A global router based on a multicommodity flow model

E. Shragowitz

Advanced CAD Department, Control Data Corporation, Minneapolis, MN 55440, U.S.A.

S. Keel

Mathematics Department, University of Chicago, Chicago, IL 60637, U.S.A.

Revised 30 October 1985

Abstract. The paper describes a new global router applicable for any object with a defined channel structure. This router can be used for the routing of chips and boards with rectilinear or nonrectilinear channel topology in a multilayer environment. This makes it more universal than other known global routers. The router is based on a multicommodity flow model in the graph form with hierarchical cost function. This model is proved to be NP-complete. An algorithm applied to this problem is based on minimax ideas. It moves from the solution optimal w/r to an initial cost function in the direction of the constraints by maximizing on each iteration the decrease in the number of channels at the highest level of overflow and the number of cells with overflown via count. If a solution exists for each iteration, then algorithm will converge in polynomially bounded number of steps to the solution of the multicommodity flow problem. If, for some iteration, a solution does not exist, then an escape procedure is applied and the process continues. Experimental results indicate that performance characteristics of this global router are not inferior to global routers applicable only to gate arrays.

Keywords. Global router, multicommodity flow problem, graph model, minimax algorithm, hierarchical objective function.

1. Introduction

The computational properties of algorithms in DA depend heavily on characteristics of mathematical models proposed for these problems. Many such models [1] belong to the class NP-complete; i.e. optimal solutions for them can be obtained only by exponential algorithms. The majority of mathematical models suggested for the routing problem are proven to be NP-complete. Many practical algorithms for routing, as for other DA problems, are based on the substitution of the original NP-complete models with a hierarchy of polynomially solvable models. Such combinations of models generally sacrifice optimality in order to obtain a quick solution which satisfies the constraints. Such methodology objectively reflects the possibility for the usage of various objective functions, for different hierarchical models or even the formulation of a routing problem as a solution of the system of constraints. The most common structure of heuristic algorithms for routing includes a decomposition step and the subsequent repeated solution of relatively simple subproblems. The decomposition step itself often is solved in a hierarchical way.

A decompositional approach to the routing problem resulted in the introduction of a global routing step in the routing process. Global wiring decomposes the original routing problem into independently solved routing problems for regions of the chip. Several major approaches to the global wiring problem can be identified.

(1) *Top-down hierarchical routing.* This strategy provides hierarchical specification of the location of routes. Implementation of this idea in [2] is based on linear integer programming (or dynamic programming) and properties of some special cases of rectilinear Steiner trees. This method is order independent but has limitations related to the requirement of uniformity for wiring substrate.

(2) *Bottom-up hierarchical routing.* Implementation of this approach in [3] is based on solving, on each hierarchical level, routing problems for arrays of 2×2

Eugene B. Shragowitz was born September 21, 1935 in Leningrad, USSR. He received his M.S. degree in electrical engineering from the College of Electrical Engineers, Leningrad, USSR, in 1958, and Ph.D from the National Scientific Research Institute, Moscow, USSR, in 1971.

He directed numerous research projects in CAD of VLSI, CAD of analog systems, CAD in architecture. From 1981 to 1986, he worked as the Technical Consultant in the CAD department of Control Data Corporation, Minneapolis. Currently he is an Associate Professor of Computer Science Department of the University of Minnesota.

Sean M. Keel was born October 2, 1962 in St. Paul, Minnesota. He received his B.S. degree in Mathematics from Carleton College, Northfield, Minnesota in 1984, and M.S. degree in Mathematics from the University of Chicago in 1985. Currently he is working on his Ph.D dissertation.

During the summer of 1984 he worked as a summer intern for the CAD department of Control Data Corporation, Minneapolis, Minnesota.

super cells. Each super cell consists of 4 cells from the previous hierarchical level. This realization of the bottom-up strategy solves the ordering problem by wiring short connections first and suffers, to a certain extent, from the negative consequences of order dependency. It has an effective rerouting scheme based on a set of heuristic rules.

(3) *Global wiring by simulated annealing.* First suggested in [4], this method does not consider constraints on the channel capacities and it has limitations on the wire shapes. Additional work is required to provide a global router based on this method.

