A Force-Directed Maze Router

Fan Mo, Abdallah Tabbara and Robert K. Brayton
Department of EECS, University of California at Berkeley

Abstract — **A new routing algorithm is presented. It is based on a multiple star net model, force-directed placement and maze searching techniques. The algorithm inherits the power of maze routing in that it is able to route complex layouts with various obstructions. The large memory requirement of the conventional maze algorithm is alleviated through successive net refinement, which constrains the maze searching to small regions. The algorithm shows advantages in routing designs with complicated layout obstructions.**

1. Introduction

With the development of integrated circuit technology, the demand for higher density of system integration on chip challenges physical design automation tools. New problems have emerged in the past few years, such as macro-cell integration, performance driven layout design etc. The trend is toward integrating tens or even hundreds of fairly heterogeneous IP blocks on one chip. To meet high performance and reliability requirements, special nets like power, ground and clock are planned and routed via special tools. The special nets are fixed in the subsequent routing; therefore they become obstructions.

To handle complicated obstructions, maze searching is the most commonly adopted technique [1,4,14,17-23,25]. A maze router is able to find the shortest path from a given source to a given target, if the path exists. However a maze router also has huge memory requirements. Although today a desktop microcomputer may have memories larger than the hard drive of a workstation in 1980's, and one can even argue that virtual memory can provide as much storage as needed, the speed of searching in such a large and increasing size of memory is not tolerable. Another severe problem with a maze router is net ordering [3,4,16,19,25], which has not been well solved yet. Finally, one must deal with strategies required when the maze routing of some nets fails. A few re-routing therapies have been proposed [7,8,10,11,13,14], however none is cheap and efficient. Re-routing does not overcome the net ordering problem. All these difficulties have canceled the benefit of maze routing, limiting its applications to sub-routines of other routing algorithms. Therefore, the problem becomes how to do maze searching efficiently, and how to overcome the net ordering difficulty. In fact these two questions are dependent. The trouble comes from routing net by net. Nets can have multiple pins distributed all around the chip. Even just routing one net may require a search over the entire chip. Not knowing how the subsequent nets are to be routed, the routing of the current net is likely to create blockage to others. The longer the net, the more possible blockages it may generate. Rip-up and re-route does not touch this weakness at all.

Our approach is to refine nets until they become short wires, which can be connected by maze searching fast and locally. The short wires are placed in such a way that they can be connected with a smaller probability of blocking other wires. All nets are refined simultaneously, and are wires placed simultaneously, which removes the global net ordering problem. A multiple star net model is suitable for this task. The stars are placed through force-directed and density-balancing techniques as in cell placement. Density reflects the blockage caused by obstructions and possible wires. By generating stars and breaking long wires into shorter ones, the net topology is successively refined. Together with other net topology optimizations on stars, nets can have an almost unlimited number of shapes. Maze routing comes into play when nets have been sufficiently refined and wires well planned by placing the stars.

In the mixed force-directed maze router, the boundary between global routing and detailed routing is blurred. The later iterations of the successive refinement of nets can be regarded as a coarse detailed routing. Moreover, nets can still have chances of changing their shapes even if maze routing has started. Although there is no theoretical proof that such mixed global and detailed routing is better than separate global and detailed routing, it is clear, intuitively, that tighter integration of the two stages can prevent inaccurate estimation made earlier due to lack of detail layout information. Another advantage is that since force-directed and density-balancing techniques have been successfully used in placement [2,24], this router is potentially capable of interacting with a force-directed placement algorithm. Although in the current version of the router, cells and pins are fixed, it is not hard to make them movable. Furthermore, if cell shapes are allowed to change, logic synthesis may take place during the routing. In addition, the router is incremental; hence it can start with partially routed designs. This feature makes the algorithm suitable for mixed manual-automatic routing.

The rest of this paper is organized as follows. In Section 2, force-directed star placement, net topology optimization and special techniques in maze routing are discussed. Experimental results and discussion are given in Section 3. Section 4 concludes and discusses future directions.

2. Algorithm

2.1. Overview

The algorithm consists of force-directed star placement, net topology optimization, maze searching and re-routing. Force-directed star placement and net topology optimization are iteratively performed. Attractive force and density-balancing force are combined to guide the placement of the stars. The attractive force is simply in accordance with Hooke's Law; while the density-balancing force is a result of the density field reflecting the congestion/sparseness distribution over the layout. Hence the combination of the two forces drives the stars to locations that minimize wire length and decrease congestion. Net topology optimization is a rule based approach and conducted during the star placement. The maze routing starts with all stars placed at optimal locations in the sense that the maze searching is short ranged and the probability of successful routing is high. When maze routing fails, several re-routing methods are attempted.

