APPROXIMATION ALGORITHMS FOR GLOBAL ROUTING IN GATE ARRAYS ‡

TEOFILO F. GONZALEZ
Department of Computer Science
University of California
Santa Barbara, CA 93106

ABSTRACT

In this paper we study the computational aspects of the gate array global routing problem. We develop an algorithm that generates a routing with congestion bounded above by $2 \cdot OPT^*$, where OPT^* is "close" to the congestion of an optimal solution. Our algorithm reduces the global routing problem to a set of linear programming problems.

INTRODUCTION

Gate arrays have been successfully used as a design tool to generate, with a fast turnaround time, computer chips. In this paper we study the computational aspects of the gate array global routing problem. A typical gate array is given in figure 1. A gate array consists of r-1 rows of active areas, i.e., where components (transistors or high level function blocks) have been placed. In each of the active areas there is only one layer for routing. All the routing wires that one may introduce in these areas are wires that run along the vertical tracks. Between every pair of adjacent active areas, there is a nonactive area or (horizontal) channel in which two layers are available for routing. The wires in one layer must run along the vertical tracks and the wires in the other layer must run along the horizontal tracks.

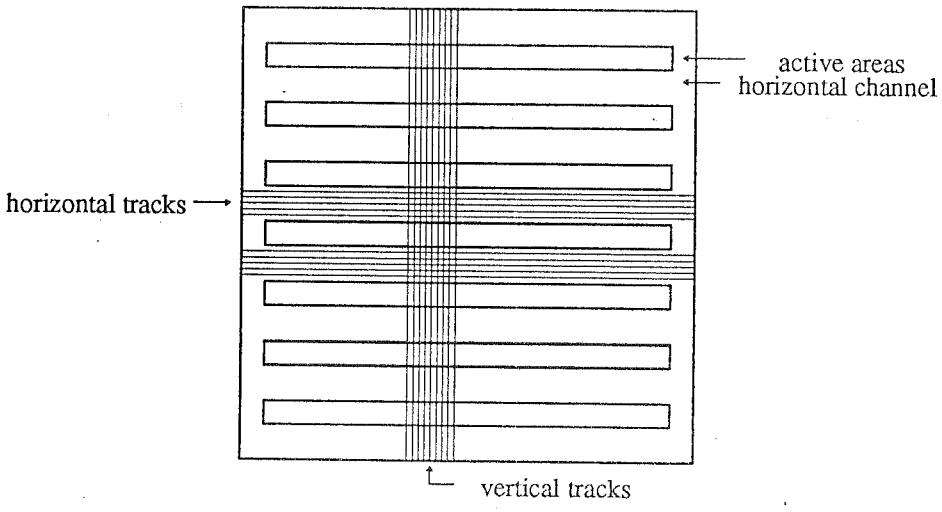


Figure 1: Typical gate array.

In the gate array design methodology, we initially place the components (transistors, high level function blocks, etc.) on the active areas and then we try to interconnect all points that have to be made

‡ This research was supported in part by the National Science Foundation under Grant DCR - 8503163

electrically common by using the tracks available for wiring. This procedure is repeated until a successful wiring is obtained, or until we have enough confidence that such wiring does not exist. In the latter case we partition our circuit into two or more computer chips, and repeat the above procedure on each of the resulting subproblems.

In this paper we present an algorithm to solve the gate array global routing problem. The typical approach for solving this routing problem begins by dividing the gate array into r by c *cells* by introducing the set of horizontal and vertical *cell dividing lines* given in figure 2. The cells are referred to as $C_{i,j}$, for $1 \leq i \leq r$ and $1 \leq j \leq c$. Cell $C_{i,j}$ contains the set of points $S_{i,j}$. The number of elements in each of these sets is at most p. The set of points $S = \cup S_{i,j}$ in all cells are partitioned into $N = \{ N_1, N_2, ..., N_n \}$. Each set N_i is called a *net*. All the points in each net have to be made electrically common by interconnecting them by wires. Each wire must follow a path in the routing regions specified above, and each path consists of alternating vertical line segments (assigned to layer one) and horizontal line segments (assigned to layer two). Note that in this design methodology the number of tracks available for wiring is an upper bound on the maximum number of wires that cross from one cell to any of its adjacent cells. The typical approach for solving the gate array routing problem consists of the following two steps:

(1) *Global Routing:* Find a global routing, R, which specifies for each net the set of cell boundaries crossed by the wire connecting it. In a global wiring the number of wires that cross the boundary between adjacent cells must not exceed the capacity of the cell, i.e., the number of tracks that cross the edge.