(3) *Independent global wiring of nets with simple rerouting.* This approach to global routing, implemented in [5,6], is similar to the strategy of detailed routing of one net at a time based on the Lee–Moore algorithm. Results of this method of global routing are order dependent.

(5) *Global routing based on minimax algorithm.* This model, described in [7], is based on the cell representation of data. The objective function is the minimum of the maximal overflow of boundary capacities subject to constraints on the number of pins inside each cell. The algorithm consists of two major steps. Initial routing assumes independent routing of each net by the use of the Lee–Moore algorithm. Iterative rerouting is then performed until no positive overflow exists. This algorithm does not prevent an increase of overflow in the process of rerouting. To get a solution with reduced overflow count, many reroutes are tried.

Our approach to global routing is more universal because, in addition to rectilinear, it also allows us to consider nonrectilinear and multilayer structures of wiring channels. This makes it applicable to multilayer chips and boards. We represent global routing as a multicommodity flow problem in the graph form, where nodes represent collections of pins to be wired, and the edges correspond to the channels through which the wires run (Fig. 1). Any required level of generalization can be achieved by such a model ranging from representation of a whole cell as one vertex of the graph and a segment of the routing channel as an edge, to representation of each pin as a vertex of the graph and each routing track as an edge. A limiting factor for implementation of such a model for detailed routing is the polynomial complexity of the global routing algorithms with respect to the size of the graph. This makes decomposition of the original routing problem by global routing procedure followed by channel routing, more computationally effective than detailed routing by global router.

It has been proven that the multicommodity flow problem itself is NP-complete [8]. This, not only the routing problem as a whole, but also the step which

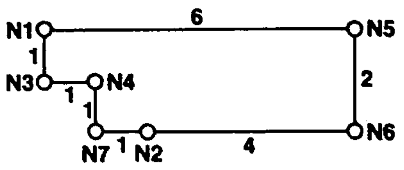

Fig. 1. Multiple objective functions for routing.

performs the decomposition of the routing problem are NP-complete problems. However, such a representation of global routing has certain advantages. It provides the opportunity to consider nonrectilinear and multilayer structures and apply such effective algorithms as the shortest path through the graph [9–11]. Although the proposed heuristic algorithm does not guarantee an existence of a solution (recall the NP-hard nature of the task), it does efficiently generate solutions which satisfy the constraints in most practical situations.

2. Multicommodity flow model of global routing

In this section, we introduce notation and initial formulation of our problem in terms of the graph theory. A directed graph $G = \{I, J\}$ consists of nonempty and finite sets of nodes $I = \{1, 2, \dots, m\}$ and arcs $J = \{1, \dots, n\}$. There are m nodes and n arcs in G. A finite set of commodities $K = \{1, \dots, t\}$ is defined on the set of nodes I. Let us assume that there is only one source and one sink for each commodity and the amount of each commodity is equal to 1. Then, a number a_{ik} is the amount of commodity k produced (consumed) at the node i, and $a_{ik} = 0, \pm 1$. Let ω be a 'nodes-arcs' incidents matrix, where $\omega_{ij} = 0, \pm 1$. Let positive integers p_j and c_j be, respectively, unit shipment cost along the arc and capacity of the arc j. Let x_{jk} be a flow of commodity k by arc j. If flow of minimal cost should be found, then the problem can be formulated in a following way:

$$\sum_j \sum_k p_j |x_{jk}| \Rightarrow \min, \tag{1}$$

$$\sum_j \omega_{ij} x_{jk} = a_{ik} \quad \forall i, k, \qquad \sum_k |x_{jk}| \leqslant c_j \quad \forall i, j, \qquad x_{jk} = 0, \pm 1 \quad \forall j, k.$$

If the arcs of graph G represent segments of the routing channels, capacities c_j represent the number of tracks available for routing in each segment, nodes of the graph represent pins and vias, and certain costs are assigned to the arcs, then model (1) can be interpreted as a model of global routing. A more realistic model of global routing does not require a 'minimal cost' solution, but rather a solution which satisfies constraints of the original multicommodity flow problem and is 'reasonably' good with regard to objective function. This fact plays a major role in the formulation of our algorithm.

