PolyPublie
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Vertex packings : (VLP)-reductions through alternate labelling
Title:

Auteurs: Jean-Claude Picard, & Maurice Queyranne
Authors:

Date: 1975

Type: Rapport / Report

Référence: Picard, J.-C., & Queyranne, M. (1975). Vertex packings : (VLP)-reductions through alternate labelling. (Rapport technique n° EP-R-75-47).
Citation: https://publications.polymtl.ca/6054/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: https://publications.polymtl.ca/6054/
PolyPublie URL:

Version: Version officielle de l'éditeur / Published version

Conditions d'utilisation: Tous droits réservés / All rights reserved
Terms of Use:

Document publié chez l'éditeur officiel
Document issued by the official publisher

Institution: École Polytechnique de Montréal

Numéro de rapport: EP-R-75-47
Report number:

URL officiel:
Official URL:

Mention légale:
Legal notice:

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

DÉPARTEMENT DE GÉNIE INDUSTRIEL

RAPPORT TECHNIQUE : EP75 - R - 47
Classification: Library of Congress

VERTEX PACKINGS: (VLP)-REDUCTIONS THROUGH ALTERNATE LABELLING

BY: Jean-Claude Picard, Ecole Polytechnique
Maurice Queyranne, Ecole Polytechnique

September 1975

Ecole Polytechnique de Montréal

CA2PQ
UP4
75R47

C.P. 501
Snowdon
Montréal 248

Bibliothèque
Ecole
Polytechnique
MONTRÉAL

CLASSIFICATION

CA2PQ
UP4
75R47

No D'ENTRÉE

77441

ECOLE POLYTECHNIQUE BIBLIOTHEQUE

ECOLE POLYTECHNIQUE - BIBLIOTHEQ
19 SEPT 1975

- dup -

A CONSULTER
SUR PLACE

77441

RAPPORT TECHNIQUE : EP75 - R - 47
Classification: Library of Congress

VERTEX PACKINGS: (VLP)-REDUCTIONS THROUGH ALTERNATE LABELLING

BY: Jean-Claude Picard, Ecole Polytechnique
Maurice Queyranne, Ecole Polytechnique

September 1975

77441

VERTEX PACKINGS: (VLP)-REDUCTIONS THROUGH ALTERNATE LABELLING

ABSTRAT

In the vertex packing problem (i.e. determine, in a graph, a maximum-weighted set of vertices, no two of them being adjacent), a (VLP)-reduction consists in fixing at the same value all the variables which are integer-valued in an optimum solution to a related linear program called (VLP). An algorithm for determining the maximum set of such variables is described; it works by alternate labelling on the vertices of the graph, the main part of this labelling being done along trees; the performances of the algorithm may be measured by the number of assigned labels: vertices corresponding to integer-valued variables need at most two labels to be identified; an upper bound on the number of labels assigned to the other vertices is of the order of half the total number of vertices in the graph.

VERTEX PACKINGS: (VLP)-REDUCTIONS THROUGH ALTERNATE LABELLING

1. INTRODUCTION

The definitions and notations given here are from |6|. Let $G = (V,E)$ be a finite, undirected, loopless graph with weights c_i on vertices $v_i \in V$. A vertex packing ($v.p$) is a subset P of V for which $v_i, v_j \in P$ implies $(v_i, v_j) \notin E$. The weight $c(P)$ of a $v.p$ is defined as $c(P) = \sum_{v_j \in P} c_j$. There is no loss of generality in assuming that $c_j > 0$ for all $v_j \in V$, and that there is no isolated vertex in G (i.e. a vertex with no edge adjacent to it).

Determining a maximum weighted $v.p.$ may be formulated as the integer program:

$$
\begin{aligned}
& \textit{Max } cX \\
& \textit{s.t.} \\
(VP) \qquad & AX \leq 1_m \\
& x_j = 0,1 \; ; \; j = 1,2,\ldots,n
\end{aligned}
$$

in which $m = |E|$, $1_m = (1,\ldots,1)$ is an m-vector of 1's and A is the $m \times n$ edge-vertex incidence matrix of G.

Relaxing the integrality constraints to $X \geq 0_n$ gives the v.p. linear program (VLP).

By the transformation:

$$U = 1_m - X \qquad (1\text{-}1)$$

we obtain the integer program:

$$
\begin{aligned}
& \textit{Min } cU \\
& \textit{s.t.} \\
(CP) \quad & AU \geq 1_m \\
& u_i = 0,1 \ ; \ i = 1,2,\ldots,n
\end{aligned}
$$

This problem is the one of finding a minimum weighted covering of edges by nodes (cf. |1|); here we simply call it the covering problem. Let (CLP) be the linear relaxation of (CP); by (1-1), we obtain (CLP) from (VLP).