(2) *Detail Routing:* For each net specify the exact position for the wires.

Of course, not all global routings have a detailed routing. However, the complexity of our problem is greatly reduced by dividing our routing problem into two subproblems.

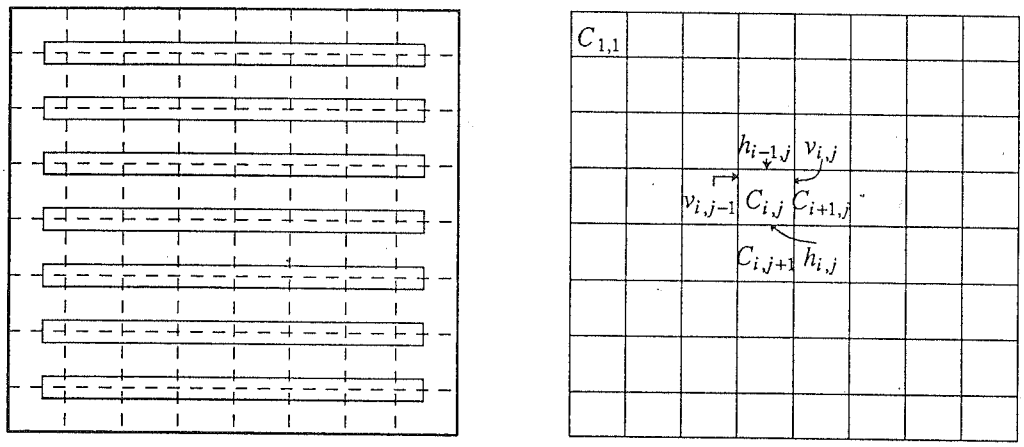


Figure 2: Partition of the gate array.

Throughout this paper we assume that all nets are of size two. An approach similar to ours can be used to solve the more general problem; however, the approximation bound in this case is larger. Figure 3 shows different global wires connecting a net. A wire is called a *t-turn wire* if it makes at most t turns, i.e., it bends t times (see figure 3). A global wiring R is called a *t-turn global wiring* if every wire is a $t-turn$ wire. The capacity of the edges adjacent to $C_{i,j}$ is given by $v_{i,j-1}$, $v_{i,j}$, $h_{i-1,j}$ and $h_{i,j}$ (see figure 2). Of course $v_{i,0} = v_{i,c} = 0$ for $1 \leq i \leq r$; and $h_{0,j} = h_{r,j} = 0$ for $1 \leq j \leq c$. For a global routing R, let $x_{i,j}$ ($y_{i,j}$) be the number of wires that cross the boundary between $C_{i,j}$ and $C_{i,j+1}$ ($C_{i,j}$ and $C_{i+1,j}$). The objective of the global routing problem is to find a global routing R such that the *congestion*, given by $B = \max\{x_{i,j}, y_{i,j}\}$, is least possible. Karp *et. all.* [KL] showed that the global routing problem is NP-hard even when $p = 1$ and all the nets are of size two. Shing and Hu [SH] developed a polynomial time algorithm to generate suboptimal solutions to the

gate array global routing problem. Raghavan and Thompson [RT] developed algorithms that guarantee provable good solution with high probability. Both of these algorithms generate a solution by reducing the global routing problem to a linear programming problem. Other algorithms for our problem appear in [GAM], [NH], [TT] and [VK]. The techniques behind these algorithms ranges from simple heuristics to simulated annealing. The underlying characteristic of all the known algorithms for global routing is that none of them have been shown to generate solutions within a fixed percentage of the optimal solution value. *That is the main difference between our algorithms and the previous algorithms for global wiring.*

