1-1-1991

Disjoint Covers in Replicated Heterogeneous Arrays

P.K. McKinley

N. Hasan

Ran Libeskind-Hadas
Harvey Mudd College

C.L. Liu

Recommended Citation

P. K. McKinley, N. Hasan, R. Libeskind-Hadas, and C. L. Liu, "Disjoint Covers in Replicated Heterogeneous Arrays," SIAM Journal on Discrete Mathematics, Vol. 4, No. 2, May 1991, pp. 281-292. DOI: 10.1137/0404026

SIAM J. DISC. MATH.
Vol. 4, No. 2, pp. 281-292, May 1991

DISJOINT COVERS IN REPLICATED HETEROGENEOUS ARRAYS*

P. K. MCKINLEY†‡, N. HASAN†§, R. LIBESKIND-HADAS†, AND C. L. LIU†

Abstract. Reconfigurable chips are fabricated with redundant elements that can be used to replace the faulty elements. The fault cover problem consists of finding an assignment of redundant elements to the faulty elements such that all of the faults are repaired. In reconfigurable chips that consist of arrays of elements, redundant elements are configured as spare rows and spare columns.

This paper considers the problem in which a chip contains several replicates of a heterogeneous array, one or more sets of spare rows, and one or more sets of spare columns. Each set of spare rows is identical to the set of rows in the array, and each set of spare columns is identical to the set of columns in the array. Specifically, an ith spare row can only be used to replace an ith row of an array, and similarly with spare columns. Repairing the chip reduces to finding a cover for the faults in *each* of the arrays. These covers must be disjoint; that is, a particular spare row or spare column can be used in the cover of at most one array. Results are presented for three fault cover problems that arise under these conditions.

Key words. reconfigurable chips, fault covers

AMS(MOS) subject classification. 94C15

1. Introduction.

As chip density increases, the likelihood of fabrication defects on chips also increases. Maintaining an acceptable yield in chip production requires the capability to repair defective chips. To this end, reconfigurable chips are fabricated with redundant elements that can be used to replace faulty elements. The *fault cover problem* consists of finding an assignment of redundant elements to the faulty elements such that all of the faulty elements are replaced.

For reconfigurable chips that consist of arrays of elements, redundant elements are configured as spare rows and spare columns [15]. Examples of such *reconfigurable arrays* include not only arrays of memory elements [19], but also arrays of processors [11], [14]. A *line* refers to a row or column of an array. In a reconfigurable array, each spare line can be activated by programming selection circuitry after fabrication to effectively replace lines containing faulty elements. The fault cover problem seeks an assignment of the spare lines to the array such that all of the faulty elements are repaired. The set of replaced lines is referred to as a *cover.*

In the model studied previously [8], [12], a row that contains faulty elements can be replaced by *any* spare row, and a column that contains faulty elements can be replaced by *any* spare column. An example of this model is shown in Fig. 1, in which X's indicate faulty elements. Assigning spare rows to rows 1 and 4 and spare columns to columns 2 and 6, marked with arrows, represents one possible repair solution for this array. The fault cover problem for this type of reconfigurable array is NP-complete [12]. Several algorithms, including exhaustive approaches and heuristics, have been developed for this problem [3], [4], [7], [12], [18], [19].

The situation in which a particular row (column) can be replaced only by a member of a proper subset of the spare rows (columns) arises when the elements in the array are not all identical. For example, consider the array shown in Fig. 2, which contains four

* Received by the editors January 16, 1990; accepted for publication (in revised form) May 8, 1990. This research was supported in part by National Science Foundation grant MIP 87-03273.

† Department of Computer Science, University of Illinois, 1304 West Springfield Avenue, Urbana, Illinois 61801.

‡ Current address, Department of Computer Science, Michigan State University, East Lansing, Michigan 48824.

§ Current address, IBM Corporation, Post Office Box 950, Poughkeepsie, New York 12602.

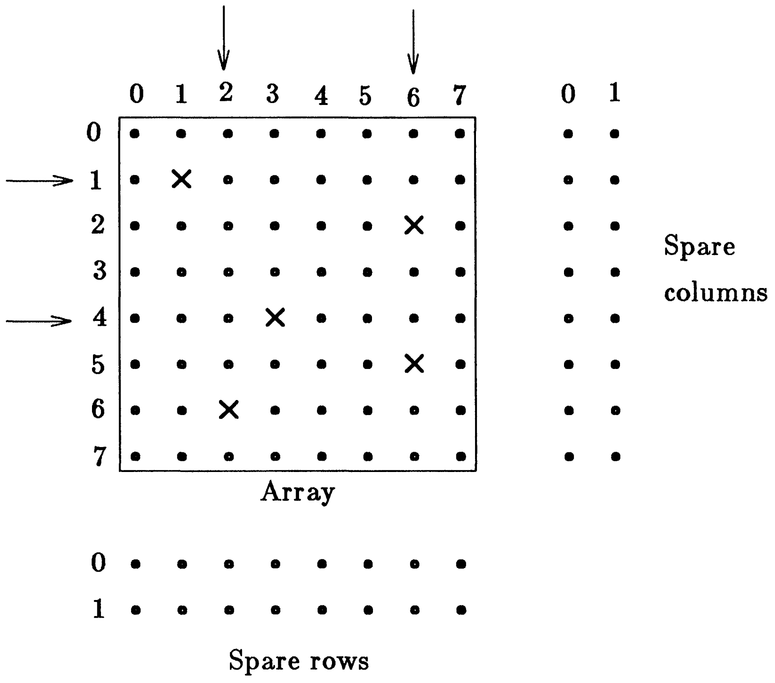


FIG. 1. *Reconfigurable* 8 × 8 *array with two spare rows and two spare columns.*

types of elements. In the configuration shown, the array comprises two types of rows and four types of columns. A spare row and column of each type is provided. Clearly, a line can be replaced only by a line of the same type.

