An Approximation Problem for the Multi-Via Assignment Problem[1]

TEOFILO F. GONZALEZ

Abstract—We consider the multi-via assignment problem for multilayered printed circuit board routing. An efficient approximation algorithm for this problem is presented. The algorithm is of (low) polynomial time complexity and guarantees solutions with no more than 3 * OPT via columns, where OPT is the number of via columns in an optimal solution. Several issues relating to the computational complexity of via and multi-via assignment problems are also discussed.

I. Introduction

LARGE-SCALE computer systems are built by interconnecting silicon chips. The interconnections are carried out by placing the chips on a multilayer printed circuit board (MPCB). The components (chips) are mounted on top of a board and their terminals are inserted in drilled-through holes called pins. Terminals from different chips are connected by printed wires located on any of the layers in the MPCB. Each set of pins that need to be made electrically common is called a *net*. In this paper we make the same basic assumptions made by So [8]. These are as follows.

(1) The pins and vias are at fixed locations. Vias appear only column-wise.

(2) Only points (pins or vias) on the same line (row or column) can be connected directly and the physical routes must be confined within the channels on both sides of the line.

(3) There are two layers in the MPCB. All row connections are on one layer and all column connections are on the other.

The *via assignment problem* (VAS) consists of adding the least number of via columns and specifying the points (pins or vias) that need to be connected directly in such a way that no two pins from the same net are connected to vias from different via columns and a routing satisfying (2) and (3) exists. Fig. 1(a) illustrates a set of via assignments and a final routing for an instance of the VAS problem. After solving the VAS finding physical routes for the printed wires (routing) reduces to solving several single-line single-layer routing problems [8]. The single-line single-layer problem can be solved by the algorithms given in [6] and [7].

The computational complexity of the VAS problem has been studied in [9] and [10]. In [9] it was shown that the VAS problem can be solved efficiently when only one via column is needed for all the interconnections. However, as pointed out in [9] and [10], when three or more via columns are required the problem becomes an NP-hard problem. This result relies on the restriction that no net is connected to vias from more than one via column. If this restriction is relaxed, the problem is called the *multi-via assignment* (*MVAS*) problem. In Fig. 1(b) we represent a set of via assignments and a final routing for an instance of the MVAS problem. As noted in [9], there are instances for which an optimal solution to the MVAS problem requires less via columns than an optimal solution to the corresponding VAS problem (one of such instances is given in Fig. 1). The converse is not true. In [9] it was conjectured that the MVAS problem is an NP-hard problem. In this paper, it is shown that the MVAS problem is NP-hard. Furthermore, it is shown that the VAS and MVAS problems are NP-hard even when only two via columns are required for all the interconnections. Our results finely separate "difficult" from "easy" cases of these problems.

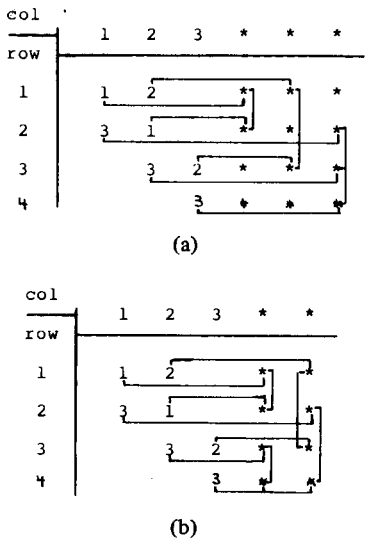


Fig. 1. (a) VAS problem. (b) MVAS problem.

A solution using n via columns, where n is the number of nets, can be obtained very quickly by just assigning one via column to each net. This solution is not desirable because it implies a huge MPCB as well as a design that is expensive to produce. On the other hand, finding a solution with the least number of via columns seems to be an extremely difficult problem. Because of these difficulties we will try to reach a compromise by turning our attention to the study of algorithms that generate near-optimal solutions quickly, i.e., every solution generated by these algorithms has a number of via

Manuscript received December 2, 1983; revised March 5, 1984.

The author was with Programs in Computer Science, The University of Texas at Dallas, Richardson, TX 75080. He is now with the Department of Computer Science, The University of California, Santa Barbara, CA 93106.

[1]A preliminary version of this paper was presented at the *1983 International Conference on Computer Aided Design.*

0278-0070/84/1000-0257$01.00 © 1984 IEEE

columns that is not far from optimal. In [9] and [10] several heuristics to solve these problems are presented. However, there is no guarantee that the objective function value of a solution generated by these heuristics is within a constant bound of the optimal one. An algorithm that generates solutions with an objective function value within a constant bound of the objective function value of an optimal solution is called an *approximation algorithm*. In what follows we discuss why it is unlikely that there exists an efficient approximation algorithm for the VAS problem. The VAS problem has the property that obtaining a solution with an objective function value within 100 percent of the optimal solution value is an NP-hard problem. This can be easily shown by using the reduction outlined in [9] and the results in [2] relating to the complexity of generating approximate solutions to the graph coloration problem. The reduction outlined in [9] can also be used to show that any approximation algorithm for the VAS problem is also an approximation algorithm for the graph coloration problem. The converse does not seem to be true. The graph coloration approximation problem has been studied in great detail [2] and it seems to be computationally intractable. Therefore, it is unreasonable to try to obtain an efficient approximation algorithm for the VAS problem before one can be found for the graph coloration problem. Because of this we turn our attention to the MVAS problem. Our main result is an efficient algorithm that guarantees solutions close to optimal for the MVAS problem.