It was mentioned earlier that many formulations of the multicommodity flow problem are NP-complete [1] and representation of global routing as a multicommodity flow problem itself, does not provide an immediate advantage. However, we can suggest an effective algorithm for the global routing as a special case of multicommodity flow problem. This algorithm has a polynomial complexity when some additional constraints are satisfied for each step of the algorithm. The algorithm is based on minimax ideas and we named it MM (minimax) algorithm for global routing.

High Level Description of MM

At a high level of abstraction, the global routing problem can be envisioned as an attempt to obtain a minimum of some cost function (which includes routing length, number of vias, etc.) over some constrained domain (in most cases this domain is defined by channel capacities). As a minimum, two approaches to the problem present themselves: we can operate within the constrained domain, attempting to find minima of the cost function among solutions which satisfy constraints, or we can begin with a situation optimal with respect to the cost function, and then move in the direction of the constraints. It is the latter technique, a descent from optimal w/r to initial cost function to acceptable w/r to constraints, which we implemented in the MM algorithm. We begin by solving the multicommodity flow problem with capacity constraints removed. This is equivalent to finding a minimum cost path for each source-sink pair, where individual connection pairs are treated in isolation from one another, and results in a situation optimal with regard to our cost function. Next, we use a minimax iterative procedure in which we choose to reroute that connection which promises to maximize the decrease in the number of edges at the highest level of overflow over the system. A real danger here is to create, as a result of rerouting, new edges with the highest level of overflow or even with the level of overflow higher than the previous highest level. To prevent this from happening, a cost function used in the rerouting procedure penalizes, most heavily, the edges which already have the highest level of overflow, and less heavily, other overflown edges. We continue to honor the initial cost function, when possible, by using it in a tie breaking capacity.

3. Formal description of algorithm

This algorithm consists of an Initial Step and a General Step.

Initial Step

Solve a multicommodity flow problem (1) with constraints on capacities of edges removed.

$$\sum_j \sum_k p_j |x_{jk}| \Rightarrow \min,$$
$$\sum_j \omega_{ij} x_{jk} = a_{ik} \quad \forall i, k, \qquad x_{jk} = 0, \pm 1 \quad \forall j, k. \tag{2}$$

The problem (2) is separable w/r to commodities, i.e. a solution of it can be obtained by an independent solution of the problem (3) for each commodity k independently.

$$\sum_j p_j |x_{jk}| \Rightarrow \min,$$
$$\sum_j \omega_{ij} x_{jk} = a_{ik} \quad \forall i, k, \qquad x_{jk} = 0, \pm 1 \quad \forall j, k. \tag{3}$$

For the situation when each commodity represents one from-to connection with $a_{gk} = 1$ for the source node g, $a_{qk} = -1$ for the sink node q, and $a_{ik} = 0$ for all intermediate nodes, the problem (3) is equivalent to the shortest path from g to q in the graph G [12]. Thus, the multicommodity flow problem in the form (2) is solved by repeated k times shortest path problem for each connection independently. There are many algorithms for shortest path problems [10,11]. An algorithm applied in this work is an adoptation of familiar Dijkstra procedure [9]. Details of our implementation of the shortest path algorithm will be discussed later.

General Step

Let r be an iteration number inside the General Step. Let $[x_{jk}^{(r-1)}]$ be a vector of variables obtained after $(r-1)$ iteration. Then the following operations should be executed at each iteration of the General Step.

Step 1. Find a maximal level of overflow M_r over the graph G after $(r-1)$ iteration

$$M_r = \max_j \left(\sum_k \left| x_{jk}^{(r-1)} \right| - c_j \right).$$

If $M_r \leqslant 0$ then stop, otherwise continue.

Step 2. Find a set of arcs J_r^0 at the maximal and one below maximal level of overflow

$$J_r^0 = \left\{ j \in J : \{ \sum_k \left| x_{jk}^{(r-1)} \right| - c_j + 1 \geqslant M_r \right\}$$

and set of arcs $J_r \in J_r^0$, so that $\sum_k \left| x_{jk}^{(r-1)} \right| - c_j = M_r$.

Remark: An existence (but not uniqueness) of $j_r \in J_r$ follows from the definition of the set J_r^0.

