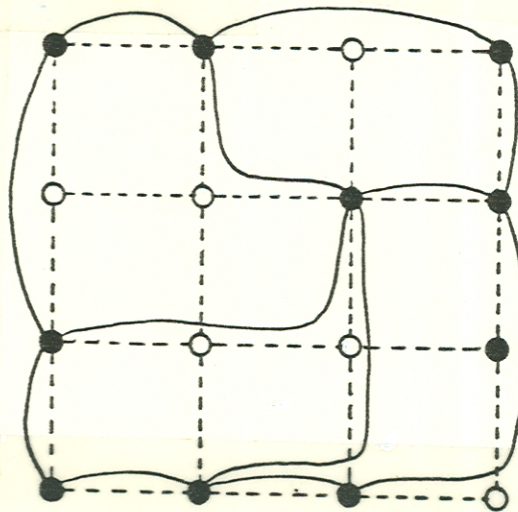MIT/LCS/TM-236

WAFER - SCALE INTEGRATION OF SYSTOLIC ARRAYS

Frank Thomson Leighton and Charles E. Leiserson

February 1983

Wafer-Scale Integration of Systolic Arrays

Frank Thomson Leighton

Mathematics Department
and
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Charles E. Leiserson

Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Abstract: VLSI technologists are fast developing *wafer-scale integration*. Rather than partitioning a silicon wafer into chips as is usually done, the idea behind wafer-scale integration is to assemble an entire system (or network of chips) on a single wafer, thus avoiding the costs and performance loss associated with individual packaging of chips. A major problem with assembling a large system of microprocessors on a single wafer, however, is that some of the processors, or *cells*, on the wafer are likely to be defective. In the paper, we describe practical procedures for integrating wafer-scale systems "around" such faults. The procedures are designed to minimize the length of the longest wire in the system, thus minimizing the communication time between cells. Although the underlying network problems are NP–complete, we prove that the procedures are reliable by assuming a probabilistic model of cell failure. We also discuss applications of this work to problems in VLSI layout theory, graph theory, fault-tolerant systems and planar geometry.

Key Words: channel width, fault-tolerant systems, probabilistic analysis, spanning tree, systolic arrays, travelling salesman problem, tree of meshes, VLSI, wafer-scale integration, wire length.

1. Introduction

VLSI technologists are fast developing *wafer-scale integration* [30]. Rather than partitioning a silicon wafer into chips as is usually done, the idea behind wafer-scale integration is to assemble an entire system (or network of chips) on a single wafer, thus avoiding the costs and performance loss associated with individual packaging of chips. A major problem with assembling a large system of microprocessors on a single wafer, however, is that some of the processors, or *cells*, on the wafer are likely to be defective. Thus a practical procedure for integrating wafer-scale systems must have the ability to configure networks "around" such faults.

This paper considers a variety of problems involving the construction of systolic arrays [15]. Systolic arrays are a desirable architecture for VLSI because all communication is between nearest neighbors. In a wafer-scale system, however, all the nearest neighbors of a processor may be *dead,* and thus the prime advantage of adopting a systolic array architecture may be lost if a long wire connects adjacent processors. In general, the longest interconnection between processors will be a communication bottleneck in the system. Of the many possible ways in which the *live* cells on a wafer can be connected to form a systolic array, therefore, the one that minimizes the length of the longest wire is most desirable from a computational standpoint because communication overhead is least.

To illustrate the subtleties inherent in configuring systolic arrays, consider the problem of constructing a linear (i.e., one-dimensional) array using all of the live cells in an N-cell wafer. Unfortunately, if we wish to minimize the length of the longest wire, the problem is NP-complete [11]. Even more discouraging is that there are some arrangements of live and dead cells for which even the optimal linear array has unacceptably long wires. Thus optimal solutions—even if they could be found quickly—are not always practical.

By assuming a probabilistic model of cell failure, however, many positive results can be proved. For example, Figure 1 illustrates a possible solution to the problem of connecting the live cells of a wafer into a linear systolic array. The live cells, which are denoted by small squares, are connected together, one after another, in a snake-like pattern. Dead cells, denoted by **X**'s, are skipped over. With probability $1 - O(1/N)$, the length of the longest wire is $O(\lg N)$, where N is the number of cells in the wafer and where each cell independently has a fifty percent chance of failure.

Figure 1. A simple means of constructing a linear systolic array from the live cells on a wafer.

This bound comes from the observation that the length of the longest wire that connects two cells in the array is just the length of the longest sequence of dead cells in the snake-like string. For a given set of k cells, the probability that all are dead is $1/2^k$, and thus the probability that any set of $2\lg N$ cells are dead is $1/N^2$. The chances are, therefore, less than one in N of having to skip more than $2\lg N$ cells in the entire snake-like path of length N, and thus with probability $1 - O(1/N)$ the maximum wire length is $O(\lg N)$.

To say that with probability $1 - O(1/N)$ the maximum wire length is $O(\lg N)$ is a substantially stronger statement than saying that the expected maximum wire length is $O(\lg N)$. Not only is the expected maximum wire length $O(\lg N)$, but the chances of it being much larger are miniscule. Furthermore, the probability can be made much higher. For example, the probability of having to skip more than $3\lg N$ dead cells in the entire snake-like path is less than one in N^2. A small adjustment to the constant within the Big Oh results in a much higher probability.

Not surprisingly, there are algorithms which, under similar assumptions of cell failure, produce far better results than the algorithm illustrated in Figure 1. For example, we will describe in Section 3 another simple procedure which, with high probability, constructs a linear array using wires of length $O(\sqrt{\lg N})$. We will also show that, up to the leading constant, the algorithm is the best possible of its kind. By relaxing the constraint that *all* live cells be connected into the linear array, however, we can do much better. In fact, we will also show in Section 3 that with high probability, a linear array containing any constant fraction (less than one) of the live cells on an N-cell wafer can be constructed using wires of at most constant length.

Although there are numerous uses for linear systolic arrays [22], two-dimensional systolic arrays are also important. Not only can the two-dimensional array be used as a powerful communications structure for parallel computation [15], but it can also serve as an all-purpose structure in which arbitrary networks can be embedded [2, 19, 21, 41, 43]. As one might expect, the problem of constructing a two-dimensional array from the live cells of a wafer is more difficult than the corresponding problem for linear arrays. Specifically, Section 4 contains a proof that with high probability a two-dimensional array that uses any constant fraction of the live cells must have wires of length $\Omega(\sqrt{\lg N})$.

Although we do not know how to construct two-dimensional arrays from most of the live cells using wires of length $O(\sqrt{\lg N})$ or channels of constant width, we can come close. We show in Section 6 that with high probability, a two-dimensional array can be constructed on an N-cell wafer using:

1) all the live cells with wires of length $O(\lg N \lg\lg N)$ and channels of width $O(\lg\lg N)$,
2) any constant fraction less than one of the live cells with wires of length $O(\sqrt{\lg N} \lg\lg N)$ and channels of width $O(\lg\lg N)$, and
3) at least $\Omega(1/\lg\lg^2 N)$ of the live cells with wires of length $O(\sqrt{\lg N})$ and channels of width 1.

The remainder of the paper is divided into seven sections. Section 2 more formally describes our model for wafer-scale integration and discusses the practicality of the modeling assumptions. The algorithms for constructing linearly connected systolic arrays are presented in Section 3. Section 4 contains the lower bound result for wire length in two-dimensional systolic arrays. In Section 5 we present a worst-case (nonprobabilistic) upper bound on the channel width necessary

to configure a two-dimensional array. This result has application to the fault-tolerant encoding of two-dimensional arrays in complete binary trees [31]. Section 6 gives algorithms for constructing two-dimensional arrays in the probabilistic model. In Section 7, we mention some related problems in geometric complexity and graph theory. The related problems are nice theoretically in that some of them have tight upper and lower bounds. They also suggest a wealth of interesting questions concerning the design of fault-tolerant systems. We conclude the paper with some additional remarks in Section 8.

2. The wafer-scale model

Laser-programming the interconnect of a wafer is a promising means of achieving wafer-scale integration. This technology was pioneered at IBM [24] and pursued in the direction of wafer-scale integration at MIT Lincoln Laboratory [30]. Figure 2 shows a scanning electron microscope photograph of a portion of a wafer with programmable interconnect. Laser welds can be made between two layers of metal, and by using the beam at somewhat higher power, wires can be cut. Defective components can thus be avoided by programming the interconnect to connect only the good components.
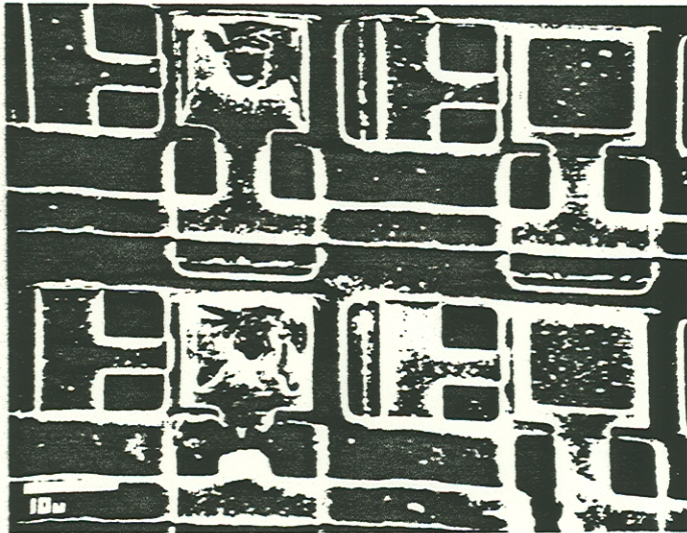
Figure 2. A close-up of laser-programmable interconnect.

