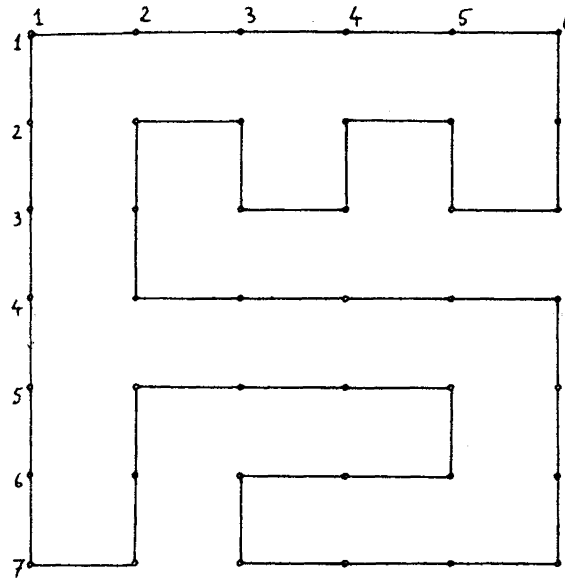
$f_1 : (1, 1) \mapsto 4.0$	$f_2 : (2, 1) \mapsto 2.0$	$C_1 = \{(2, 2)\}$
$(1, 2) \mapsto 2.5$	$(2, 2) \mapsto 0.6$	$C_2 = \{(1, 4), (1, 3)\}$
$(1, 3) \mapsto 1.5$	$(2, 3) \mapsto 2.5$	$C_3 = \{(2, 1), (2, 3), (2, 5), (2, 6)\}$
$(1, 4) \mapsto 0.8$	$(2, 4) \mapsto 3.2$	$C_4 = \{(1, 2), (1, 5)\}$
$(1, 5) \mapsto 3.0$	$(2, 5) \mapsto 2.0$	$C_5 = \{(2,4)\}$
	$(2, 6) \mapsto 2.2$	$C_6 = \{(1, 1)\}$

États successifs de la pile P au cours de l'algorithme de reconnaissance (le chiffre le plus à gauche est le sommet de P).

x	P ($i = 1$)
1	6
2	46
3	246
4	46
5	6

x	P ($i = 2$)
1	3
2	3
3	0
4	5
5	35
6	5

P ne contenant rien d'autre que 6 ou 5 à la fin de chaque série, la condition est satisfaite. L'algorithme de construction donnera le résultat suivant : (les arêtes de type 1 sont horizontales).



REMARQUE : On aura remarqué que $f(1, 2) = f(2, 3)$. On a choisi ici l'ordre : $(2, 3) \prec (1, 2)$. Un choix différent aurait conduit à un REJET.

BIBLIOGRAPHIE

1. C. BERGE, *Graphes et hypergraphes*, Dunod, Paris, 1970.
2. P. ROSENSTIEHL : *L'arbre minimal d'un graphe*. Théorie des graphes (Journées internationales d'études, Rome, 1966) Dunod, Paris, 1967, p. 357-368.
3. C. F. PICARD, *Graphes et questionnaires*, Gauthier-Villars, Paris, t. 1, chap. 7, 1972.
4. C. F. PICARD, *Arbre minimal d'une somme de graphes*, Cahiers du C.E.R.O., vol. 17 n^{os} 2, 3, 4, 1975, p. 351-368.
5. I. TOMESCU, *Un algorithme pour l'obtention d'une chaîne hamiltonienne en partant de l'arbre minimal d'un graphe*, R.A.I.R.O. vol. 3, octobre 1975, p. 5-12.