Step 3. Assign new costs to the arcs of the graph G: Let

$$p_j^r = \infty \quad \text{for } j \in J_r^0, \qquad p_j^r = p_j^{(r-1)} \quad \text{for } j \notin J_r^0.$$

Step 4. Define a set of connections K_r^0, so that

$$K_r^0 \left\{ k_r \in K,\ j \in J_r^0 : \{ \left| x_{jk}^{(r-1)} \right| = 1 \right\};$$

and subset $K_r \in K_r^0$, so that:

$$K_r = \left\{ k_r \in K_r^0,\ j \in J_r : \{ \left| x_{jk}^{(r-1)} \right| = 1 \right\}.$$

Step 5. Solve shortest path problems for all $k \in K_r$. Let us denote solutions of these problems x_{jk*}^r. If a solution of the shortest path problem satisfies condition $\min \sum_j p_j^r \left| x_{jk*}^r \right| \neq \infty$ as minimum for one $k \in K_r$, then go to Step 6. Otherwise, go to Step 9.

Step 6. Find connection $k^0 \in K_r$ such, that

$$\sum_j \left(p_{j^0}^r \left| x_{jk}^r \right| - p_j^{(r-1)} \left| x_{jk}^{(r-1)} \right| \right) = \min_{k^* \in K_r} \sum_j \left(p_j^r \left| x_{jk*}^r \right| - p_j^{(r-1)} \left| x_{jk*}^{(r-1)} \right| \right).$$

Step 7. Assign $x^r_{jk} = x^r_{jk^0}$ $\forall j$ for $k = k^0$; $x^r_{jk} = x^{(r-1)}_{jk}$ $\forall j$ for $k \neq k^0$.
Step 8. Increase an iteration counter $r = r + 1$ and go to Step 1.
Step 9. Check condition

$$x^{(r-1)}_{pk} \times x^{(r-1)}_{qk} = 0 \ \forall p,\ q \in J^0_r (p \neq q), \forall k \in K^0_r \qquad (*)$$

(this condition will be discussed later in Lemma 5). If condition $(*)$ is satisfied, then Stop (routing problem does not have a solution for a given set of data). Otherwise, execute the Escape Procedure (see next section) and go to Step 1.

4. Convergence, number of iterations and related subjects

It will be shown in this section that all constraints of the original problem of type $\Sigma_k |x_{jk}| \leqslant c_j$ can actually be satisfied as a result of the execution of Algorithm MM, although the minimum of the objective function need not necessarily be achieved.

Earlier, we denoted by J_r a subset of the set J of such edges, that

$$\sum_k \left| x^{(r-1)}_{jk} \right| - c_j = M_r \quad j \in J_r.$$

A number of elements in J_r is denoted as $m(r)$.

The following two lemmas will be proven together.

Lemma 1. $M_{r+1} \leqslant M_r$.

Lemma 2. *If* $M_{r+1} = M_r$, *then* $m(r+1) \leqslant m(r) - 1$.

Proof. These two lemmas are formulated under the assumption that the solution of the shortest path problem, which satisfies condition $\min \Sigma_j p^r_j |x^r_{jk^*}| \neq \infty$, exists as minimum for one $k \in K_r$ on each execution of Step 5 of the Algorithm MM. Then, it can be stated that according to Steps 4 and 7 of Algorithm MM

(a) $\displaystyle\sum_k |x^r_{jk}| = \sum_k \left| x^{(r-1)}_{jk} \right| - 1$ for $j = j_r$,

(b) $\displaystyle\sum_k |x^r_{jk}| \leqslant \sum_k \left| x^{(r-1)}_{jk} \right|$ for $j \in J^0_r$, $j \neq j_r$,

(c) $\displaystyle\sum_k |x^r_{jk}| \leqslant \sum_k \left| x^{(r-1)}_{jk} \right| + 1$ for $j \notin J^0_r$.

Since $\Sigma_k \left| x^{(r-1)}_{jk} \right| - c_j \leqslant M_r - 2$ for $j \notin J^0_r$, then $\Sigma_k \left| x^r_{jk} \right| - c_j \leqslant M_r - 1$. From (a) and (b) follows, that M_{r+1} is not bigger than M_r. If $M_{r+1} = M_r$, then from (a) follows, that $m(r)$ decreased by 1, from (b) follows, that $m(r)$ did not increase and from (c) follows, that $m(r)$ did not increase. This completes the proof. □

Lemma 3. $M_{r+m(r)} \leqslant M_r - 1$.