Figure 3 shows a typical organization of a wafer-scale system with programmable interconnections. The components are organized as a matrix of cells, and between the cells are channels through which the interconnect runs. Figure 4 is a close-up of the channel structure. At the intersection of a horizontal and vertical channel, laser-programmable connections can make a horizontal and a vertical wire electrically equivalent. Between two cells, connections can be made from the wires in the channel to the inputs and outputs of the two cells. Given that the interconnect is programmable, we shall adopt a usage of the term "wire" to mean an electrically equivalent portion of the programmable interconnect.
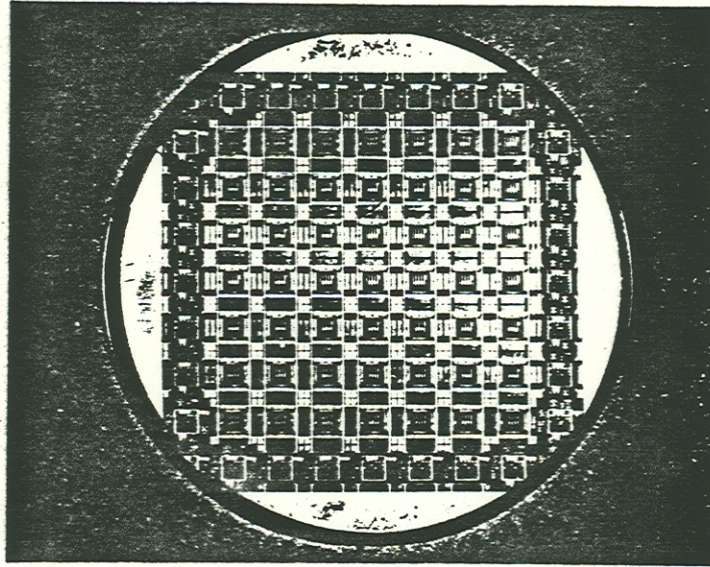
Figure 3. A wafer-scale system of cells and programmable interconnect.
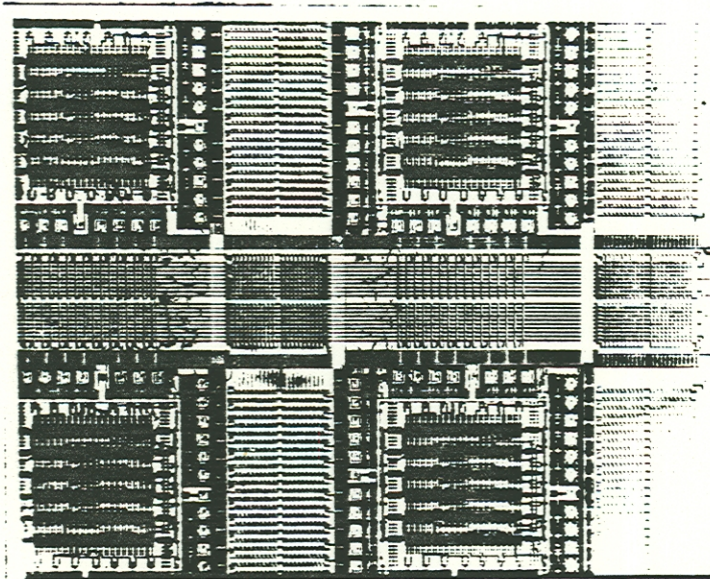

Figure 4. The channel structure of a wafer-scale system.

The preassignment of wire segments to layers such that wires in one layer run horizontally and the other vertically is called *Manhattan* wiring [16]. This wiring model has been studied extensively, but in this paper the details of the wiring are not the central issue. It will be sufficient to understand one fact about Manhattan wiring. The width of a channel need only be a constant factor larger than the maximum number of wires that occupy any portion of the channel.

4

A natural question to ask about the use of programmable interconnections to avoid defective cells is, *"If cells are unreliable, why might not the interconnect fail also?"* The answer is that, indeed, interconnect does fail. But the reliability of the interconnect is much higher than the reliability of the cells. The interconnect in the MIT Lincoln Laboratories project, for example, takes three masking steps to fabricate, but manufacturing the active devices requires well over a dozen steps. This project is targeting yields of fifty percent for cells and over ninety-five percent for wires. And even if a wire fails at one point, it is often possible to break it into two usable pieces.

In this paper we shall assume that the interconnect has sufficient redundancy so that the inability to interconnect cells arbitrarily is a rare phenomenon. In this sense, we are making the same assumption that is used to substantiate redundancy in any fault-tolerant system. The idea is not that the system will be completely reliable, but that its failure will depend on the failure of the most reliable component instead of the least reliable component.

Another assumption that must be examined more closely is that the probability of cell failure is independent and the same for all cells. Failures can be attributed to one of two causes—materials defects during manufacturing, and mask misalignment. Materials defects are spread uniformly, but the size of the region affected by a defect is a separate random variable. This means that if one point on the wafer is flawed, neighboring points are also likely to be flawed. Nevertheless, independence of cell failures is quite a reasonable assumption because the area of a cell is substantially larger than the expected area of a defect.

Mask misalignment is a somewhat more serious problem with respect to our modeling assumptions. The reason is that misalignment is a global failure mode. Misalignment due to translation of the axes of one mask relative to the others poses no real problem in terms of the modeling assumptions, however, because the effect is the same for all cells. The real problem is misalignment due to angular rotation of one mask with respect to the others. Those cells near the center of rotation are much more likely to be good than those far from the center. Experimental evidence indicates, however, that the effects from angular rotation that cannot be accounted for by our model are minimal.

The two cost functions we shall examine in this paper are *channel width* and *maximum wire length*. Minimizing channel width is important because the available wafer area is essentially fixed. If the channel width is large, the size of the system, and hence its functionality, is reduced. In addition, large channel widths often lead to long wires, and minimizing the length of the longest wire is our other cost criteria.

Minimizing the length of the longest wire in a wafer-scale system is important because communication delays can be the limiting factor of the performance of the system. Since both resistance and capacitance increase with the length of wire, the time required to drive a wire can grow as fast as the square of the length of the wire [27]. (See [4] for a discussion of propagation delays through wires.) In particular, a designer that chooses a two-dimensional systolic array architecture is counting on low overhead for communication, and will not want communication down a long wire to degrade the performance of the system. Furthermore, for reasons of electrical correctness, cells must be designed with signal buffers capable of driving the maximum length wire. Since the size of buffers varies with the size of the load being driven, substantial area in a cell can be saved if the maximum length wire is known to be short. As was argued previously, this savings in area translates to larger systems with greater functionality.

Throughout the paper, we will consider cells which occupy an s-by-s square region on the wafer and which have (independently) a probability p of failure. Unless specifically stated to the contrary, we will assume for simplicity that $s = 1$ and $p = 1/2$. As we will later observe, these restrictions have little bearing on the analysis. In addition, we will use the term *"high probability"* to mean *"with probability at least $1 - O(1/N)$,"* where N is the number of cells on the wafer.

We conclude this section with a simple result that places the rest of this paper in a proper context. Given a circuit composed of active components and wires, it is possible to construct a wafer of not much more area (asymptotically) which is fault tolerant. If there are N active components, expand the layout of the circuit in each dimension by $c\sqrt{\lg N}$, where c is a constant chosen large enough that $2\lg N$ copies of a given active component fit in the space designated to that component in the original circuit. The probability that every one of the $2\lg N$ copies is bad is $1/N^2$, and thus with high probability, one of the copies of every component is good. It only remains to hook them up in the space left for wires.

This scheme works even if components are different. The results in this paper are better for systolic arrays, however, because we can utilize substantially more of the live cells at less cost. Since the number of cells on a wafer might typically be between 100 and 1000, $\lg N$ is a considerable fraction of N. Some of our algorithms use all of the live cells, and others use a considerable proportion.

3. Wafer-scale integration of linearly connected systolic arrays

The snake-like scheme described in the introduction connects with high probability all the live cells on an N-cell wafer into a linear array with wires of length at most $O(\lg N)$. This section substantially improves and generalizes this result. We commence by showing that this bound can be improved to $O(\sqrt{\lg N})$, which is optimal to within a constant factor.

Theorem 1. *With probability $1 - O(1/N)$, the live cells on an N-cell wafer can be connected in a linear array using wires of length $O(\sqrt{\lg N})$. Up to the leading constant, this bound is the best possible.*

Proof. We first show how to construct a linear array using wires of length $O(\sqrt{\lg N})$. Partition the wafer into square regions containing $2\lg N$ cells each as is shown by the dashed lines in Figure 5. The probability that each of the $2\lg N$ cells are dead in one or more of the squares is at most

$$\frac{N}{2\lg N}2^{-2\lg N} = \frac{1}{2N\lg N},$$

which is less than $1/N$. Thus with probability $1 - O(1/N)$, each of the squares contains at least one live cell.

Construct a linear array out of the live cells in each square using the "transpose" of the algorithm from Section 1, except that when an empty column is encountered, the column is skipped. In Figure 5, these connections are shown with solid lines. Since any pair of cells in the same square can be linked with a wire of length at most $2\sqrt{2\lg N}$, the wires in each array have length $O(\sqrt{\lg N})$. Next, add wires, shown by dotted lines in the figure, which connect the small arrays into one large array. Because each region contains at least one live cell, these connections
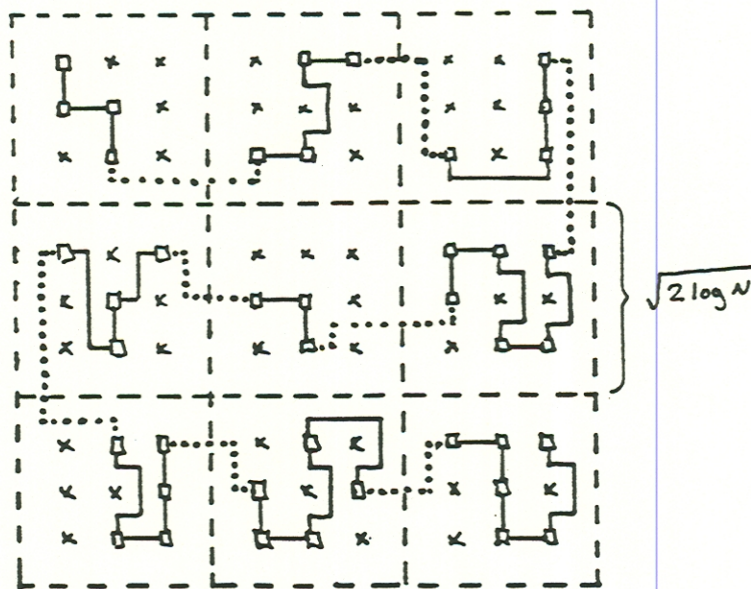
Figure 5. A scheme for constructing linear arrays from all live cells on a wafer with wires of length $O(\sqrt{\lg N})$ and constant channel widths.

can be made with wires of length at most $3\sqrt{2\lg N}$. Thus every wire in the completed linear array has length $O(\sqrt{\lg N})$ with high probability.

That the bound cannot be improved by more than a constant factor is due to the observation that with high probability, some live cell will be at the center of a region of $\Omega(\lg N)$ dead cells. Thus a wire of length $\Omega(\sqrt{\lg N})$ will be required to link the isolated live cell to any other live cell. To demonstrate this bound more formally, we again partition the wafer into square regions, but this time the squares are rotated by forty-five degrees in the plane to form diamond-shaped regions containing $\lg N - 2\lg\lg N$ cells each, as is shown in Figure 6.
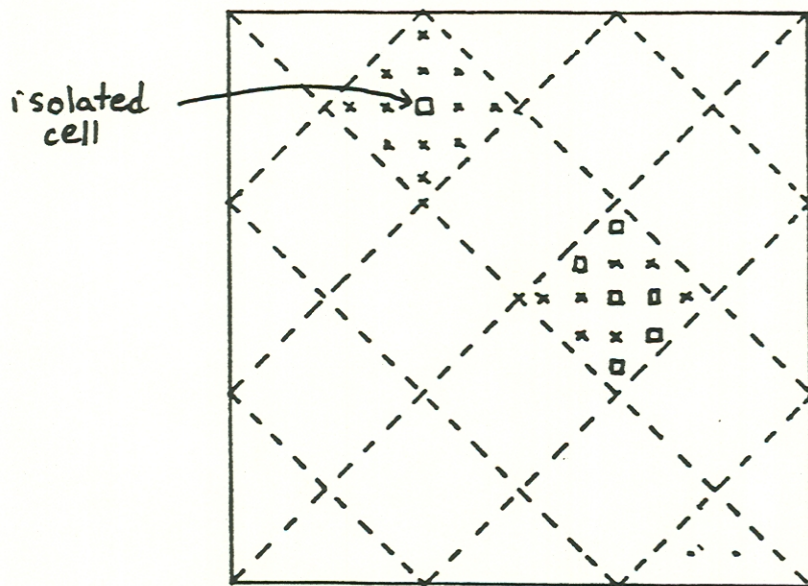
Figure 6. An example of an isolated cell.