Our contribution, on the theoretical aspects of the VAS and MVAS problem, is to settle some issues relating to their computional complexity. On the practical side, for the MVAS problem we present an efficient algorithm that generates solutions with an objective function value that in the worst case is not far from the objective function value of an optimal solution.

In the next paragraph we define the via assignment problem and in Section II we show that some versions of this problem are NP-complete. An approximation algorithm for the MVAS problem is presented in Section III and in Section IV it is shown that our algorithm can be implemented to run in polynomial time by obtaining efficient algorithms to solve a couple of graph problems. These algorithms are used as internal procedures by our algorithm.

Let P be a multilayered printed circuit board. Board P consists of r rows and c columns for the pins. At the intersection of a row with a column there is a pin. Each pin belongs to at most one net. The nets are represented by $N_1, N_2, \cdots, N_n$ and each N_i consists of a set of pins that need to be made electrically common. Vias can be added for interlayer connections, however vias appear only column-wise. Wire segments can only be used to connect points (pins or vias) located on the same line (row or column). It is assumed that all the wire segments connecting points on the same row are on one layer and the ones connecting points on the same column are on the other layer. Two wires on different layers are said to be connected directly if both of them are connected to the same point (pin or via). The VAS problem consists of adding the least number of via columns in such a way that all pins in each net can be made electrically common by adding a set of wires that satisfy the above requirements and no two pins from the same net are connected directly to vias from different via columns. The MVAS problem is defined similarly, except that the restriction imposed on the location of the vias used for the interconnection of all the pins in each net is relaxed, i.e., it is now possible to use vias from two or more via columns for the interconnection of all the pins in a net. Since every feasible solution for the VAS problem is also a feasible solution for the MVAS problem, the MVAS problem has an optimal solution value that is never worse than the optimal solution value for the corresponding VAS problem. It is simple to show that the converse is not true [9] (another example of this is given in Fig. 1). An instance of the VAS problem is shown in Example 1.1 (Fig. 2) and one of its solutions is depicted in Fig. 4.

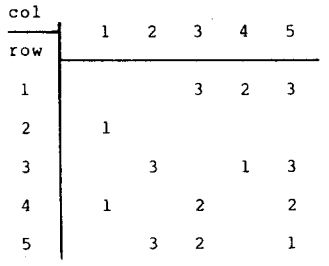


Fig. 2. The VAS or MVAS problem given in Example 1.1.

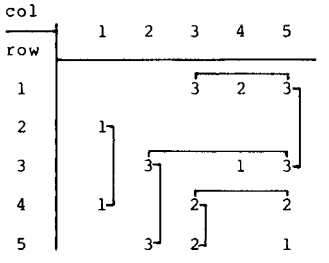


Fig. 3. All the possible net connections for Example 1.1 without the use of via columns.

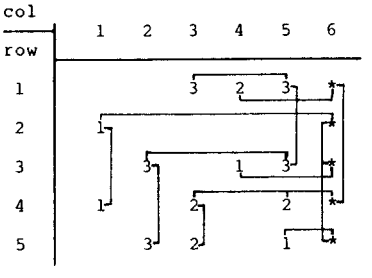


Fig. 4. A solution to the VAS problem given in Example 1.1 (column 6 is a via column).

Example 1.1: An instance of the VAS or MVAS problem.

$r = 5, c = 5, n = 3$

$N_1 = \{(2, 1), (3, 4), (4, 1), (5, 5)\}$

$N_2 = \{(1, 4), (4, 3), (4, 5), (5, 3)\}$

and

$N_3 = \{(1, 3), (1, 5), (3, 2), (3, 5), (5, 2)\}$.

Our NP-completeness results are established by reducing the exact cover by 3-sets (XC3) problem to the VAS and MVAS problems. The XC3 problem was shown to be NP-complete by Karp [5] and it is defined as follows:

Exact Cover by 3-Sets (XC3):

Input: A finite set $X = \{x_1, x_2, \cdots, x_{3q}\}$ and a collection $C = \{(x_{i_{l,1}}, x_{i_{l,2}}, x_{i_{l,3}}) | 1 \leqslant l \leqslant m\}$ of 3-element subsets of X. In each triplet all the x_i's are different.

Question: Is there a subcollection $C' \subseteq C$ such that every element in X occurs in exactly one member of C'?

II. NP-Completeness Results

In this section it is shown that the VAS and MVAS problems are NP-complete even when only two via columns are needed for all the interconnections. The first result is obtained by reducing the XC3 problem to the VAS problem. The NP-completeness of the MVAS problem is established by using the reduction for the VAS problem and observing that the instance constructed has no solution with two via columns in which a net is connected to vias from more than one via column. The problem of deciding whether an instance of the VAS problem can be routed by using at most two via columns is referred to as the *2-VAS* problem. The *2-MVAS* problem is defined similarly.

Theorem 2.1: The 2-VAS problem is NP-complete.