2.2. Star generation

Initially there are no stars but pins. The initial topology of a net is a spanning tree. Stars are generated when wires meet the criteria described in section 2.4.

2.3. Force-directed star placement

The only movable objects are the stars. A force-directed placement technique is adopted. The formulation of the forces is quite similar to [2,24], so we focus on the unique aspects in the star placement.

2.3.1. Attractive force. Like all other force-directed methods, the attractive forces are in linear form:

$$f_A(s_1, s_2) = k \cdot d(s_1, s_2)$$

where function $d(\cdot)$ gives the Euclidean distance of the two points, and k is a constant.

2.3.2. Density-balancing force. The function of density-balancing force is to guide the star such that the wires attached to the star can get away from congested places. Density is the measure of congestion in a region. We discretize the whole layout by a 3-dimensional array of cubes. Cubes are coarse grids that have the size of about 10 to 50 real routing grids, and size in Z direction is 1. From now on, density refers to the density of a cube. Cells and fixed wires are deterministic sources of density while wires, for which their routing has not been implemented, contribute to density in probability. So, $D(P)$ ($P \in Z^3$), the density of a cube is:

$$D(P) = \sum_r A_R(P, r) + \sum_m A_M(P, m) + \sum_w A_W(P, w)$$

where $A_R(\cdot)$ is the function counting the portion of cell r covering this cube, $A_M(\cdot)$ is the function for fixed wires, and $A_W(\cdot)$ is the function describing probable wires passing this cube. $A_R(\cdot)$ and $A_M(\cdot)$ are trivial, and we concentrate on $A_W(\cdot)$.

The density computation of $A_W(\cdot)$ should reflect the real routing strategy as much as possible. We want to choose one closer to maze searching while the estimation should be simple and efficient. We try to maze route the connection of two points in coarse grids, that is, the cubes. Consider the wave propagation within a rectangular cube array diagonally bounded by the two points. If there is no over-dense cube on a wave front, let all the cubes on this wave front share the probability of wiring. If some cubes are fully occupied by cells and/or fixed wires, the remaining cubes on the wave front share the probability. Sometimes all the cubes on the same wave front are fully occupied, then we still use the average sharing as if there is no blockage at all. This certainly results in cube densities higher than 1, which create pushing forces and get the movable points away from their original locations. One may ask if the two points are both pins, there is no way of getting through the full blockage. As will be shown in section 2.4, a star is to be generated to split the problematic wire and then the star can move and get the two wires away from the congested region.

With density of all cubes computed, the density-balancing force generated at a cube is:

$$f(P) = \sum_{P \neq P'} \frac{\Phi[D(P')](P' - P)}{|P' - P|^2}$$

in which function $\Phi(\cdot)$ is:

$$\Phi(v) = \begin{cases} v - 1.0 & v > 1.0 \\ \phi(v - 1.0) & v \leq 1.0 \end{cases}$$

where ϕ is around 0.1. The idea is that we would like to keep a little effect for the function in the $v \leq 1.0$ region such that less-dense cubes can generate small pulling forces. If we cut the tail by letting $\Phi(v)=0$ when $v \leq 1.0$, the stars may only get information about areas to keep away from, but have no idea about where to go. Maintaining small pulling forces will tell the stars where empty space exists.

We have also experimented with modeling cube density via the estimation of wires crossing cube edges [5]. Results show no improvement as compared to the simple formulation given above. The edge flow model however takes longer computation time. If the cube size is not chosen too small, the simple density formulation is reasonable and efficient.

2.3.3. Total force and star displacement. The total force applied to a star consists of attractive and density-balancing forces. The displacement of a star is guided by the total force. At iteration i, ω_A and ω_B, the weights put on attractive forces and density-balancing forces satisfy:

$$\begin{cases} \omega_A + \omega_B = 1.0 \\ \omega_A = 1.0 - \dfrac{i}{I} \cdot \psi_A \end{cases}$$

where I is the total iteration number, and ψ_A controls the decreasing rate of ω_A, ranging from 0.9 to 1.0.

2.4. Net topology optimization

Four rules apply:

Rule I (Splitting due to length): When the Euclidean distance of a wire exceeds a given threshold, the wire is split into two wires and a star is generated to connect the two new wires. The star is placed at the mid-point of the original wire.

