Decomposition of Two-Dimensional Microlaser Patterns

Hans Raj Nahata and Miles Murdocca*

Department of Computer Science, Hill Center

Rutgers University, New Brunswick, NJ 08903

(908) 445-2654 (phone), (-0537 fax)

hnahata@paul.rutgers.edu (Nahata)

murdocca@cs.rutgers.edu (Murdocca) **now: murdocca@iiusa.cc**

* Corresponding author

Submitted (July 1995) to Applied Optics feature issue on Optical Computing

Revised and resubmitted (November 1995)

NOTE: This paper appeared as: **Nahata, H. R. and M. Murdocca, "Decomposition of Two-Dimensional Microlaser Patterns,"** ***Applied Optics*****, vol. 35, (Mar. 10, 1996).**

Miles Murdocca is currently with IIUSA: murdocca@iiusatech.com

Abstract

For an ordinary *individually addressable* microlaser array, a separate control line is used for each microlaser, which requires a large number of control lines for even a small array. An organization that reduces the width of the control stream and simplifies packaging is *matrix addressing*, in which microlasers are arranged at the crossings of horizontal and vertical control lines.

We consider the problem of decomposing arbitrary two-dimensional (2D) microlaser patterns into matrix addressable patterns that are applied time sequentially to realize the target pattern. We present a mathematical model for the decomposition process, and present an algorithm for optimal decomposition. We also consider *bake factor*, in which no more than N microlasers in a neighborhood of M (where $N < M$) are enabled, which avoids thermal overload by limiting the density of enabled microlasers. We conclude with a case study and show that for completely arbitrary 2D patterns, that the average number of time sequential patterns is less than the number of rows in a square array.

1. Introduction

Two-dimensional (2D) arrays of microlasers are manufactured in two primary configurations: individually addressable [1], and matrix addressable [2], as illustrated in Figure 1. Each microlaser in the individually addressable array has a ground (n) terminal and a positive (p) terminal. All of the microlasers share the same ground, but a separate p contact is provided for each microlaser. An 8×8 array thus requires 64 p contacts, as indicated by the numbered bonding pads at the edges of the array. For small arrays individual addressing works well, but the complexity becomes unmanageable as the arrays scale to large sizes, and so an alternative configuration is needed that scales more gracefully.

For the matrix addressable array, each row of microlasers shares the same ground. For the 8 rows shown in Figure 1b, there are 8 independent n lines that are each connected to a distinct bonding pad. The p lines are connected to the columns in a similar manner, and so there are 8 independent p lines, which connect the p contacts of the 8 microlasers in a column. In order to enable a microlaser at location (i, j), in which i identifies a row and j identifies a column, the corresponding i row and j column bonding pads must be enabled. The n ground is applied to the row pad and the p potential is applied to the column pad. If a potential is applied to more than one pad, then the corresponding collection of microlasers is enabled. In Figure 1b, a potential is applied across rows 2, 3, and 5 and columns 3, 4, and 7, which enables the nine microlasers at the corresponding crosspoints. Notice that only six bonding pads are used, as opposed to the nine bonding pads for the same individually addressable configuration shown in Figure 1a.

An advantage of the matrix addressable configuration is that for an N^2 increase in the size of an array, the bonding pad complexity increases by only $2N$, which allows for a simplified electronic interface. A disadvantage is that the user loses a degree of freedom in selecting combinations of microlasers to enable or disable. For example, in Figure 1b, there is no combination of bonding pads that can be powered so that a checkerboard pattern is formed on the array. Despite the limited number of possible on/off combinations for a matrix addressable array, the complexity of the electronic addressing is simplified, which is an important practical consideration.

Here, we describe an algorithm for decomposing arbitrary patterns for a 2D microlaser array into a set of matrix addressable subpatterns, that when applied in succession achieve the desired target pattern. For discussion purposes, we assume there is a flip-flop that maintains the state of each microlaser (allowing time sequential pattern buildup) that can be set, reset, or toggled. The flip-flop is not needed if the pattern buildup is recorded by another means (such as with film, or a detector array). We begin by describing a mathematical model for the problem. We then develop an algorithm for the optimal decomposition of arbitrary patterns into matrix addressable subpatterns. We relate the decomposition method to the outer product of matrix multiplication and to singular value decomposition. We conclude with a case study of decomposition applied to randomly generated patterns for an 8×8 array, and show that the expected number of matrix addressable subpatterns that compose an arbitrary pattern is approximately 5.6 for an 8×8 array.

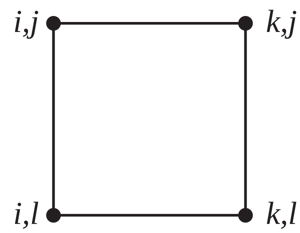
2. Background

The decomposition method is based on *binary matrices* and *symmetric patterns*, as defined below:

Binary Matrix: $B_{n \times n} = (b_{i,j})$ is a matrix in which each element $b_{i,j}$ is either 1 or 0. A binary matrix is also called a 0-1 matrix. A one-dimensional (1D) case of a binary matrix is a *binary vector*.

Symmetric Patterns: $P_{n \times n} = (p_{i,j})$ is a binary matrix with the following constraint [3], for all pairs of (i,j) and (k,l):

$$\forall (i,j),(k,l) \quad \begin{array}{ccc} p_{i,j} = 1 & & p_{i,l} = 1 \\ \text{and} & \Leftrightarrow & \text{and} \\ p_{k,l} = 1 & & p_{k,j} = 1 \end{array}$$



$\forall$ = "For all"
$\Leftrightarrow$ = "Implies in both directions"

That is, for all (i,j) and (k,l) pairs that form diagonal corners of a rectangle, there is another pair (i,l) and (k,j) that is also part of the pattern. Notice that a symmetric pattern may have intervening points between the corners that are not part of the pattern, that it may have zero extent in any dimension, and that it may have any number of points as long as the above relationships are satisfied.

Examples of symmetric patterns are shown in Figure 2a. Any matrix with a single 1 cell, or a single column of 1's, or a single row of 1's is a symmetric pattern. For comparison, Figure 2b shows a binary matrix that is not a symmetric pattern. Notice that all non-zero rows of a symmetric pattern are the same, as are all non-zero columns.