We are concerned with problems in which such heterogeneity of array elements implies that the *i*th rows of all the arrays share one or more spares, and similarly for the

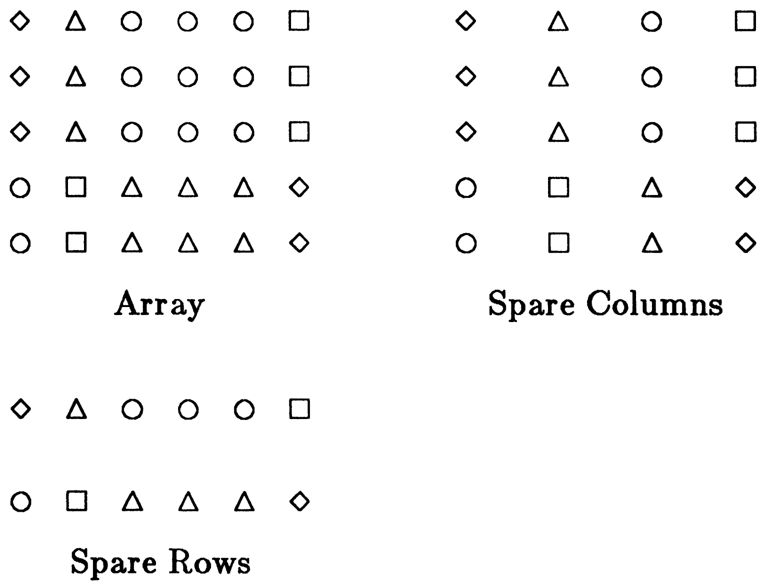


FIG. 2. *Heterogeneous reconfigurable array.*

jth columns of all the arrays. In other words, the set of spare rows is identical to the set of rows in the array, and the set of spare columns is identical to the set of columns in the array. When the chip contains a single array of elements, the problem of repairing faults is trivial. In fact, each fault can be covered in either of two ways, with a spare row or with a spare column. When the chip contains multiple copies of an array, however, repairing the chip reduces to finding a cover for the faults in *each* of the arrays. Figure 3 shows three copies of an array whose faults must be covered by lines from one set of spare rows and one set of spare columns. A spare line can be assigned to only one of the three arrays; that is, the three covers for the arrays must be *disjoint*.

To formulate the problem of finding disjoint covers for replicated heterogeneous arrays, we model each array as a (0, 1)-matrix, a 0 indicating a nonfaulty element and a 1 indicating a faulty element. Figure 4 shows an instance of the problem for the arrays depicted in Fig. 3. Each of the arrays contains two faulty elements. One solution to the cover problem, indicated with arrows, is the following: spare columns 1 and 2 are assigned to array 1; spare row 2 is assigned to array 2; spare column 4 is assigned to array 3. The covers are disjoint and all of the faults are covered.

In this paper, we present results for three fault cover problems for reconfigurable arrays in which the use of spare lines is constrained in the manner described above. These problems are the *feasibility problem*, the *disjoint minimum cover problem*, and the *multiple spare array problem*. In the first two problems, the chip is assumed to comprise t replicates of an array and one set each of spare rows and columns. The feasibility problem asks whether or not the chip can be repaired; the disjoint minimum cover problem seeks a feasible solution but with the stipulation that the individual cover of each array be minimum, that is, consisting of a minimum number of spare lines. In §§ 2 and 3, respectively, we show that these problems can be solved in polynomial time. The multiple spare array problem is a generalization of the feasibility problem in which more than two arrays of spares are available for use in covering faults. In § 4, we show that the multiple spare array problem is NP-complete. In § 5, we briefly discuss other potential applications for our fault cover model, and in § 6 we summarize our results.

