Hindawi Publishing Corporation
Modelling and Simulation in Engineering
Volume 2014, Article ID 127359, 7 pages
http://dx.doi.org/10.1155/2014/127359

Research Article

Simulated Annealing Technique for Routing in a Rectangular Mesh Network

Noraziah Adzhar[1] and Shaharuddin Salleh[1,2]

[1]*Department of Mathematical Sciences, Faculty of Science, Universiti Teknologi Malaysia (UTM), 81310 Johor Bahru, Johor, Malaysia*
[2]*Centre for Industrial and Applied Mathematics, Universiti Teknologi Malaysia (UTM), 81310 Johor Bahru, Johor, Malaysia*

Correspondence should be addressed to Noraziah Adzhar; noraziah_adzhar@yahoo.com

Received 21 July 2014; Revised 8 December 2014; Accepted 9 December 2014; Published 22 December 2014

Academic Editor: Min-Chie Chiu

In the process of automatic design for printed circuit boards (PCBs), the phase following cell placement is routing. On the other hand, routing process is a notoriously difficult problem, and even the simplest routing problem which consists of a set of two-pin nets is known to be NP-complete. In this research, our routing region is first tessellated into a uniform $N_x \times N_y$ array of square cells. The ultimate goal for a routing problem is to achieve complete automatic routing with minimal need for any manual intervention. Therefore, shortest path for all connections needs to be established. While classical Dijkstra's algorithm guarantees to find shortest path for a single net, each routed net will form obstacles for later paths. This will add complexities to route later nets and make its routing longer than the optimal path or sometimes impossible to complete. Today's sequential routing often applies heuristic method to further refine the solution. Through this process, all nets will be rerouted in different order to improve the quality of routing. Because of this, we are motivated to apply simulated annealing, one of the metaheuristic methods to our routing model to produce better candidates of sequence.

1. Introduction

In electronic design automation (EDA), wire routing or simply called routing is one of the most important steps in the design of VLSI integrated circuits and robot path planning. The connections for each pair of pins (sometimes called terminals or nodes) on the circuit must satisfy the design rules. High quality routing will give a great impact on the chip performance. Due it its importance and pervasive applications, researchers have shown a high interest in this problem and are studying extensively to improve the optimality and efficiency. An optimal routing should provide minimum cost, shortest distance, or lowest running time.

Suppose we are given a netlist and each net in the list consists of a pair of processing elements. This processing element is also sometimes referred to as node, processing node, or pin in the literature. For N^2 number of pins, there will be at most $N/2$ nets to be routed. However, in real problem, most of the time, the number of connections is usually less than $N/2$ nets because sometimes some of the pins are not assigned to be connected to any other pins. But we restrict ourselves to have maximum number of nets which is $N/2$ and propose a method to achieve 100% routing while satisfying the routing requirements.

However, it is almost impossible to have all $N/2$ nets routed in one layer especially when N is large. In this paper, our goal is to propose a method to minimize layers in rectangular mesh network while satisfying the routing requirements. In order to do this, we need to maximize number of connections in each layer and thus it is important to have each net routed in shortest way. The complexity of a routing region is bounded by limitations of the number of processors and the energy level, and this energy level highly depends on the netlist.

Routing problem is an interesting topic and is being studied extensively among researchers. Several problems that have been addressed in this field include providing deadlock-free routing scheme [1], obstacle-aware routing problem with net length constraints [2], and also longest path for each net with the presence of obstacles [3]. Most of routing algorithm

in this field is an extension of Lee's algorithm [4–6]. Even though it guarantees to find minimum cost possible path if it exists, its searching nature based on wave propagation is slow. Therefore, we are motivated to propose another routing algorithm based on stochastic optimization using simulated annealing technique and Dijkstra's algorithm to solve shortest path part of the problem. Our proposed routing algorithm can be applied to problems of any dimensions.

This paper can be organized as follows. Section 2 states the problem and discusses our routing layout model. Section 3 explains the shortest path problem which uses Dijkstra's algorithm and the implementation of simulated annealing technique to the problem. The simulated work and whole results are discussed in Section 4. The paper concludes with conclusion in Section 5.

2. Problem Statement

Routing in printed circuit board (PCB) is a process of determining and prescribing paths between various electronic components in order to establish a connection between given source point and its target. Routing in modern chip is a notoriously difficult problem, and even the simplest routing problem which consists of a set of two-pin nets is known to be NP-complete [7].

In this problem, our routing region is assumed to be divided uniformly into $N_x \times N_y$ square cells and each cell contains p pins. Unless stated otherwise, p is assumed to be 1. With focus on wiring, we shall assume the location of pins has been specified. This process is as illustrated in Figure 1.