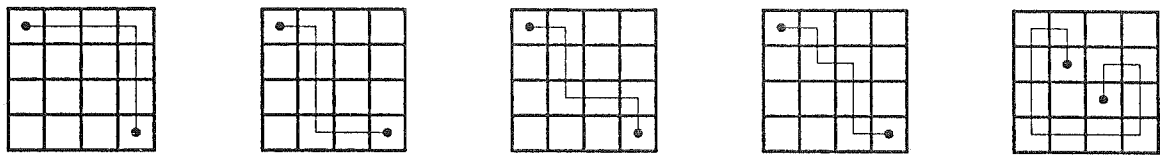
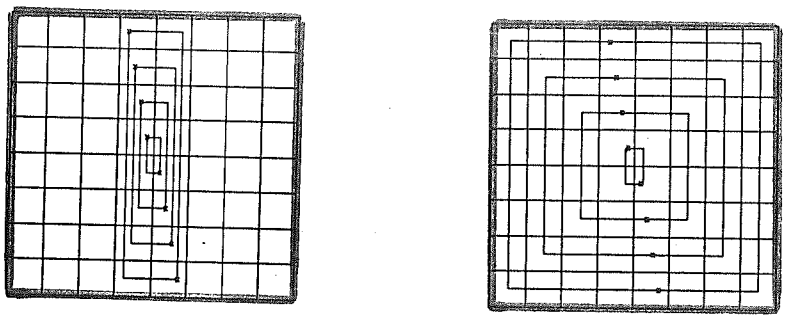


Figure 3: $t-turn$ global wires.

Let OPT_k be the congestion of an optimal $k-turn$ routing R and let OPT be the congestion of an optimal routing R (there is no limit on the number of bends in any wire). Clearly, $OPT_{k+1} \leq OPT_k$ and $OPT \leq OPT_k$. Karp *et. all.* [KL] also showed that for some problem instances $OPT_1 = ((r+c)/4)OPT$. Figure 4 gives one of these problem instances. The algorithm developed by Raghavan and Thomson [RT] generates a routing with congestion at most $2 \cdot OPT_1$. Clearly, this does not imply that such routing has a congestion bounded by $2 \cdot OPT$.



(a) OPT_1 (b) OPT

Figure 4: Instance with $OPT_1 = 4$ and $OPT = 1$.

A wire is said to be a *t-turn falling wire* if it is a t-turn wire and every horizontal cell boundary line intersects the wire at most once. All the wires in figure 4, except for the $6-turn$ wire, are falling wires. A global routing R is said to be a *t-turn falling routing* if all the wires are $t-turn$ falling wires. Let OPT'_k be the congestion of an optimal $k-turn$ falling routing R and let OPT' be the congestion of an optimal falling routing R (there is no limit on the number of bends in each wire). Clearly, $OPT_k \leq OPT'_k$ and $OPT \leq OPT'$. In this paper we develop an algorithm that generates a global routing with congestion bounded above by $2 \cdot OPT'$. It is interesting to note that the wirings generated by our algorithm have the same provable good behavior as the ones generated by the algorithm in [RT]. Our algorithm reduces the global routing problem to a set of linear programming problems which can be solved by the classical algorithms [GA] or the newer algorithms [Ka], [K], and [Va].

TRANSFORMATION TO THE VIA PLACEMENT PROBLEM

Let us now outline our approach in detail. First we transform our global routing problem to a via placement problem. In the new problem we also have $r \cdot c$ cells denoted by $C_{i,j}$. Associated with cell $C_{i,j}$ there is a region for placing via columns referred to as *via slot* $V_{i,j}$ (see figure 5). Let us assume for the moment that net N_i consist of points located in cells $C_{1,1}$ and $C_{4,4}$. In this case we introduce three via columns (see figure 6). Each via column has to be placed in one of the *via column slots* formed by the via slots located in the same column. The *ith* via column has a pin in row i and a pin in row i+1. In figure 7 (1–*turn*) all the via columns are placed in via column slot 4; in figure 7 (2–*turn*) all the via columns are placed in via column slot 2; in figure 7 (3–*turn*) the first two via columns are placed in via column slot 2 and the last one in via column slot 4; in figure 7 (4–*turn*) the first via column is placed in via column slot 2 and the other two in via column slot 3; and in figure 7 (5–*turn*) the *ith* via column is placed in the i+1*st* via column slot. In the new problem we only care to count the wires that cross the thick lines (see figure 7). Note that any t–*turn* falling wire can be obtained by placing the via columns in the appropriate via column slots. The global routing for the original problem is obtained by ignoring the via points and considering only the "thick" boundaries crossed by the wires.