The dual of (CLP) is:

$$
\begin{aligned}
& \textit{Max } 1_m \, Y \\
& \textit{s.t.} \\
(MLP) \quad & Y A \leq c \\
& y_j \geq 0 \ ; \ j = 1,2,\ldots,m
\end{aligned}
$$

These linear relaxations have some well-known properties (|6|,|7|):

(i) any basic feasible solution to (VLP) and (CLP) are $(0,\frac{1}{2},1)$-valued

(ii) solving (MLP), and hence (CLP) and (VLP) may be done by a good algorithm

(iii) any integer-valued variable in an optimum solution to (VLP) keeps the same value in an optimum solution to (VP).

Using the (VLP)-reduction suggested by (iii), i.e. fixing at the same value all these integer-valued variables and discarding from the graph the corresponding vertices and all the edges adjacent to them, leads to the problem of determining an optimum solution to (VLP) having

the maximum set of integer-valued variables.

In |7| , a labelling procedure, derived from sensitivity analysis in a maximum-flow problem, was described. The purpose of this paper is to give an improved version of this labelling procedure.

In part 2., some results about (MLP) are recalled. Two algorithms for solving (MLP) have appeared in the literature; the simplex-based first if from E. Johnson |3|; the second uses a maximum flow in a related bipartite graph, and was given by J. Edmonds and W. Pulleyblank (see |6|). In part 3. the labelling procedure of |7| is recalled and the symmetry property of the maximum flow in the Edmonds and Pulleyblank graph leads to a number of results about the labelling process. The structure of a basic solution in the simplex solution is also used to simplify the labelling. These results are used in part 4. where the improved labelling procedure is given and illustrated on an example.

2. (MLP) SOLUTIONS

Consider first the network-based approach, from Edmonds and Pulleyblank: Let V' be a copy of the vertex set V of G, in which $v' \in V'$ corresponds to $v \in V$. Let $W = V \cup V' \cup \{s,t\}$, where s(resp.t) is an artificial source (resp.sink) and $H = (W, F, \bar{c})$ a network whose arcs are:

(s,v_j) with capacity c_j for all j

(v_j,v'_k) with infinite capacity for all edges $(v_j,v_k) \in E$

(v'_k,t) with capacity c_k for all k

A maximum flow F in H defines an optimum solution Y to (MLP), by:

$$y_{ij} = f_{ij}$$

A minimum cut $(S;\bar{S})$ in H defines an optimum solution x to (VLP), by:

$$x_j = 1 \text{ if } v_j \in S \text{ and } v'_j \in \bar{S}$$
$$0 \text{ if } v_j \in \bar{S} \text{ and } v'_j \in S$$
$$\tfrac{1}{2} \text{ otherwise}$$

On the other hand, since (MLP) is a linear program, it may be solved by the simplex algorithm. A specific version of it, is the "Alternating Path Algorithm" deviced by E. Johnson and described p. 39-42 of |3|. Essentially, it is a specialization of the simplex algorithm using graphic properties. It may be seen as an efficient implementation of the max-flow routine on H, with these additional specifications:

(i) it works directly on G; the flow F is always kept symmetric:

$$f_{ij} = f_{ji}$$

(ii) it keeps a basic solution: after a node is labelled, all the arcs a having flow strictly between the upper and lower bound (i.e. $f_a > 0$ and $f_a < c_j$ if $a = (s,v_j)$ or (v_j,t)) are considered first in the labelling step (see |3|, |4|).

A basic solution of (MLP) defines a subgraph G_b of G, whose connected components are:

(i) trees with a "slack" on a vertex (called "root"), these trees are called "rooted trees"; the corresponding variables in (VLP) have value 1 for the root and all the vertices of the tree such that the unique path connecting them (in the tree) to the root contains an even number of edges, and 0 for the other vertices (located at an "odd" distance from the root)

(ii) pseudo-trees, i.e. subgraphs with the same number of vertices and

edges, containing exactly one odd cycle; the corresponding (VLP) variables have the value $\frac{1}{2}$ see(|3|, |5|).

We may first discard from the graph all the vertices in rooted trees, since the corresponding variables are integer-valued. In the remaining graph consider the subgraph G_f defined by the edges such that the (MLP) variables are positive; we may call them "edges with flow". The connected components of G_f are:

(i) pseudo-trees

(ii) (unrooted) trees, obtained by deletion of edges connecting these subgraphs to the odd cycle in the pseudo-tree, or by deletion of edges in the odd cycle itself.