Suppose a linear array can be constructed using wires of length at most $\sqrt{\frac{1}{2}\lg N} - 1$. Then in any given diamond, the center cell is not the only live cell in the diamond. The probability that every diamond avoids this condition is at most

$$\left(1 - 2^{-\lg N + 2\lg\lg N}\right)^{\frac{N}{\lg N - 2\lg\lg N}} = \left(1 - \frac{\lg^2 N}{N}\right)^{\frac{N}{\lg N - 2\lg\lg N}}$$

$$\leq e^{\left(-\frac{\lg^2 N}{N}\right)\left(\frac{N}{\lg N - 2\lg\lg N}\right)}$$

$$= e^{-\frac{\lg^2 N}{\lg N - 2\lg\lg N}}$$

$$\leq e^{-\lg N}$$

$$\leq \frac{1}{N}.$$

Thus the probability that the optimal linear array has a wire of length $\Omega(\sqrt{\lg N})$ is at least $1 - O(1/N)$.∎

If all the cells are incorporated in a linear array, then the maximum wire length is $\Theta(\sqrt{\lg N})$ with high probability. But the proof of the lower bound suggests that isolated cells induce the long wires. Instead of insisting that *all* live cells be incorporated in the linear array, suppose we only require that *most* of the live cells be included. A linear array that incorpoates most of the live cells can be constructed with constant-length wires. The proof is indirect, and depends on the following lemma. (The lemma is essentially equivalent to the result of Sekanina [36] which states that the cube of a nontrivial connected graph always has a Hamiltonian circuit. This result was later reproved by Karaganis [12] and Rosenberg and Snyder [35].)

Lemma 2. *A spanning tree T with maximum wire length L can be transformed into a linear array with maximum wire length $6L$.*

Proof. We show that, without regard for wire widths, the linear array can be constructed using wires of length $3L$ by tracing over wires in T no more than twice. The larger $6L$ bound comes because the channel widths need to be doubled to accomodate the extra wires.

Choose a node v to be the root of T, and let $T_1, T_2, \ldots, T_m$ be the subtrees of v as is shown in Figure 7. (Degenerate cases not like Figure 7 are easily handled, but we do not include the details here.)
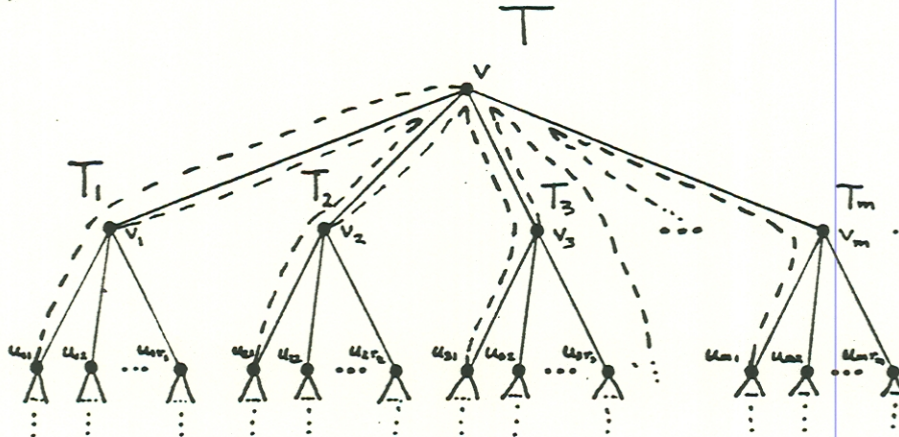

Figure 7. Constructing a linear array from a spanning tree.

Assume as an inductive hypothesis that we have constructed linear arrays on the nodes of $T_1, T_2, \ldots, T_m$ such that no wire has length greater than $3L$, and so that the end points of the array in T_i are v_i and u_{i1} for $1 \leq i \leq m$. Join the arrays in the subtrees by adding the following wires: (v, u_{11}), (v_1, u_{21}), (v_2, u_{31}), ..., (v_{m-1}, u_{m1}). (These wires are shown as dashed lines in Figure 7.) Each of these wires has length at most $3L$, and the resulting network is a linear array on the nodes of T with endpoints v and v_m, which completes the induction. For completeness, we remark that the basis of the induction is easily verified. ∎

The problem of constructing a linear array with constant maximum wire length that contains most of the live cells has now been reduced to the problem of constructing a spanning tree with constant maximum wire length that contains most of the live cells. The next lemma shows that such a spanning tree can be formed with high probability.

Lemma 3. *There exists a positive constant c such that for any d (which might be a function of N), with probability $1 - O(1/N)$, at least $1 - O(2^{-cd^2})$ of the live cells on an N-cell wafer can be connected in a spanning tree using wires of length at most d. Up to constants, this is the best possible bound.*

Proof. We first show that up to constants, the bound is the best that one could hope for. In fact, we show something stronger—that for any constant $\hat{c} > 2$ with probability $1 - O(1/N)$, no more than $1 - O(2^{-\hat{c}d^2})$ of the live cells on an N-cell wafer can be connected in *any* network using wires of length at most d. The proof is based on showing that with high probability, there are $\Omega(N/d^2 2^{2d^2})$ live cells, each of which is located at the center of a region of dead cells whose radius is at least d.

Partition the wafer into diamond-shaped regions as was done in Figure 6 to prove the lower bound of Theorem 1, except make the size of each region be $2d^2$ cells. The probability that any particular region consists of an isolated live cell at the center of $2d^2 - 1$ dead cells is 2^{-2d^2}. The probability that T or fewer of the $N/2d^2$ regions are like this is thus

$$\sum_{x=0}^{T} \binom{N/2d^2}{x} \left(2^{-2d^2}\right)^x \left(1 - 2^{-2d^2}\right)^{\frac{N}{2d^2}-x} \leq \left(1 - 2^{-2d^2}\right)^{\frac{N}{2d^2}} \sum_{x=0}^{T} \frac{\left(\frac{N}{2d^2}\right)^x 2^{-2d^2 x}}{\left(1 - 2^{-2d^2}\right)^x x!}$$

$$\leq e^{-\frac{N}{2d^2 2^{2d^2}}} \sum_{x=0}^{T} \frac{\left(N/d^2 2^{2d^2}\right)^x}{x!}.$$

When T assumes the value $N/8d^2 2^{2d^2}$, the largest term in the series occurs for $x = T$, and thus the preceding expression can be bounded above by

$$\frac{(T+1)}{T!} e^{-\frac{N}{2d^2 2^{2d^2}}} \left(\frac{N}{d^2 2^{2d^2}}\right)^T = O(1/N).$$

In order to prove the upper bound, consider the graph whose vertices are live cells on the wafer and whose edges connect cells which are within distance d of each other on the wafer. In what follows, we will show that there is one *main connected component* in this graph, and that

the total size of all other *isolated components* is a small fraction of N. More specifically, we will show that there exist constants c and c' such that the probability that more than $c'2^{-cd^2}N$ live cells are isolated is $O(1/N)$.

The approach will be to find a crude upper bound on the number of paths which can define the *outer boundary* of an isolated region. (See Figure 8.) For any given path which defines the outer boundary of a potentially isolated region, we will show that the probability is very small that all the cells are dead in the corresponding width-d *boundary region*. In particular, the longer the path that defines a potentially isolated region, the smaller the probability that the region is actually isolated.
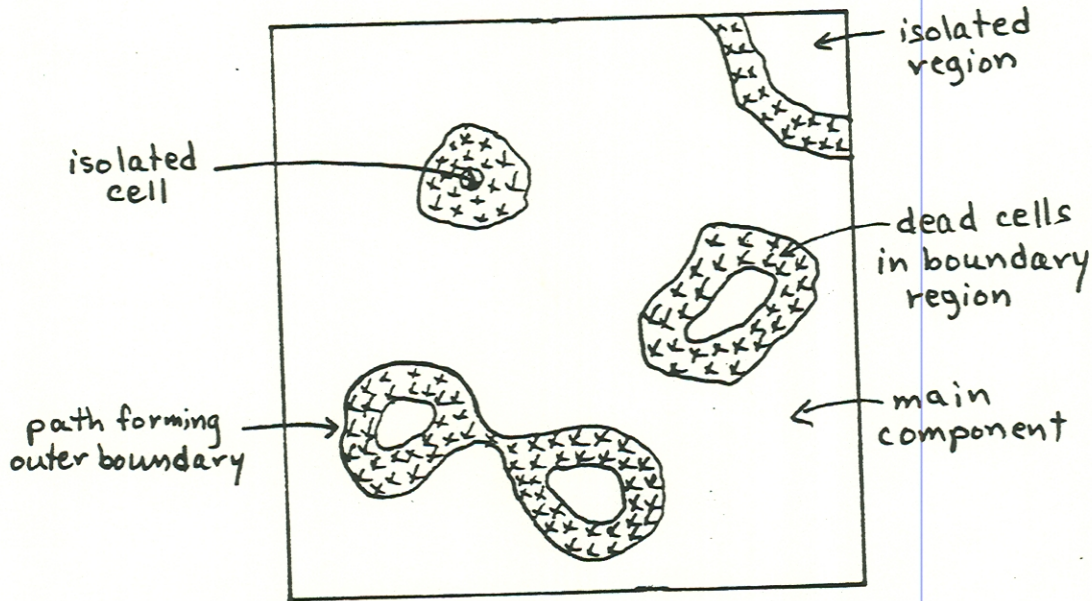
Figure 8. Examples of isolated regions.

Because there are N positions at which a path can start and at most four ways it can continue at each step, there are at most $N4^r$ paths consisting of r consecutive cells. Thus there are at most

$$\binom{N4^r}{k} \leq \frac{N^k 4^{rk}}{k!}$$

sets of k different paths of length r.

The number of paths of length r is quite a formidable number, and at first glance it seems unlikely that our approach will work. The probability is quite small, however, that each of k given paths actually defines a region which both is isolated and contains at least one live cell. For a region to be isolated, its boundary region must consist of at least $rd/8$ dead cells, where $r \geq d$. The probability that all $krd/8$ cells are dead in the boundary regions of k potentially isolated regions with a boundary of length r is $2^{-krd/8}$. Thus the probability that there are actually k isolated regions, each containing one or more live cells, with outer boundaries of length r is at most $(N^k 2^{2rk-krd/8})/k!$, which for $k \geq eN/2^{rd/16}$ and $d > 32$ is less than $1/N^2$.

Observe that a region with an outer boundary of length r contains $O(r^2)$ live cells. Thus for $d > 32$, with probability $1 - O(1/N)$ at most

$$\sum_{r=d}^{N} \frac{eN}{2^{rd/16}} O(r^2) = O\left(\frac{d^2 N}{2^{d^2/16}}\right)$$

live cells are isolated from the largest component on the wafer, which implies that for $c < 1/16$ at most $O(2^{-cd^2}N)$ live cells are isolated. For $d < 32$ the same result holds by simply adjusting the constant hidden by the Big Oh. ∎

By choosing d to be a sufficiently large constant, Lemma 3 ensures that with high probability, a constant fraction of the live cells on the wafer can be connected into a spanning tree with constant wire length. Because we know all wires will be constant length, Prim's minimum spanning tree algorithm [28] can be modified to run in linear time instead of the normal $O(N^2)$.