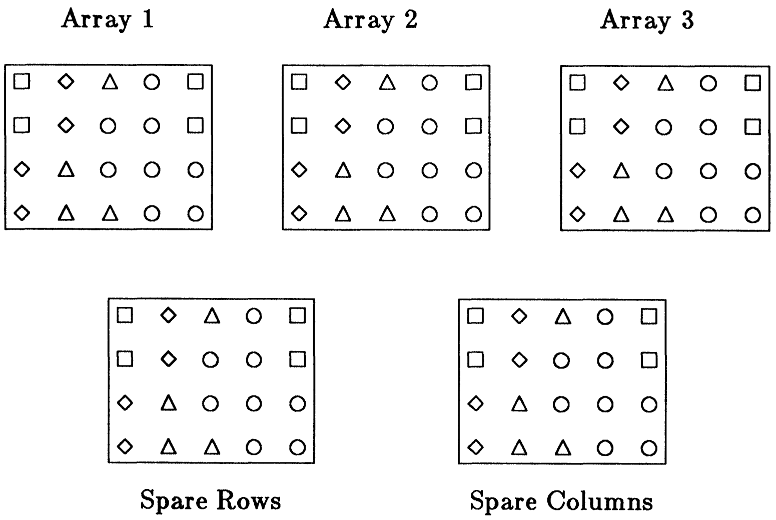


FIG. 3. *Replicated heterogeneous arrays and spares.*

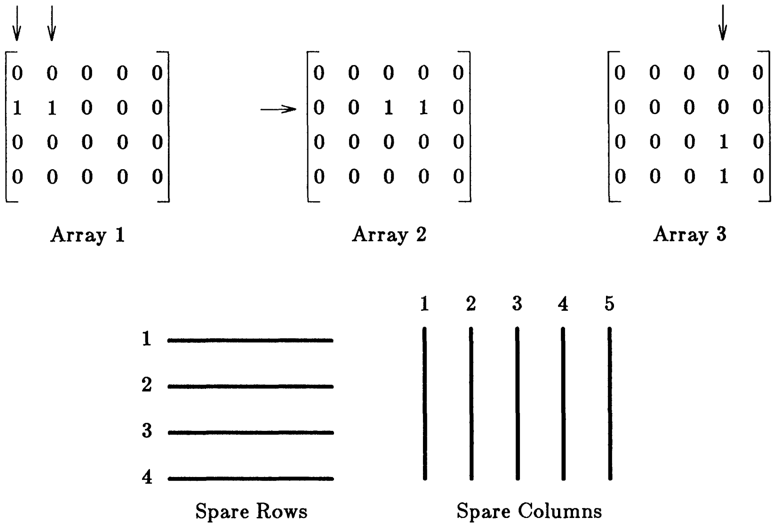


FIG. 4. *Disjoint covers of arrays.*

2. Feasible disjoint covers. Given t copies of an $R \times C$ array, each containing zero or more faulty elements, an array of R spare rows and an array of C spare columns, the *feasibility problem* seeks an assignment of the spare lines to the arrays such that all faults are covered and no spare line is assigned to more than one array. The ith spare row may only be assigned to cover the ith row of an array, and the jth spare column may only be assigned to cover the jth column. To solve the feasibility problem, we show that the problem can be formulated as a multigraph coloring problem. This problem in turn can be reduced to the 2-satisfiability (2SAT) problem in which, given a set U of Boolean variables and a conjunction of 2-clauses over U, we seek an assignment of values to variables such that each of the clauses is *true*. The 2SAT problem is solvable in polynomial time using any one of several known algorithms [2], [5]. The multigraph is constructed as follows.

CONSTRUCTION 1. *Given a set of replicated arrays* $A_1, A_2, \cdots, A_t$, *we represent each fault i with a vertex* v_i *in a multigraph G. With each vertex* v_i, *we associate the label* $(a_i{:}r_i, c_i)$, *representing the array, row, and column, respectively, of the ith fault. For each pair of faults not in the same array that lie in the same rows of their respective arrays, we add a red edge to G between the vertices representing the faults. Similarly, we add a black edge for pairs of faults in the same columns of their respective arrays. Let V be the set of vertices and E the set of edges in G.*

The multigraph G may not be connected. In fact, if there exists a fault in row i and column j of an array, and if there are no faults in row i or column j of all other arrays, then this fault will be represented by an isolated vertex. Next, we consider the problem of assigning the colors red and black to the vertices of such a multigraph. We say that a coloring is *feasible* if every vertex is colored, no black edge has two black endpoints, and no red edge has two red endpoints.

THEOREM 1. *A feasible coloring for a multigraph resulting from Construction* 1 *exists if and only if there exist disjoint covers* $K_1, K_2, \cdots, K_t$ *for the arrays* $A_1, A_2, \cdots, A_t$.

Proof. In the following, $i, j, k, l \in \{1, 2, \cdots, |V|\}$. Assume that $K_1, K_2, \cdots, K_t$ are disjoint covers for $A_1, A_2, \cdots, A_t$. For each faulty element i, if spare row r_i is contained in K_{a_i}, then we color vertex v_i red; otherwise we color the vertex black. We claim that this coloring is feasible. If not, then there must exist a red edge whose endpoints, labeled $(a_i{:}r_i, c_i)$ and $(a_j{:}r_j, c_j)$, with $r_i = r_j$, are both red or a black edge whose endpoints, labeled $(a_k{:}r_k, c_k)$ and $(a_l{:}r_l, c_l)$, with $c_k = c_l$, are both black. The former case implies that row r_i is contained in two different covers, K_{a_i} and K_{a_j}. This contradicts our assumption that the covers are disjoint. The latter case implies that column c_k is contained in two different covers, K_{a_k} and K_{a_l}. Again, this is a contradiction.