Since $c_i > 0$ for all i, and no slack remains after discarding the rooted trees, each vertex v_i is adjacent to some edge in G_f.

3. ALTERNATING PATHS AND CYCLES

The Labelling Procedure was defined in |7| as follows:

Labelling procedure:

(1) Discarding: discard all the vertices $v_i \in V$, such that x_i is integer-valued.

(2) Initiating: choose an unscanned vertex $v_i \in V$, label v_i and go to step 3; if all the remaining vertices v_i of V are scanned, terminate.

(3) Direct labelling: label all the vertices v_k' such that there exists an edge $(v_j, v_k') \in F$ and v_j is labelled.

(4) Test: if v_i' is labelled, then v_i is scanned and go to step (2).

(5) Reverse labelling: label all the vertices v_k such that there exists a labelled vertex v'_j and $f_{jk} > 0$.
If there is no new labelling go to step (6), otherwise go to step (3).

(6) Solution modification: the set S of all labelled vertices, together with $\bar{S}$, defines a cut $(S,\bar{S})$ in H. Redefine X by (3-1) and go to step (1).

In order to study this labelling process, suppose first that the source s, the sink t and all the arcs adjacent to them, be deleted from H; the remainder is a symmetric bipartite graph B_G with values $f_{ij} \geq 0$ on the arcs (v_i,v_j); from now on, F is supposed to be symmetric, i.e. $f_{ij} = f_{ji}$ for all edges $(v_i,v_j) \in E$. If $f_{ij} > 0$, the edge e_{ij} will be said "containing a positive flow".

Consider a path (u,w) from u to w in B_G; u and w may belong either to V or to V'; a path is a succession of arcs such that any one has one vertex together with the previous arc and the other vertex together with the following arc; u (resp. w) is the vertex adjacent to the first (resp. last) edge and not to the second (resp. previous). The arcs in the path (u,w) are alternately forward arcs and reverse arcs (see |2| p. 12). We call this path an "alternating path" from u to w if all the reverse arcs (v_i,v_j) contain a positive flow f_{ij}. Alternating path are considered here, for the vertices are labelled, in the above Labelling Procedure, along alternating paths issued in v_i.

By using the symmetry property of the graph B_G and the flow , we may derive more about this labelling process. Define first the "image" $(\hat{w},\hat{u})$

of a path (u,w) as follows:

(i) $\hat{u}$ (resp. $\hat{w}$) is the "image" of u (resp. w) i.e.

$$\hat{u} = v'_j \quad \text{if} \quad u = v_j$$

$$\text{and} \quad \hat{u} = v_j \quad \text{if} \quad u = v'_j$$

(ii) to a forward arc (v_j,v'_k) in (u,w) corresponds a forward arc (v_k,v'_j) in $(\hat{w},\hat{u})$

(iii) to a reverse arc (v'_k,v_j) in (u,w) corresponds a reverse arc (v'_j,v_k) in $(\hat{w},\hat{u})$.

Example 1: The image of the path (v_1,v'_2,v_3,v'_4) is the path (v_4, v'_3, v_2, v'_1).

LEMMA 1

A path (u,w) is an alternating path if and only if its image $(\hat{w},\hat{u})$ is an alternating path.

PROOF: It follows obviously from the part (iii) of the above definition. #

LEMMA 2

Consider the labelling process initiated from v_i in step 2 of the Labelling Procedure; if both a vertex v_j and its copy v'_j are labelled, then v'_i may be labelled and no solution modification occurs.

PROOF: If v_j was labelled first, this labelling has been done along an alternating path (v_i,v_j); since v'_j is labelled and (v'_j,v'_i) is an alternating path (lemma 1), v'_i may be labelled.

If v'_j was labelled before v_j, the same result may be shown by exchanging v_j and v'_j in the above argument. #

Now we use the structure of the subgraph G_f.

LEMMA 3

If v_i belongs to a pseudo-tree in G_f there is an alternating path from v_i to v'_i and an alternating path from v'_i to v_i.

PROOF: If v_i belongs to the odd cycle in the pseudo-tree, then describe this cycle from v_i to v_i, in any direction: thus we define, in the bipartite graph B_G, an alternating path from v_i to v'_i (all the arcs contain a positive flow).

If v_i belongs to a branch, there is a path connecting v_i to the cycle. If this path is (v_i,v_j) in G, it defines an alternating path (v_i,v_j) (resp. (v_i,v'_j)) if (v_i,v_j) contains an odd (resp. even) number of edges; thus v_j (resp. v'_j) may be labelled, and, following the cycle as in the proof, we may label v'_j (resp. v_j); then, we may apply lemma 2. to exhibit an alternating path (v_i,v'_i).