Suppose we are given a set of routing requirements which consist of two-pin nets $N = \{N_1, N_2, \ldots, N_n\}$, where $N_1 = (S_1, T_1)$, $N_2 = (S_2, T_2), \ldots, N_n = (S_n, T_n)$ with S and T being source and target pins, respectively. The ultimate goal of this routing problem is to minimize number of layers needed to perform complete routing. In order to do this, we first seek the maximum number of interconnected structures for set of nets N for each layer(s) while minimizing level of congestion throughout the region. The objective function for this problem can be defined as

$$\text{Max } R = \sum_{i=1}^{m!} \sum_{j=1}^{m} q_{ij} a_{ij},$$

$$\text{where } q_{ij} = \begin{cases} 1 & \text{nonblocking} \\ 0 & \text{blocking,} \end{cases} \tag{1}$$

$a_{ij} = m \times m$ matrix representing order and pair,

m = total number of nets, i = order, j = nets.

Minimizing energy level is also one of the objectives. In order to calculate the energy level in the routing region, we simply replace a_{ij} with d_{ij}, so the energy function becomes

$$E = \sum_{j=1}^{m} q_{ij} d_{ij}, \quad \text{for } i = 1, 2, 3, \ldots, m, \tag{2}$$

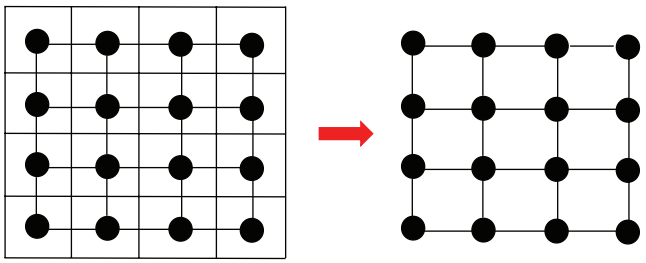

Figure 1: Partitioned layout into $N_x \times N_y$ rectangular array.

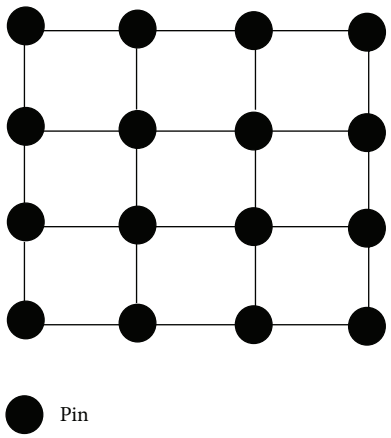

Figure 2: Routing region of size 4×4.

where i is the sequence (i can be referred to as number of iterations too).

Complete wiring on this mesh network should obey the design rules and satisfy the necessary conditions as follows.

(i) Consider

$$N_i \cap N_j = \varnothing, \quad i \neq j. \tag{3}$$

(ii) For each N_i, there must be exactly one connection only.

(iii) All paths can cross but should not overlap each other.

(iv) Each connection will be made using the communication links with no specific direction. This allows for a simpler representation of the routing configuration, even though it reduces freedom during routing.

As a result of its importance to the industry, we are motivated to produce significant method to perform optimal routing. Figure 2 illustrates the conceptual structure of 4×4 rectangular mesh model.

The major features of this grid model are described as follows.

(i) We define $p \times p = p^2$ as the network size. For 4×4 model, we designated an ID for each pin in the first row as 1, 2, 3, and 4, respectively, with the upper left corner as 1 and the upper right corner as 4. Similarly, the ID for each pin in the last row were 13, 14, 15,

and 16, respectively, with the lower left corner as 13 and the lower right corner as 16. Therefore, each pin in the network will have a unique ID.

(ii) All pins can communicate with each other to the left, right, upward, or downward direction as long as the path is not overlapped.

(iii) Each connection is independent of other connections.

(iv) Each pin will belong to only one net.

(v) Only one pin could log data in a given subbus at a time. This operation was performed sequentially and depended highly on the net ordering. Therefore, earlier connection would block later paths.

(vi) For the sake of simplicity, each link in the network held a value of 1 as its weight.

3. Routing Model

Shortest path problem is one of the components in our research work. Basically, this problem seeks the solution to this question: How can processor i be connected to processor j in such a way that it has the minimum distance travelled? The prototype of most nowadays shortest path problem was actually the travelling salesman problem (TSP). In TSP, the salesman needs to perform a complete tour starting from a point, passing each station only once, and coming back to the starting point while minimizing the distance travelled.