2.1. Symmetric Patterns Compose Arbitrary Patterns

The key relationship between symmetric patterns and the decomposition of 2D microlaser patterns is that any pattern that can be applied to a 2D array of matrix addressable microlasers in a single step is a symmetric pattern. Conversely, every symmetric pattern can be applied to a 2D array of matrix addressable microlasers in a single step. Our objective is to decompose an arbitrary pattern into the minimal number of symmetric patterns that when applied in succession enable the target pattern. As an example, consider the 5×5 matrix shown in Figure 3, in which 16 of the 25 microlasers are enabled, forming a non-symmetric pattern. The original pattern can be decomposed into three symmetric patterns as shown.

An arbitrary pattern can be expressed as a sum of symmetric patterns. A decomposition always exists; observe that a pattern with a single non-zero point is a symmetric pattern, by definition. Therefore, any pattern can be expressed as a sum of symmetric patterns each having only one non-zero point. This is illustrated in Figure 4, for one example. Note also that a decomposition need not be unique. In Figure 5, a binary matrix is decomposed in two different ways.

The principal algebra used for the decomposition process in the sections that follow is Boolean AND-OR. That is, for two Boolean variables a and b, the relationships shown in Figure 6 must hold. The AND operation produces a 1 when a and b are both 1, and produces a 0 otherwise. The OR operation (also referred to as the *logical sum*) produces a 0 when a and b are both 0, and produces a 1 otherwise. This algebra is applied over binary matrices and binary vectors, in a component-wise sense.

2.2. Equivalence Between Symmetric Patterns and the Outer Product

The above definition of a symmetric pattern is exact, but it does not expose the inherent structure of a pattern. We now define symmetric patterns in an equivalent manner that makes a more direct connection to matrix addressing.

The *outer product* is a matrix product of two vectors. Let $\vec{R}_{1\times n} = (r_i)$ and $\vec{C}_{1\times n} = (c_j)$ be two binary (row) vectors. The outer product $Q_{n\times n}$ of these two vectors is $R^T \times C$, where R^T is the transpose of a row vector into a column vector. The notation for the outer product is: $Q = [R,C]$ $(= R^T \times C)$.

As an example, consider the outer product of R and C shown below:

$$\begin{Bmatrix} R = (1\ \ 1\ \ 0\ \ 0) \\ C = (0\ \ 1\ \ 0\ \ 1) \end{Bmatrix}$$

$$Q = [R,C] = R^T \times C = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \times (0\ \ 1\ \ 0\ \ 1) = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Notice that all of the non-zero rows (and non-zero columns) of Q are the same. Furthermore, the number of non-zero rows (or columns) is directly related to the non-zero entries of R (or C).

Every outer product (Q) of two binary vectors is a symmetric pattern (P) and every non-zero symmetric pattern (P) can be expressed uniquely as an outer product of two binary vectors. In essence, the row vector (R) and the column vector (C) identify the bonding pads that need to be activated to enable a symmetric pattern onto a matrix addressable array. As an example of this relationship, consider the following:

$$\begin{pmatrix} 0\,1\,0\,1\,1\,1 \\ 0\,1\,0\,1\,1\,1 \\ 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,1\,1\,1 \\ 0\,0\,0\,0\,0\,0 \end{pmatrix} \Leftrightarrow \begin{Bmatrix} R = (1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0) \\ C = (0\ \ 1\ \ 0\ \ 1\ \ 1\ \ 1) \end{Bmatrix}$$

$$\begin{pmatrix} 0\,0\,0 \\ 0\,0\,1 \\ 0\,0\,0 \end{pmatrix} \Leftrightarrow \begin{Bmatrix} R = (0\ \ 1\ \ 0) \\ C = (0\ \ 0\ \ 1) \end{Bmatrix}$$

2.3. Properties of Symmetric Patterns

The outer product formulation exposes properties of symmetric patterns that are important for the decomposition process, as described below.

2.3.1. Equivalence of Two Symmetric Patterns

Let $P_1 = [R_1, C_1]$ and $P_2 = [R_2, C_2]$ be two non-zero symmetric patterns. Then:

$$P_1 = P_2 \Leftrightarrow R_1 = R_2 \text{ and } C_1 = C_2$$

Therefore, it follows that an ordered pair of two binary vectors $(\vec{R}, \vec{C})$ is equivalent to a pattern P ($P = Q = [R, C] = R^T \times C$), and *vice-versa*.

2.3.2. Number of Enabled Cells in a Symmetric Pattern

The number of 1's in the pattern P is the same as the product of the number of 1's in R and the number of 1's in C, where $P = [R, C]$.

2.3.3. Permutations on Symmetric Patterns

By permuting the rows and columns of P we can always convert it to the form:

$$\begin{pmatrix} 1's & 0 \\ 0 & 0 \end{pmatrix}$$

That is, for any given pattern P, there exist two permutation matrices Π and Φ such that:

$$\Pi \times P \times \Phi = \begin{pmatrix} 1's & 0 \\ 0 & 0 \end{pmatrix}$$

2.3.4. Number of Symmetric Patterns

The total number of distinct symmetric patterns is $2^n(2^n - 2) + 2$, where n is the number of row (or column) elements in a square matrix. This is given by the following expression:

$$\underset{\substack{\nearrow \\ \text{Number of non-} \\ \text{zero } R \text{ vectors}}}{(2^n - 1)} \quad \times \quad \underset{\substack{\uparrow \\ \text{Number of non-} \\ \text{zero } C \text{ vectors}}}{(2^n - 1)} \quad + \quad \underset{\substack{\nwarrow \\ \text{The zero} \\ \text{symmetric pattern}}}{1}$$

Contrast this with the total number of distinct binary matrices, which is 2^{n^2}!

The total number of distinct symmetric patterns, discounting the transposed patterns is given by:

$$(2^n - 1) \times (2^n - 2) + 1$$

Define the size of a symmetric pattern as the number of 1's it contains. The number of symmetric patterns of a given size m is given by the following expression:

$$\sum_{\left\{ \forall(p,q) \,\middle|\, \substack{0 < p,q \leq n \\ p \times q = m} \right\}} \binom{n}{p} \times \binom{n}{q}$$

Again, n is the extent of a row or column in a square matrix, and p and q are the number of 1's in the row and column vectors, respectively.