Proof: It is simple to see that the 2-VAS problem is in NP. We now show that the XC3 problem reduces to it. Let (X, C) be any instance of the XC3 problem. Let l_i be the number of occurrences of element x_i in C. We may assume without loss of generality that all the l_i's are greater than one as otherwise an "equivalent" instance for it with all the l_i's > 1 can be easily obtained. The new instance can be obtained in polynomial time and the size of the new problem is polynomially related to the size of the original instance. Let

$$L = \sum_{i=1}^{3q} l_i.$$

From any instance of the XC3 problem we construct the following 2-VAS problem (INS). The problem INS consists of $n = m + 2$ nets and the board is of size $r = L$ rows by $c = 3 * m + L$ columns for the pins. The rows are labeled (i, j) for $1 \leqslant i \leqslant 3 * q$ and $1 \leqslant j \leqslant l_i$ and the columns are labeled with the integer k for $1 \leqslant k \leqslant 3 * m + L$. By $((i, j), k)$ we denote the pin located at the intersection of row (i, j) with column k. The nets are defined as follows:

i) Net N_i, $1 \leqslant i \leqslant m$, respresents the ith 3-element subset in C. Let $(x_r, x_s, x_t,)$ be this triplet from C. Net N_i has a pin on each of the following positions:

$$((r, j), 3 * (i - 1) + 1), \quad \text{for} \quad 1 \leqslant j \leqslant l_r$$

$$((s, j), 3 * (i - 1) + 2), \quad \text{for} \quad 1 \leqslant j \leqslant l_s$$

and

$$((t, j), 3 * (i - 1) + 3), \quad \text{for} \quad 1 \leqslant j \leqslant l_t.$$

ii) Net N_{m+1} is an enforcer that consists of pins located on each of the following points:

$$((i, l_i), 3 * m + i) \quad \text{for} \quad 1 \leqslant i \leqslant 3 * q.$$

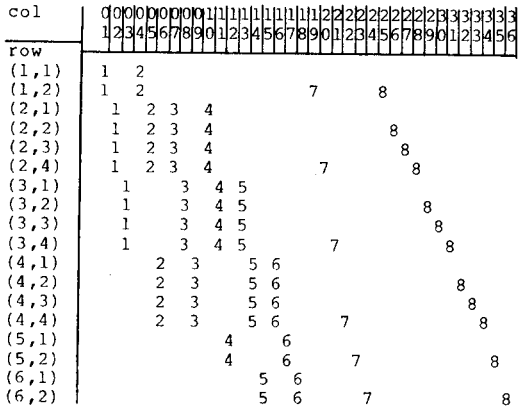


Fig. 5. INS for XC3 in Example 2.1.

iii) Net N_{m+2} is also an enforcer with a pin located on each of the following positions:

$$((i, j + 1), 3 * m + 3 * q + \sum_{z=1}^{i-1} (l_z - 1) + j),$$

$$\text{for} \quad 1 \leqslant i \leqslant 3 * q \quad \text{and} \quad 1 \leqslant j \leqslant l_i - 1.$$

Example 2.1: Given the instance (X, C) for the XC3 problem, where

$$X = \{x_1, x_2, \cdots, x_6\} \quad \text{and}$$

$$C = \{(x_1, x_2, x_3), (x_1, x_2, x_4), (x_2, x_3, x_4), (x_2, x_3, x_5), (x_3, x_4, x_6), (x_4, x_5, x_6)\}$$

the above procedure constructs the following instance of the 2-VAS problem (INS) see Fig. 5.

Fig. 6 shows a solution for INS with two via columns.

Clearly, INS can be constructed in polynomial time. In order to complete the proof of the theorem it is only required to show that the instance INS can be routed using only two via columns if and only if XC3 has an exact cover. This is shown in two parts:

a) if XC3 has an exact cover then INS can be routed using two via columns.

Proof (for part a): Let C' be any exact cover for XC3. Clearly, $|C'| = q$. Let $i_1, i_2, \cdots, i_q$ be the set of indices of the nets that represent the 3-element subsets in C'. Each of these nets and net N_{m+2} are connected to one of the via columns. The rows used by these nets on via column number one are as follows:

i) Net N_{m+2} uses rows $(i, j + 1)$, $1 \leqslant i \leqslant 3 * q$ and $1 \leqslant j \leqslant l_i - 1$.

ii) Let net N_{i_j} represent $(x_r, x_s, x_t) \in C$. Net N_{i_j} uses rows $(r, 1), (s, 1)$ and $(t, 1)$.

It is simple to show that each row on this via column will be used by exactly one net.

The second via column will be used for the connection of the remaining nets. The rows on the second via column used by these nets are as follows:

i) Net N_{m+1} uses rows (i, l_i) for $1 \leqslant i \leqslant 3 * q$.

ii) Let $j_1, j_2, \cdots, j_{m-q}$ be the indexes of the nets that represents 3-element subsets that are in C but not in C'. Let

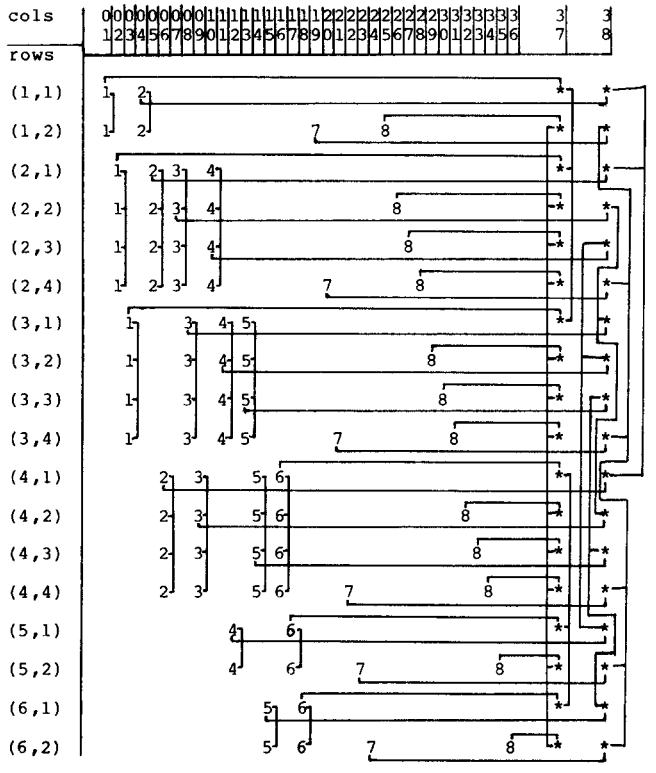


Fig. 6. Solution for the problem depicted in Fig. 5 (columns 37 and 38 are via columns).