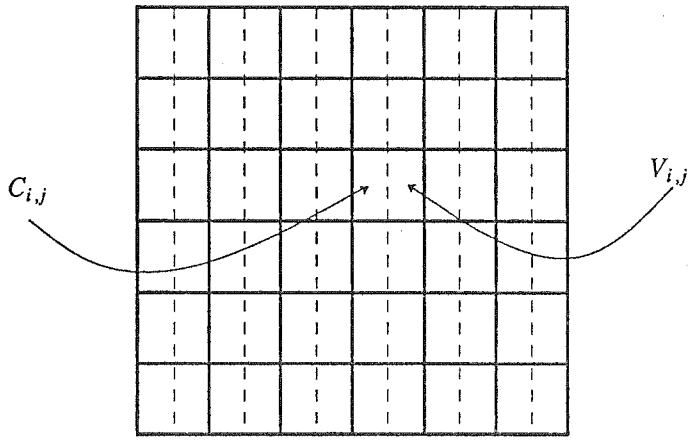


Figure 5: Component cells and via slots.

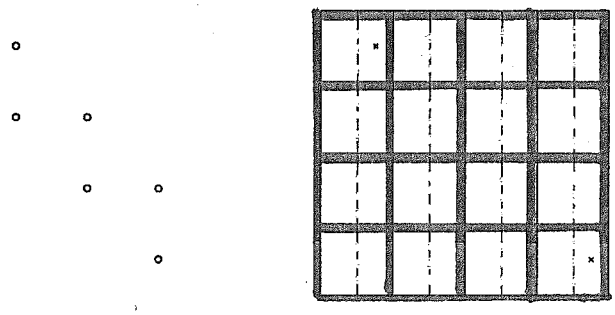


Figure 6: Component cells, via slots, and via columns.

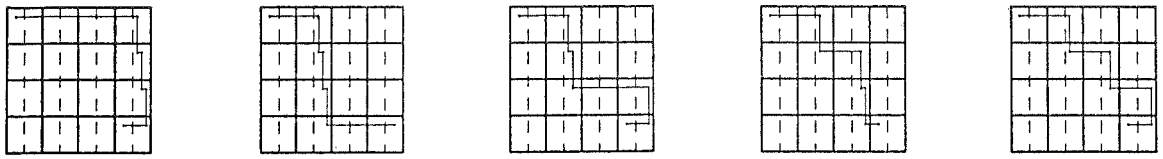


Figure 7: *t−turn* global wires after placement of the via columns.

At this point the original problem has been transformed to the "generalized" via placement problem. A simplified version of this problem, the via placement problem for MPCBs, has been studied in [GK]. The main difference between these two problems is that in the via placement problem for MPCBs we only minimize the row congestion. The column congestion is ignored. Gonzalez and Kurki-Gowdara [GK] present a polynomial time algorithm for the via placement problem that generates solutions that are within twice the optimal row congestion. The main problem now is how to modify their algorithm to achieve also a small column congestion.

Let us now explain Gonzalez and Kurki-Gowdara's [GK] algorithm. Let P be a via placement problem. The algorithm begins by finding an optimal solution to the integer-relaxed version of P, which we call P'. The main difference between problems P and P' is that in problem P' one is allowed to place fractions of via columns at different via column slots. As a result of this there will be fractional wires throughout the wiring. An optimal solution to problem P' may be obtained by solving a set of linear programming problems. The optimal solution to P' is used to obtain a (suboptimal) solution to the via placement problem P. The suboptimal solution to P is such that each wire in each region can be associated with a fractional wire of thickness at least 1/2 (or *half wire*). That is why we can prove the approximation bound of two. In figure 8a we show the placement of via columns in P. Each wire delineated by the wiggle lines (figure 8a) in P has associated with it a *skeleton wire* in P' with the property that there is at least a half wire in the region delineated by the wiggle line and there are fractional wires elsewhere in the row. Gonzalez and Kurki-Gowdara's algorithm can be modified so that an optimal solution to P' also takes into account the column congestion by artificially assigning the skeleton wire to the horizontal cell boundaries just under it. This modification would work on row one in example 8a; however, it will not work on rows 2, 3 and 4. The reason for this is that the skeleton wire in row two would contribute at least a half wire in two horizontal cell boundaries. In row three of the example given in figure 8b it is even worse, because a wiggle line crosses two thick boundaries. This overestimation has the side effect that the optimal congestion for P' might not be smaller than the optimal congestion for P, and as a result of this, the approximation bound of two will not hold. In the linear programming formulation it is not possible to just artificially assign to one horizontal cell boundary a skeleton wire. A skeleton wire is either assigned to all the horizontal cell boundaries in a row or to none. This limitation arises from the reduction to a set of linear programming problems. Can we modify our construction rules so that all problem instances are of the type given in row one in figure 8a? The answer is yes!