Theorem 4. *With probability $1 - O(1/N)$, any constant fraction (less than 1) of the live cells on an N-cell wafer can be connected in a linear array with constant-length wires.*

Proof. Straightforward from Lemmas 2 and 3. ∎

To conclude this section, we provide a theorem which states our results on constructing linear arrays in their fullest generality. The proof is similar to that of Lemma 3, and is not included here.

Theorem 5. *With probability $1 - O(1/N)$, at least $1 - \epsilon$ of the live cells on an N-cell wafer can be connected in a linear array using wires of length $O(s\sqrt{\log_p \epsilon})$ and channels of width 2, where p is the probability of a particular cell dying, s is the side length of each cell, and $1/N \leq \epsilon \leq p < 1$. This bound cannot be improved by more than a constant factor for any p, ϵ, or s.*

4. A lower bound for wafer-scale integration of two-dimensional systolic arrays

The problem of linking the live cells on a wafer to form a square two-dimensional array is substantially more difficult than the corresponding problem for linear arrays. The main difficulty with constructing two-dimensional arrays is that constant length wires no longer suffice when we throw away some of the live cells. In this section we provide a lower bound on the length of the longest wire required by a two-dimensional array. This bound was first discovered by Greene and Gamal [8]. Our proof (which is similar to but more general than that in [8]) was obtained independently from an idea due to Joel Spencer [40].

Theorem 6. *With probability $1 - O(1/N)$ every realization of any m-cell two-dimensional array on an N-cell wafer has a wire of length $\Omega(\sqrt{\lg m})$, for all $m = \Omega(\lg^2 N)$.*

Proof. The proof consists of two parts. In the first, we show that with high probability, the wafer contains a large number of regularly spaced square regions of $\frac{1}{4}\lg m$ cells, each of which is dead. In the second part of the proof, we show that any realization of an m-cell two-dimensional array must contain a cycle of four cells that surrounds the center of one of these dead regions. Thus one of the wires in the 4-cycle will have length $\frac{1}{2}\sqrt{\lg m}$.

First, partition the N-cell wafer into square regions with $m/32$ cells each, and then partition each of these regions into square subregions with $\frac{1}{4}\lg m$ cells each. We claim that with high probability, every $m/32$-cell region contains a $\frac{1}{4}\lg m$-cell subregion in which every cell is dead, as is illustrated in Figure 9.
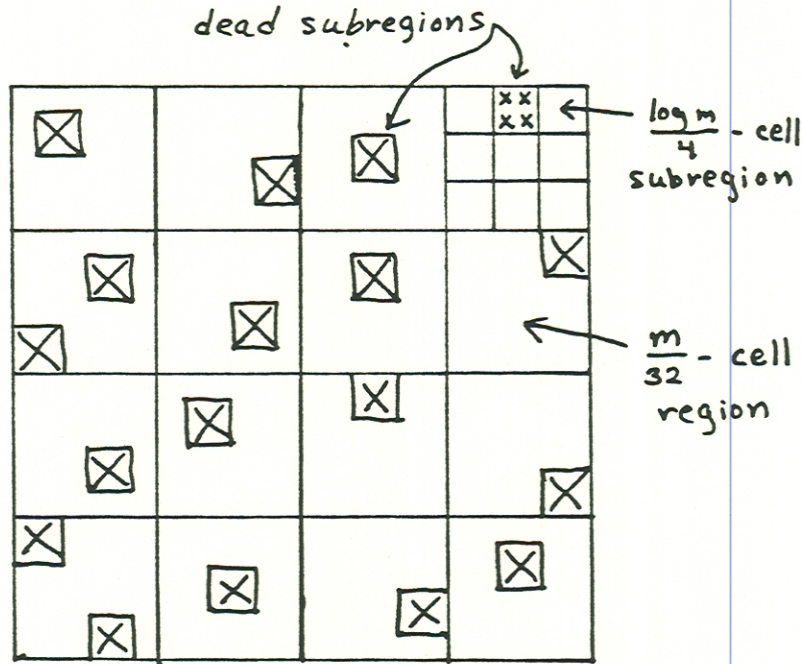

Figure 9. The distribution of dead $\frac{1}{4}\lg m$-cell subregions.

The probability that any particular $\frac{1}{4}\lg m$-cell subregion contains at least one live cell is $1 - m^{1/4}$. Thus the probability that each of the $\frac{1}{4}\lg m$-cell subregions in a particular $m/32$-cell region contains at least one live cell is

$$\left(1 - m^{-1/4}\right)^{m/8\lg m} \le e^{-m^{3/4}/8\lg m},$$

since $1 + x \le e^x$ for all x. The probability that one or more of the $32N/m$ $m/32$-cell regions fails to contain a totally dead $\frac{1}{4}\lg m$-cell subregion is at most

$$\frac{32N}{m}e^{-m^{3/4}/8\lg m} = O(1/N),$$

for $m = \Omega(\lg^2 N)$, which completes the first half of the proof.

If we can show that a 4-cycle of the two-dimensional array encloses the center of one of the $\frac{1}{4}\lg m$-cell dead regions, the proof will be complete because one of the wires of the 4-cycle will have length at least $\frac{1}{2}\sqrt{\lg m}$. More generally, however, if *any* cycle in the array surrounds the center of a dead subregion, then *some* wire in the array must have length $\frac{1}{2}\sqrt{\lg m}$. This observation follows because

12

1) every directed cycle in a two-dimensional array can be decomposed into the sum of directed 4-cycles, and

2) the number of times a cycle "wraps" around a point in the plane is equal to the sum of the number of wraps for each 4-cycle in its decomposition.

Thus a two-dimensional array with a cycle that encloses the center of a dead region must also contain a 4-cycle that surrounds the center of the dead region.

. We must now show that with high probability, every realization of every m-cell two-dimensional array contains a cycle that encloses the center of a square region of $\frac{1}{4} \lg m$ dead cells. We already know that with high probability a wafer contains contains a dead subregion of this size in every square region of $m/32$ cells. Assume for the purposes of contradiction that an m-cell two-dimensional array can be realized on such a wafer so that no cycle of the array surrounds the center of one of the dead regions. Consider a line drawn between the centers of two dead regions. If any wires cross this line, their removal will disconnect the two-dimensional array into two or more components, as is shown in Figure 10.
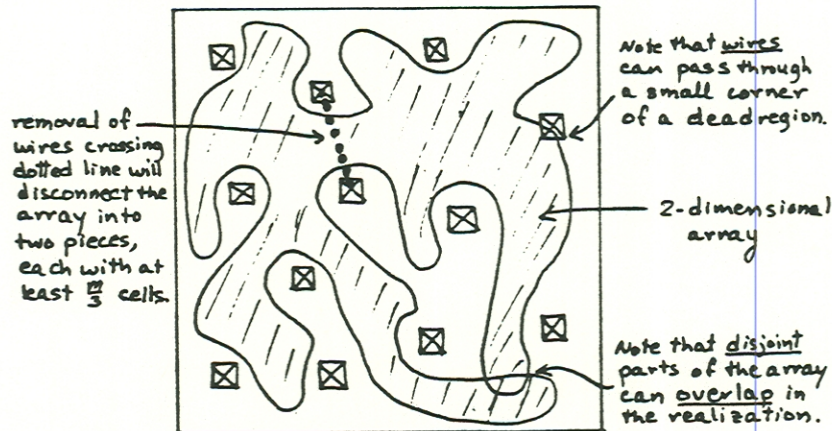
Figure 10. Disconnecting a two-dimensional array.

Among all pairs of neighboring dead regions (*i.e.*, pairs contained in $m/32$-cell regions that share an edge or corner), there is at least one pair for which removal of the wires passing between them disconnects the array into two pieces, each with at least $m/3$ cells. Since at most $4\sqrt{m/32} = \sqrt{m/2}$ wires can cross the line between the centers of two neighboring dead regions, by removing only $\sqrt{m/2}$ wires, we can disconnect an m-cell two-dimensional array into two pieces, each with at least $m/3$ cells. But it is well known that any such disconnection requires $\sqrt{m}$ wires to be removed, and we have obtained the contradiction that completes the proof.∎

The most interesting case of Theorem 6 is when the two-dimensional array to be constructed has $m = \Theta(N)$ cells.

Corollary 7 *With probability $1 - O(1/N)$ every realization of any two-dimensional array that utilizes any constant fraction of the live cells on an N-cell wafer has a wire of length $\Omega(\sqrt{\lg N})$.*

13

5. A divide-and-conquer method for constructing two-dimensional systolic arrays

The principal focus of this paper is the construction of systolic arrays on wafers such that the maximum wire length is minimized. In this section we ignore maximum wire length as a cost measure and look at the problem of constructing systolic arrays when only channel width is at issue. In doing so, we shall extend the general VLSI layout results of [19] and [21] to the wafer-scale situation where some of the cells may be faulty. Furthermore, the analysis of this section is *worst case* and not probabilistic, and thus all possible configurations of live and dead cells, however unlikely, can be handled.

The basic result of this section is that a two-dimensional array can always be constructed from all the live cells of an N-cell wafer if the channels have width $\Omega(\lg N)$. This result will be used in the next section as a subroutine in methods that achieve better bounds for wire length. The divide-and-conquer technique used in the construction is similar to general VLSI layout methods based on *separators* [21] and *bifurcators* [19].

We first prove a result on encoding two-dimensional arrays in complete binary trees à la Rosenberg [31] where some of the leaves may be dead. An *encoding* of a graph $G = (V, E)$ in a tree T is a one-to-one mapping f from the vertices V to the leaves of T. In our case, f must map V to live leaves of T. Such a mapping can be extended naturally to map E to the paths of T, where f maps (u, v) to the unique simple path connecting $f(u)$ to $f(v)$.

Lemma 8. *Let T be a complete binary tree with each of its N leaves labeled as either "live" or "dead," and let M be the number of live leaves. Then for any M-element two-dimensional array G, there exists an encoding f of G in T such that only $O(\sqrt{k})$ edges of E are mapped by f to an edge of T that has k descendent leaves.*

Proof. We rely on the fact that every tree has a *weighted one-separator theorem* [23]. That is, if the vertices of the tree are given arbitrary weights, removal of a single vertex will partition the tree into two components, each with less than two-thirds the weight. In our case, we weight the internal nodes and dead leaves of the tree T with zero and the live leaves with one. Then, in fact, a single edge of the tree can be removed to split the tree into two components, each with at least $M/3$ leaves.

The construction of f is obtained by a divide-and-conquer algorithm. We start by attempting to encode the original two-dimensional array G in the original tree T. Of course, the number of live cells on a wafer will rarely be a perfect square, and the subarrays corresponding to internal nodes of the tree will not be square either. We shall allow the original array to be missing some cells from the bottommost row and the rightmost column. Any subarray that is generated may be missing some cells from each of its four edges, and will in general have the shape shown in Figure 11.