2.4. Merging of Symmetric Patterns

Symmetric patterns P_1 and P_2 are said to be *mergeable* if and only if their logical sum P (equal to $P_1 + P_2$) is also a symmetric pattern. Two symmetric patterns P_1 (equal to $R_1^T \times C_1$) and P_2 (equal to

$R_2^T \times C_2$) are mergeable (*i.e.* $P_1 + P_2$ is also a symmetric pattern) if and only if $R_1 = R_2$ or $C_1 = C_2$. Furthermore, the structure of the resultant symmetric pattern P (equal to $P_1 + P_2 = R^T \times C$) is the following [3]:

$$R_1 = R_2 \Rightarrow \left\{ \begin{matrix} R = R_1 \\ C = C_1 + C_2 \end{matrix} \right\} \qquad C_1 = C_2 \Rightarrow \left\{ \begin{matrix} C = C_1 \\ R = R_1 + R_2 \end{matrix} \right\}$$

3. Decomposition Algorithm for Uniform Cost Model

In the first of two decomposition methods, we assume that all symmetric patterns are equally desirable. That is, the cost/penalty associated with each symmetric pattern is the same. This differs from the second approach (described in Section 4) in which localized thermal loads are reduced.

3.1. The Algorithm

The problem of finding the most desirable decomposition is essentially an optimization problem. Given a binary matrix B (that is, a pattern), we want to decompose B into a sum of the minimum number of symmetric patterns (P_i's). The objective function is:

$$\min_{P_i\text{'s}} w$$

The constraint is:

$$\sum_{i=1}^{w} P_i = B$$

We define the matrix inclusion relation ($\subseteq$) as follows: A Boolean matrix A is said to be included in B if and only if $A + B = B$. That is,

$$A \subseteq B \Leftrightarrow B = A + B$$

The decomposition algorithm has two phases. Given the binary matrix B,

Phase 1. Find all symmetric patterns (P_i's) included in B. Let this set be I.

Phase 2. Compute $C \subseteq I$, such that C covers B (that is, $\Sigma_{P \in C} P = B$).

Notice that I can be a very large set. Therefore we re-define I so that none of its members can be contained in other members of I:

- $P \in I \Rightarrow P \subseteq B$

- For every non-trivial (i, j, k) triplet, there is no symmetric pattern P_k that also contains two other symmetric patterns P_i and P_j. That is, each symmetric pattern subsumes all symmetric patterns that it contains:

$$P_i, P_j, P_k \in I \Rightarrow P_i + P_j \neq P_k$$

- $P \subseteq B \Rightarrow \exists Q \in I$ such that $P \subseteq Q$.

Let each element of I be called a *prime implicant* (symmetric pattern) and I be the set of prime implicants. Notice that:

- I is not unique.

- I covers B, that is, $\Sigma_{P \in I} P = B$.

- A minimal cover C for B need not be unique.

- Phase 1 of the algorithm (finding the set of prime implicants) is closely related to the Quine-McCluskey method of finding prime implicant terms for Boolean expressions [4]. This is covered in Section 3.2.

5. Phase 2 of the algorithm is a case of the *set covering problem*, which is covered in Section 3.3.

3.2. Generating the Prime Implicants

In order to generate the set of prime implicants for B we need to first generate the set of *base patterns H*. Let H be the set of symmetric patterns for B, with the following properties:

1. $h \in H \Rightarrow h \subseteq B$ (A member of symmetric pattern set H is contained in binary matrix B.)

2. $P \subseteq B \Rightarrow \exists K \subseteq H$ such that $\Sigma_{h \in K} h = P$. (If pattern P, which may or may not be symmetric, is contained in binary matrix B, then there exists a subset of symmetric patterns K that is contained in H that composes P.)

3. $h_1, h_2 \in H \Rightarrow h_1 + h_2 \notin H$ (For any two symmetric patterns in H, there is no pattern in H that contains both.)

Notice that:

1. The base set for any binary matrix is unique.

2. By definition, H covers B (that is, $\Sigma_{h \in H} h = B$).

The base set H for B is the set of all single 1 (symmetric) patterns included in B. An example is:

$$\begin{pmatrix} 1\ 0\ 1 \\ 0\ 0\ 0 \\ 1\ 0\ 1 \end{pmatrix} = \begin{pmatrix} 1\ 0\ 0 \\ 0\ 0\ 0 \\ 0\ 0\ 0 \end{pmatrix} + \begin{pmatrix} 0\ 0\ 1 \\ 0\ 0\ 0 \\ 0\ 0\ 0 \end{pmatrix} + \begin{pmatrix} 0\ 0\ 0 \\ 0\ 0\ 0 \\ 1\ 0\ 0 \end{pmatrix} + \begin{pmatrix} 0\ 0\ 0 \\ 0\ 0\ 0 \\ 0\ 0\ 1 \end{pmatrix}$$

The following procedure generates an optimal decomposition. It closely resembles the classical Quine-McCluskey method [4]. The central idea is to use the merging-rule on symmetric patterns in a way that produces the smallest number of symmetric patterns that reconstructs the original binary matrix.

Decomposition Algorithm for Uniform Cost Model

1. INPUT: The binary matrix B.

2. PREPROCESSING STEP: Compute the base set H for B.

3. Let $W = H$. Mark each element of W *uncovered*.

4. While there exists two mergeable patterns P_1, P_2 in W, such that $P_1 \not\subseteq P_2$ and $P_2 \not\subseteq P_1$:

    • Let $P = P_1 + P_2$.

• Mark P_1 and P_2 *covered*.

• Let $W = W \cup P$.

• Mark P *uncovered*.

End

Let I be the set of all uncovered patterns of W. This is the set of prime implicants for B.

Having computed the set of prime implicants I, we compute the minimal cardinality set $C \subseteq I$ which covers B, or equivalently which covers the base set H. That is, the smallest number of symmetric patterns that composes B:

$$\min_{C \in 2^I} \left(|C| \;\middle|\; \sum_{P \in C} P = \sum_{h \in H} h = B \right)$$

3.3. The Set Covering Problem

Ref. [5] describes the non-weighted case of the set covering problem in the hypergraph setting and presents greedy heuristics. We cast the set covering problem in the form of a 2D matrix (different from the microlaser matrix) in which the m rows in the matrix correspond to elements in the base set H, and the n columns correspond to elements in the prime implicant set I. The problem is to find the set A that contains the fewest columns that intersect each row at least once, taking the weights w_i for each member of the base set into account. The weights are all equal to 1 for the uniform cost model.

In the standard form of any linear programming (LP) problem, the variables are assumed to be non-negative real numbers. In integer linear programming (ILP), the variables are assumed to be non-negative integers. The standard ILP formulation for weighted set-covering [6] is:

Given the weight (cost) vector $W \in R^m$, $A \in \{0, 1\}^{m \times n}$,

$$\min_{X \in \{0, 1\}^m} \left(W^T X \,|\, AX \geq e \right)$$

where $e = \{1\}^m$ (a vector of all 1's). This is simply a formal way of stating that a solution contains the fewest prime implicants and minimizes the overall cost. For the uniform cost model, this reduces to the fewest prime implicants that cover the base set.