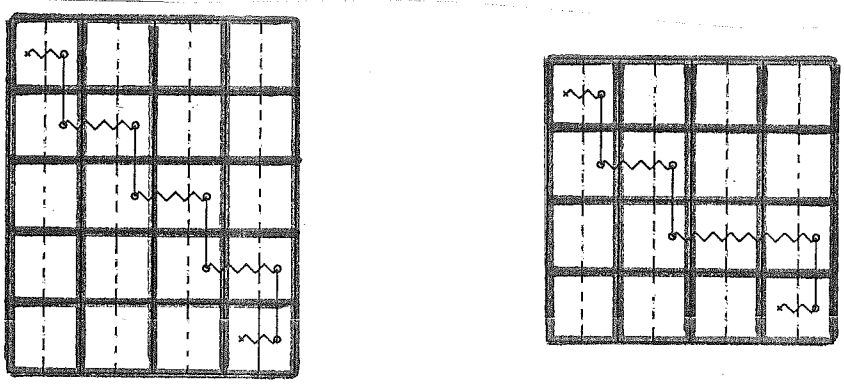


Figure 8: Placement.

Let us now modify the previous transformation to achieve minimal column congestion. The main idea is to granulate the via slots and introduce a large number of via columns in such a way that the integer-relaxed version of this problem, P', has an optimal solution in which whenever a vertical wire crosses a thick line, the wiggle line in P associated with its topmost point does not cross any thick lines (see figure 10). For this situation it is simple to see that when the skeleton wire is added to the horizontal cell boundaries immediately below them, only one half wire will be added to one horizontal cell boundary. Note that solutions that overestimate, i.e., wiggle lines crossing more than one thick boundary, are possible. The reason why this new formulation is correct is that a solution that overestimates will always be worse than one that does not; and there is always a solution that does not overestimate. This implies that the optimal congestion value for P' is smaller than that for P. Hence, an algorithm similar to the one in [GK] generates a solution with objective function value within twice OPT'.

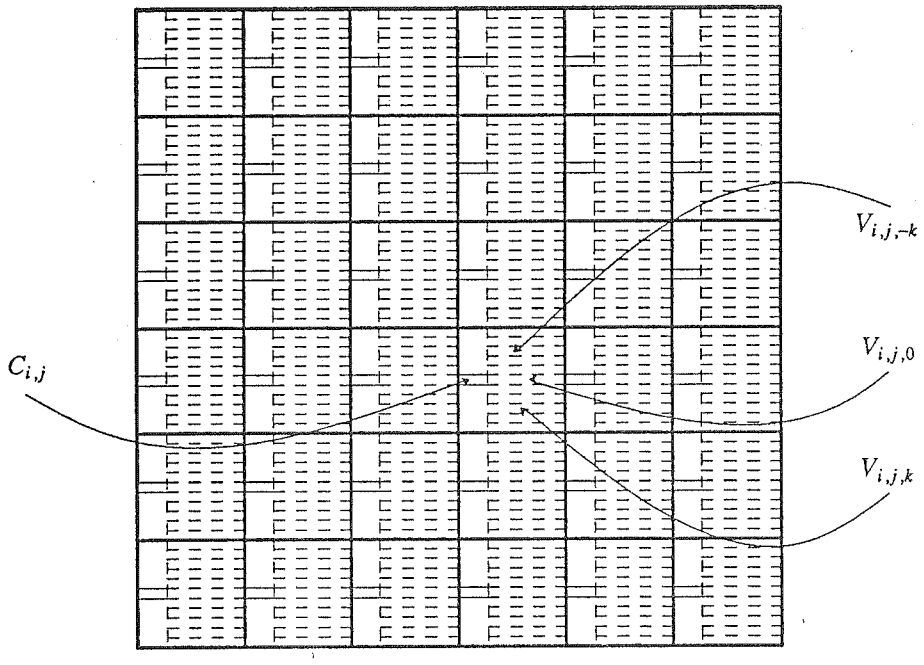


Figure 9: Component cells and via slots that allow "short" wires.