N_{j_i} represent $(x_r, x_s, x_t) \in C$. For net N_{j_i}, let k_i, k_i', and k_i'' be the number of times variables x_r, x_s, and x_t appear in the 3-element subsets of X represented by nets $N_{j_1}, N_{j_2}, \cdots, N_{j_i}$. Net N_{j_i} is routed using rows (r, k_i), (s, k_i') and (t, k_i'').

It is simple to show that each row on via column two will be used by exactly one net and that each of the nets can be interconnected by using the vias specified above. This completes the proof of part *a*.

b) If INS can be routed using two via columns then XC3 has an exact cover.

Proof (for part b): Let R be a routing for INS. Clearly, nets N_{m+1} and N_{m+2} need at least one via for their connection and both of these nets cannot use vias from the same via column. Let $N_{i_1}, N_{i_2}, \cdots, N_{i_l}$ be the nets connected through the columns used by net N_{m+2}. Clearly, the 3-element subsets of X represented by nets $N_{i_1}, N_{i_2}, \cdots, N_{i_l}$ form a subcover for X. In order to show that this is an exact cover for X, it is only required to show that $l = q$. A proof for this can be obtained by using the following observations: in any feasible solution all rows on both via columns must be used, net N_{m+2} leaves $3 * q$ unused vias on the via column it uses for its connection and each of the nets $N_1, \cdots, N_m$ use exactly three rows for their connection on one of the via columns. This completes the proof of part *b* and Theorem 2.1. ■

Theorem 2.2: The 2-MVAS problem is NP-complete.

Proof: The proof of this theorem is based on the reduction used in the proof of Theorem 2.1. The proof of the theorem follows the same arguments as the ones used in the previous proof after showing that the instance INS does not have a feasible solution in which a net is connected to vias from the two via columns. ■

III. Approximation Algorithm

In this section we present an efficient approximation algorithm for the MVAS problem. The input to the algorithm is an instance, Y, of the MVAS problem. The solution that it generates has no more than $3 * \text{OPT}$ via columns, where OPT is the number of via columns in an optimal solution. The algorithm consists of three major steps.

In the first two steps an instance of the VAS problem, X, in which all nets have exactly two pins is constructed. Problem X has the property that the maximum number of nets sharing the same row in the MPCB is not greater than $2 * \alpha$, where α is a lower bound on the number of via columns in an optimal solution to Y. Furthermore, a solution to Y can be easily obtained from the solution to X. This solution has the same number of via columns as the one in the solution to X. The specific computations performed by these steps are the construction of a bipartite graph and the identification of a "special" subset of edges in it. The "special" subset of edges and a decomposition of the nets in Y are used to define the nets for X.

The third step of the algorithm consists of finding an approximate solution to X. An approximate solution to this problem is required since it is an NP-hard problem. The specific computations performed by this step are to construct a multigraph from X, then the edges of this multigraph are "colored" and from any such coloration we obtain a solution to X. The solution to X that we construct has no more than $(\frac{3}{2}) * \beta$ via columns, where β is the maximum number of nets sharing the same row in X.

Before presenting our algorithm we outline in more detail the three major steps in it. Additional definitions as well as some intermediate computations have been introduced to clarify our procedure.

Step 1: The set of pins in each net N_i is partitioned into the sets $N_{i,1}, N_{i,2}, \cdots, N_{i,l_i}$ in such a way that all the pins in each set $N_{i,j}$ can be made electrically common without the use of vias and the pins in each set $N_{i,j}$ cannot be made electrically common with a pin in $N_{i,k}$ $(k \neq j)$ without the use of vias. We shall refer to the sets $N_{i,j}$ as *subnets*. For $N_{i,j}$ we define $R_{i,j}$ as the set of rows on which the pins in $N_{i,j}$ lie. The sets $N_{i,j}$ and $R_{i,j}$ for the MVAS problem given in Example 1.1 are as follows:

$$N_{1,1} = \{(2,1),(4,1)\}, \quad R_{1,1} = \{2,4\},$$
$$N_{1,2} = \{(3,4)\}, \quad R_{1,2} = \{3\},$$
$$N_{1,3} = \{(5,5)\}, \quad R_{1,3} = \{5\},$$
$$N_{2,1} = \{(1,4)\}, \quad R_{2,1} = \{1\},$$
$$N_{2,2} = \{(4,3),(4,5),(5,3)\}, \quad R_{2,2} = \{4,5\},$$
$$N_{3,1} = N_3 \quad \text{and} \quad R_{3,1} = \{1,3,5\}.$$

Every net with $l_i = 1$ can be eliminated since it needs no via column for its connection. If all nets are eliminated then our solution is an optimal solution since it includes no via columns. In Example 1.1, net N_3 is the only net eliminated. Hereafter, n represents the number of nets not yet been eliminated.