This lemma follows from Lemmas 1 and 2 and holds under the same assumptions.

Corrolary 1. *Algorithm MM will converge in $M_r = 0$ under assumptions of Lemmas 1–3.*

An upper bound for the number of steps is $[J] * M_I$, where $[J]$ is an upper bound for $m(r)$ and M_I is a level of overflow after initial step of MM.

Remark 1. From Lemma 3 and Corrollary 1 follows, that the number of steps of Algorithm MM depends on the maximal level of overflow created by the Initial Step of the algorithm. This initial level of overflow can be decreased by choosing for each connection among the routes with equal cost, one which has the minimal number of edges in common with the other routes.

It is possible, that there is no solution to the global routing problem for the given set of data. This will result in the failure of r step of the algorithm MM to find a route of finite cost for all connections, which belong to the set K_r. The same effect may be caused by the wrong topology of certain nets resulting in an incorrect choice of pairs to be connected. Some additional facts can be stated to distinguish these situations.

Let us denote F_k^r minimal cost for the route $k \in K_r$, obtained as a solution of shortest path problem by Step 5 of Algorithm MM.

Lemma 4. *If $F_k^r = \infty$ $\forall k \in K_r$, then J_r^0 is a cutset of the graph G.*

The statement of the lemma immediately follows from Step 5 of Algorithm MM and definition of cutset.

Lemma 5. *If* (a) $M_r > 0$; (b) J_r^0 *is a cutset and* (c) $x_{pk}^{(r-1)} \times x_{qk}^{(r-1)} = 0$ $\forall k$, p, $q \in J_r^0$ $(p \neq q)$; *then the global routing problem does not have a solution.*

Proof. From (a) follows, that vector $[x_{jk}^{(r-1)}]$ does not satisfy constraints of (1). From (b) and (c) follows, that each route crosses cutset J_r not more then once. Thus, any redistribution of routes will not decrease the maximal overflow through cutset J_r^0. This proves the lemma. □

Lemma 5 provides a sufficient condition for nonexistence of solution for (1). A check of this condition is performed by Step 9 of Algorithm MM. If Step 5 of algorithm MM generates infinite solutions only, but conditions of Lemma 5 are not satisfied, then the escape procedure is activated. The function of the escape procedure is to find nets which have wrong topology and are blocking improvement of the route distribution and change their topology by changing pairs of connected nodes.

Escape Procedure

Let a set of arcs J_{rk} for a connection k be a subset of J_r^0, such as $J_{rk} = \{ j \in J_r^0 : \{ |x_{jk}^{(r-1)}| = 1\}$. Let a number of elements in J_{rk} be $v(r, k)$. Then, the escape procedure can be formulated as follows:

Step 1. Find connection $k' \in K_r^0$, such as $v^0(r, k') = \max_k v(r, k)$;

Step 2. Rip-off connection k'.

Step 3. Route a new connection, which will preserve continuity of the net by connecting another pair of terminals of the net broken by removal of connection k'.

Step 4. Add this route to the list of routes and go to Step 1 of General Step of Algorithm MM.

5. Details of implementation of algorithm MM

The MM algorithm takes as input a description of the graph model for the global routing problem and a list of from-to pairs which are to be connected. The data is organized so that we can obtain for each node: (1) position in the grid; (2) arcs incident at the node. For each arc: (1) nodes which it runs between; (2) length; (3) capacity. This information is simply an input to the problem.

To facilitate processing a data base is maintained consisting of

(1) Route list: list of connections (edges comprising routes).

(2) Edge list: list of edges where for each edge is stored: (a) usage (number of routes using the edge); (b) list of routes using the edge.

(3) Overflow list: An array representing the current level of overflow throughout the graph. This includes for each level of overflow: (a) number of edges at this level; (b) list of edges at this level.

With respect to a data base our algorithm can be described as follows:

Step 1. Choose the initial routes for all connections.

Step 2. If the overflow list is empty, then routing is completed, otherwise:

Step 3. For each arc at the highest level of overflow, attempt to reroute one of the routes using the edge, in a manner which maximally decreases the number of edges at the highest level of overflow. Break any ties by consulting the cost function. If the reroute is not found, the escape procedure is invoked and processing is continued at Step 2.