Each recursive iteration attempts to encode an m-element subarray $\hat{G}$ of G in a subtree $\hat{T}$ of T. Using the weighted separator for trees, we determine a single edge whose removal partitions the tree $\hat{T}$ into two subtrees, each with at least $m/3$ live leaves. The two-dimensional array $\hat{G}$ is then cut into two subarrays with the corresponding number of live cells. By cutting parallel to the short dimension, we can ensure that the subarrays of $\hat{G}$ have perimeter $O(\sqrt{m})$, which is important for the analysis later on. Finally, the two subarrays of $\hat{G}$ are encoded recursively in
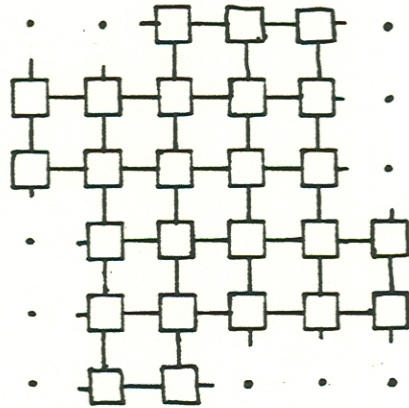
Figure 11. A 6-by6 array that is missing some border cells.

the two subtrees of $\hat{T}$. The recursion terminates when the subarray to be encoded consists of a single vertex v of G. When this happens, the edge immediately above the single live leaf of the corresponding subtree of T is removed, and the vertex v is mapped to the live leaf.

It remains to be shown that this encoding maps only $O(\sqrt{k})$ edges of the graph G to any edge e leading out of any k-leaf subtree T' of T. Look at e and those edges beneath in T' as they were cut during the execution of the algorithm. (See Figure 12.) The first cut of one of these edges partitions some subtree $\hat{T}$ which contains T' into two portions, each with at least one-third the live leaves of $\hat{T}$. One of the two pieces is a subtree T'' of T' into which a subarray is encoded. The number of connections from this subarray that pass through e is at worst the perimeter of the subarray, and the number of elements in the subarray is at most k. (This worst case occurs when $T' = T''$ and e is the first edge cut.)
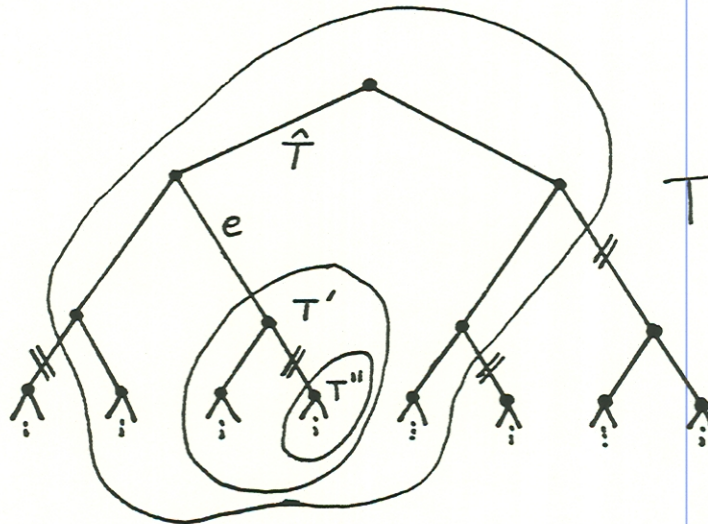
Figure 12. The relationships among trees in the proof of Lemma 8.

15

Therefore, only $O(\sqrt{k})$ connections from this subarray can possibly pass through e to $\hat{T} - T''$. No other edges from within T'' pass through e to the rest of $\hat{T}$, but some in $T' - T''$ might. The subtree $\hat{T} - T''$, however, has at most two-thirds the live leaves of $\hat{T}$, and thus only $O(\sqrt{\frac{2}{3}k})$ additional connections corresponding to the second cut can pass through e. By induction, the sum of the perimeters of all arrays that could pass through e is bounded by $O(\sqrt{k})$ because the series decreases geometrically. ∎

The encoding of a two-dimensional array in an N-leaf complete binary tree corresponds naturally to an embedding of the array in an $O(N)$-leaf *tree of meshes* [17, 18, 19]. Figure 13 shows a 16-leaf tree of meshes. The root of the complete binary tree has $O(\sqrt{N})$ connections passing through it from one side to the other. In the corresponding tree of meshes, the switching of these connections is accomplished by a $\Theta(\sqrt{N})$-by-$\Theta(\sqrt{N})$ mesh at the root. The two subtrees of the root of the complete binary tree correspond recursively to the two subtrees of the root of the tree of meshes. The leaves of the complete binary tree will be embedded in small meshes at most a constant distance from the leaves of the tree of meshes because the mesh at the root of the tree of meshes is a constant factor larger than $\sqrt{N}$-by-$\sqrt{N}$.
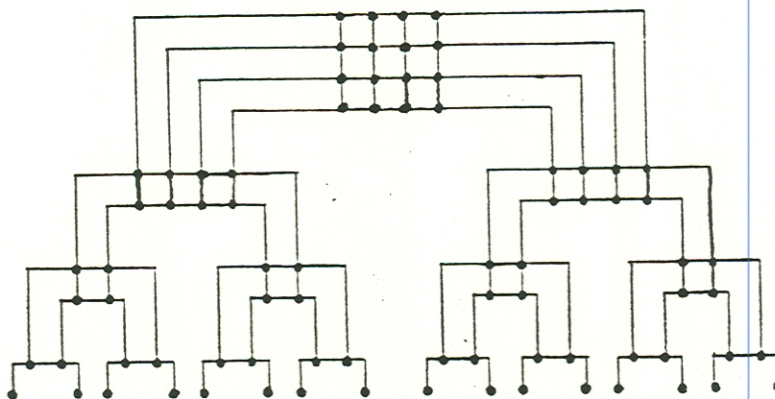

Figure 13. The 16-leaf tree of meshes.

The upper level meshes of the tree of meshes contain only wires, the bottom level meshes are empty, and small meshes near the bottom contain the cells of the two-dimensional array. If we chop off the unused lower level meshes, we obtain a *shortened tree of meshes* whose leaves correspond to the cells of the two-dimensional array. The next lemma shows that a shortened tree of meshes can be embedded on a wafer with channels of width $O(\lg N)$.

Lemma 9. *An N-leaf shortened tree of meshes can be constructed on an N-cell wafer that has a uniform channel width of $\Theta(\lg N)$ so that the leaves of the shortened tree of meshes correspond in a one-to-one manner with the cells of the wafer.*

Proof. The first step is to construct a $\Theta(\lg N)$-layer three-dimensional layout [20, 32] of the shortened tree of meshes. Fold the connections between the root of the shortened tree of meshes and each of its two sons so that the sons fit naturally on a second layer over the root. Fold the

16

connections to each of the grandsons so that they fit naturally over the sons on a third layer, and so forth. This generates a $\Theta(\lg N)$-layer three-dimensional layout where each layer has linear area. By projecting the three-dimensional layout onto a single layer in the manner of [42, pp. 36–38], channels with a uniform width of $\Theta(\lg N)$ are obtained.∎

The next theorem is the major result of this section.

Theorem 10. *Any M-cell two-dimensional array can be constructed from any subset of the live cells on an N-cell wafer using wires of length $O(\sqrt{N} \lg N)$ and channels of width $O(\lg N)$.*

Proof. Immediate from Lemmas 8 and 9.∎

By using *two-color bisectors* [3] or *fully balanced bifurcators* [19], the results of this section can be generalized to the encoding or embedding of classes of graphs other than two-dimensional arrays. The general idea is to use these tools to bound the number of external connections from a subgraph during the divide-and-conquer algorithm in the proof of Lemma 8. The only subtlety is that proportional cuts are required, which involves several applications of the two-color bisector or fully balanced bifurcator. All of the bounds on areas of graphs reported in [19] and [21] can then be obtained in the wafer-scale model where channels have uniform width and cells can be defective. (For a more complete description of how these techniques can be used to embed arbitrary graphs in a fault-tolerant manner, see [2].)

6. Upper bounds for wafer-scale integration of two-dimensional systolic arrays

Theorem 6 from Section 4 gives a lower bound of $\Omega(\sqrt{\lg N})$ on the length of a wire in any realization of a two-dimensional systolic array that utilizes all or most of the live cells of an N-cell wafer. We do not know how to achieve this lower bound, but we can come close. This section gives three nontrivial upper bounds for wire length and channel width. Of the three methods, however, only the algorithm in the proof of Theorem 13 achieves the lower bound of Theorem 6. Unfortunately, this algorithm utilizes only $m = \Theta(N/\lg\lg^2 N)$ of the live cells.

We first present a divide-and-conquer algorithm that constructs a square two-dimensional array using all the live cells on a wafer. In the first stage, the wafer is recursively bisected, and the number of live cells in each half is counted. Based on the count of live cells in each half of the wafer, the algorithm computes the dimensions of the two subarrays that must be constructed, and then recursively constructs the subarrays. The two subarrays are then linked together to form the complete array.

The algorithm remains in the first stage until subproblems with $\Theta(\lg N)$ cells are encountered. At this point the techniques used in Theorem 10 are used to complete the wiring of a $\Theta(\lg N)$-cell subarray. The exact crossover point between the first and second stages can be set at subproblems of size $c \lg N$, where c is any constant sufficiently large to ensure that with high probability, every $c \lg N$-cell region contains $\Omega(\lg N)$ live cells. (For example, a choice of $c = 2$ will suffice.)

Figures 14 through 17 illustrate the divide-and-conquer procedure. Figure 14a shows a 64-cell wafer which contains 36 live cells. In what follows, we step through the algorithm as it constructs a 6-by-6 array, which is identified as the "overall target" in Figure 14b.
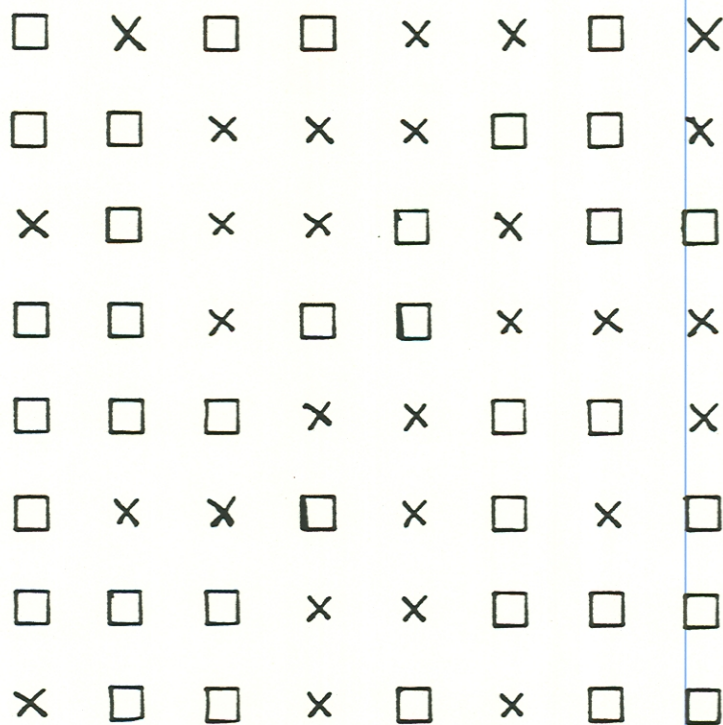
17

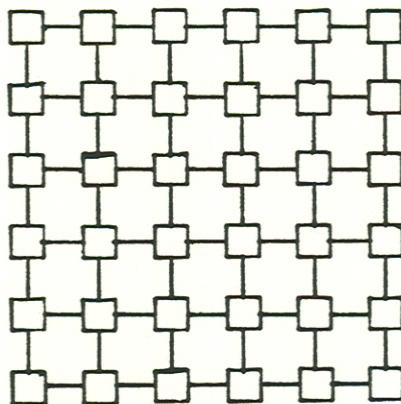**Figure 14a.** A 64-cell wafer that contains 36 live cells.