Next, let C: $V \rightarrow \{red, black\}$ be a feasible coloring of the multigraph. We construct a set of covers $K_1, K_2, \cdots, K_t$ as follows. For each vertex v_i that is colored red, we include spare row r_i in cover K_{a_i}. For each vertex v_j that is colored black, we include spare column c_j in cover K_{a_j}. We claim that $K_1, K_2, \cdots, K_t$ are disjoint covers of $A_1, A_2, \cdots, A_t$. Since each vertex represents a faulty element, and for each vertex v_i at least one of row r_i and column c_i is included in cover K_{a_i}, it follows that $K_1, K_2, \cdots, K_t$ constitute covers for $A_1, A_2, \cdots, A_t$, respectively. Assume that $K_1, K_2, \cdots, K_t$ are not pairwise disjoint. If a row r_i is included in both K_{a_i} and K_{a_j}, then there must exist two vertices labeled $(a_i{:}r_i, c_i)$ and $(a_j{:}r_j, c_j)$, with $r_i = r_j$, both colored red and connected by a red edge, a contradiction. Similarly, if a column c_k is included in both K_{a_k} and K_{a_l}, then there must be two vertices labeled $(a_k{:}r_k, c_k)$ and $(a_l{:}r_l, c_l)$, with $c_k = c_l$, both colored black and connected by a black edge. Again, this is a contradiction. □

Figure 5 shows the multigraph corresponding to the arrays and their faults shown in Fig. 4. Red edges are depicted with solid lines, black edges with dashed lines. If vertices v_1, v_2, v_5, and v_6 are colored black and vertices v_3 and v_4 are colored red, then the coloring is feasible. From this solution, we generate disjoint covers for the arrays as follows: for each red vertex labeled $(a_i{:}r_i, c_i)$, we assign spare row r_i to cover row r_i in array a_i; for each black vertex labeled $(a_j{:}r_j, c_j)$, we assign spare column c_j to cover column c_j in array a_j. That is, spare columns 1 and 2 are assigned to array 1; spare row 2 is assigned

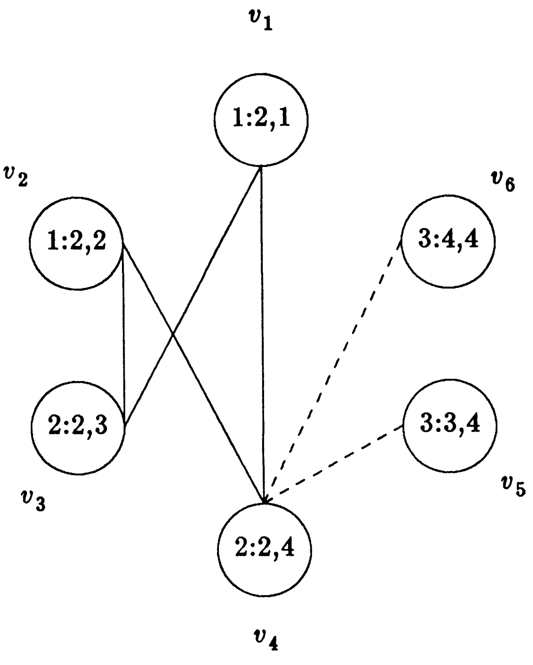


FIG. 5. *Multigraph for feasibility problem.*

to array 2; spare column 4 is assigned to array 3. This solution is indicated with arrows in Fig. 4.

To solve the feasibility problem, we require an algorithm to solve this multigraph coloring problem. Construction 2 shows that this problem can be formulated as an instance of 2SAT, solvable in polynomial time [2], [5].

CONSTRUCTION 2. *Given a multigraph $G = (V, E)$ with red edges and black edges, we construct a conjunction of clauses as follows: For each vertex $v_i \in V$, we introduce a Boolean variable s_i. For each red edge (v_i, v_j), we include the clause $(\bar{s}_i \vee \bar{s}_j)$; for each black edge (v_k, v_l), we include the clause $(s_k \vee s_l)$.*

Intuitively, setting s_i to *true* means that the fault represented by s_i is covered by a row. Similarly, setting s_i to *false* means that the fault represented by s_i is covered by a column.

THEOREM 2. *The conjunction of clauses resulting from Construction 2 is satisfiable if and only if a feasible coloring exists for the multigraph G.*

Proof. Let $C: V \rightarrow \{red, black\}$ be a feasible coloring of the multigraph G. We assign values to the Boolean variables as follows: For each vertex v_i that is colored red, we assign variable s_i the value *true*. For each vertex v_j that is colored black, we assign variable s_j the value *false*. Each red edge (v_i, v_j) has at least one black endpoint, so the clause $(\bar{s}_i \vee \bar{s}_j)$ is *true*. Each black edge (v_k, v_l) has at least one red endpoint, so the clause $(s_k \vee s_l)$ is *true*. Therefore, all the clauses in the conjunction are *true*.

Next, let $TA: \{s_i\} \rightarrow \{true, false\}$ be a truth assignment satisfying the conjunction of clauses. We color the multigraph as follows: For each *true* variable, we color its corresponding vertex red. For each *false* variable, we color its corresponding vertex black. Note that an isolated vertex will not be represented in the conjunction of clauses. For completeness, we color each isolated vertex red. Each clause of the form $(\bar{s}_i \vee \bar{s}_j)$ is *true*, so the red edge (v_i, v_j) it represents must have at least one black endpoint. Each clause of the form $(s_k \vee s_l)$ is *true*, so the black edge (v_k, v_l) it represents must have at least one red endpoint. Therefore, the coloring is feasible. □

As an example, we give the 2SAT formulation for the set of arrays shown in Fig. 4. Using the numbering of the vertices in Fig. 5, the conjunction of clauses is: $(\bar{s}_1 \vee \bar{s}_3) \wedge (\bar{s}_1 \vee \bar{s}_4) \wedge (\bar{s}_2 \vee \bar{s}_3) \wedge (\bar{s}_2 \vee \bar{s}_4) \wedge (s_4 \vee s_5) \wedge (s_4 \vee s_6)$. An example of a satisfying truth assignment is constructed by setting s_3 and s_4 to be *true* and setting s_1, s_2, s_5, and s_6 to be *false*.