Step 4. Add this route to the data base (update the various lists) and continue processing at Step 2. The implementation relies heavily on a routine MinCost-Path. The routine (described further on) solves the problem of a minimum cost path (w/r to the given function) between source and sink through the current state of the graph. Step 1 is completed by calling MinCostPath with a cost function assigning arc cost equal to the arc length. Step 2 requires traversing the list of arcs at the highest level of overflow. A list of routes using this arc is examined and an attempt is made to find rerouting of the connections.

This attempted rerouting constitutes a call on MinCostPath with a cost function that weights an edge according to its level of overflow.

As it was stated earlier, the algorithm of MinCostPath is an adaptation of Dijkstra procedure, which aims to label the sink node with the cost of the optimal path from the source. The routine maintains, at all times, a list of edges L and relies on the notion of an edge's 'arrow head distance' (AD) from the source. If c1 is the cost of an optimal path from the source to the starting node of the edge, and if c2 is the cost of traversing the edge, then the edges 'arrow head distance' from the source is the sum c1 + c2. With this, the routine can be described by:

Step 1. L starts out empty and the start node is labeled with cost 0. Let n be this node.

Step 2. For each of the edges incident on vertex n set AD and place the edge in L.

Step 3. If L is empty, no path exists. Otherwise, let e be the edge in L with minimal AD. Remove e from L.

Step 4. Let n be the node at the tip of e. If n is already labeled, continue processing at Step 3. Otherwise go to Step 5.

Step 5. Label n with the AD of 3. If n is the sink node, then stop. Otherwise, continue processing at Step 2.

If processing terminates successfully, then a list of decreasing labels can be followed to obtain an optimal path from sink to source.

It is clear from the description of MinCostPath that the fundamental loop is executed, at most, once for each node of the graph, and that the processing in the single loop is heavily dominated by the chore of adding elements to L, and retrieving edges to L with minimal AD. In order to obtain optimal performance, it is necessary to select a representation for L which facilitates addition and deletion of edges based on AD. We selected a balanced binary tree representation, which allows for the deletion or addition of an element with the order of $\log(|L|)$ operations. Thus, the complexity of MinCostPath is $o(m\log n)$, where m is a number of nodes and n is a number of edges for graph G.

6. Multiple objective functions for routing

Two characteristics of routing render the consideration of methods for multiple-goal functions important. First, there is often a set of routes between a given source-sink pair which are all optimal w/r to objective function; secondly, a number of independent or semi-independent objective functions can be applied for ranking routes. Thus, for each function, may exist a number of optimal routes, and for any set of routes, a number of functions for determining optimality. Simultaneously, although each objective function may have a set of optimal routes, it is possible that these sets do not overlap. That is, it is conceivable that no route exists which is optimal w/r to each objective function independently.

Formally: let M be a set of routes between a given source and sink and g_z be an objective function. Let us define a set

$$Q(M, g) = \left\{ R \in M \mid g_z(R) = \min_{r \in M} g_z(M) \right\}, \quad (z = 1, \dots, \alpha)$$

Then possibly $\bigcap_z Q(M, g_z) = \emptyset$. As an example, suppose $g_1(R)$ is the length of a route and $g_2(R)$ is the number of edges in the route. Suppose we consider routes between N1 and N2 (Fig. 1). Then $Q(M, g_1) =$ N1-N3-N4-N7-N2, $Q(M, g_2) =$ N1-N5-N6-N2, and $Q(M, g_1) \cap Q(M, g_2) = \emptyset$. Given this possibility, a reasonable alternative is to rank the functions $g_1, g_2, \dots, g_\alpha$ according to heuristically defined importance and then seek a route in the set S_α where S_1 is a set of all routes optimal w/r to g_1, S_2 is the subset of S_1 of all routes optimal w/r to g_2, etc. In effect, objective functions $g_2, \dots, g_\alpha$ can be used successively to break ties among the routes optimal w/r to g_1, hence obtaining a route with 'ranked optimality'.

A straight forward method of determining a route r in S_α is to decompose the problem into the series of smaller problems of form:

Given objective function g and set of routes M, find $Q(M, g)$. Thus, S_α can be obtained by:

given g_1 and M, obtain $S_1 = Q(M, g_1)$;
given g_2 and S_1, obtain $S_2 = Q(S_1, g_2)$;
etc.