Rule II (Splitting due to angle): When a wire has an angle within the range of $45° \pm \theta_v$ where θ_v is a given value, about 10° or so, the wire is split and a star is generated and placed at the mid-point of the original wire. The reason is that such wires cause large wave fronts in the cube maze searching and thus scatter the routing probability to many cubes. Thus the difference of estimating a cube being taken and it being actually taken is large. In contrast, a wire closer to an axis direction is highly probable to be consistent with its final routing. This is why wires with small angles with the axes are preferred. But to prevent generating too many stars by this rule, we limit the number of stars generated this way.

Rule III (Splitting due to congestion): As mentioned in Section 2.3.2, when a wire crosses a dense region, it is split. The criterion is that if over 90% of cubes on the same wave front have density over 1, the wire is split and the star is placed at the intersection point of the wire and the wave front. If more than one wave front has over 90% dense cubes, the wire is only split at the one with the largest number of dense cubes.

Rule IV (Merging): If two points of the same net are within a given range, the following action is taken. If they are both pins, do nothing; if one of them is a star and the other is a pin, the star is absorbed by the pin; and if both are stars, they are merged into one star to be placed at the mid-point of the original two stars.

The four rules should not be used simultaneously or very frequently, otherwise unnecessary oscillation may occur. And wire connection relations would need to be re-processed after topologies are changed.

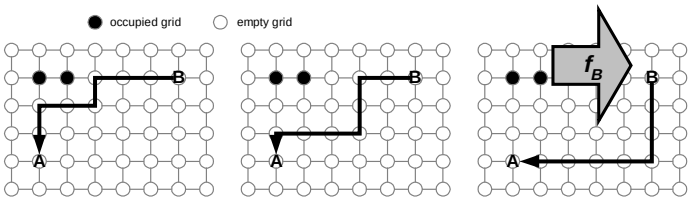
2.5. Maze routing

After generation, merging and placement, the stars are supposed to reach locations that make successful maze routing highly probable. Moreover, the stars break nets into short wire segments, thus the maze connection becomes quite local. This means fast speed and less memory requirements. For simplicity, only uniform grids are considered. Extension to non-uniform grids may not be difficult, since it does not conflict with the star model and the force-directed star placement.

2.5.1. Constrained wave propagation. The main difference of the wave propagation in this algorithm and that in conventional maze routers is that we constrain the search region to a box slightly larger than the bounding box formed by the two points to be connected. This is motivated by the fact that within such constrained region the connection can be finished successfully with high probability. Due to the existence of the stars, it is not exactly a point-to-point search, because there is no reason to force the wire to reach the exact location of a star. Stars are guides for wire connections rather than an actual point that the wire must arrive at. So whenever the searching involves stars, we only require the maze algorithm searches from and/or to a rectangular region centered with the star.

2.5.2. Back tracing. Usually less attention is paid to the back tracing stage in maze routing, because the task of back tracing is thought to be trivial. Since the program written to implement the back-tracing is definite, a definite shape of the final path is derived, given the obstruction. For example, when writing the back tracing part, if the programmer wrote trace-X prior to trace-Y, then the algorithm will always generate a path like the one shown in Figure 1(a). In our

algorithm, we propose two alternative ways of back tracing. One is called local obstruction guided back tracing, as shown in Figure 1(b). When there is more than one choice at a certain grid during back tracing, we choose the one with fewer neighboring blockages. The motivation is to make the subsequent routing easier. This method is quite useful in the case where pins are densely arranged. If care is not taken, early routing to some pins may easily block the routing for the rest. The other approach, as shown in Figure 1(c), is called global density-balancing force guided back tracing. It is similar to the local obstruction guided method in the sense that it tries to leave space, when possible, for the following routing wires. However the difference is that the guidance comes from global congestion distribution, that is, the density-balancing force. In our algorithm the local obstruction and global density-balancing force guided back tracings are combined to indicate the "better" choice from a set of candidates.



(a) normal (b) local obstruction guided (c) global density-balancing force guided

Figure 1. Back tracing

2.6. Re-routing

The difficulty of a maze routing algorithm often boils down to the re-routing strategies. In our algorithm, the maze searching, as described in section 2.5, is constrained locally, so we could extend maze searching for the failed nets to the entire chip. This sounds reasonable, however the cost is very high for large designs, so we do this only when the following strategies fail.