3.1. Shortest Path Procedure. In our problem, it is important to have all pins connected in shortest way since reducing the energy level in routing region is one of our objectives. In this way, we can provide larger routing space to route remaining nets, thus maximizing number of successive nets on each layer. Several shortest path algorithms have been proposed in literature such as Dijkstra's algorithm which solves single-source problem [8]. However, all the weights in the graph should be given in positive value. Otherwise, it will lead to acyclic graphs in which most of the time cannot provide the right shortest path. Bellman and Ford improve Dijkstra's algorithm deficiency by producing Bellman-Ford algorithm which works on negative weights [9, 10]. Floyd-Warshall algorithm solves all pairs of shortest path and uses matrix to update all the path values [11, 12]. We decided to use Dijkstra's algorithm as the tool to solve shortest path problem due to several reasons.

(i) Our connected graph which is in the form of rectangular array will not use negative values as its weight.

(ii) We only need to find a single path connecting processors i and j in a time; therefore Floyd-Washall's algorithm may not be suitable.

In Dijkstra's algorithm, suppose we are given a connected directed or undirected graph $G = \{V, E\}$, where V is the set of all vertices and E is the weight at every edge representing cost or time. This algorithm works by solving subproblem k, which searches the path between the source vertex and all adjacent vertices. Dijkstra's algorithm begins by assigning initial value to each node in the graph and improves them step by step.

3.2. Heuristics Improvement. Traditionally, routing in rectangular array can be performed sequentially or concurrently. Sequential routing is perhaps the most straightforward strategy where one has to select specific net ordering and route the nets one by one according to the order. In this research, we would like to apply the sequential routing approach. Therefore, the quality of the routing solution will greatly depend on the net ordering and the task to find such net ordering has proven to be NP-hard [13]. While Dijkstra's algorithm guarantees to find shortest path for a single net, each routed net will form obstacles for later paths. This will add complexities to route later nets and make its routing longer than optimal path or sometimes impossible to complete. Today's sequential routing often applies heuristic method used to conduct rip-up and reroute process to further refine the solution. Through this process, the connections for some nets will be removed and rerouted in different order to improve routing quality. Because of this, we are motivated to apply simulated annealing, one of the metaheuristic methods to our routing model to produce better candidates of sequence.

3.2.1. Simulated Annealing Routing Algorithm. In this research, a heuristic method called simulated annealing is proposed to produce better quality net ordering. It is a probabilistic method which is first simulated by Metropolis et al. and Kirkpatrick et al. [14, 15]. This algorithm now has become a very useful tool in solving a variety of combinatorial optimization problems.

This method is motivated from the theory of annealing in solids. The term simulated annealing derives from the roughly analogous physical process of heating and then slowly cooling a substance to obtain a strong crystalline structure [14]. The parameters that have been used in this study are as follows.

(1) Initial permutation $L_0 = \{1, 2, 3, \ldots, n\}$.

(2) Initial temperature $T_0 = 100^{\circ}$C. This high temperature is to allow a move to almost all potential neighbourhood solutions.

(3) Final temperature $T_f \cong 0^{\circ}$C. When the temperature reaches zero or near to zero, the probability of acceptance also will be near to zero and there is almost no chance of accepting worst moves.

(4) Temperature reduction rules are $T_k = \alpha T_{k-1}$. The temperature decrement is set to geometric decrement where T_k is current temperature and T_{k-1} is previous iteration temperature. In [6], experience has shown that the best suitable value of α should be $0.8 < \alpha < 0.99$. If α value is set too small, the system will rapidly cool and may contain trapped local minimum. If α value is set too high, the cooling procedure will take a longer time as the system will be running more iterations until it reaches the stopping criteria. Therefore, here we choose $\alpha = 0.95$.

(5) Moves are based on pairwise interchange for two nets randomly.

(6) Only one solution is chosen as neighbouring solution at a time.

(7) Stopping criteria are $T_k < T_f$.

During the simulation, the temperature will be lowered gradually until the system "freezes" and no further changes occur. At each temperature, the simulation must proceed long enough for the system to reach steady state or thermal equilibrium. This method avoids being trapped at local minima by accepting sometimes uphill moves. This acceptance is determined by using Boltzman probability:

$$P(\Delta E) = e^{-\Delta E/T_i}, \tag{4}$$

where ΔE is the difference costs between the current solution and previous one while T_i is current temperature. For a given annealing schedule of temperature $T = \{t_1, t_2, \ldots\}$, our implemented simulated annealing algorithm is given as follows.

(1) Determine an initial sequence for all to be routed nets, called L_0. Set $L = L_0$.

(2) Then Dijkstra's algorithm is applied to compute shortest path for each net. For each S_i, assign to every node a tentative distance value: set it to zero for our initial nodes and to infinity for all other nodes.