3. Disjoint minimum covers.

The disjoint minimum cover problem seeks a feasible solution to the fault cover problem, with the stipulation that the individual cover of each array be minimum. Finding minimum covers is one way to reduce the cost of repairing the chip [3]. To show that the disjoint minimum cover problem can be solved in polynomial time, we must first provide some background results. A minimum cover of a (0, 1)-matrix is a minimum set of lines that contain all the 1's. The problem may be represented by a graph. For a given (0, 1)-matrix, we construct a bipartite graph G, which consists of two sets of vertices, X and Y, and a set of edges E. For each row r_i of the matrix there is a vertex $x_{r_i} \in X$. For each column c_j of the matrix there is a vertex $y_{c_j} \in Y$. There is an edge between vertices x_{r_i} and y_{c_j} if there is a 1 in position (r_i, c_j) in the matrix. This construction is illustrated in Fig. 6. A cover of G is a set of vertices $K \subseteq X \cup Y$ such that every $e \in E$ is adjacent to some vertex $k \in K$.

A *matching* in a graph is a subset of the edges such that no two edges in the matching have a common endpoint. A *maximum matching* is a matching of maximum cardinality. The bold edges in Fig. 6 constitute a maximum matching. Given a graph G and a matching in G, a vertex is said to be *matched* if it is adjacent to an edge in the matching; otherwise,

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & & 1 & 1 & & \\ 2 & 1 & & 1 & & 1 \\ 3 & & & & & 1 \\ 4 & 1 & & & 1 & \end{array}$$

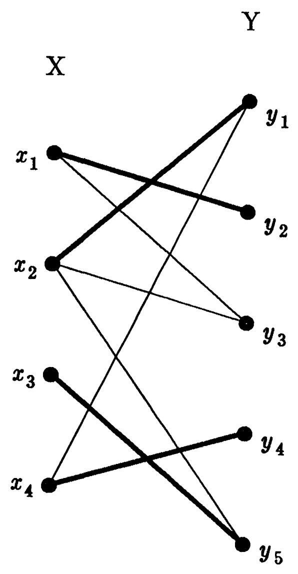


FIG. 6. (0, 1)-*matrix and corresponding bipartite graph.*

it is said to be *unmatched*. The König–Egerváry theorem [13] states that the size of a minimum cover in a (0, 1)-matrix is the same as the size of a maximum matching in the corresponding bipartite graph. Finding a maximum matching in a bipartite graph can be done in time $O(m\sqrt{n})$, where m is the number of edges and n is the number of vertices [10].

LEMMA 1. *Given a bipartite graph $G = (X \cup Y, E)$, a maximum matching M in G, and a minimum cover K of G, exactly one endpoint of each edge in M belongs to K.*

Proof. By definition, at least one endpoint of every edge must be included in K. Since edges in M have no endpoints in common, no vertex can cover two edges in M. So at least one endpoint of each edge in M must be in K. From the König–Egerváry theorem [13], we know that $|K| = |M|$. Therefore, at most one endpoint of each edge in M can be in K. □

LEMMA 2. *Given a bipartite graph $G = (X \cup Y, E)$ and a maximum matching M in G, an unmatched vertex does not belong to any minimum cover K of G.*

Proof. By Lemma 1, there must be at least $|M|$ matched vertices in the cover. Since $|K| = |M|$, the cover K can contain no other vertices. □

Using these two lemmas, we now show that the disjoint minimum cover problem, like the feasibility problem, can be reduced to 2SAT. The conjunction of clauses is formed using the following construction.

CONSTRUCTION 3. *Let R and C be the number of rows and columns, respectively, in each of t (0, 1)-matrices, $A_1, A_2, \cdots, A_t$. Let G_i be the bipartite graph corresponding to A_i. Let M_i be a maximum matching for A_i. For each row r_i, $1 \leq r_i \leq R$, we introduce t Boolean variables, $r_{i,1}, r_{i,2}, \cdots, r_{i,t}$. For each column c_i, $1 \leq c_i \leq C$, we introduce t Boolean variables, $c_{i,1}, c_{i,2}, \cdots, c_{i,t}$. The conjunction consists of four types of clauses*:

(1) *For each* 1, *we include the clause $(r_{i,k} \vee c_{j,k})$, where r_i, c_j, and A_k are the row, column, and array, respectively, that contain the* 1.

(2) *Next, for each row r_i that contains a* 1 *in one or more of the arrays, and for each unordered pair of matrices, A_k and A_l, we include the clause $(\bar{r}_{i,k} \vee \bar{r}_{i,l})$. For each column c_j that contains a* 1 *in one or more of the arrays, and for each unordered pair of*

matrices, A_k and A_l, we include the clause $(\bar{c}_{j,k} \vee \bar{c}_{j,l})$. Hence, for each line that contains a 1 in one or more of the arrays, $t(t-1)/2$ clauses are included in the conjunction.