Figure 14b. The target: a 6-by-6 systolic array.

The first step is to bisect the wafer vertically, which gives 19 live cells in the left half and 17 in the right. We wish to construct a 19-cell subarray in the left half wafer and a 17-cell subarray in the right half wafer. Since we want the two subarrays to fit together nicely after they have been constructed, we choose the shapes of the two subarrays that are determined by the partition of the 6-by-6 array shown in Figure 15.

18
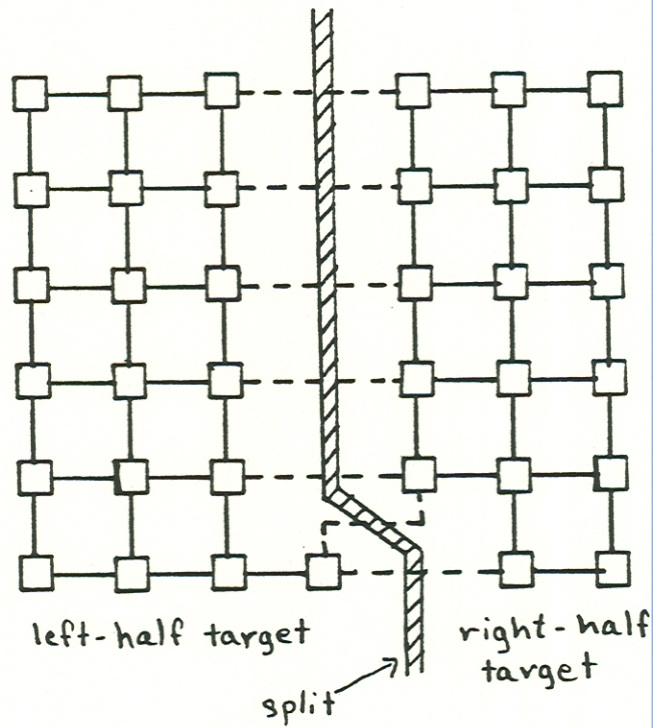
Figure 15. Partitioning the target.

We now invoke the procedure recursively on the two subarrays, but this time we bisect each of the halves horizontally. For example, when the left half wafer is bisected, the 19 live cells are divided into 9 cells above and 10 cells below, as displayed in Figure 16. The algorithm continues in this fashion, alternating between horizontal and vertical divisions, until the wafer and the target have been partitioned into $\Theta(\lg N)$-cell regions, at which point the algorithm proceeds to the second stage, and the technique of Theorem 10 applied.
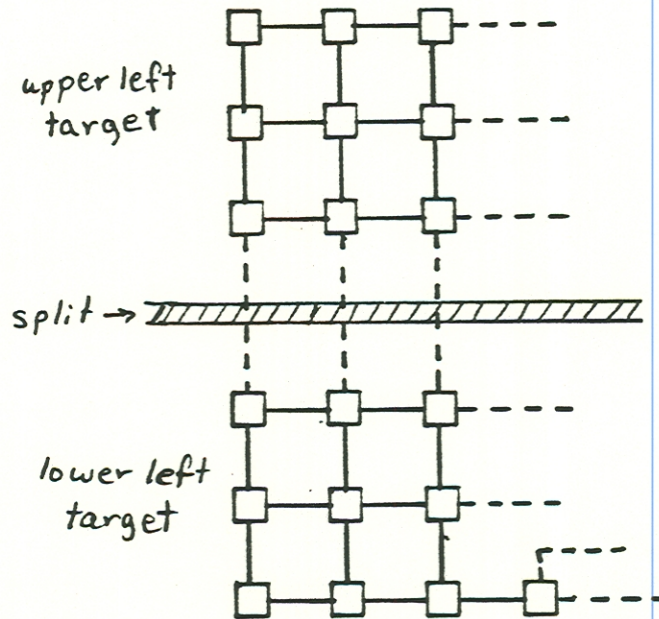
Figure 16. Partitoning the left target.

19

In this example the number of cells is small enough that the second stage construction can be performed by inspection. The inspection strategy can be used effectively in practice. Since the second stage operates on regions of size $\Theta(\lg N)$, the routings of this size can be precomputed. The second stage then consists of a single table lookup. At worst, this strategy costs polynomial time and space.

Figure 17 shows the final solution to the problem in Figure 14. For clarity the wires have not been routed within the channels of the wafer. Notice that each quadrant contains the specified targets for second level of recursion. The dashed lines represent wires that connect cells in different quadrants of the wafer.
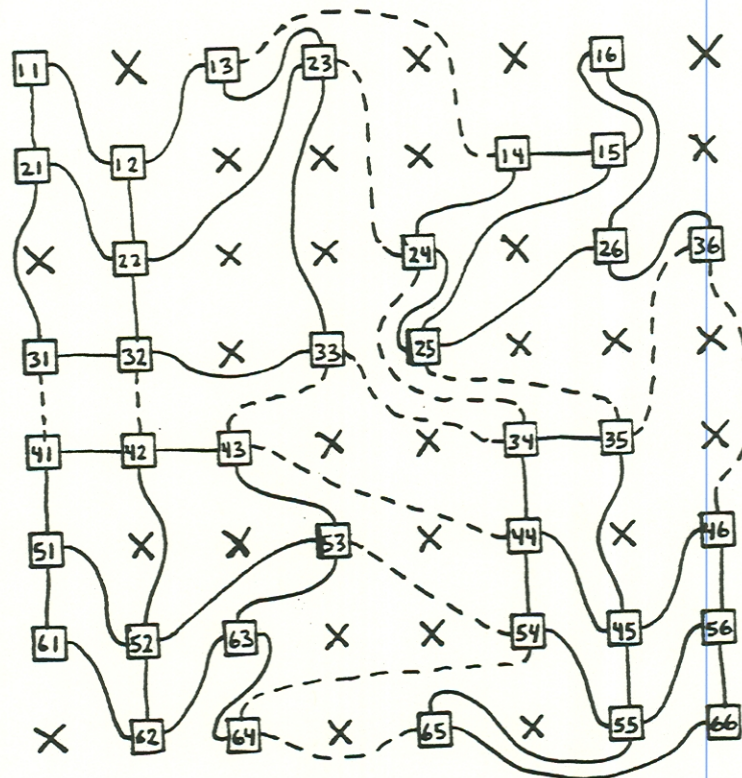
Figure 17. Completed cell assignment and wiring of the 6-by-6 array.

The next theorem describes how well the divide-and-conquer algorithm performs with respect to wire length and channel width.

Theorem 11. *With probability $1 - O(1/N)$ a two-dimensional array can be constructed from all the live cells on an N-cell wafer using wires of length $O(\lg N \lg\lg N)$ and channels of width $O(\lg\lg N)$.*

Proof. The divide-and-conquer algorithm just described provides the bounds in the theorem. The analysis is divided into two parts corresponding to the two stages of the algorithm.

We begin at the first level of recursion. Consider the wires that link a cell in the left subarray to a cell in the right subarray, as is illustrated by the two examples in Figure 18. For the most part, the connecting wires can be routed in the channel that separates the left and right halves

20

of the wafer. The length of the longest wire in the channel, as well as the width of the channel itself, is proportional to the longest vertical distance that a single wire must traverse.

The length of the longest wire in the center channel depends on the distribution of cells in each quadrant. For example, if we are extremely lucky and the live cells are regularly spaced, the longest wire may have constant length, as in Figure 18a. But if we are very unlucky, half the live cells might occur in the upper right quadrant and the other half in the lower left quadrant (Figure 18b). To connect the two halves, some wire will have length $\Omega(\sqrt{N})$.
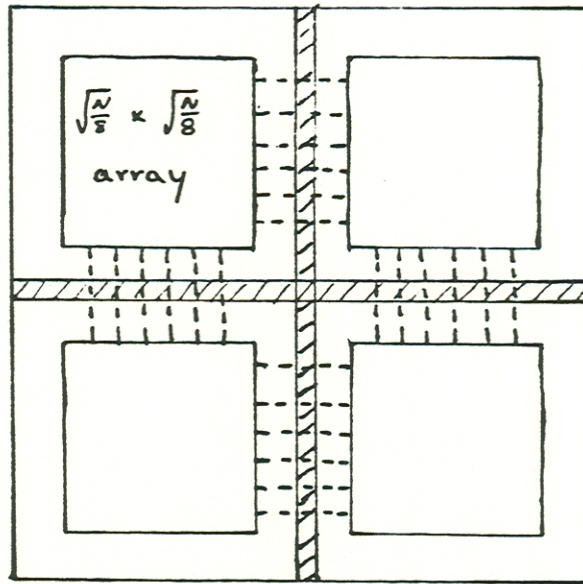
Figure 18a. A distribution of live cells which might allow a narrow center channel.
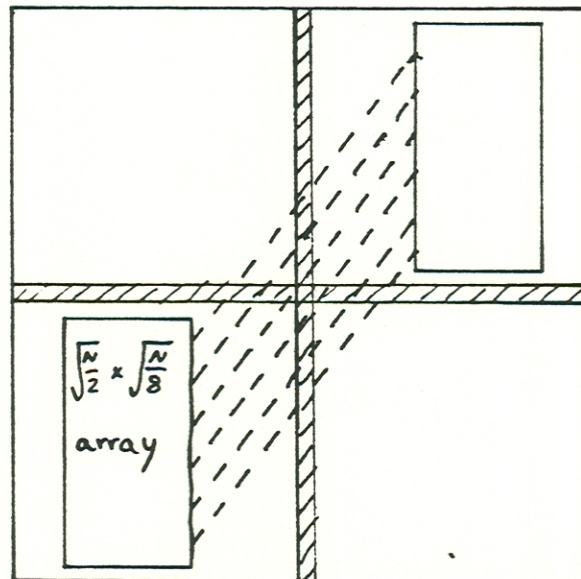

Figure 18b. A distribution of live cells which requires a wide center channel.

The length of the longest wire in the center channel can also be influenced by the distribution of cells within a quadrant. For example, if the upper left quadrant contains $\sqrt{N/8}$ live cells (about the right number), but they are distributed as in Figure 19, then the center channel still contains a wire of length $\Omega(\sqrt{N})$.
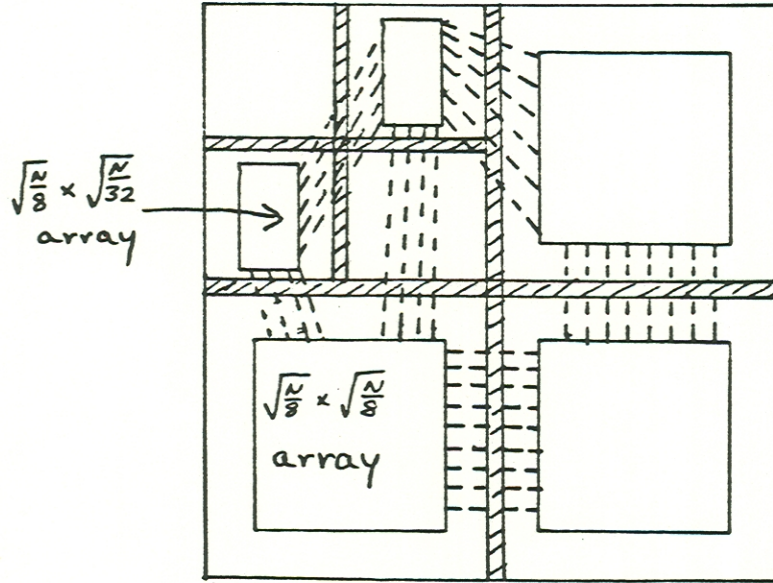

$\sqrt{\frac{N}{8}} \times \sqrt{\frac{N}{32}}$ array

$\sqrt{\frac{N}{8}} \times \sqrt{\frac{N}{8}}$ array

Figure 19. Another distribution of live cells which requires a wide center channel.