Now replace all the vertices in (v_i,v'_i) by their image: the path (v'_i,v_i) obtained is alternating since all its edges drive a positive flow. #

COROLLARY 4

In the labelling process initiated from a vertex v_i, if a vertex v_j or its copy v'_j, belonging to a pseudo-tree in G_f is labelled, then v'_i may be labelled.

LEMMA 5

Consider the labelling process initiated from a pseudo-tree vertex v_i.

If a vertex v'_j is labelled, then there is an alternating path (v_j,v'_j).

PROOF: The image of (v_i,v'_j) is the alternating path (v_j,v'_i); then, in the labelling process initiated from v_j, we may label v'_i and the result

follows from corollary 4. #

The odd cycle in a pseudo-tree of G_f is an instance of a more general pattern called "alternating cycle". It is defined as a pair of two alternating paths (v_i,v'_i) and (v'_i,v_i) and will be denoted by $((v_i,v'_i))$.

Lemmas 3 and 5, and corollary 4 are generalized by the following results.

LEMMA 6

If a vertex v_i or its copy v'_i belongs to an alternating cycle $((v_j,v'_j))$ (i.e. belongs to (v_j,v'_j) or to (v'_j,v_j) there is an alternating path (v_i,v'_i) and an alternating path (v'_i,v_i).

PROOF: Four cases are to be considered in order to show the existence of (v_i,v'_i):

(i) v_i belongs to (v_j,v'_j); this alternating path contains the alternating path (v_i,v'_j); if we initiate the labelling process from v_i, we may label v'_j; then v_j may be labelled along (v'_j,v_j) and the result follows from lemma 2.

(ii) v'_i belongs to (v_j,v'_j); thus v'_i belongs belongs to the image (v_j,v'_j) of (v_j,v'_j) and the above argument may be applied.

In the two last cases:

(iii) v_i belongs to (v'_j,v_j)

(iv) v'_i belongs to (v'_j,v_j), exchange v_j and v'_j in the above arguments. The existence of (v'_i,v_i) is established in considering that v_i belongs to $(u,\hat{u})$ implies that v'_i belongs to the image $(u,\hat{u})$. #

COROLLARY 7

In the labelling process initiated from a vertex v_i, if a vertex v_j or its copy v'_j, belonging to an alternating cycle $((v_k,v'_k))$, is labelled, then v'_i may be labelled.

LEMMA 8

Consider the labelling process initiated from an alternating cycle vertex v_i; if a vertex v'_j is labelled, then there is an alternating path (v_j,v'_j).

PROOF: The image of the alternating path (v_i,v'_j) is the alternating path (v_j,v'_i); in the labelling process initiated from v_j, v'_i may be labelled and the result follows from corollary 7.

4. ALTERNATE LABELLING

The Labelling Process, working in the symmetric bipartite graph , may be implemented directly on the original graph G . We use labels"+ and -":

labelling v_i with + corresponds to labelling v_i in B_G;

labelling v_i with - corresponds to labelling v'_i in B_G.

The basic labelling rule is reduced to:

(i) "forward labelling": label with - all the vertices adjacent to a + labelled vertex;

(ii) "reverse labelling": label with + all the vertices v_j such that there is a - labelled vertex v_i and the flow f_{ij} is positive.

Notice that "reverse labellings" are only done along edges with flow, i.e. in the connected components of G_f.

Consider a tree T in G_f; if a vertex $v_i \in T$ is labelled, all the vertices v_j in T may be labelled in the following way:

"v_j has the same label as v_i if (and only if) the path connecting v_i and v_j in T, has an even number of edges".

By "labelling the tree T" we mean assigning labels to all the vertices in T. We will not go further in the computational aspects of this tree-labelling, its implementation depending on the way the tree T is stored (see |4|, |8|).

The results in section 3. about pseudo-trees in G_f allow the following "pseudo-tree reduction". In the run of this procedure, a vertex v_i may be in one of the four states; unlabelled, + labelled, - labelled, and"+ and -" labelled (this last case occuring when a second distinct label is assigned to v_i).

ROUTINE 1 (PSEUDO-TREE REDUCTION):

0. Let S be the set of all the vertices belonging to some pseudo-tree in G_f. Label all these vertices with + and go to step 1.
1. If there is an edge connecting a + labelled or"+ and -"labelled vertex to an unlabelled or + labelled vertex $v_i \notin S$, then go to step 2, else go to step 3.
2. Label v_i with - ; label the tree T containing v_i; go to step 1.
3. (no more labelling may be done)

   Set $x_i = \frac{1}{2}$ for all the pseudo-tree vertices and all the"+ and -"labelled vertices v_i.