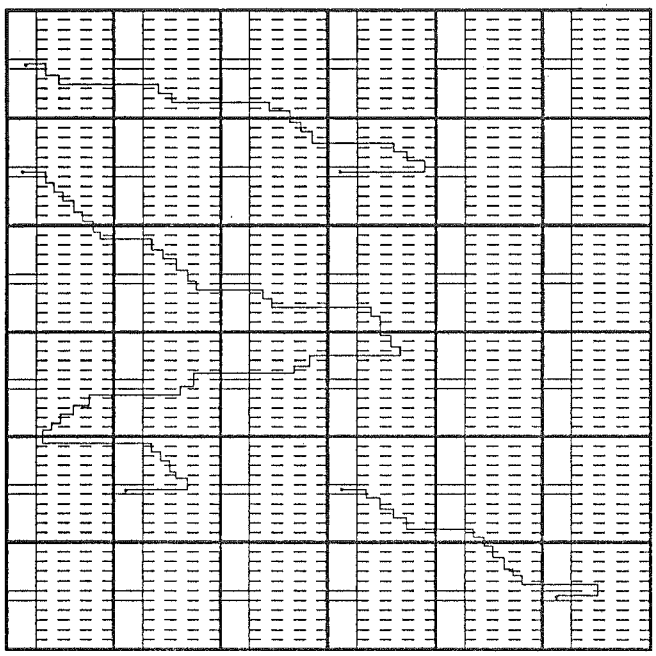
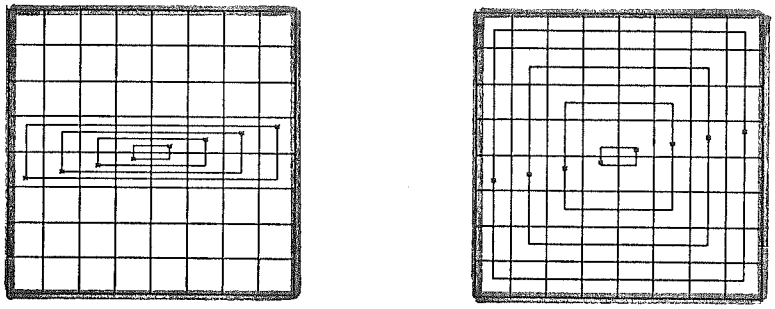


Figure 10: Global wirings using via column slots that allow short wires.



OPT' OPT

Figure 11: $OPT' = 4$ and $OPT = 1$.

How far is OPT' from OPT? The problem instance given in figure 11 (figure 4 rotated 180 degrees) is such that $OPT' = 4$ and $OPT = 1$, i.e., OPT' is as bad as OPT_1. Let us try to reduce this bound. A wire is said to be a *t-turn left-to-right wire* if it is a t-turn wire and every vertical cell boundary line intersects the wire at most once. A global routing R is said to be a *t-turn left-to-right routing* if all the wires are $t-turn$ left-to-right wires. Let OPT_k'' be the congestion of an optimal $k-turn$ left-to-right routing R and let OPT'' be the congestion of an optimal left-to-right routing R (there is no limit on the number of bends in each wire). Clearly, $OPT_k \leq OPT_k''$ and $OPT \leq OPT''$. Our algorithm generates a global routing with congestion value bounded by $2 \cdot OPT''$. Unfortunately, the example in figure 4 has the property that $OPT'' = 4$ and $OPT = 1$, i.e., OPT'' is as bad as OPT_1 and OPT'. Let the points from net N_k be located in cells $C_{i,j}$ and $C_{i',j'}$. The *horizontal span* for net N_k is $|i-i'|$ and its *vertical span* is $|j-j'|$. A global routing R is said to be a *t-turn l-span routing* if all the wires that connect a net with horizontal span smaller than its vertical span are $t-turn$ falling wires; otherwise, the net is connected by a $t-turn$ left-to-right wire. Let OPT_k''' be the congestion of an optimal $k-turn$ l-span routing R and let OPT''' be the congestion of an optimal l-span routing R (there is no limit on the number of bends in each wire). Clearly, $OPT_k \leq OPT_k'''$ and $OPT \leq OPT'''$.