In the above setting, $x_i = 1$ means the i^{th} prime implicant is in the covering set C, and $a_{i,j} = 1$ means the j^{th} prime implicant covers the i^{th} element of the base set.

We describe a method for computing the minimal cover even when the costs are not uniform (in which case it is the minimal cost set cover problem). A brief sketch of the strategy is as follows [7]:

1. First relax the ILP to LP. (Relax the integrality constraint.)

2. Add slack variables to A. (Change the inequalities to equalities.)

3. Find a primal cover X^*. This is the subset K of the set of prime implicants I such that:

    - K covers B

    - No proper subset of K covers B

    (The greedy strategy can be adapted here to generate primal covers.)

4. Construct the corresponding basis β to LP. (Every primal cover corresponds to a basis).

5. Invert β to get β'. (This step is highly optimized. Due to the special structure of A and the property of the primal cover, β' could be obtained merely by permuting the rows and the columns of A [7, 8]).

6. Compute the dual solution.

7. Check the dual feasibility:

    - If the solution is dual feasible, then STOP since the present solution is an optimal solution.

    - Otherwise,

(a) Compute the set of indices Q that violate the dual feasibility conditions.

(b) To the present LP add a constraint of the form $\Sigma_{i \in Q} X_i \geq 1$.

(c) Go to the step of computing the primal cover.

Note the following:

- All intermediate solutions are integral.
- The actual computation is in proving the optimality, so heuristics may be appropriate.
- The inversion of the basis matrix β is almost a trivial task [8].

4. Decomposition Algorithm for Non-Uniform Cost Model

In a system, not all symmetric patterns may be equally desirable due to thermal loads or other practical considerations. A symmetric pattern that has dense non-zero clusters gives rise to local hot spots that may need to be reduced to meet system constraints. Here, we present a model that incorporates thermal "bake factor" penalties associated with symmetric patterns.

Let P (= $(p_{i,j})$) be a symmetric pattern, where $w(P)$ is the bake factor penalty associated with P. There is more than one way to measure bake factor. We define two classes of measures:

Measure 1. t-Moment Weight Distribution over Square Templates (of size k):

$$w_k^t(P) = \sum_{x=1}^{n-k+1} \sum_{y=1}^{n-k+1} \left(\sum_{i=x}^{x+k} \sum_{j=y}^{y+k} p_{i,j} \right)^t$$

That is, within a neighborhood of size $k \times k$ on an $n \times n$ microlaser array, add up all of the 1 elements within the neighborhood, raise that sum to the t power, and add it into the total sum $w_k^t(P)$. Variable t determines the degree of variance from the desired value. For example:

$$w_2^3 \begin{pmatrix} 1&0&1&0 \\ 1&0&1&0 \\ 0&0&0&0 \\ 1&0&1&0 \end{pmatrix} = 30 \qquad w_2^3 \begin{pmatrix} 1&1&0&0 \\ 1&1&0&0 \\ 1&1&0&0 \\ 0&0&0&0 \end{pmatrix} = 153$$

Measure 2. Threshold (T) on Square Templates (of size k):

$$w_{k,T}(P) = \sum_{x=1}^{n \pm k+1} \sum_{y=1}^{n \pm k+1} \min\left\{1, \max\left\{0, \sum_{i=x}^{x+k} \sum_{j=y}^{y+k} p_{i,j} \pm T\right\}\right\}$$

That is, $w_{k,T}(P)$ is 0 if the threshold T is not exceeded within any neighborhood, and is greater than 0 by the sum of the amounts that neighborhoods exceed T. For example:

$$w_{2,3} \begin{pmatrix} 1&0&1&0 \\ 1&0&1&0 \\ 0&0&0&0 \\ 1&0&1&0 \end{pmatrix} = 0 \qquad w_{2,3} \begin{pmatrix} 1&1&0&0 \\ 1&1&0&0 \\ 1&1&0&0 \\ 0&0&0&0 \end{pmatrix} = 2$$

Since the set covering method described in the previous section is general enough to take the weights into account, only the first part of the decomposition algorithm needs to be modified for the non-uniform (weighted) cost model:

1. Consider all symmetric patterns included in B.

2. Choose some measure of penalty, such as $w_k^t(P)$ or $w_{k,T}(P)$, and compute the weight of each pattern.

3. Continue the decomposition process as before, by setting up a (larger) set covering problem and solve it by the method sketched in the earlier sections.

5. Exclusive-OR Algebra: Toggling the Patterns

In the previous sections, we assumed that an AND-OR algebra is used. That is, a symmetric pattern is enabled as a collection (a logical OR) of microlasers that are enabled if and only if their corresponding row AND column bonding pads are activated.

Here, we look into a different exclusive-OR (XOR) algebra in which the state of a microlaser is toggled (changed from enabled to disabled, or *vice versa*) if its corresponding row and column bonding pads are activated. Physically, a flip-flop or some other device should be attached to each microlaser so that its state can be retained and the XOR can be determined locally. Except for this assumption, the underlying physical model is the same as for the previous sections.

A summary of the XOR algebra is given in Figure 7. Notice that:

- The two-input XOR function is true (produces a 1) if and only if its inputs are unequal.

- In conventional digital electronics, the AND-OR algebra is related to D flip-flops (data latches), whereas XOR relates to T flip-flops (toggle latches).

- XOR defines an Abelian-group, and along with logical AND defines a finite field (Galois Modulo-2 field). As an example:

$$\begin{pmatrix} 1\,1\,1\,0 \\ 0\,0\,1\,1 \\ 1\,1\,0\,1 \\ 1\,1\,0\,1 \end{pmatrix} = \begin{pmatrix} 1\,1\,0\,1 \\ 0\,0\,0\,0 \\ 1\,1\,0\,1 \\ 1\,1\,0\,1 \end{pmatrix} \oplus \begin{pmatrix} 0\,0\,1\,1 \\ 0\,0\,1\,1 \\ 0\,0\,0\,0 \\ 0\,0\,0\,0 \end{pmatrix}$$

In order to develop an optimal decomposition method for the XOR algebra, we need to borrow an important concept from matrix theory, which is covered in the next section.