   Set $x_i = 1$ for all the + labelled vertices v_i.

   Set $x_i = 0$ for all the - labelled vertices v_i.

Example 2:

Consider the graph shown in Fig. 1., with weights $c_i = 1$ for all $i = 1,\ldots,21$. A (MLP) solution is pictured on Fig. 1. by dotted lines (v_i,v_j) for $y_{ij} = \frac{1}{2}$ and thick lines (v_i,v_j) for $y_{ij} = 1$. This solution is described by the odd cycle (v_1,v_2,v_3) and the matching (v_{2k},v_{2k+1}) for all $k = 2,\ldots,10$; the unique pseudo-tree is the odd cycle (v_1,v_2,v_3) and Routine 1 is applied to it:

step	edge	vertex	label
0		1	+
		2	+
		3	+
1	(v_1,v_4)		
2		v_4	-
		v_5	+
1	(v_1,v_5)		
2		v_5	- +
		v_4	+ -
1	(v_1,v_6)		
		v_6	-
		v_7	+
1	(v_2,v_8)		
		v_8	-
		v_9	+
3	Solution modification:		$x_1 = x_2 = x_3 = x_4 = x_5 = \frac{1}{2}$
			$x_6 = x_8 = 0$
			$x_7 = x_9 = 1$

THEOREM 9 (Validation of Routine 1)

(i) there is no optimum solution to (VLP) in which a variable x_i, corresponding to a pseudo-tree or"+ and -"labelled vertex v_i after completion of Routine 1., may be integer valued.

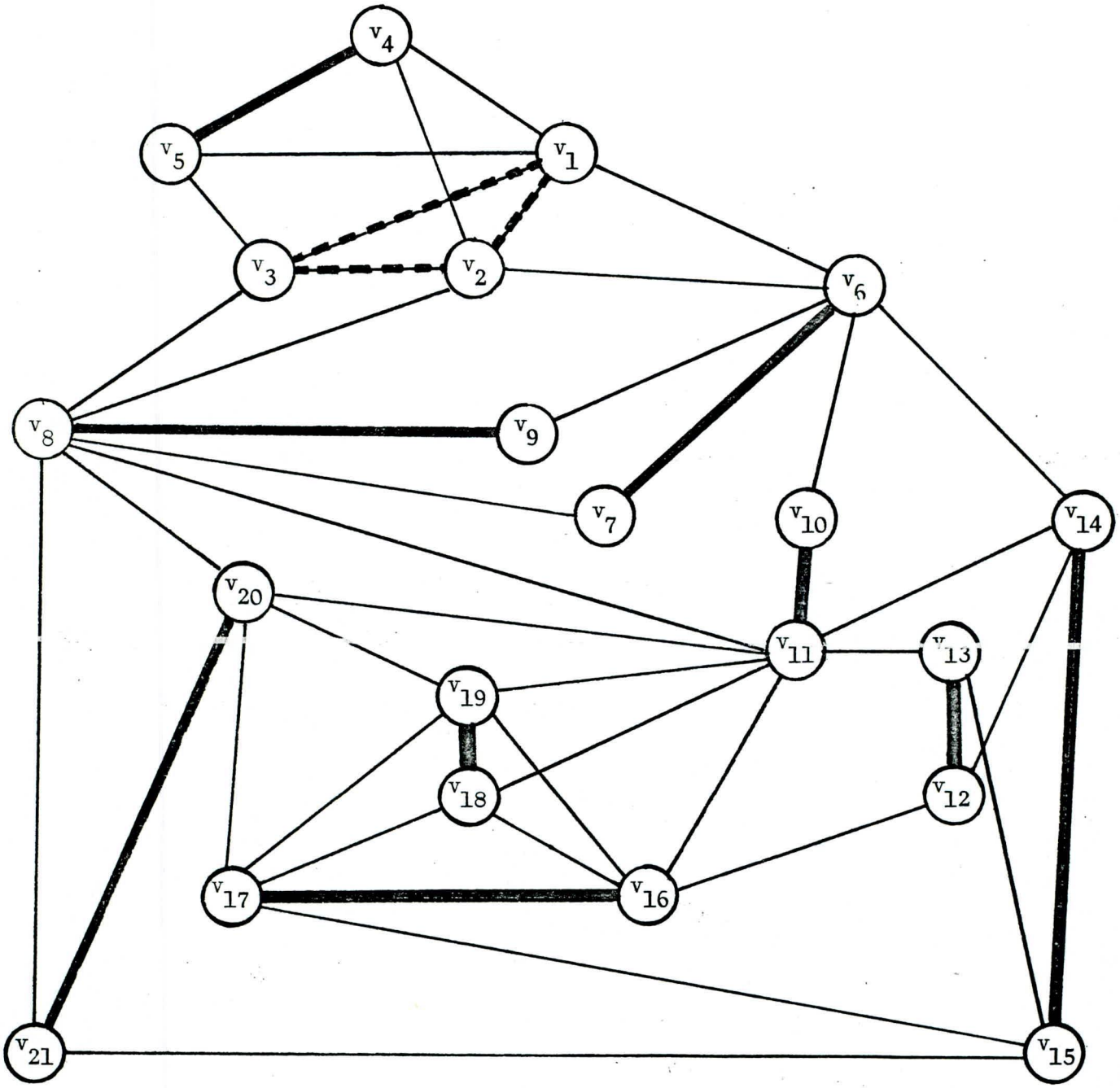
v4
v5
v1
v3
v2
v6
v8
v9
v7
v10
v14
v20
v11
v13
v19
v18
v12
v17
v16
v21
v15