In this step we select an element from each $R_{i,j}$. Any pin

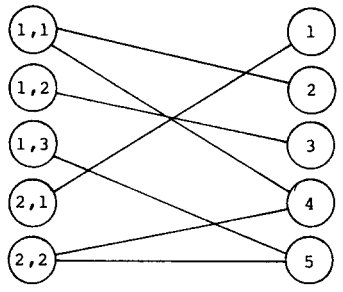


Fig. 7. Bipartite graph constructed by Step I for Example 1.1.

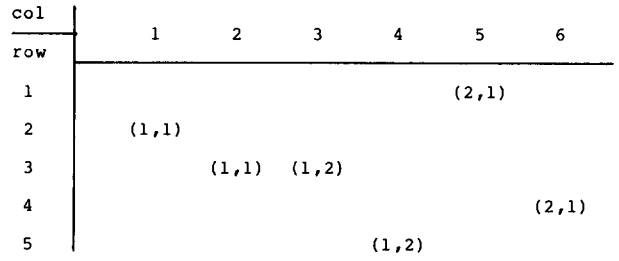


Fig. 8. Instance X constructed by Step II for Example 1.1.

from subnet $N_{i,j}$ located on the row selected from $R_{i,j}$ will be used for the connection of $N_{i,j}$ to all other subnets of net N_i. The selection of these elements is performed in such a way that the maximum number of subnets sharing the same row for their connection, is minimized. We shall refer to this minimum value as α. The value for α can be shown to be a lower bound on the number of via columns in an optimal solution to the original MVAS problem.

The selection of the row from each $R_{i,j}$ is made by finding an optimal complete s-matching in a bipartite graph. This is explained in what follows. Let $S = \{(i, j) | N_{i,j}$ is a subnet$\}$, $T = \{1, 2, \cdots, r\}$, and $E = \{((i, j), k) | k \in R_{i,j}\}$. Clearly $G = (S \cup T, E)$ is a bipartite graph. The bipartite graph obtained for the MVAS problem given in Example 1.1 is depicted in Fig. 7.

An *s-matching*, I, for G is a subset of edges such that no two edges in it are incident to the same node in S. A *complete s-matching* is an s-matching with cardinality equal to $|S|$. For a complete s-matching, I, we define $M(I)$ as the maximum number of edges in I incident to any node in set T. We say that the complete s-matching, I, is an *optimal complete s-matching (ocs-matching) for* G if every complete s-matching Z for G has $M(Z) \geq M(I)$. The set of edges $I = \{((1, 1), 2), ((1, 2), 3), ((1, 3), 5), ((2, 1), 1), ((2, 2), 4)\}$ is an ocs-matching for the bipartite graph depicted in Fig. 7. The problem of selecting a row for the connection of each subnet reduces to the problem of finding an ocs-matching for G. In the next section we present an efficient algorithm to construct an ocs-matching for G. Given an ocs-matching, I, for G we find $r_{i,j}$, the row to be used for the connection of subnet $N_{i,j}$ to all other subnets of net N_i, as follows: $r_{i,j} = k$ if edge $((i, j), k)$ is in the ocs-matching I. Clearly, from the definition of an ocs-matching we can see that each subnet has one and only one row selected and in such a row there is at least one pin from this subnet. Also, if "I" is an ocs-matching for G then $\alpha = M(I)$. The optimality of the complete s-matching guarantees that there is no feasible solution to Y with less than α via columns, where α is defined above. The $r_{i,j}$ values for the ocs-matching obtained above are: $r_{1,1} = 2, r_{1,2} = 3, r_{1,3} = 5, r_{2,1} = 1$, and $r_{2,2} = 4$.

The computations performed by this step can be described as follows:

1.1) Construct $G = (S \cup T, E)$ from G;

1.2) Obtain an optimal complete s-matching, I, for G;

1.3) Define $r_{i,j}$ from I. ■

Step II: At this point we construct an instance, X, of the VAS problem, such that from a solution to X it is simple to construct a solution to Y with the same number of via columns as the one in the solution to X. Problem X has the property that the number of pins that need to be connected on each row is at most $2 * \alpha$.

Let

$$L = \sum_{i=1}^{n} l_i.$$

The instance X consists of $n' = L - n$ nets and the board has $r' = r$ rows and $c' = 2n'$ columns for the pins. We shall refer to the n' nets as $N'_{i,1}, \cdots, N'_{i,l_i-1}$ for $1 \leq i \leq n$. Net $N'_{i,j}$ has two pins, one is located at position $(r_{i,j}, k)$ and the other pin at position $(r_{i,j+1}, k + 1)$, where $k = 2 * (j - 1 + \Sigma_{z=1}^{i-1} (l_z - 1)) + 1$. Fig. 8 shows the instance X constructed for the problem given in Example 1.1.

$$r = 5, c = 6, n = 3$$
$$N'_{1,1} = \{(2, 1), (3, 2)\}$$
$$N'_{1,2} = \{(3, 3), (5, 4)\}$$

and

$$N'_{2,1} = \{(1, 5), (4, 6)\}.$$

The proof for the claim that from any solution to X one can construct a solution to Y, with the same number of via columns as the one in the solution to X, is straightforward.

In this step, the initial computation is:

2.1) Construct X from $r_{i,j}$.

Once the solution to X (S_X) has been obtained (Step III) we perform the following computation:

2.2) Obtain the solution S_Y for Y from solution S_X. ■

Step III: In this step we find an approximate solution to X, the restricted VAS problem obtained in Step II. This solution is obtained from a coloration of the edges in a multigraph. The multigraph that we use is $M = (V', E')$, where V' is the set of nodes and E' is a multiset of edges, is defined as follows:

$$V' = \{i | i \text{ is a row in } X\}$$

and

$$E' = \{\{i, j\} | \text{ a net in } X \text{ has a pin on rows } i \text{ and } j\}.$$

The multigraph M constructed from X given in Fig. 8 is shown in Fig. 9.