Such decomposition methodology is implicit in the number of tie-breaking routines currently in use. It will be demonstrated here that instead of decomposing the problem, we can transform it into a single MinCostPath problem with an auxiliary objective function f composed of a weighted sum of the g_z. We begin with a proof of the existence of such function f.

Let functions g_z be bounded and D_z be max of g_z over M, and $R > |r|$ for all $r \in M$. Suppose

$$w_z > \sum_{\lambda = z+1}^{\alpha} D_\lambda w_\lambda R, \quad z = 1, \dots, (\alpha - 1).$$

Lemma 6. *If* $g_z(R_1) < g_z(R_z)$ *then*

$$\sum_{\lambda = z}^{\alpha} g_\lambda(R_1) w_\lambda < \sum_{\lambda = z}^{\alpha} g_\lambda(R_2) w_\lambda.$$

Proof.

$$\sum_{\lambda = z}^{\alpha} [g_\lambda(R_2) - g_\lambda(R_1)] w_\lambda$$

$$= [g_z(R_2) - g_z(R_1)] w_z + \sum_{\lambda = z+1}^{\alpha} [g_\lambda(R_2) - g_\lambda(R_1)] w_\lambda$$

$$> w_z - \sum_{\lambda = z+1}^{\alpha} g_\lambda(R_1) w_\lambda > w_z - \sum_{\lambda = z+1}^{\alpha} D_\lambda R w_\lambda > 0. \qquad \square$$

Theorem 1. *Assuming w_{λ} satisfies the conditions of the Lemma 6, route optimal w/r to function $f = \Sigma_{\lambda} w_{\lambda} g_{\lambda}$ belongs to subset S_{λ}, $\lambda = 0, \ldots, \alpha$; i.e.*

$$Q(M, f) \subset \left\{ S_{\alpha} = \bigcap_{\lambda=1}^{\alpha} S_{\lambda} \right\}.$$

Proof. Proof is by induction. For $\lambda = 0$, $S_{\lambda} = M$, so result is evident. Let assume inductively that $Q(M, f) \subset \{\cap_{\varphi=1}^{\lambda-1} S_{\lambda}\}$. Let $r \in Q(M, f)$ and suppose $r \notin S_{\lambda}$. Then by induction base $r \in S_{\lambda-1}$, but $r \in Q(S_{\lambda-1}, g_{\lambda})$, thus $\exists r^1 \in S_{\lambda-1}$ such that $g_{\lambda}(r^1) < g_{\lambda}(r)$. From definition of f follows:

$$f(r) - f(r^1) = \sum_{\varphi=1}^{\lambda-1} \left[g_{\varphi}(r) - g_{\varphi}(r^1) \right] w_{\varphi} + \sum_{\varphi=\lambda}^{\alpha} \left[g_{\varphi}(r) - g_{\varphi}(r^1) \right] w_{\vartheta}.$$

As r, $r^1 \in S_0, \ldots, S_{\lambda-1}$ then $g_{\varphi}(r) = g_{\varphi}(r^1)$ for $\varphi = 1, \ldots, \lambda - 1$. Also by the Lemma 6 $\Sigma_{\varphi=\lambda}^{\alpha}[g_{\lambda}(r) - g_{\lambda}(r^1)]W_{\varphi} > 0$. Thus, $f(r) > f(r^1)$ contradicting the assumption, that $r \in Q(M, f)$. This completes the induction. □

7. Details of implementation of several goal functions

The existence of w_{λ} satisfying the condition of Lemma 6 follows from the fact that any objective function has a finite domain, and hence is bounded. As an indication of how the weights can be obtained in a practical environment, we provide an example which is similar to the actual implementation of the MM algorithm.

For an edge j,

$g_1(j) = 1$ if j is overflowed at the maximum level, 0 otherwise;

$g_2(j) = 1$ if j is overflowed below the maximum level, 0 otherwise;

$g_3(j) = 1$ if j is within some box, 0 otherwise.

Thus, g_1 discourage the use of edges at maximum overflow, g_2 discourages the use of overflowed edges at below maximum level, and g_3 discourages the search for paths outside a box around source and sink. Suppose we want a path with a minimum number of edges at the maximum level of overflow, and among these, one on the minimum general overflow, and among these, one with a minimum number of edges outside of the bounding box.