6. Singular Value Decomposition

Singular value decomposition (SVD) is a method of matrix decomposition that is appropriate for creating XOR symmetric patterns. We make use of results from classical matrix analysis:

- A matrix $Q \in R^{m \times m}$ is said to be orthogonal if $Q^T Q = I$, where Q^T is the transpose of Q and I is the identity matrix (all 1's along the diagonal and 0's elsewhere).

- If A is a real $m \times n$ matrix, then there exist orthogonal matrices U and V:

$$U = [u_1, u_2, ..., u_m] \in R^{m \times m}$$

$$V = [v_1, v_2, ..., v_n] \in R^{n \times n}$$

such that

$$U^T A V = \Sigma = diag(\sigma_1, \sigma_2, ..., \sigma_p) \in R^{m \times n}$$

$$p = min\{m, n\}$$

where

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_p \geq 0$$

Terminology: σ_i's are the singular values of A and the vectors u_i and v_i are the i^{th} left singular vector and the i^{th} right singular vector, respectively.

Notation: $\sigma_i(A)$ = the i^{th} largest singular value of A.

• If SVD is given by the above definition and we define r by:

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > \sigma_{r+1} = ...\sigma_p = 0$$

then we have:

– r is the rank of A.

– A could be expressed as a weighted linear combination of outer products:

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

7. SVD Based Decomposition Algorithm

In order to translate these results to our problem domain, we make a few observations:

1. XOR with AND defines a field (Galois, Modulo-2 field).

2. Therefore, any result that holds true for a real field applies to a modulo-2 field also.

3. In our case A, U, and V are all binary matrices.

4. If the rank of A is r, then:

$$\sigma_1 = \sigma_2 = \ldots \sigma_r = 1$$

$$\sigma_{r+1} = \ldots \sigma_p = 0$$

5. In particular we are dealing with square matrices, therefore $p = n$ and $A, U, V \in B^{n \times n}$.

6. In the absence of a specialized algorithm for finite fields (in particular for a mod-2 field), we can use the algorithm available for real fields, with a slight modification. The crucial and only change required regards the notion of "+" (addition, in real numbers). For a mod-2 field we use "⊕" (addition, or XOR, as defined in congruent modulo-2), in place of "+".

7. Observe that the minimum number of outer products required to express the given matrix (in XOR algebra) is the same as the rank of the matrix. Therefore, we can break the overall process into two steps. Given a pattern A,

   (a) Compute the rank of A.

   (b) Depending on the complexity of the SVD algorithm, the rank of A, and the dimension of A, decide if the outer product expansion is economical (as opposed to a simple row-by-row XOR decomposition, which is no greater than the number of rows).

8. Refs. [9, 10] describe an algorithm to compute the SVD for real fields.

8. Connection Between AND-OR and XOR Decompositions

In spite of the simplicity of the XOR decomposition, neither the AND-OR nor the XOR decomposition is generally better than the other, in terms of the minimal number of symmetric patterns needed to reconstruct the initial pattern. As an example in which the AND decomposition is worse than the XOR decomposition, consider the following initial pattern:

$$\begin{pmatrix} 1&1&1&0 \\ 0&0&1&1 \\ 1&1&0&1 \\ 1&1&0&1 \end{pmatrix}$$

A minimal AND decomposition requires three symmetric patterns:

$$\begin{pmatrix} 0&0&0&0 \\ 0&0&0&0 \\ 1&1&0&1 \\ 1&1&0&1 \end{pmatrix} + \begin{pmatrix} 1&1&1&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \end{pmatrix} + \begin{pmatrix} 0&0&0&0 \\ 0&0&1&1 \\ 0&0&0&0 \\ 0&0&0&0 \end{pmatrix}$$

In contrast, a minimal XOR decomposition requires only two symmetric patterns:

$$\begin{pmatrix} 1&1&0&1 \\ 0&0&0&0 \\ 1&1&0&1 \\ 1&1&0&1 \end{pmatrix} \oplus \begin{pmatrix} 0&0&1&1 \\ 0&0&1&1 \\ 0&0&0&0 \\ 0&0&0&0 \end{pmatrix}$$

Conversely, as an example in which the XOR decomposition is worse than the AND decomposition, consider the following initial pattern:

$$\begin{pmatrix} 1&1&1&0 \\ 1&1&1&1 \\ 1&1&1&1 \\ 0&1&1&1 \end{pmatrix}$$

A minimal AND decomposition requires two symmetric patterns:

$$\begin{pmatrix} 1&1&1&0 \\ 1&1&1&0 \\ 1&1&1&0 \\ 0&0&0&0 \end{pmatrix} + \begin{pmatrix} 0&0&0&0 \\ 0&1&1&1 \\ 0&1&1&1 \\ 0&1&1&1 \end{pmatrix}$$

In contrast, a minimal XOR decomposition requires three symmetric patterns:

$$\begin{pmatrix} 1&1&1&0 \\ 1&1&1&0 \\ 1&1&1&0 \\ 0&0&0&0 \end{pmatrix} \oplus \begin{pmatrix} 0&0&0&0 \\ 0&1&1&1 \\ 0&1&1&1 \\ 0&1&1&1 \end{pmatrix} \oplus \begin{pmatrix} 0&0&0&0 \\ 0&1&1&0 \\ 0&1&1&0 \\ 0&0&0&0 \end{pmatrix}$$

9. Discussion

Although the matrix addressable approach scales more gracefully in terms of pin count, the drive requirements per pin are greater, since an entire row or column may need to be enabled, unless this constraint is addressed by taking bake factor into account. The effect of taking bake factor into account may increase the number of symmetric patterns in the decomposition, however. Further, the total energy expended with these methods of decomposition may be greater than for individual addressing. For example, the initial binary matrix shown in Figure 3 has only 16 enabled microlasers, but the decomposition into symmetric patterns requires 19 enabled microlasers, although not all at once.

In comparing the matrix addressable approach to the individually addressable approach, the question arises as to how much more time it takes to set up a matrix addressable array than it takes for an individually addressable array (which takes only one step regardless of the initial pattern). Figure 8 shows a plot for an 8×8 array in which the number of enabled microlasers varies from 1 to 63, and 1000 random patterns are generated for each sample point. For an individually addressable array, the number of activation patterns is only 1 for each case since an arbitrary pattern can be applied directly. For a matrix addressable approach, the best case is a single activation pattern if the target pattern is symmetric, and the worst case is 8 activation patterns (if eight enabled microlasers are on a diagonal with disabled microlasers elsewhere, for instance). As shown in the plot, the average for any number of enabled microlasers is always better than the worst case of 8. If we take the average of the averages, assuming that any number of enabled microlasers is equally likely, then the expected number of activation patterns is approximately 5.6.