(3) *For each 1 that is represented by an edge in M_k, we include the clause $(\bar{r}_{i,k} \vee \bar{c}_{j,k})$, where r_i and c_j are the row and column, respectively, that contain the 1.*

(4) *Finally, for each row r_i whose representative vertex in G_k is not matched, we include the clause $(\bar{r}_{i,k})$. For each column c_j whose representative vertex in G_k is not matched, we include the clause $(\bar{c}_{j,k})$.*

THEOREM 3. *The conjunction of clauses resulting from Construction 3 is satisfiable if and only if there exist disjoint minimum covers $K_1, K_2, \cdots, K_t$ for matrices $A_1, A_2, \cdots, A_t$.*

Proof. Assume that there exist disjoint minimum covers $K_1, K_2, \cdots, K_t$ for matrices $A_1, A_2, \cdots, A_t$. We assign truth values to variables as follows: For each spare row $r_i \in K_k$, we set $r_{i,k}$ to be *true*; for each spare column $c_j \in K_l$, we set $c_{j,l}$ to be *true*. Each 1 in A_k is covered by its row r_i or its column c_j, so its corresponding clause $(r_{i,k} \vee c_{i,k})$ must be *true*. Since each spare row r_i can be assigned to at most one cover, every clause $(\bar{r}_{i,k} \vee \bar{r}_{i,l})$ must be *true*. Since each spare column c_j can be assigned to at most one cover, every clause $(\bar{c}_{j,k} \vee \bar{c}_{j,l})$ must be *true*. By Lemma 1, we know that, for each edge in a matching M_k, exactly one of its endpoints must be included in a minimum cover of G_k. Therefore, every clause $(\bar{r}_{i,k} \vee \bar{c}_{j,k})$ must be *true*. Finally, by Lemma 2, an unmatched vertex in a bipartite graph G_k cannot be in a minimum cover of G_k, so all 1-clauses must be *true*. Hence, using the truth assignment above, the conjunction of clauses is *true*.

Conversely, assume the conjunction is satisfiable. Then there exists a truth assignment that forces every clause to be *true*. For each *true* variable $r_{i,k}$, include spare row r_i in cover K_k. For each *true* variable $c_{j,l}$, include spare column c_j in cover K_l. The clauses from step 1 imply that each 1 is covered. The clauses from step 2 imply that the covers are disjoint. The clauses from steps 3 and 4 imply that the covers are minimum. □

We omit the details of the conjunction for the example shown in Fig. 4. We note, however, that while there exists a solution to the feasibility problem for this example, there does not exist a set of disjoint minimum coverings for the three arrays shown. Such a set could involve no more than three spare lines, but the faults in the arrays cannot be covered with fewer than four.

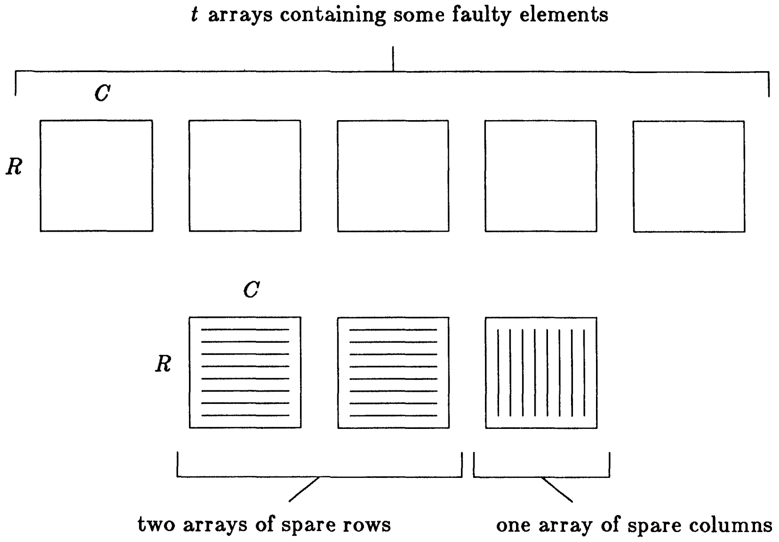
4. Disjoint covers using multiple spare arrays.

The multiple spare array problem is an extension of the feasibility problem discussed in § 2 to include the case in which the chip contains more than one set of spare rows, more than one set of spare columns, or both. Multiple sets of spares offer potential increases in chip yield because more defects can be successfully covered. Of course, the increase in yield must be balanced against the increase in fabrication and materials costs accompanying the use of additional spares.

The multiple spare array problem can be stated as follows: Given t (0, 1)-arrays of R rows and C columns each, SR $R \times C$ arrays of spare rows and SC $R \times C$ arrays of spare columns, the problem is to find an assignment of the spares to the arrays such that all the ones in the arrays are covered. An example of the multiple spare array problem, in which $SR = 2$ and $SC = 1$, is depicted in Fig. 7. Unfortunately, finding such disjoint covers is much more difficult than is the original problem, in which $SR = SC = 1$.

THEOREM 4. *The multiple spare array problem is* NP-*complete.*

Proof. The problem is in NP because we can guess an assignment of the spares to the arrays and check in polynomial time whether or not all the faulty elements are covered. Next, we want to use a reduction from a known NP-complete problem to the multiple spare array problem to show that the latter is NP-complete. Our reduction is



t arrays containing some faulty elements
C
R
C
R
two arrays of spare rows
one array of spare columns

FIG. 7. *Multiple spare array problem.*

from the *vertex cover* problem, which is as follows: Given a graph $G = (V, E)$, and an integer $K \leqq |V|$, does there exist a subset $V' \subseteq V$ with $|V'| \leqq K$ such that for each edge $(u, v) \in E$, at least one of u and v belongs to V'? The vertex cover problem is NP-complete [6].