Fig. 1.

(ii) there is an optimum (VLP) solution in which the variables assigned to integer values in step 3 of Routine 1., keep the same values.

PROOF:

(i) Since the Labelling Procedure builds an optimum (VLP) solution having the maximum set of integer-valued variables, we just have to prove that after its completion, $x_i = \frac{1}{2}$ for all the pseudo-tree or + and - labelled vertices v_i.

Suppose first that $x_i = 1$; this solution modification occured only if a labelling process, initiated from a vertex v_i (possibly $v_j = v_i$) failed to label v'_j after v_i was labelled. If v_i is a pseudo-tree vertex, this is inconsistent with corollary 4. If v_i was - labelled or"+ and -"labelled after completion of Routine 1, lemma 5 asserts the existence of an alternating path (v_i, v'_i). v_i may also be-labelled in the Labelling Process and lemma 2 asserts that v'_j may be labelled.

On the other hand, if $x_i = 0$ after completion of the Labelling Procedure, then a neighbour v_j of v_i was such that $x_j = 1$. But, in the run of Routine 1, v_j was labelled with - (since one of its neighbours v_i was + labelled) and lemma 5 may be applied again as above.

(ii) Consider a + labelled vertex v_i, after completion of Routine 1. There is no alternating path (v_i, v'_i) (otherwise v_i would be + and - labelled) so the labelling process initiated from v_i at step 2 of the Labelling Procedure, fails to label v'_i; hence x_i may be assigned the value 1 in an optimum (VLP) solution having

the maximum set of integer valued variables.

Consider a - labelled vertex v_j and the tree T in G_f, containing v_j. All the neighbours v_i of v_j in T (there is at least one) are + labelled and are not"+ and -"labelled. Hence there is a solution in which the corresponding variables $x_i = 1$ and, consequently, $x_j = 0$. #

In order to define a similar "alternating cycle reduction" using lemmas 6 to 8, we need to detect the alternating cycles. To this end, two successive labelling stages are required: the first to discover an alternating path (v_i, v'_i) and the second to discover (v'_i, v_i); after the first stage succeeded, we need to store the assigned labels before assigning new labels; in both stages, a vertex v_i may be in one these three states: unlabelled, - labelled and + labelled (see lemma 2).

ROUTINE 2 (ALTERNATING PATH DETECTION):

0. Choose a remaining vertex v_i and go to stage 1.
   If there are no remaining vertices, terminate.
1. First labelling stage:
   1.1 Label v_i with + ; label the tree containing v_i and to to step 1.2.
   1.2 If there is an edge connecting two + labelled vertices then go to second stage, else go to step 1.3.
   1.3 If there is an edge connecting a + labelled vertex to an unlabelled vertex v_j then go to step 1.4, else go to step 1.5.
   1.4 Label v_j with - ; label the tree containing v_j and go to step 1.2.
   1.5 Solution modification:
   set $x_j = 1$ for all + labelled vertices v_j
   $x_j = 0$ for all - labelled vertices v_j

and discard all these vertices and the edges adjacent to them;

to to step 0.

2. Second labelling stage:

2.1 Label v_i with - ; label the tree containing v_i and go to step 2.2.

2.2 If there is an edge connecting two + labelled vertices then go to Routine 3 (Alternating cycle reduction), else go to step 2.3.

2.3 If there is an edge connecting a + labelled vertex to an unlabelled vertex v_j then go to step 2.4, else go to step 2.5.

2.4 Label v_j with - ; label the tree containing v_j and go to step 2.2.

2.5 Solution modification:

set $x_j = 1$ for all + labelled vertices v_j

$x_j = 0$ for all - labelled vertices v_j

and discard all these vertices and the edges adjacent to them;

delete the stored labels and go to step 0.