The expected value (5.6) is better than the worst case (8 activation patterns), but still requires a relatively large number of activation patterns. Depending on the application, however, the target patterns may not be randomly distributed as they are for the plot shown in Figure 8. In fact, truly random distributions are a rare behavior in communication and computation in general, and we expect there will be a great deal of locality in applications for microlasers, especially for displays, in which case simple updates to the existing patterns may further reduce the number of activation patterns. Since locality changes over time, the gains made possible by locality will not be greatly offset by the effects of laser aging, which is sometimes handled by removing locality from the data stream (which is not necessary for this application).

The method as presented relies on set coverage, which is a computationally hard problem (of the class NP) and is therefore not scalable to large arrays. However, the decomposition problem is tractable for small arrays (such as 8×8), and simple changes to the wiring pattern can ensure that a large array behaves as an array of tractable sized arrays. For example, the row-column addressing scheme can be changed to a two_row-column addressing scheme with a nearly equivalent pin count but with a greatly simplified decomposition. This can be done without increasing the drive requirements for the pins by applying bake factor constraints to the decomposition process.

10. Conclusion

We presented an optimal method for decomposing arbitrary 2D patterns into the minimal set of patterns that can be applied time sequentially to a matrix addressable microlaser array. Two methods are explored: AND-OR, in which the target pattern is built up from a cleared array in an additive manner; and XOR, in which the target pattern is built up from a cleared array in an exclusive-OR (toggled) manner. Neither approach is clearly better than the other when starting from a cleared array. A study of time sequential buildup for an 8×8 array shows that the expected number of activation patterns for completely randomly occurring patterns is 5.6, which is better than the worst case of 8. An open issue is how the expected value improves for real patterns using updates to the existing patterns.

11. References

[1] Jewell, J. L., J. P. Harbison, A. Scherer, Y. H. Lee, and L. T. Florez, "Vertical-cavity surface-emitting lasers: Design, growth, fabrication, characterization," *IEEE J. Quan. Electron.*, **27**, 1332-1346, (Jun. 1991).

[2] Morgan, R. A., "Vertical Cavity Surface Emitting Lasers," 1993 Annual SPIE meeting, SPIE vol. 1992, Paper no. 9, (SPIE, Bellingham, Washington), (1993).

[3] Nahata, H. R., and M. J. Murdocca, "Decomposition Method for Matrix Addressable Microlaser Arrays," in *Optical Computing*, vol. 10, 1995 OSA Technical Digest Series, (Optical Society of America, Washington, DC, 1995), Salt Lake City, pp. 26-28.

[4] McCluskey, E. J., *Introduction to the Theory of Switching Circuits*, McGraw-Hill, (1965).

[5] Cormen, T. H., R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*, The MIT Press, McGraw-Hill, (1990).

[6] Balas, E. and M. Padberg, “Set Partitioning,” SIAM Review, vol. 18, pp. 710-760, (1976).

[7] Balas, E., “Some valid inequalities for the set partitioning problem,” *Studies in Integer Programming*, edited by I. L. Johnson, B. Korte, and G. L. Nemhauser, North Holland, pp. 13-47 (1977).

[8] Bellmore and Ratliff, “Set covering and Involutary bases,” *Management Science*, vol. 18, pp. 194-206, (1971).

[9] Golub, G. H. and C. F. van Loan, *Matrix Computations*, 2/e, The Johns Hopkins University Press, (1989).

[10] Strang, G., *Linear Algebra and its Applications,* Harcourt, Brace, and Jovanovich Publishers, (1980).

Acknowledgments

Dr. Ravindra Athale and Dr. Kannan Raj of George Mason University are gratefully acknowledged for connecting the pattern decomposition methods to singular value decomposition. Dr. Konrad Zürl of the University of Erlangen is gratefully acknowledged for drawing our attention to the greater drive requirements of matrix addressable arrays over individually addressable arrays. This work was supported by the National Science Foundation on grants MIP 92-24707 and ECS 93-12625.