2.6.1. Star re-placement. If the failed wire involves at least one star, it is conceivable that the star may be placed at a wrong location, because the estimation is based on probability. However when a lot of other wires have been finished successfully, their contribution to the density shifts from probability to reality. Using realistic and thus more accurate wiring, the densities and the balance forces can be re-computed. The star of the failed connection is placed again, and then another run of constrained maze routing is tried.

2.6.2. Rip-up and re-route. The key problem of rip-up and re-route is the selection of the set of nets to be ripped. The heuristic used in this router is based on the concept of blocking likelihood. The more a net lies perpendicular to the connection line, the more likely it will block the connection. On the other hand, with same number of occupied grids and same distribution direction, the one lying close to one of the points to be connected is more likely to generate blockage. For each net across the same bounding box as in the constrained maze routing, calculate the distribution of its distance to *s*, one of the two points to be connected:

$$\mu(n) = \frac{1}{N(n)} \sum_{\substack{s_n \in \\ net\_n}} d(s_n, s)$$

$$\sigma^2(n) = \frac{1}{N(n)} \sum_{\substack{s_n \in \\ net\_n}} \left[d(s_n, s) - \mu(n) \right]^2$$

where $N(n)$ is the number of grids within the bounding box that are occupied by net n, and $d(\cdot)$ is the distance between two points. Then define a blocking effect value:

$$e(n) = \frac{\left| \frac{\mu(n)}{d(s,t)/2} - 1.0 \right| + \kappa_1}{\frac{\sigma(n)}{d(s,t)/2} + \kappa_2} \cdot N(n)$$

where $d(s,t)/2$ is the half distance of the two points to be connected, and κ_1 and κ_2 are constants.

It is obvious that we postpone the global maze routing until the last minute because of its high cost. The re-placement is used at first since it is cheap. When it fails, it is very likely that we have to rip up some other nets and route this failed net first. All the involved nets have their blocking effect values calculated. The nets whose effect values exceed a given threshold will be ripped. Then we route the failed net first, and the ripped nets next. When this fails, global maze routing is attempted, or we simply report failure to the user. We also conjecture that push-and-shove might be a good way of doing re-routing, for it is "mild" as compared to rip-up. However push-and-shove has not been implemented in the current version of our router.

3. Experimental results

Three testing examples were chosen from MCNC macro-cell placement benchmarks. The examples are placed by a force-directed macro-cell placer [2], which outputs layouts in standardized LEF/DEF/verilog formats. All these examples use 4 metal layers and a uniform routing grid. Each example has three difficulty levels with respect to the distribution of the obstructions. In level A, a cell occupies metal1 and metal2, and its pins are on metal2. In level B, a set of rectangles on metal3 and 4 are randomly generated and stacked within the cell area. Level C is the hardest, in which the space between cells are partially occupied by metal1 and metal2 objects that simulate power, ground and pre-routed nets. One additional testing example is mcc1-75 obtained from the MCM benchmark set. This example has no obstruction.

The characteristics of the testing examples are given in Table 1. Results for the three MCNC examples are given in Table 2, in which comparison is made between our algorithm and Cadence's Warp Router (Silicon Ensemble, version 5.0) and InternetCAD's Itools Router (Itools, version 1.4.0). Table 3 lists the results for the MCM example, and we cited the results of an MCM router called V4R [12]. All the software except V4R was run on a Sun Ultra 2 workstation with 256MB memory and 2 CPUs. The operating system is SunOS 5.5.1. Our algorithm is programmed with JAVA. The interpreter is Sun JAVA jdk1.2.

Table 1. Characteristics of the examples

examples	#nets	#pins	#grids	#cells	#rectangles
mcc1-75	802	2495	599×599×4	0	0
ami33A	119	279	240×180×4	33	66
ami33B	119	279	240×180×4	33	165
ami33C	119	279	240×180×4	33	297
ami49A	408	933	720×960×4	49	98
ami49B	408	933	720×960×4	49	245
ami49C	408	933	720×960×4	49	441
playoutA	1611	3761	640×880×4	62	124
playoutB	1611	3761	640×880×4	62	310
playoutC	1611	3761	640×880×4	62	558

Table 2. Results of the MCNC examples (1)