Given an instance of the vertex cover problem, we want to construct an instance of the multiple spare array problem. Given $V = \{v_1, v_2, \cdots, v_n\}, E = \{e_1, e_2, \cdots, e_m\}$, and a positive integer $K \leqq n$, we construct n $1 \times m$ arrays, $A_1, A_2, \cdots, A_n$, K $1 \times m$ arrays of spare rows, and one $1 \times m$ array of spare columns. That is, $SR = K$ and $SC = 1$. An entry $(1, j)$ in array A_i is 1 if and only if vertex v_i is one of the endpoints of edge e_j. This means that the sum of the number of faulty elements in each column over all the arrays is exactly 2.

Now we want to show that there exists a solution to the instance of the vertex cover problem if and only if there exists a solution to the instance of the multiple spare array problem. If there is a solution to the instance of the vertex cover problem, then there is a subset $V' = \{v'_1, v'_2, \cdots, v'_l\}$ of V such that $l \leqq K$ and every edge in E has at least one endpoint in V'. For each vertex v'_i in V', we assign a spare row to the first (and only) row of array A_i. We use at most SR spare rows, because $SR = K$. Since every edge has at least one of its endpoints in V', the sum of the number of faulty elements that have not been covered by spare rows, in each column over all the arrays, is at most 1. This means that a spare column can be used to cover each of these 1's.

Suppose there is a solution to the instance of the multiple spare array problem. Let $A_{a_1}, A_{a_2}, \cdots, A_{a_l}$, where $l \leqq SR$, be the arrays to which spare rows are assigned. Let V' be $\{v_{a_1}, v_{a_2}, \cdots, v_{a_l}\}$. Since we have only one array of spare columns, this means that the sum of the number of faulty elements left uncovered by the spare rows in each column over all the arrays is at most 1. Recall that initially this number was 2. This means that the set V' contains at least one endpoint of each edge in E. $\square$

Although chip yield may be increased with the use of multiple sets of spares, our NP-completeness result implies that heuristic algorithms are likely to be the only viable approach to the problem. The investigation of such heuristics is a potential area for future research.

5. Other fault cover applications. The results presented here have potential application in two other VLSI contexts. First, as the density of reconfigurable arrays continues to increase, with a corresponding increase in the number of elements in the arrays, repairing a chip will require a larger number of spare lines. Often, however, an individual row or column contains only a small number of faulty elements [17]. This implies that, for a reconfiguration method such as shown in Fig. 1, in which entire rows or columns are replaced, the spare elements will be used inefficiently because most of them will replace correctly functioning elements. The number of such wasted redundant elements increases as the size of the array increases and limits the number of chips that can be repaired.

One way to increase efficiency in the use of spares, and thus increase yield, is to replace the single large array with an array of smaller subarrays. The redundant elements are arranged such that rows and columns of individual subarrays may be replaced, independent of other subarrays, achieving the desired higher efficiency. Allocating spare lines for each subarray may be expensive. Alternatively, allowing a spare line to be used anywhere on the chip is not an attractive solution because the cost of wiring and the size of programmable decoders increases with the partitioning of the array. A compromise solution, used in [9], [16], is to limit the number of subarrays to which a particular spare line may be assigned. Figure 8 shows how our model may be used in this manner. The array has been partitioned into 16 subarrays. The spare elements have been arranged as one array of spare rows and one array of spare columns.

Another potential application of our model stems from recent interest in three-dimensional VLSI design [1]. Consider the situation depicted in Fig. 9, in which eight arrays are sandwiched between an array of spare rows and an array of spare columns. Arranging redundant elements in this manner, and requiring that a spare row be used to replace only one of the rows directly below it, and that a spare column be used to replace only a column directly above it, offers one way to reduce the circuit complexity in reconfigurable three-dimensional devices.

In both applications just described, the arrays may be homogeneous. Our model for heterogeneous arrays is applicable because it is assumed that the need to simplify wiring for reconfiguration imposes constraints on the use of spares.