Example 2 (continued):

The remaining set of vertices is $\{v_{10}, v_{11}, \ldots, v_{21}\}$; the application of Routine 2 is shown below:

stage	step	edge	vertex	label
0			v_{10}	
1	1.1		v_{10}	+
			v_{11}	-
	1.2			
	1.3			
	1.5	Solution modification:	$x_{10} = 1$	
			$x_{11} = 0$	
		v_{10} and v_{11} are discarded.		

stage	step	edge	vertex	label
0			v_{12}	
1	1.1		v_{12}	+
			v_{13}	-
	1.2			
	1.3	(v_{12}, v_{14})		
	1.4		v_{14}	-
			v_{15}	+
	1.2			
	1.3	(v_{12}, v_{16})		
	1.4		v_{16}	-
			v_{17}	+
	1.2	(v_{15}, v_{17})		
2	2.1		v_{12}	-
			v_{13}	+
	2.2			
	2.3	(v_{13}, v_{15})		
	2.4		v_{15}	-
			v_{14}	+
	2.2			
	2.3			
	2.5	Solution modification: $x_{13} = x_{14} = 1$		
				$x_{12} = x_{15} = 0$
		v_{12}, v_{13}, v_{14} and v_{15} are discarded.		
0			v_{16}	
1	1.1		v_{16}	+
			v_{17}	-
	1.2			
	1.3	(v_{16}, v_{18})		
	1.4		v_{18}	-
			v_{19}	+
	1.2	(v_{16}, v_{19})		
2	2.1		v_{16}	-
			v_{17}	+

stage	step	edge	vertex	label
	2.2			
	2.3	(v_{17},v_{18})		
	2.4		v_{18}	-
			v_{19}	+
	2.2	(v_{17},v_{19})		

Before defining Routine 3. (Alternating cycle reduction), we give the following justification of Routine 2.:

THEOREM 10 (Validation of Routine 2):

There is an optimum (VLP) solution in which the variables assigned to integer values in steps 1.5 or 2.5 of Routine 2., keep the same value.

PROOF: Occurence of step 1.5 means that the labelling process, initiated in v_i, fails to label v'_i. The above solution modification is the one done in the Labelling Procedure.

Occurence of step 2.5 means that the labelling process, initiated from any neighbour v_j of v_i in the tree containing v_i, fails to label v'_j; the solution modification is again the one done in the Labelling Procedure. #

ROUTINE 3 (ALTERNATING PATH REDUCTION):

0. Merge the stored and new labels (i.e. as indicated by Fig. 2).
1. If there is an edge connecting a + labelled or"+ and -"labelled vertex to an unlabelled or + labelled vertex v_j, then go to step 2, else go to step 3.
2. Label v_j with - ; label the tree containing v_j; go to step 1.

3. (No more labelling may be done)

Set $x_j = \frac{1}{2}$ for all the"+ and -"labelled vertices v_j.

Set $x_j = 1$ for all + labelled vertices v_j.

Set $x_j = 0$ for all - labelled vertices v_j.

go to Routine 2.

New labels \ Stored labels	none	+	-	+ and -
none	none	+	-	+ and -
+	+	+	+ and -	+ and -
-	-	+ and -	-	+ and -
+ and -	+ and -	+ and -	+ and -	+ and -

Fig. 2.: Label merging

Example 2 (end):

step	edge	vertex	label
0		v_{16}	+ and -
		v_{17}	+ and -
		v_{18}	-
		v_{19}	+
1	(v_{16}, v_{19})		
2		v_{19}	+ and -
		v_{18}	+ and -
1	(v_{17}, v_{20})		
2		v_{20}	-
		v_{21}	+
1			
3	Solution modification:	$x_{16} = x_{17} = x_{18} = x_{19} = \frac{1}{2}$	
		$x_{20} = 0$	
		$x_{21} = 1$	

THEOREM 11 (Validation of Routine 3)

(i) there is no optimum (VLP) solution in which the variables x_j, corresponding to a"+ and -"labelled vertex v_j after completion of Routine 3., may be integer valued.

(ii) there is an optimum solution in which the variables assigned to integer values in step 3 of Routine 3., keep the same values.

PROOF: We first prove that all the vertices in the alternating paths (v_i,v'_i) and (v'_i,v_i) are"+ and -"labelled.

Let v_j and v_k be the adjacent + labelled vertices detected in step 2 of the first stage; so v_j and v_k are also - labelled after edge (v_j,v_k) was considered twice in step 1 of Routine 3.

Along the alternating paths (v'_j,v'_i) and $(v'_kv'_i)$ (images of (v_i,v_j) and (v_i,v_k) opposite labels are assigned to all the vertices in the alternating path (v_i,v'_i).