Most often, we are not so unlucky that a wire in the center channel has length $\Omega(\sqrt{N})$, but neither are we lucky enough that all wires are constant length. We now show that with high probability, we are more lucky than unlucky because the length of the longest wire in the center is $O(\lg N)$. The key to the analysis is to prove that the live cells are distributed so evenly that with high probability, the total vertical distortion of the wires in the center channel (over all subproblems of size $\Omega(\lg N)$) is $O(\lg N)$. In order to do so, we first observe that for all positive r, with probability $1 - O(e^{-2r^2})$ the four quadrants in the ith subproblem each have $m/4 \pm O(r\sqrt{m})$ live cells, where $m = N/2^{i+1} \pm O(\sqrt{N \lg N/2^i})$. Thus with probability $1 - O(e^{-2r^2})$, a subproblem contributes at most $O(r)$ distortion of wires in the center channel that are connected to the subarray at the level of the subproblem. There are $\Theta(\lg N)$ subproblems that can contribute to the distortion of a given wire in the center channel. Using standard combinatorial arguments involving sums of random variables, it is now possible to show that the worst case of the sum of the distortions is $O(\lg N)$ with probability $1 - O(1/N)$.

The same observations can be used to prove a high-probability bound of $O(\sqrt{\lg N \lg m})$ on the distortion of wires that connect subarrays of size $m = \Omega(\lg N)$. Thus it is sufficient that the channels between subproblems with m cells have width $O(\sqrt{\lg N \lg m})$. By summing over all $\Omega(\lg N)$-sized subproblems, it can be checked that at this point, the *average* channel width on the wafer is $O(1)$, which is because the channels inside $O(\lg N)$-sized subproblems have not been used at all. The constant average channel width can be achieved as a maximum without increasing the length of any wire by more than $O(\lg N)$. The idea is to distribute the $\Theta(\sqrt{\lg N \lg m})$-width channels across neighboring unused channels. As the details of this argument are somewhat tedious, we have omitted them. This concludes the first stage of the analysis.

The analysis of wires that link cells within a $\Theta(\lg N)$-cell subproblem differs substantially from the preceding analysis because live cells within a small region can have arbitrarily irregular distributions with high probability. The regions of irregularity are small enough, however, that the worst-case distributions are not really all that bad. For example, if a $\Theta(\lg N)$-cell region has the structure shown in Figures 18b or 19, then the maximum distortion of a wire at the top level of the recursion is just $O(\sqrt{\lg N})$.

In fact, the analysis of Section 5 ensures that the algorithm constructs a two-dimensional array in each $m = \Theta(\lg N)$-cell region using wires of length $O(\sqrt{m}\,\lg m) = O(\sqrt{\lg N}\,\lg\lg N)$ and channels of width $O(\lg m) = O(\lg\lg N)$. Thus the entire two-dimensional array is constructed using wires of length $O(\lg N \lg\lg N)$ and channels of width $O(\lg\lg N)$. The extra $\lg\lg N$ factor in the wire length bound comes about because a wire with $O(\lg N)$ distortion crosses $O(\lg N)$ channels, each of width $O(\lg\lg N)$. ∎

The wire length analysis of the *algorithm* in Theorem 11 is fairly tight. For example, the algorithm requires wires of length $\Omega(\lg N)$ with high probability. Thus, if the lower bound in Theorem 6 is to be achieved, a different algorithm must be discovered. It may be possible to improve the channel width bound, however. Any improvement in Theorem 10 would directly lead to an improvement in the channel width bounds in both Theorem 11 and the next theorem, which shows how to construct a two-dimensional array from *most* of the live cells on a wafer.

Theorem 12. *With probability $1 - O(1/N)$ a two-dimensional array can be constructed from any constant fraction (less than 1) of the live cells on an N-cell wafer using wires of length $O(\sqrt{\lg N}\,\lg\lg N)$ and channels of width $O(\lg\lg N)$.*

Proof. The key idea is to partition the wafer into $N/c\lg N$ square regions, each containing $m = c\lg N$ cells, where c is a sufficiently large constant. With probability $1-O(1/N)$, each of the regions contains at least $m' = \frac{1}{2}c(1 - 2/\sqrt{c})\lg N$ live cells. Using the technique of Theorem 10, we can therefore construct an m'-cell two-dimensional array in each region using wires of length $O(\sqrt{m}\,\lg m) = O(\sqrt{\lg N}\,\lg\lg N)$ and channels of width $O(\lg m) = O(\lg\lg N)$. The $N/c\lg N$ two-dimensional arrays are then connected together into one large array with $\frac{1}{2}N(1 - 2/\sqrt{c})$ live cells. The added wires have length at most $O(\sqrt{\lg N}\,\lg\lg N)$, and the channel width is not substantially increased. ∎

For each of the two previous results, the channels on the wafer have width $O(\lg\lg N)$. The next theorem shows that with high probability a two-dimensional array can be constructed from many of the live cells on a wafer using channels of unit width. Furthermore, the lower bound of $\Omega(\sqrt{\lg N})$ on wire length given in Theorem 6 is achieved by this construction.

Theorem 13. *With probability $1 - O(1/N)$ at least a fraction $\Omega(1/\lg\lg^2 N)$ of the live cells on an N-cell wafer can be connected into a two-dimensional array using wires of length $O(\sqrt{\lg N})$ and channels of unit width.*

Proof. The proof is similar to that of Theorem 12. As before, we partition the wafer into square regions with $c\lg N$ cells each. The constant c must be chosen large enough to ensure that with high probability, each region contains $\lg N$ live cells. We next partition each $c\lg N$-cell region into square subregions with $c'\lg\lg^2 N$ cells each. Consider all pairs of indices i and j in

23

the range $1 \leq i, j \leq \sqrt{c'} \lg \lg N$. For a given region of $c \lg N$ cells, at least one pair (i, j) satisfies the condition that at least $1/c$ of the cells in the (i, j) positions of of the subregions are alive. (Otherwise, it is impossible for $\lg N$ of the cells in the region to be alive.) Notice that by ignoring those cells not in the (i, j) positions of a subregion, the (i, j)-positioned cells together with all of the channels of the region form a "subwafer" with

1) $m = c \lg N / c' \lg \lg^2 N$ cells total,
2) at least $m/c = \lg N / c' \lg \lg^2 N$ live cells, and
3) channels of width $\sqrt{c'} \lg \lg N = O(\lg m)$.

By choosing c' large enough, the technique of Theorem 10 can be applied to construct within each $c \lg N$-cell region, a two-dimensional array with $\lg N / c' \lg \lg^2 N$ cells using wires of length $O(\sqrt{m} \lg m) = O(\sqrt{\lg N})$. These arrays can then be easily connected together to form a two-dimensional array with $N / c c' \lg \lg^2 N$ cells and wires of length $O(\sqrt{\lg N})$. ∎

By setting $m = \Omega(N / \lg \lg^2 N)$, it can be checked that Theorem 13 achieves the lower bounds for wire length proved in Theorem 6. The cell utilization, however, leaves something to be desired.

We have summarized the results of this section in the following table. Each bound is achieved with probability $1 - O(1/N)$, where N is the number of cells on the wafer, p is the probability that a particular cell is dead, and s is the side length of each cell. (Wires are assumed to have width one.)

Table I. Bounds on wire length and channel width for two-dimensional arrays.

Portion of live cells used	Wire length	Channel width
All	$O\left(\log_{1/p} N \left(s + \log_2 \log_{1/p} N\right)\right)$	$O\left(\log_2 \log_{1/p} N\right)$
Constant fraction (less than one)	$O\left(\sqrt{\log_{1/p} N} \left(s + \log_2 \log_{1/p} N\right)\right)$	$O\left(\log_2 \log_{1/p} N\right)$
$\Omega\left(1 / \left(\log_2 \log_{1/p} N\right)^2\right)$	$O\left(\sqrt{\log_{1/p} N}\right)$	1

7. Related models and problems

The problem of incorporating all the live cells of a wafer into a linear array so that the maximum wire length is minimized has been studied in more standard graph-theoretic models and has come to be known as the *Bottleneck Traveling Salesman* problem [7]. In addition, the wafer-scale model of N cells which fail independently with probability $1/2$ is essentially equivalent

to the well-studied geometric model in which N points are thrown down randomly in a unit square [1, 9, 13, 29, 37, 44]. Thus the algorithms for constructing linear arrays described in Section 3 can also be applied to the Bottleneck Traveling Salesman problem in the geometric unit-square model. For example, our results can be modified to show that with high probability, all of the points in the unit square can be joined into a hamiltonian path using wires of length $O(\sqrt{\lg N}/\sqrt{N})$, the least possible. In addition, *most* of the points can be joined in a linear array using wires of length $O(1/\sqrt{N})$, again the least possible. Although neither of these results have been explicitly stated in the literature, the first result is really just a minor extension to the prior work of Karp [13] and Bentley, Weide, and Yao [1]. The latter result of joining most of the points differs substantially from previous work, however. To the best of our knowledge, the only result of a similar nature is due to Erdös and Renyi [5] who showed that most graphs with N vertices and N edges have large connected components.

Channel widths do not play an important role in the unit-square model because the lines drawn between points are infinitesimally narrow. Thus the algorithms in the proofs of Theorems 11 and 12 can be modified to construct with high probability, two-dimensional arrays containing:

1) each of N points thrown randomly into a unit square using edges of length no more than $O(\lg N/\sqrt{N})$, and

2) any constant fraction (less than one) of N points thrown randomly into a unit square using edges of length no more than $O(\sqrt{\lg N}/\sqrt{N})$.

Since the lower bound of Theorem 6 can be extended to the unit-square model, the second result above is optimal, and thus there is no need to extend the result of Theorem 13.