An edge coloration for multigraph M consists of assigning one color to each edge in E in such a way that no two edges incident to the same node are assigned the same color. An

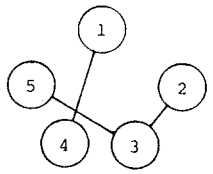


Fig. 9. Multigraph constructed by Step III for Example 1.1.

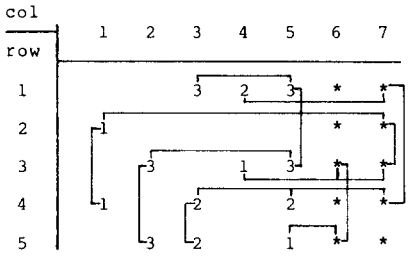


Fig. 10. Solution constructed by our algorithm for Example 1.1 (columns 6 and 7 are via columns).

edge coloration for the multigraph depicted in Fig. 9 is: edge $\{3, 5\}$ is colored "one" and all the remaining edges are colored "two." A solution to X can be obtained from a coloration for M by simply connecting a net through the ith via column if the edge in M that represents it is assigned color "i." Clearly, the number of via columns in our solution for X is the same as the number of different colors in the coloration for M. It is simple to show that the number of colors in any edge coloration for M is at least β, where β is the maximum number of nets sharing the same row in the board for X. In the next section we present an algorithm that finds an edge coloration for M with no more than $(\frac{3}{2}) * \beta$ different colors. Hence, in our solution to the original MVAS problem there are at most $3 * \alpha$ via columns. The solution obtained for the MVAS problem given in Example 1.1 is shown in Fig. 10. In this case the solution obtained can be easily transformed into an optimal solution. We cannot claim that this is always possible because there are nonoptimal solutions for which any efficient "transformation" process will not decrease the number of via columns.

The computations performed in this step are:

3.1) Use X to construct $M = (V', E')$;

3.2) Find an edge coloration, C, for M;

3.3) Use C to construct S_X for X.

Our algorithm is illustrated in Fig. 11. Additional definitions as well as intermediate steps have been introduced to clarify the exposition of the algorithm.

Theorem 3.1: Algorithm APPROX constructs a solution to Y by introducing at most $3 * \text{OPT}$ via columns, where OPT is the number of via columns in an optimal solution to Y.

Proof: By the above discussion. ■

In the next section we show that an ocs-matching for a bipartite graph $G = (S \cup T, E)$ can be obtained in $O(es^{3/2} t^{1/2} \log s)$ time, where $s = |S|$, $t = |T|$, and $e = |E|$. In Section IV it is also shown that a coloration with no more than $(\frac{3}{2}) * d$ different colors for a multigraph $M = (V', E')$ can be obtained in $O(m^2 n)$ time, where d is the maximum degree of any node in M, $n = |V'|$ and $m = |E'|$. In order to simplify the notation in the next section, the above bounds are in terms of the number of elements (nodes and edges) in the graphs and not in terms of the number of rows and nets in the board. Using the above results we compute the overall time complexity of the algorithm.

```
Algorithm APPROX ( Y //an instance of the MVAS
                     problem// )

1.1   Use Y to construct
            G = ( S (_) T , E );

1.2   Obtain an optimal complete s-matching, I,
        for G;

1.3   Define r_{i,j} from I;

2.1     Construct X from r_{i,j};

3.1       Use X to construct M = ( V' , E' );

3.2       Find a coloration, C, for M;

3.3       Use C to construct solution S_X for X;

2.2     Use S_X to obtain the solution S_Y for Y;

      Return ( S_Y );

    End of Algorithm;
```

Fig. 11. Algorithm APPROX.

Theorem 3.2: Algorithm APPROX can be implemented to execute in no more than $O(r^{1/2} p^{5/2} \log p)$ time, where $p = \Sigma |N_i|$ and r is the number of rows in the board.

Proof: First let us find a bound for the maximum size of G. Set S can have at most $O(p)$ elements since each subnet must have at least one pin and no pin is in more than one subnet. In T there is one node per row in the board, hence T has $O(r)$ nodes. The maximum number of edges in E is $O(p)$.

Using these bounds and Theorem 4.2 we get that lines 1, 2, and 3 take $O(p)$, $O(r^{1/2} p^{5/2} \log p)$ and $O(p)$ time, respectively.

Line 4 takes no more than $O(p)$ time since X contains no more than $p/2$ nets. The multigraph M contains r vertices (one per row in the original board) and the number of edges is $O(p)$ (one per net in X). Using these bounds and Theorem 4.3 we have that line 5 takes $O(p)$ time and line 6 takes $O(p^2 r)$ time. The remaining lines can be easily shown to require $O(p)$ time.

Now if $r > p$ then some rows have no pins that need to be connected. These rows can be eliminated from the beginning. Hence, one can assume that $r \leq p$ and the upper bound on the time complexity for the algorithm can be easily obtained by combining the individual upper bounds on the time required by each of the steps. This completes the proof of the theorem. ■

IV. OCS-Matchings and Colorations

In Section IV-A we present an algorithm to construct an ocs-matching for a bipartite graph and in Section IV-B we present an algorithm that finds an edge coloration for a multigraph. In order to simplify our notation we derive all of our bounds in terms of the number of elements (nodes and edges) in the graph rather than using the number of rows and nets in the board.