Our algorithm generates a global routing with congestion value bounded by $2 \cdot OPT'''$. We conjecture that OPT''' is close to OPT. A global routing R is said to be a *t-turn e-span routing* if all the wires that connect a net can be connected with either $t-turn$ falling wires and $t-turn$ left-to-right wires. Let $OPTB_k$ be the congestion of an optimal $k-turn$ e-span routing R and let $OPTB$ be the congestion of an optimal l-span routing R (there is no limit on the number of bends in each wire). Clearly, $OPT_k \leq OPTB_k$ and $OPT \leq OPTB$. We conjecture that $OPTB$ is close to OPT. However, our algorithm can only generate a global routing with congestion bounded above by $4 \cdot OPTB$.

DISCUSSION

We presented an algorithm for the global routing problem in gate arrays. Our algorithm transforms the problem into a set of linear programming problems, which can be solved efficiently. The main advantage of our algorithm over previously known algorithms is that our algorithm generates solutions that are provably good. The time complexity of our algorithm and the ones in [RT] and [SH] is dominated by the time required to solve the linear programming problem. The main difference is that in our algorithm there are more variables and inequalities. From the practical point of view, our algorithms are important because they provide us with a "good" estimate of the optimal solution value. In practical situations our algorithm should be run concurrently with the other algorithms and then select the best of the solutions. With our bound on the optimal solution value we can estimate the quality of our solutions.

REFERENCES

[GAM] Gamal, A El, "Two-Dimensional Stochastic Model for Interconnections in Master-Slice Integrated Circuits," *IEEE Transactions on Circuits and Systems,* Vol. CAS-28, pp. 127 - 137, Feb. 1981.

[GA] Gass, S.,"Linear Programming," Mc-Graw Hill, 1969.

[GK] Gonzalez, T. and S. Kurki-Gowdara, "An Approximation Algorithm for the Via Placement and Related Problems", Proceedings of the Twenty-Fifth Annual Allerton Conference on Communications, Control and Computing, Oct. 1987, to appear in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*

[Ka] Karmarkar, N., "A New Polynomial-Time Algorithm for Linear Programming," *Combinatorica,* Vol. 4, 1984, pp. 373 - 395.

[KL] Karp R. M., F. T. Leighton, R. L. Rivest, C. D. Thompson, U. Vazirani and V Vazirani, "Global Wire Routing in Two-Dimensional Arrays," Proceedings of the Twenty-fourth Annual Symposium on Foundations of Computer Science, Oct. 1986, pp. 453 - 459.

[K] Khachiyan, L. G., "A Polynomial Algorithm in Linear Programming," *Doklady Akademiia Nauk SSSR,* 244:S (1979), pp. 1093 - 1096, translated in *Soviet Mathematics Doklady,* 20:1 (1979), pp. 1 - 12.

[NH] Nair, R., S. J. Hong, S. Liles, and R. Villani, "Global Routing On a Wire Routing Machine," Proceedings of the Nineteenth Design Automation Conference, June 1982, pp. 224 - 231.

[RT] Raghavan P. and C. D. Thompson, "Provably Good Routing in Graphs: Regular Arrays," Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computation, May 1985, pp. 79 - 87.

[SH] Shing M. T. and T. C. Hu, "Computational Complexity of Layout Problems," *Layout Design and Verification,* T. Ohtsuki (Editor), Elsevier Science Publishers, 1986, pp. 267 - 294.

[TT] Ting, B. S. and B. N. Tien, "Routing Techniques for Gate Array," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol. CAD-2, No 4, Oct. 1983, pp. 301 - 312.

[Va] Vaidya, P. M., "An Algorithm for Linear Programming which Requires $O(((m+n)n^2 + (m+n)^{1.5} n))$ Arithmetic Operations," Proceedings of the Nineteenth Annual ACM Symposium on Theory of

Computation, May 1987, pp. 29 - 38.

[VK] Vecchi M. R. and S. Kirkpatrick, "Global Wiring by Simulated Annealing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol. CAD-2, No. 4, Oct. 1983, pp. 215 - 222.