The problems considered heretofore in this paper also have an interpretation in a purely graph theoretic model. Suppose we are given a two-dimensional grid graph, and assume that each node in the grid has independently a probability p of of being *bad*. We wish to find a subgraph of the grid that contains only *good* nodes and that forms a smaller two-dimensional grid. For example, Figure 20 illustrates the embedding of a good three-by-three grid in a partially bad four-by-four grid.
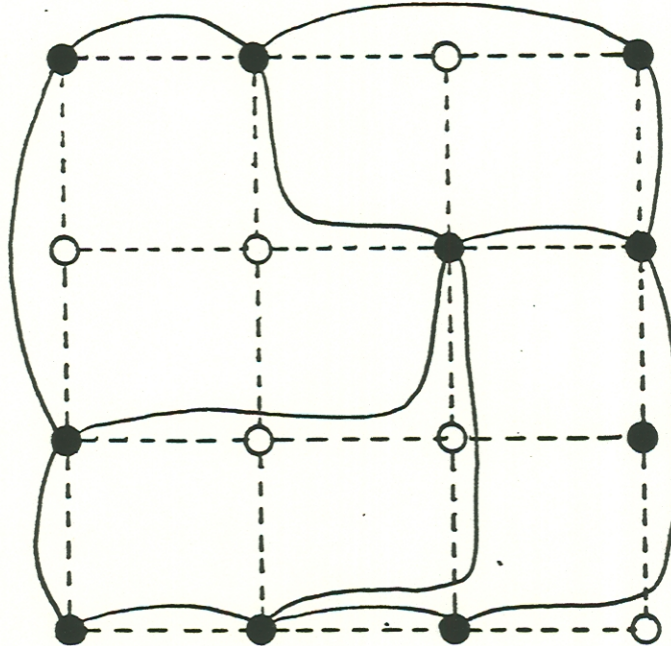

Figure 20. A good 3-by-3 grid formed in a partially bad 4-by-4 grid. Good nodes are denoted by black dots.

The objectives we might choose to optimize in such a problem are:
1) maximizing the size of the good grid,
2) minimizing the maximum distance between neighbors,
3) minimizing the total distance between all pairs of neighbors, and
4) minimizing the maximum number of times an edge in the partially bad grid is utilized.

These parameters roughly correspond in the wafer-scale model to the usage of live cells, maximum wire length, total wire length, and maximum channel width, respectively.

The beauty of the graph theoretic model, however, is that it generalizes naturally to broader classes of graphs. For example, the same kinds of questions can be reasonably asked about the class of k-dimensional grids for any k, the class of complete binary trees, or the class of hypercubes. In each case, the appropriate problem might be:

"A network in the class is given, but some portion of the nodes fail. How do we use the edges and good nodes of the network to construct a somewhat smaller network of the same type?"

For linear graphs the answer to the question is straightforward. This paper provides a starting point for two-dimensional grids. For other classes, the answers are as yet unknown. Also of interest is the problem of embedding a graph from one class in a partially bad graph from a different class. Research in this area should lead to a greater understanding of the fault tolerance of networks.

8. Concluding remarks

For all the theoretical analysis in this paper, some of the algorithms described are quite practical. Not only are they fast, but they produce good results because the constants are small. For example, the methods of Section 3 can be used to show that there is a simple, linear-time algorithm to connect most of the live cells on an N-cell wafer into a linear array using wires of length 1, 2, or 3 and channels of width at most 2. The method from Section 6 for connecting all the live cells into a two-dimensional array, modified to do table lookup on small subproblems, appears to be substantially better than what has been used in practice [38].

In addition to providing algorithms for constructing one- and two-dimensional arrays, the techniques used in this paper can also be used to construct arbitrary networks on integrated circuit wafers. There are two ways this can be done. First, one could embed the desired network in a two-dimensional array using the methods described in [2, 19, 21, 42, 43] and then construct the two-dimensional array using the procedures from Section 4. Alternatively, one could apply the divide-and-conquer process directly to the network. For example, the latter approach can be used to construct with high probability a complete binary tree from the live cells using constant channel width and edges of length $O(\sqrt{N}/\lg N)$, the least possible.

Some of the problems mentioned in this paper have been studied independently by Greene and Gamal. In their recent paper [8], they prove most of the results found in Section 3 as well as the lower bound in Section 4. Their analysis of linearly connected arrays is somewhat different from ours, however, as they rely on percolation theory from statistical physics.

In addition, Manning [25, 26], Hedlund [10], Koren [14], and Fussell and Varman [6] look at the basic problem of constructing arrays from defective arrays. Each gives algorithms but little theoretical or statistical analysis. Rosenberg [33, 34] has also investigated issues of fault tolerance.

Acknowledgments

References

[1] J. Bentley, B. Weide, and A. Yao, "Optimal expected-time algorithms for closest point problems," *ACM Transactions on Mathematical Software*, Vol. 6, No. 4, December 1980, pp. 563–580.

[2] S. N. Bhatt, F. T. Leighton and C. E. Leiserson, "A framework for solving VLSI graph layout problems," submitted.

[3] S. N. Bhatt and C. E. Leiserson, "How to assemble tree machines," *Proceedings of the Fourteenth ACM Symposium on Theory of Computation*, May 1982, pp. 77–84.

[4] G. Bilardi, M. Pracchi, and F. P. Preparata, "A critique and appraisal of VLSI models of computation," *Proceedings of the CMU Conference on VLSI Systems and Computations*, edited by H. T. Kung, R. Sproull, and G. Steele, October 1981, pp. 81–88.

[5] P. Erdös and A. Renyi, "On the evolution of random graphs," *Magyar Tud. Akad. Math. Kut. Int. Kozl.*, Vol. 5, 1960, pp. 17–61.

[6] D. Fussell and P. Varman, "Fault-tolerant wafer-scale architectures for VLSI," *Proceedings of the 9th Annual IEEE/ACM Symposium on Computer Architecture*, April 1982, pp. 190–198.

[7] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, San Francisco, 1979.

[8] J. Greene and A. Gamal, "Area and delay penalties for restructurable VLSI arrays," manuscript submitted for publication, Stanford University, May 1982.

[9] J. Halton and R. Terada, "A fast algorithm for the Euclidean traveling salesman problem, optimal with probability one," *SIAM Journal on Computing*, Vol. 11, No. 1, February 1982, pp. 28–46.

[10] K. Hedlund, personal communication.

[11] A. Itai, C. H. Papadimitriou, and J. L. Szwarcfiter, "Hamiltonian paths in grid graphs," *SIAM Journal of Computing*, Vol. 11, No. 4, November 1982, pp. 676–686.

[12] J. J. Karaganis, "On the cube of a graph," *Canad. Math. Bull.*, No. 11, 1968, pp. 295–296.

[13] R. M. Karp, "Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane," *Mathematics of Operations Research*, Vol. 2, 1977, pp. 209–244.

[14] I. Koren, "A reconfigurable and fault-tolerant VLSI multiprocessor array," *Proceedings of the 8th Annual IEEE/ACM Symposium on Computer Architecture*, May 1981, pp. 425–431.

[15] H. T. Kung and C. E. Leiserson, "Systolic arrays (for VLSI)," *Sparse Matrix Proceedings 1978*, edited by I. S. Duff and G. W. Stewart, Society for Industrial and Applied Mathematics, pp. 256–282.

[16] C. Y. Lee, "An algorithm for path connection and its applications," *IRE Transactions on Electronic Computers*, Vol. EC–10, No. 3, September 1961, pp. 346–365.

[17] F. T. Leighton, *Layouts for the Shuffle-Exchange Graph and Lower Bound Techniques for VLSI*, Ph.D. dissertation, Mathematics Department, M. I. T., September 1981. Also appears as MIT/LCS Technical Report 274 and as MIT/VLSI Memo 82–103.

[18] F. T. Leighton, "New lower bound techniques for VLSI," *Proceedings of the Twenty-Second IEEE Symposium on Foundations of Computer Science*, October 1981, pp. 1–12.

[19] F. T. Leighton, "A layout strategy for VLSI which is provably good," *Proceedings of the Fourteenth ACM Symposium on Theory of Computing*, May 1982, pp. 85–98.

[20] F. T. Leighton and A. L. Rosenberg, "Three-dimensional circuit layouts," MIT-VLSI Technical Memo 82–102.

[21] C. E. Leiserson, "Area-efficient graph layouts (for VLSI)," *Proceedings of the Twenty-First Annual IEEE Symposium on Foundations of Computer Science*, October 1980, pp. 270–281.

[22] C. E. Leiserson, *Area-Efficient VLSI Computation*, Ph.D. dissertation, Department of Computer Science, Carnegie-Mellon University, October 1981. Also appears as CMU Technical Report CMU-CS-82-108.

[23] R. J. Lipton and R. E. Tarjan, "A separator theorem for planar graphs," *A Conference on Theoretical Computer Science*, University of Waterloo, August 1977.

[24] J. Logue, W. Kleinfelder, P. Lowy, J. Moulic, and W. Wu, "Techniques for improving engineering productivity of VLSI designs," *Proceedings of the IEEE International Conference on Circuits and Computers*, 1980.

[25] F. Manning, *Automatic Test, Configuration, and Repair of Cellular Arrays*, Ph.D. dissertation, MIT Project MAC, June 1975.

[26] F. Manning, "An approach to highly integrated, computer-maintained cellular arrays," *IEEE Transactions on Computers*, Vol. c-26, June 1977.

[27] C. A. Mead and L. A. Conway, *Introduction to VLSI Systems*, Addison-Wesley, Reading, Massachusetts, October 1980.

[28] R. C. Prim, "Shortest connection networks and some generalizations," *Bell System Tech. J.*, Vol. 36, 1957, pp. 1389–1401.

[29] M. O. Rabin, "Probabilistic algorithms," in *Algorithms and Complexity: New Directions and Recent Results*, edited by J. F. Traub, Academic Press, New York, 1976, pp. 21–39.

[30] J. I. Raffel, "On the use of nonvolatile program links for restructurable VLSI," *Proceedings of the Caltech Conference on VLSI*, January 1979, pp. 95–104.

[31] A. L. Rosenberg, "Encoding data structures in trees," *JACM*, Vol. 26, No. 4, October 1979, pp. 668–689.

[32] A. L. Rosenberg, "Three-dimensional integrated circuitry," *Proceedings of the CMU Conference on VLSI Systems and Computations*, edited by H. T. Kung, R. Sproull, and G. Steele, October 1981, pp. 69–80.

[33] A. L. Rosenberg, "The Diogenes approach to testable fault-tolerant networks of processors," Technical Report CS-1982-6.1, Department of Computer Science, Duke University, May 1982.

[34] A. L. Rosenberg, "On designing fault-tolerant arrays of processors," Duke University Technical Report CS-1982-14.

[35] A. L. Rosenberg and L. Snyder, personal communication, 1982.

[36] M. Sekanina, "On an ordering of the set of vertices of a connected graph," *Publications of the Faculty of Science, University Brno*, Czechoslovakia, No. 412, 1960, pp. 137–142.

[37] M. I. Shamos, *Computational Geometry*, Ph.D. dissertation, Yale University, May 1978.

[38] J. Siskind, private communication, December 1981.

[39] L. Snyder, "Overview of the CHiP computer," *VLSI 81*, edited by J. Gray, Academic Press, New York, August 1981, pp. 237–246.

[40] J. Spencer, personal communication 1982.

[41] C. D. Thompson, "Area-time complexity for VLSI," *Proceedings of the Eleventh ACM Symposium on Theory of Computing*, May 1979, pp. 81–88.

[42] C. D. Thompson, *A Complexity Theory for VLSI*, Ph.D. dissertation, Department of Computer Science, Carnegie-Mellon University, 1980.

[43] L. G. Valiant, "Universality considerations in VLSI circuits," *IEEE Transactions on Computers*, Vol. C–30, No. 2, February 1981, pp. 135–140.

[44] B. W. Weide, *Statistical Methods in Algorithm Design and Analysis*, Ph.D. dissertation, Department of Computer Science, Carnegie-Mellon University, August 1978.