Let R be a bound on the longest route possible. (The dimensions of the graph provide a reasonable number). Let $D_1 = D_2 = D_3 = 1$. The requirements of the Lemma 6 are that $w_1 > R * w_2 + R * w_3$ and $w_2 > R * w_3$. For large R (i.e. for large graphs) $w_1 = R * R * R$, $w_2 = R * R$ and $w_3 = 1$ will suffice.

The transformation based approach is preferable to the decomposition in two respects. Most visibly, the decomposition creates and processes a number of paths which are abandoned on subsequent tie breaking stages while the transformation obtains a 'ranked-optimal' path directly from a single call to the min-cost routine. In addition, the number of iterations of the min-cost algorithm is decreased by

Table 1
Experimental results

Number of vertices in graphs	50	400	400	400	900	900
Number of edges	84	760	760	760	1740	1740
Number of connections	41	811	987	1229	367	514
Channel capacities	3	7	7	7	10	10
CPU (sec) initial routing	4	301	352	411	653	786
CPU (sec) rerouting	20	143	307	394	105	515
CPU (sec) on Apollo DN300	24	444	659	905	758	1301

the transformation. The fundamental step of min-cost consists of labeling a node with the cost of an optimal path from the source, essentially completing a path to the node. The nodes are labeled in increasing order, hence one iteration is performed for any node which can be reached by a path whose cost is less than the cost of an optimal path to the target node. The effect of the weighted objective function is to increase the cost of some paths optimal w/r to the primary objective function. As a result of the increase, these never expanded, since the goal is reached first by expansion of some unaltered path, hence processing is decreased. Such weighting will not eliminate any potential path, as min-cost will attempt all possible path before failing, nor will any path suboptimal w/r to the primary function be accepted if an optimal w/r to primary exists, since the primary function is weighted above all else (refer to Lemma 6 and Theorem 1 for justification).

Finally, there is a sort of methodological justification for the transformation approach. Since an efficient algorithm exists for the MinCostPath, if the problem can be efficiently transformed to one of MinCostPath, then decomposition, traditionally justified by overcomplexity, is unnecessary and undesirable.

The MM algorithm was implemented into a program written in the C-language and executed on an Apollo DN300. Experimental data for several problems are given by Table 1. Our computer is 10–14 times slower then VAX 11/780 used by many researchers [3]. With this number in mind, we can state that a performance of our general global router is not inferior to the performance of some global routers specialized for gate arrays.

References

[1] Garey, M.R. and D.S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, CA, 1979).

[2] Burstein, M. and R. Pelavin, Hierarchical wire routing, *IEEE Trans. Computer-Aided Des.* 4 (1983) 223–234.

[3] Marec-Sadowska, M., Global router for gate array, Proc. ICCD 84, New York, 1984, pp. 332–337.

[4] Vecci, M.P. and S. Kirkpatrick, Global wiring by simulated annealing, *IEEE Trans. Computer-Aided Des.* **4** (1983) 215–222.

[5] Goto, S., T. Matsuda, K. Taksmizawa, T. Fujita, H. Mizumura, H. Nakamoura and F. Kitajama, Lamda, an Integrated master-slice LSI CAD system, *Integration, the VLSI Journal* **1** (1983).

[6] Nair, R., S.J. Hong, S. Liles and R. Villani, Global wiring on a wire routing machine, Proc. 19th Design Automation Conf., 1982, pp. 224–231.

[7] Ting, B.S. and B.N. Tien, Routing techniques for gate array, *IEEE Trans. Computer-Aided Des.* **4** (1983) 301–312.

[8] Even, S., A. Itai and A. Shamir, On the complexity of timetable and multicomodity flow problems, *SIAM. J. Comput.* **4** (1976) 691–703.

[9] Dijkstra, E.W., A note on two problems in connection with graphs, *Numer. Math.* **4** (1959) 269–271.

[10] Johnson, D.B., Efficient algorithms for shortest path in sparse networks, *J. ACM* **1** (1977) 1–13.

[11] Wagner, R.A., A shortest path algorithm for edge-sparse graphs, *J. ACM* **1** (1976) 50–57.

[12] Ford, L.R. and D.R. Fulkerson, *Flows in Networks* (Princeton University Press, Princeton, New Jersey, 1962).