	Total wire length (k)			# Vias			Run time (min'sec)		
examples	SE	Itools	This	SE	Itools	This	SE	Itools	This
ami33A	13.1	14.0	13.1	572	579	552	0'6	1'10	0'59
ami33B	13.6	14.5	13.4	583	593	571	0'6	1'01	1'10
ami33C	13.8	16.8	13.6	624	682	609	0'15	1'35	1'10
ami49A	173	182	175	2473	2485	2395	1'07	3'03	15'
ami49B	201	214	208	3212	2991	2954	1'15	3'12	20'
ami49C	223	231	222	3401	3354	3258	1'49	2'58	32'
playoutA	2041	2219	2108	7958	8127	8014	2'05	14'	45'
playoutB	2557	f(?)	2497	9107	—	8973	4'58	19'	44'
playoutC	f(15)	f(?)	f(2)	—	—	—	4'49	18'	58'

(1) f(#) represents # nets fail; f(?) means failure but without detail information

Table 3. Results of mcc1-75

	Total wire length (k)				#Vias				Run time (min'sec)			
layer#	V4R	SE	Itool	This	V4R	SE	Itool	This	V4R	SE	Itool	This
4	394	370	399	392	6993	6373	6207	6392	(1) 0'14	3'40		35'
3	—	373	f(?)	f(5)	—	6453	—	—	— 0'41	4'15		35'
2	—	420	f(?)	f(30)	—	8768	—	—	— 1'28	4'08		37'

(1) We get the results of V4R directly from [12]. The run time is not comparable because the hardware platforms are different.

Table 4. Statistics of our algorithm

		% of successful connections					
		constr. maze		rip-up & reroute		global maze	
examples	# stars	net	wire	net	wire	net	wire
ami33A	125	98.7	97.2	100.0	100.0	—	—
ami33B	143	95.4	95.4	100.0	100.0	—	—
ami33C	172	93.1	92.8	100.0	100.0	—	—
ami49A	1744	97.1	96.5	100.0	100.0	—	—
ami49B	2013	95.2	94.7	99.9	99.9	100.0	100.0
ami49C	2094	91.0	89.1	99.9	99.9	100.0	100.0
playoutA	6044	92.8	90.8	99.9	99.9	100.0	100.0
playoutB	6258	91.2	90.8	99.5	99.4	100.0	100.0
playoutC	6306	90.7	90.0	99.4	99.4	99.9	99.9
mcc1-75 4-layer	4281	91.6	90.3	100.0	100.0	—	—
mcc1-75 3-layer	5099	83.0	78.1	99.7	97.2	99.4	97.5
mcc1-75 2-layer	5518	56.0	50.3	93.3	91.7	96.2	96.1

For the MCNC examples, our algorithm shows its power in routing with complicated layouts (Level C) in terms of routing completeness and total wire length. For simpler layouts like those in Level A and Level B, this router can give results comparable to Warp Router. The run time of our algorithm is large, partially because it is completely a JAVA program. The experiments on the MCM example disclose the characteristics of our router. The MCM example has no obstruction at all, which means the power of maze searching is heavily reduced. So it is not surprising to find that our router had a hard time in finishing the work decently. The really surprising thing is that the Warp Router finished the 100% routing with only two layers in one and half minute. This shows that our router is not expected to become a universal routing tool. On the contrary, it has its unique application area where the power of maze searching can be fully utilized.

We list the statistics of our algorithm in terms of the rate of successful trials at different stages. It is obvious that, except for the MCM example, the constrained maze routing usually has over 90% successful connection in one run. To our best knowledge, there is no other maze style router that can achieve such high success rate in a single run.

4. Conclusion

We presented a router based on the combination of force-directed placement and maze searching techniques. The multiple star net model enables routing estimation and constrained maze routing. The memory and net ordering problems with classical maze routers have been greatly reduced. Experimental results show that the new router performs well in the routing of layouts with complicated obstructions and pre-routed objects. Future work includes enhancing the router with timing-driven features. Since the multiple star net model represents the net topology explicitly, timing analysis and optimization is easy to apply. Furthermore, by dropping the constraint that cells and pins are not allowed to move, an integration of this router with force-directed macro-cell placer is expected.

Acknowledgement

This work was supported by GSRC (grant from MARCO/DARPA 98DT-660, MDA972-99-1-0001).