A. OCS-Matchings

We present an algorithm to construct an ocs-matching for G, where $G = (S \cup T, E)$ is a bipartite graph and an ocs-match-

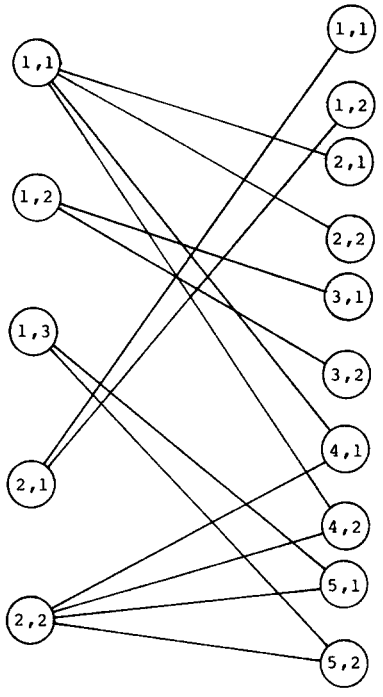


Fig. 12. H graph obtained from G (Fig. 7) with $l = 2$.

ing is defined in the previous section. Let $S = \{x_1, x_2, \cdots, x_s\}$, $T = \{y_1, y_2, \cdots, y_t\}$ and $e = |E|$. Note that $s = |S|$ and $t = |T|$. For a complete s-matching Z, let $M(Z)$ be the maximum degree of any node in $G' = (S \cup T, Z)$. It is simple to show that for any ocs-matching I, $1 \leqslant M(I) \leqslant |S| = s$. Our procedure performs a binary search on the set of integers $\{1 \cdots s\}$ to find the least value, l, such that G has a complete s-matching, I, with $M(I) \leqslant l$. Clearly, at most $O(\log s)$ of these tests are required. The problem of testing whether G has a complete s-matching, I, with $M(I) \leqslant l$ can be reduced to the problem of testing whether a bipartite graph $H = (S \cup T', E')$ has a complete matching for S, where $T' = \{y_{i,j} \mid y_i \in T \text{ and } 1 \leqslant j \leqslant l\}$ and $E' = \{\{x_k, y_{i,j}\} \mid x_k \in S, y_{i,j} \in T' \text{ and } \{x_k, y_i\} \in E\}$. See [1] for the definition of a complete matching for S in H. Fig. 12 shows the bipartite graph H obtained from the bipartite graph G given in Fig. 7 with $l = 2$.

Theorem 4.1: Let G, H, and l be as defined above. G has a complete s-matching, I, with $M(I) \leqslant l$ if H has a complete matching for S.

Proof: Since the proof is straightforward it will be omitted. ■

Theorem 4.2: An ocs-matching for G can be obtained by the above procedure in $O(es^{3/2} t^{1/2} \log s)$ time.

Proof: Testing whether a bipartite graph has a complete matching for S (the set of nodes is $S \cup T$) can be accomplished by constructing a maximum matching and then checking if its cardinality is equal to the cardinality of set S. A maximum matching for a bipartite graph can be obtained in $O(en^{1/2})$, where n is the number of nodes and e is the number of edges (see [4]). When we perform this test for some value of l, H has $|S| = s$, $|T'| = tl$ and $|E'| = el$. Hence, such a test can be performed in $O(el(s + tl)^{1/2})$. Since l is $O(s)$, this bound becomes $O(es^{3/2} t^{1/2})$. As established above, the maximum number of such tests is $O(\log s)$. Multiplying these last two bounds gives us the desired result. This completes the proof of the theorem. ■

In a subsequent paper we will present a faster algorithm to construct ocs-matchings.

B. Coloring the Edges in a Multigraph

Any multigraph M with maximum node degree, d, can be labeled with no more than $(\frac{3}{2}) * d$ different colors. The proof of this fact is based on a generalization of Vizing's theorem [10] (see [1]). It is simple to design an algorithm that constructs such a coloration. The algorithm colors one edge at a time. When considering an edge for coloration, the algorithm first checks if it is possible to color it without recoloring any previously colored edge. This operation takes $O(d)$ time. If some edges need to be recolored, this is accomplished by following a procedure similar to the one used in the proof of Vizing's theorem outlined in [1]. In the worst case, the time complexity for this recoloration is $O(dn)$. Since d is $O(m)$, then the time complexity is $O(nm)$. As there are m edges in M, the overall time complexity of our procedure is $O(m^2 n)$.

Theorem4.3: an edge coloration for a multigraph $M = (N, E)$ with no more than $(\frac{3}{2}) * d$ different colors can be obtained in $O(m^2 n)$ time, where d is the maximum degree of a node in M, $n = |N|$ and $m = |E|$.

Proof: By the above discussion. ■

V. Discussion

It has been shown that the VAS and MVAS problems are NP-hard even when only two via columns are needed for the interconnections. On the practical side, we have shown that for the MVAS problem one can construct a solution that is "close" to optimal. Our algorithm is of polynomial time complexity. One can easily improve on the performance of our algorithm without increasing the upper bound on its time complexity by using some simple heuristics in the construction of problem X as well as in some postprocessing transformations. However, it seems highly unlikely that a smaller worst case approximation bound can be obtained without a considerable increase on the time complexity bound. An approximation bound of $3 - 3/n$ can be achieved on problem instances with n nets, for $n \geqslant 3$. One of the interesting features of our algorithm is that it generates a lower bound for the number of via columns in an optimal solution. This allows the practitioner to get some idea of how good is the solution he intends to use.