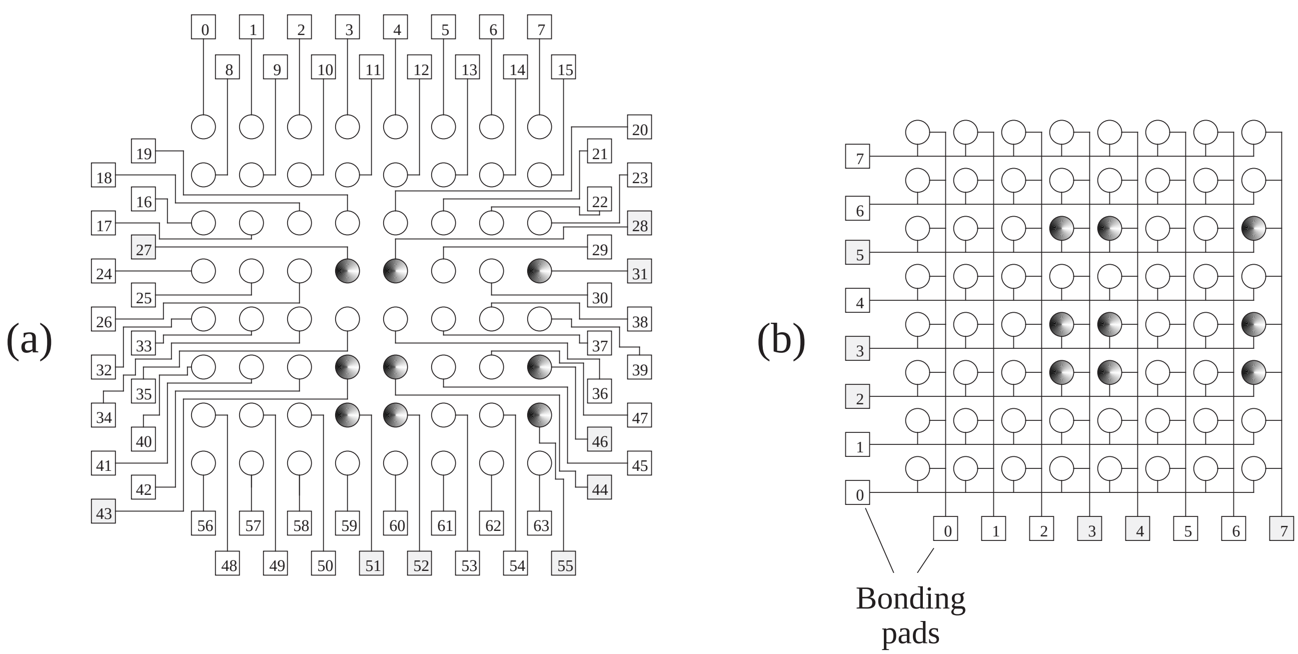
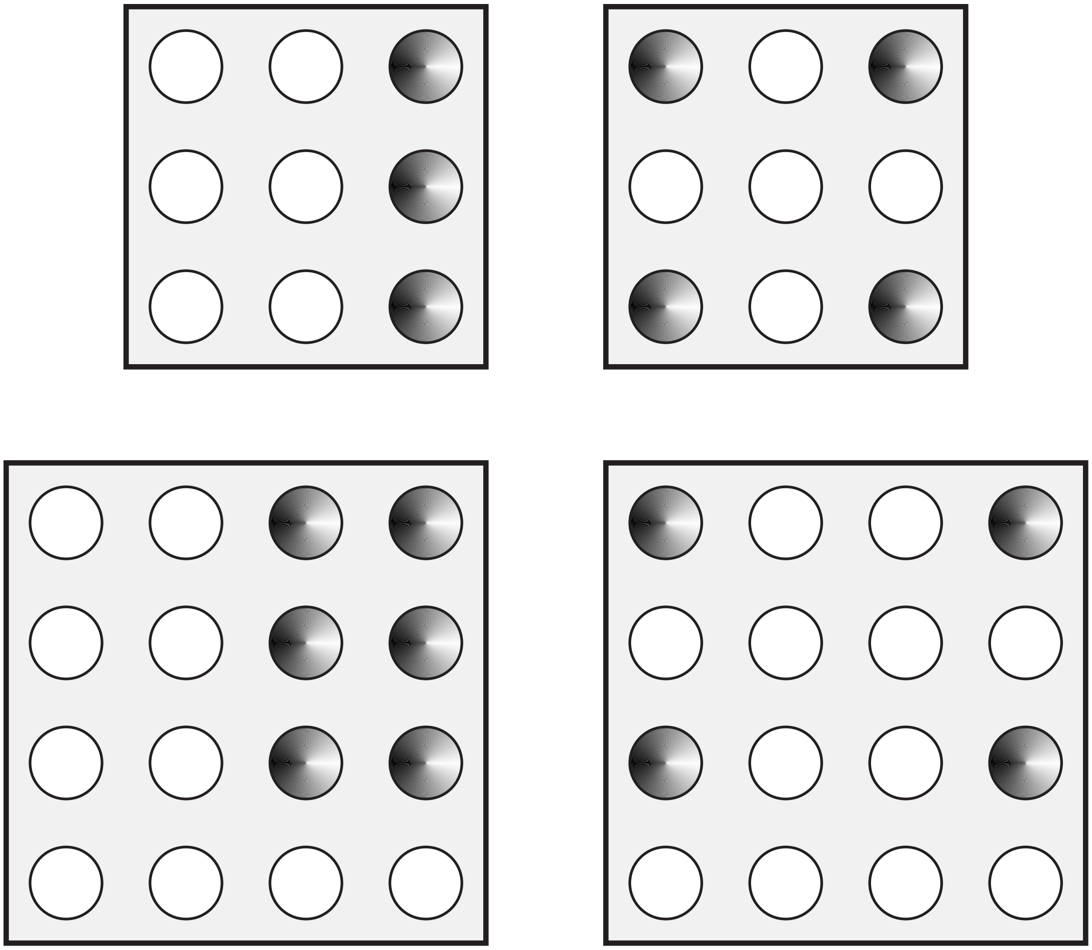


Figure 1: *(a) Individually addressable microlaser array; (b) matrix addressable microlaser array.*



(a)

(b)

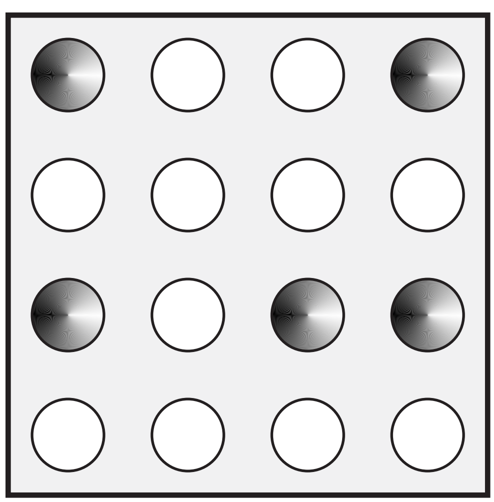


Figure 2: *(a) Examples of symmetric patterns; (b) a binary matrix that is not a symmetric pattern.*