The same is true for the second stage (just exchange v_i and v'_i). All the vertices in the alternating path (v'_i,v_i) are "+ and -" labelled.

The remainder of the proof is similar to the proof of theorem 9, using lemmas 6 to 8 in place of lemmas 3 to 5. #

The alternate labelling algorithm is defined as follows:

ALTERNATE LABELLING ALGORITHM:

Routine 1 (Pseudo-tree reduction)

Routine 2 (Alternating cycle detection)

Routine 3 (Alternating cycle reduction)

THEOREM 12 (Performances of the Alternate Labelling Algorithm)

Let k be the number of occurences of step 2.5 in the second stage of Routine 2.:

(i) $k \leq \frac{1}{2}n$ (n is the number of vertices in G)

(ii) the number of labels assigned to any vertex is less than or equal to $k+2$

(iii) if the unique optimum (VLP) solution in G is $x_i = \frac{1}{2}$ for all $i = 1,\ldots,n$ then the number of assigned labels is exactly 2 for each vertex.

PROOF:

(i) is obvious since each vertex has at least one neighbour in its tree. Hence, at each occurence of step 2.5 at least 2 vertices are integer valued and discarded.

(ii) if no occurence of step 2.5 appears in the completion of the Alternate Labelling Algorithm, exactly 2 labels are assigned to v_j for having $x_j = 1$, and 1 for having x_j integer-valued in Routine 1 or 3.

The only unused labels are those assigned in the first stage of Routine 2, when Solution Modification Step 2.5 has to occur (see, for instance, the labelling process initiated from v_{12} in Example 2, where unused labels are assigned to $v_{14}, v_{15}, v_{16}, v_{17}$).

(iii) follows from the above arguments. #

After completion of the Alternate Labelling Algorithm, the set of $\frac{1}{2}$-valued vertices defines a subgraph of G called "the remaining graph". Detecting the connected components of the remaining graph is interesting,

for instance in order to simplify the solution of (V.P.) (see |6|). After completion of Routine 1, the $\frac{1}{2}$-valued vertices are obviously disconnected from the unlabelled vertices, but they may define several connected components. An obvious modification of Routine 1 avoids this (put, at step 0, one pseudo-tree in S, and incorporate pseudo-trees as soon as one of their vertices is labelled). This problem does not arise with Routine 3; each occurence of step 3 in which vertices are $\frac{1}{2}$-valued defines a connected component of the remaining graph.

It may also be interesting to restore an optimal basis in the remaining graph. The basis structure may be destroyed in a connected component of the old basis in which some vertices, but not all, were discarded. If this discarding occured in Routine 1, the odd cycle was preserved; if it is due to an alternating cycle, keep one odd cycle detected either in stage 1 or 2 of Routine 2; for the other vertices (not in an odd cycle), store the edges which permitted their first labelling. Thus all the edges with flow are kept and no unnecessary cycle is introduced. Moreover a slight modification in the alternate labelling routines insures having the least change in the basis, that is in labelling steps, we use the basic (zero-valued) edges first. We have not included these topics in order to simplify the description of the algorithm.

REFERENCES

|1| BALINSKI M.L. (1970) "On Maximum Matching, Minimum Covering and their Connections" in Proceedings of the Princeton Symposium on Mathematical Programming, H. Kuhn ed., Princeton University Press.

|2| FORD L.R. and FULKERSON D.R. (1962) Flows in Networks, Princeton University Press.

|3| JOHNSON E.L. (1965) "Programming in Networks and Graphs" ORC 65.1, University of California, Berkeley.

|4| JOHNSON E.L. (1966) "Networks and Basic Solutions" Operations Research 14, 619-623.

|5| LORENTZEN L.C. (1966) "Notes on Covering of Arcs by Nodes in an Undirected Network" ORC 66.1, University of California, Berkeley.

|6| NEMHAUSER G.L. and TROTTER L.E. (1974) "Vertex Packings: Structural Properties and Algorithms" Mathematical Programming, Vol 8, No 2 (April 1975) 232-248.

|7| PICARD J.C. and QUEYRANNE M. (1975) "On the Integer-Valued Variables in the Linear Vertex Packing Problem" Tech. Report EP75-R-35 Ecole Polytechnique - Université de Montréal.

|8| SRINIVASAN V. and THOMPSON G.L. (1972) "Accelerated Algorithms for Labelling and Relabelling of Trees, with Applications to Distribution Problems" Journal of A.C.M. 19, 712-726.

ÉCOLE POLYTECHNIQUE DE MONTRÉAL
3 9334 00288777 4