The total area of the board can also be used as the objective function for the via assignment problem. Under this function, finding an optimal solution is still an NP-hard problem. However, the complexity of the corresponding approximation problem is very different from the one of the problems studied in this paper. The same comment applies when we are allowed to introduce via rows. In a subsequent paper we present some results relating to this and other variations of the via assignment problem.

Acknowledgment

The author would like to thank Sing-Ling Lee for his comments and suggestions concerning this paper. The problem

instances that achieve the approximation bound of $3 - 3/n$ for $n \geqslant 3$ were obtained by him.

References

[1] J. Bondy and U.S.R. Murty, *Graph Theory with Applications.* Amsterdam, The Netherlands: Elsevier North Holland, 1976.

[2] M. Garey and D. Johnson, "The complexity of near-optimal graph coloring," *J. ACM*, vol. 23, no. 23, pp. 43–49.

[3] T. Gonzalez, "Remarks on the via assignment problem," Tech. Rep. 129, Univ. of Texas at Dallas, Apr. 1983.

[4] J. Hopcroft and R. Karp, "An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs," *SIAM J. Comput.*, 1973, pp. 225–231.

[5] R. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York: Plenum, 1972, pp. 85–104.

[6] E. Kuh, T. Kashiwabara, and T. Fujisawa, "On optimal single-row routing," *IEEE Trans. Circuits Syst.*, vol. CAS-26, June 1979.

[7] R. Raghavan and S. Sahni, "Optimal single row router," *IEEE Trans. Computers*, vol. C-32, Mar. 1983, pp. 209–220.

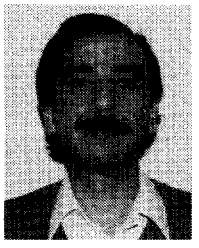
[8] H. So, "Some theoretical results on the routing of multilayer printed-wiring boards," in *IEEE Symp. Circuits and Systems*, pp. 296–303, 1974.

[9] B. Ting, E. Kuh, and A. Sangiovanni-Vincentelli, "Via assignment problem in multilayer printed circuit board," *IEEE Trans. Circuits Syst.*, vol. CAS-26, Apr. 1979.

[10] S. Tsukiyama, I. Shirakawa, and S. Asahara, "An algorithm for the via assignment problem in multilayer backboard wiring," *IEEE Trans. Circuits Syst.*, vol. CAS-26, June 1979.

[11] V. G. Vizing, "On an estimate of the chromatic class of a p-graph," *Diskret. Analiz.*, (in Russian), 3, 1964, pp. 25–30.

*



Teofilo F. Gonzalez was born in Monterrey, Mexico, on January 26, 1948. He recieved the B.S. degree in computer science from the Instituto Tecnologico de Monterrey, in 1972 and the Ph.D. degree in computer science from the University of Minnesota, Minneapolis, in 1975.

He has been a member of the Computer Science faculty at the University of Oklahoma, the Pennsylvania State University, the Instituto Tecnologico de Monterrey and the University of Texas at Dallas. He is now Associate Professor of Computer Science at the University of California, Santa Barbara. His major research interests are in the design and analysis of algorithms, computational aspects of computer aided design, and scheduling theory.

Dr. Gonzalez is a member of the Association for Computing Machinery and the Operations Research Society of America.

A Class of Cellular Architectures to Support Physical Design Automation

ROB A. RUTENBAR, STUDENT MEMBER, IEEE, TREVOR N. MUDGE, MEMBER, IEEE, AND DANIEL E. ATKINS, MEMBER, IEEE

***Abstract*—Special-purpose hardware has been proposed as a solution to several increasingly complex problems in design automation. This paper examines a class of cellular architectures called *raster pipeline subarrays–RPS* architectures–applicable to problems in physical DA that are (1) representable on a cellular grid, and (2) characterized by local functional dependencies among grid cells. Machines with this architecture first evolved in conventional cellular applications that exhibit similarities to grid-based DA problems. To analyze the properties of the RPS organization in context, machines designed for cellular applications are reviewed, and it is shown that many DA machines proposed/constructed for grid-based problems fit naturally into a taxonomy of cellular machines.**

The implementation of DA algorithms on RPS hardware is partitioned into *local* issues that involve the processing of individual cell neighborhoods, and *global* issues that involve strategies for handling complete grids in a pipeline environment. Design rule checking and routing algorithms are examined in an RPS environment with respect to these issues. Experimental measurements for such algorithms running on an existing RPS machine exhibit significant speedups.

From these studies are derived the necessary performance characteristics of RPS hardware optimized specifically for grid-based DA. Finally, the practical merits of such an architecture are evaluated.

Manuscript received February 20, 1983; revised February 28, 1984. This work was supported in part by the National Science Foundation under Grant MCS-8009315 and under Grant MCS-8007298.

The authors are with the Department of Electrical Engineering and Computer Science, and the Computing Research Laboratory, University of Michigan, Ann Arbor, MI 48109.

I. Introduction

THE successful implementation of increasingly complex integrated systems has been made possible only because of the existence of increasingly sophisticated DA tools. Traditional DA research–for example, mathematical analysis of DA algorithms and data-structures, application of software structuring techniques to chip layout, and use of databases to manage the design process–has produced software tools running on conventional serial computers. These tools are limited in three fundamental ways: by the inherent complexity of the problem, by the efficiency of the coded implementation, and by the resources of the machine on which the code runs. To overcome these three limitations recent attention has focused

0278-0070/84/1000-0264$01.00