References

[1] M.H.Arnold and W.S.Scott, "An Interactive Maze Router with Hints", the 25th Design Automation Conference, 1988, pages 672-676

[2] F.Mo, A.Tabbara and R.K.Brayton, "A Force-directed Macro-cell Placer", In the *Proceedings of International Conference on Computer-aided Design*, 2000, pages 177-180

[3] S.Das, S.Nandy and B.Bhattacharya, "High Performance MCM Routing: A New Approach", In the *Proceedings of The twelfth International Conference on VLSI Design*, 1999, Pages 564 -569

[4] J.Cong and P.H.Madden, "Performance Driven Multi-Layer General Area Routing for PCB/MCM Designs", In the *Proceedings of Design Automation Conference*, 1998 , Page(s): 356 -361

[5] Kusnadi and J.D.Carothers, "Routability Checking for General Area Routing Problems", In the *Proceedings of the Twelfth Annual IEEE International Conference on ASIC/SOC*, 1999, Page(s): 201 -205

[6] H.P.Tseng, L.Scheffer and C.Sechen, "Timing and Crosstalk Driven Area Routing", In the *Proceedings of Design Automation Conference*, 1998, Page(s): 378 -381

[7] H.Shirota, S.Shibatani, and M.Terai, "A New Rip-up and Rereoute Algorithm for Very Large Scale Gate Arays", IEEE Customer Integrated Circuits Conference, 1996, Page(s) 9.3.1-9.3.4

[8] A.Hetzel, "A Sequential Detailed Router for Huge Grid Graphs", In the Proceedings of *Design, Automation and Test in Europe*, 1998, Page(s): 332 -338

[9] P.S.Tzeng and C.H.Sequin, "Codar: A Congestion-Directed General Area Router", Digest of Technical Papers, IEEE International Conference on Computer-Aided Design, 1988, Page(s): 30 -33

[10] H.Shin and A.Sangiovanni-Vincentelli, "Mighty: A Rip-up and Reroute Detailed Router", Iternational Conference on Computer-aided Design", 1986, pages 2-5

[11] E.Rosenberg, "A New Iterative Supply/demand Router with Rip-up and Reroute Strategy", The 28th Design Automation Conference, 1987, pages 721-726

[12] K.Y.Khoo and J.Cong, "An Efficient Multilayer MCM Router based on Four-Via Routing", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol 14, Oct. 1995, Page(s): 1277 -1290

[13] M.Bartholomeus and M.Raith, "A New Graph Theoretical Approach to the Selection of Rip-Ups", Euro ASIC '91, 1991, Page(s): 224 -229

[14] C.Y.Lee, "An Algorithm for Path Connection and its Applications", IRE Trans. on Electroinic Computers, EC-10(3), 1961, pages 346-365

[15] F.Curatelli, "Switchbox routing with Rerouting Capabilities in VLSI Design", IEE Proceedings G, Circuits, Devices and Systems, vol 137 3, June 1990, pages 210 -218

[16] P.Groeneveld, "Wire Ordering for Detailed Routing", IEEE Design & Test of Computers, vol 6, dec. 1989, pages 6 -17

[17] E.P.Huijbregts, J.T.S.van Eijndhoven and J.A.G.Jess, "On Design Rule Correct Maze Routing", European Design and Test Conference, 1994. EDAC, The European Conference on Design Automation. ETC European Test Conference. EUROASIC, The European Event in ASIC Design, 1994, pages 407 -411

[18] K.W.Lee.and C.Sechen, "A Global Router for Sea-of-Gates Circuits", In the *Proceedings of European Conference on Design Automation. EDAC*, 1991, pages 242 -247

[19] N.Mani.and N.H.Quach, "Heuristics in the routing algorithm for circuit layout design", IEE Proc.-Comput. Digit. Tech., vol 147, no. 2, March 2000

[20] M.Johann and R.Reis, "Net by net routing with a new search algorithm", In the *Proceedings of the 13th Symposium on Integrated Circuits and Systems Design*, 2000, pages 144 -149

[21] N.Mani and B.Srinivasan, "Optimising ZIg-Zags in Maze-Routing Algorithm", IEEE International Conference on Systems, Man and Cybernetics, vol 4, 1998, pages 3949 -3952

[22] S-W.Hur, A.Jagannathan and J.Lillis, "Timing-Driven Maze Routing", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol 19, Feb. 2000, pages 234 -241

[23] J.Cong, J.Fang and K-Y.Khoo, "Via Design Rule Consideration in Multilayer Maze Routing Algorithms", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol 19, Feb. 2000, pages 215 -223

[24] H.Eisenmann and F.M.Johannes, "Generic Global Placement and Floorplanning", In the *Proceedings of the 35th Design Automation Conference*, 1998, pages.269-274

[25] Routing in the Third Dimension from VLSI chips to MCMs, edited by N.Sherwani, S.Bhingarde, A.Panyam, IEEE Press,1995