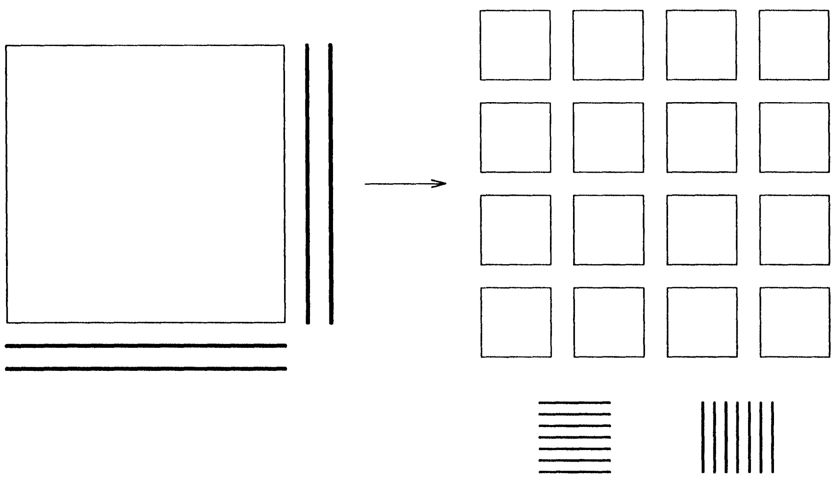


FIG. 8. *Reconfiguration of spares.*

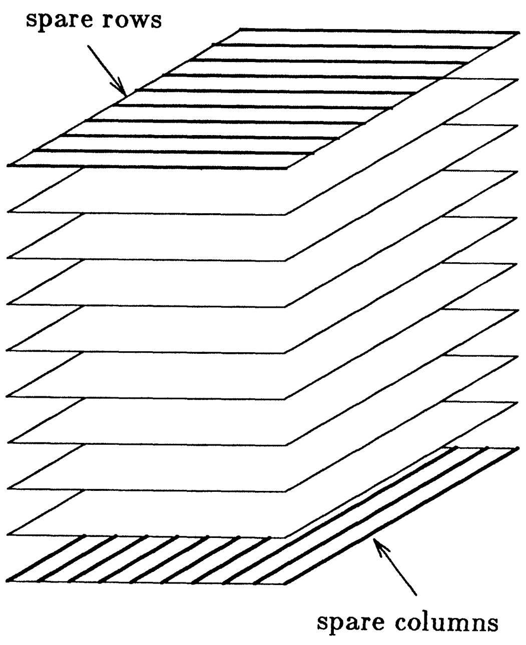


FIG. 9. *Three-dimensional arrays and spares.*

6. Summary. We have presented results for a set of fault cover problems in replicated, heterogeneous arrays of elements. First, a polynomial-time solution was given for the problem of finding a set of disjoint covers, if one exists, for the arrays using one set of spare rows and one set of spare columns. Second, a polynomial-time algorithm was given to find a feasible set of disjoint covers such that each is minimum. Finally, the problem of finding a feasible solution when multiple sets of spare lines are available was shown to be NP-complete. We briefly discussed two other potential applications of this work. We are currently studying extensions of the problems discussed here.

REFERENCES

[1] *Topical Meeting on Three Dimensional Integration*, Miyagi-Zao, Japan, May 30–June 1, 1988.
[2] M. DAVIS AND H. PUTNAM, *A computing procedure for quantification theory*, J. Assoc. Comput. Mach., 7 (1960), pp. 201–215.
[3] R. J. DAY, *A fault-driven comprehensive redundancy algorithm*, IEEE Design and Test, 2 (1985), pp. 35–44.
[4] R. C. EVANS, *Testing repairable* RAMS *and mostly good memories*, in Proc. IEEE Int. Test Conference, Philadelphia, PA, 1981, pp. 49–55.
[5] S. EVEN, A. ITAI, AND A. SHAMIR, *On the complexity of timetable and multicommodity flow problems*, SIAM J. Comput., 5 (1976), pp. 691–703.
[6] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W. H. Freeman, New York, 1979.
[7] R. W. HADDAD AND A. T. DAHBURA, *Increased throughput for the testing and repair of* RAMs *with redundancy*, in Proc. Intl. Conf. on Computer-Aided Design, Santa Clara, CA, 1987, pp. 230–233.
[8] N. HASAN AND C. L. LIU, *Minimum fault coverage in reconfigurable arrays*, in Proc. 18th Intl. Symp. on Fault-Tolerant Computing, Tokyo, Japan, June 27–30, 1988, pp. 348–353.
[9] Y. HAYASAKA, K. SHIMOTORI, AND K. OKADA, *Testing system for redundant memory*, in Proc. IEEE Int. Test Conference, Philadelphia, PA, 1982, pp. 240–244.

[10] J. HOPCROFT AND R. M. KARP, *An $n^{5/2}$ algorithm for maximum matching in bipartite graphs*, SIAM J. Comput., 2 (1973), pp. 225–231.

[11] I. KOREN AND D. K. PRADHAN, *Yield and performance enhancement through redundancy in* VLSI *and* WSI *multi-processor systems*, Proc. of IEEE, Vol. 74, No. 5 (1986), pp. 699–711.

[12] S. Y. KUO AND W. K. FUCHS, *Efficient spare allocation for reconfigurable arrays*, IEEE Design and Test, 4 (1987), pp. 24–31.

[13] D. KÖNIG, *Graphen und Matrizen*, Mat. Fiz. Lapok, 38 (1931), pp. 116–119.

[14] F. LOMBARDI, R. NEGRINI, M. G. SAMI, AND R. STEFANELLI, *Reconfiguration of* VLSI *arrays*: *A covering approach*, IEEE 17th Intl. Symp. on Fault-Tolerant Computing, Pittsburgh, PA, 1987, pp. 251–256.

[15] W. R. MOORE, *A review of fault-tolerant techniques for the enhancement of integrated chip yield*, Proc. of IEEE, Vol. 74, No. 5 (1986), pp. 684–697.

[16] Y. NISHIMURA, M. HAMADA, H. HIDAKA, H. OZAKI, AND K. FUJISHIMA, *A redundancy test-time reduction technique in* 1-Mbit DRAM *with a multibit test mode*, IEEE J. Solid-State Circuits, 24 (1989), pp. 43–49.

[17] S. E. SCHUSTER, *Multiple word/bit redundancy for semiconductor memories*, IEEE J. Solid-State Circuits, SC-13 (1978), pp. 698–703.

[18] M. TARR, D. BOUDREAU, AND R. MURPHY, *Defect analysis system speeds test and repair of redundant memories*, Electronics, January, 1984, pp. 175–179.

[19] C. L. WEY AND F. LOMBARDI, *On the repair of redundant* RAMS, IEEE Trans. Computer-Aided Design, Cad-6 (1987), pp. 222–231.