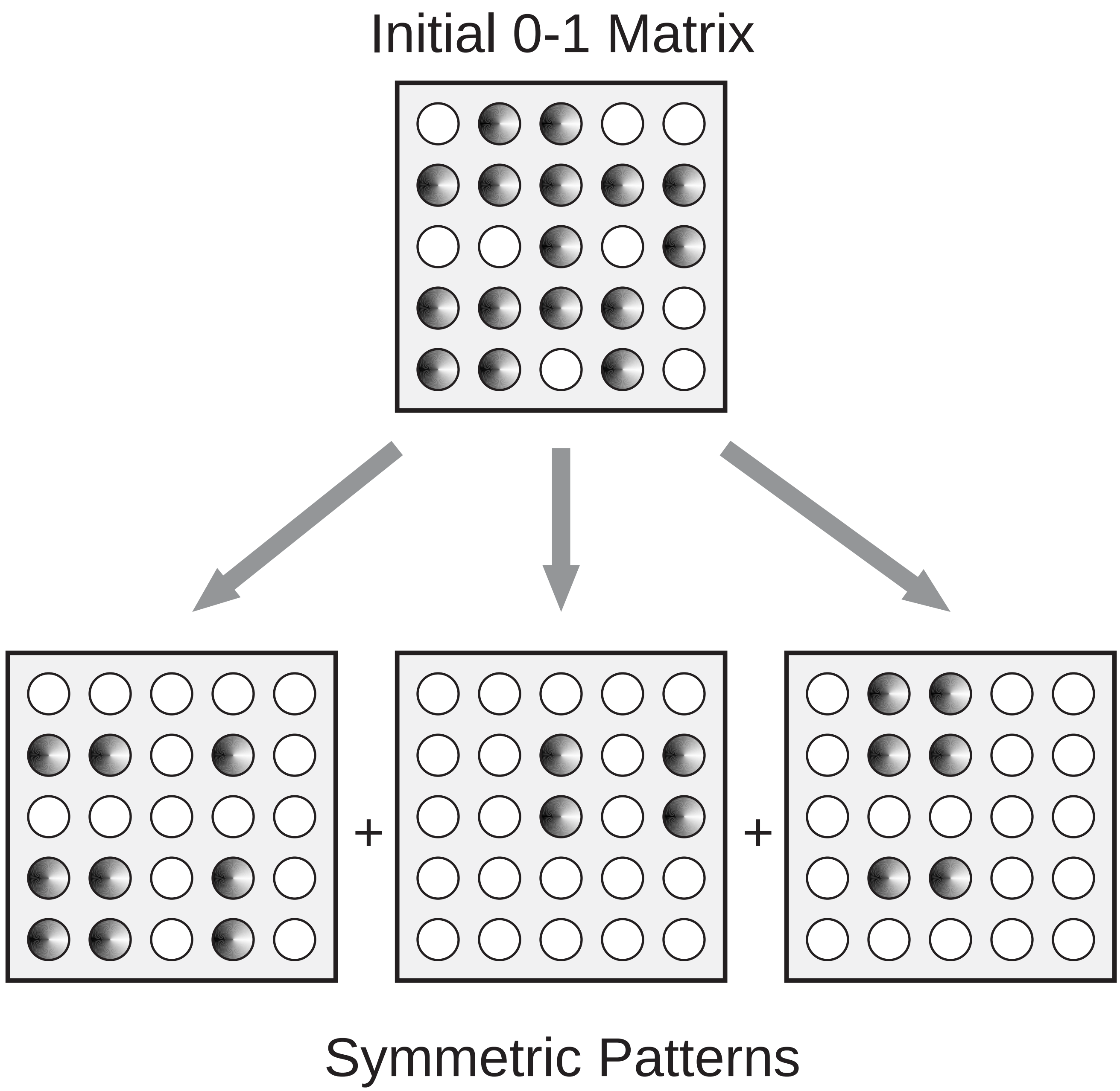


Figure 3: *An example of decomposition of a non-symmetric pattern into three symmetric patterns.*

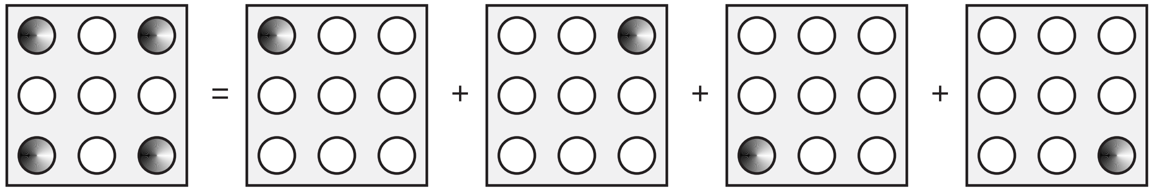


Figure 4: *Existence of decomposition shown as a sum of symmetric patterns having only a single non-zero point.*

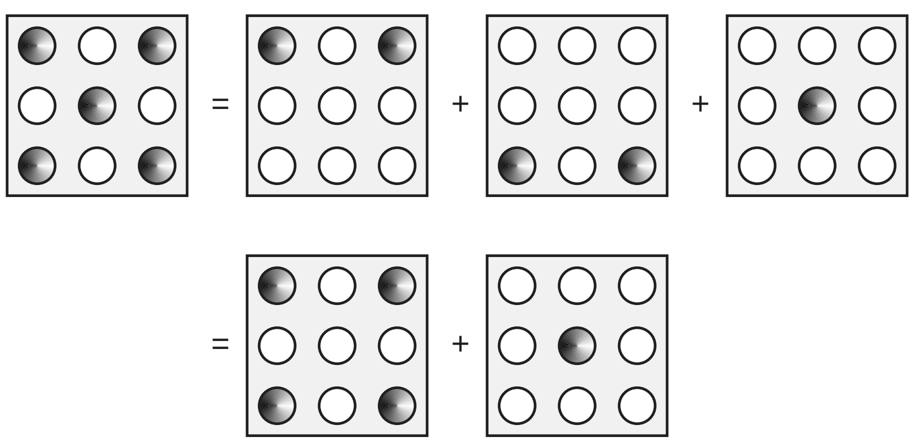


Figure 5: *A decomposition need not be unique.*

a	b	$a \cdot b$ (AND)	$a+b$ (OR)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Figure 6: *AND-OR Boolean algebra.*

a	b	$a \oplus b$ (XOR)
0	0	0
0	1	1
1	0	1
1	1	0

Figure 7: *XOR Boolean algebra.*

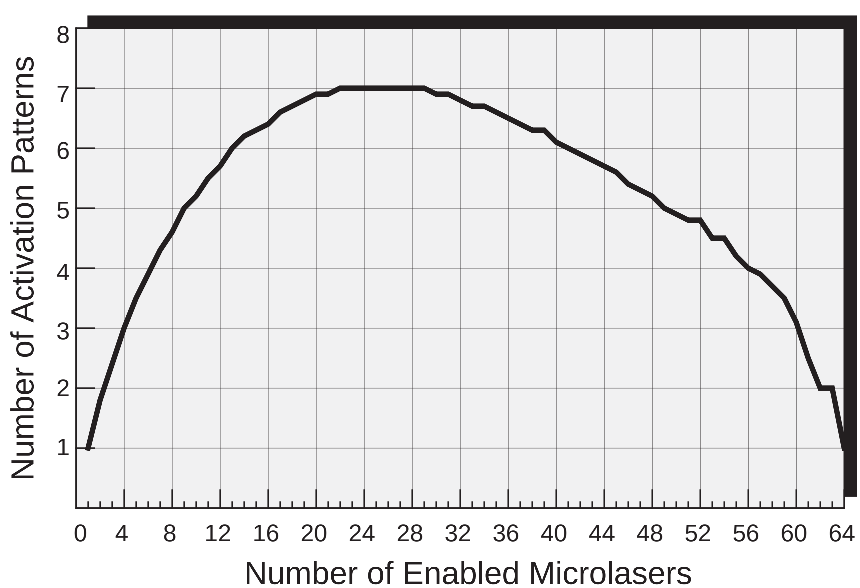


Figure 8: *Decomposition results for randomly generated patterns on an 8×8 matrix. 1000 sample points are taken for each number of enabled microlasers (1 – 63).*