(3) Mark all nodes as unvisited. Set the initial node as current. For the current node, consider all of its unvisited neighbors and calculate their tentative distances. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one.

(4) When we are done considering all of the neighbors of the current node, mark the current node as visited. A visited node will never be checked again.

(5) If T_i has been marked as visited, then compute $d(S_i, T_i)$. If T_i is not reached, therefore a blockage has occurred. Abandon the net and continue with the next net in L. Repeat the process for every net in L.

(6) Compute initial energy, E_0, using (2). Compute the number of successful routed nets, R_i. Mark that sequence as "accept."

(7) From L, generate new sequence by swapping any two different elements randomly.

(8) Set $L = L'$ and repeat Steps 2–5. Evaluate the new energy for the new sequence, $E(L')$ and the new $R(L')$.

(9) If $R(L') > R(L_0)$, proceed to Step 13.

(10) If $R(L') < R(L_0)$, reject the sequence and repeat Step 7.

(11) If $R(L') = R(L_0)$, compute the energy change, $E(L')$. If $E(L) - E(L') < 0$, proceed to Step 13. Otherwise, go to Step 12.

(12) Apply Boltzmann's probability. If $P(\Delta E) = e^{-\Delta E/T_i} > \varepsilon$, where $\varepsilon \sim U(0, 1)$, go to Step 13. Otherwise, reject the move and repeat Step 7.

Table 1: Calculation for maximizing number of R.

i	j	N_i	q_{ij}	a_{ij}	$R = \sum_{i=1}^{m!}\sum_{j=1}^{m} p_{ij}a_{ij}$
1	1	N_1	1	$\begin{bmatrix}1&0&0\\0&0&0\\0&0&0\end{bmatrix}$	$R = \begin{bmatrix}1&0&0\\0&1&0\\0&0&0\end{bmatrix}$ $^*N_1, N_2$ survive.
	2	N_2	1	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	
	3	N_3	0	$\begin{bmatrix}0&0&0\\0&0&0\\0&0&1\end{bmatrix}$	
2	1	N_2	1	$\begin{bmatrix}0&0&0\\1&0&0\\0&0&0\end{bmatrix}$	$R = \begin{bmatrix}1&0&0\\0&1&0\\0&0&0\end{bmatrix}$ $^*N_1, N_2$ again survive.
	2	N_3	0	$\begin{bmatrix}0&0&0\\0&0&0\\0&1&0\end{bmatrix}$	
	3	N_1	1	$\begin{bmatrix}0&0&1\\0&0&0\\0&0&0\end{bmatrix}$	

(13) Accept the candidate sequence as a current solution, and set $L = L', E(L) = E(L')$ and $T = T_k$. Using reducing parameter, $\alpha = 0.95$, update the temperature counters and parameters. Set $k = k + 1$, $T_{k+1} = \alpha T_k$ and repeat Step 7. If the total number of iterations or epochs is less than M, where M is the specified number to control the number of elapsed iterations, go to Step 7. Otherwise, stop.

4. Simulation Results

Through the objective function in (1), we would like to illustrate how the equation works. Suppose we are given a routing region of size 3×3 and the net requirement is as follows:

$$N_1 = (2, 8), \quad N_2 = (3, 7), \quad N_3 = (1, 9). \tag{5}$$

Table 1 illustrates the calculation for number of R (successful routed nets in certain order).

For example, suppose we are given a routing region of size 4×4 and the net requirement is as follows:

$$\begin{aligned} &N_1 = (3, 9), \quad N_2 = (2, 13), \\ &N_3 = (4, 14), \quad N_4 = (1, 15), \quad N_5 = (6, 16). \end{aligned} \tag{6}$$

Table 2 illustrates the process. At the second iteration, N_1 is swapped with N_5 producing the sequence N_5, N_2, N_3, N_4, N_1. Through this sequence, the number of blocked nets is reduced; thus number of R is increased even though E is increasing too. However, as the process continues, the value of E will gradually be lowered into an optimal and acceptable result.

Table 2: Calculation of energy level in a routing region.

i	j	N_i	q_{ij}	d_{ij}	R	E
	1	N_1	1	$d(3,9) = 4$		
	2	N_2	1	$d(2,13) = 4$		
1	3	N_3	1	$d(4,14) = 5$	3	13
	4	N_4	0	$d(1,15) = 0$; *path is blocked		
	5	N_5	0	$d(6,16) = 0$; *path is blocked		
	1	N_5	1	$d(6,16) = 4$		
	2	N_2	1	$d(2,13) = 4$		
2	3	N_3	0	$d(4,14) = 0$; *path is blocked	4	19
	4	N_4	1	$d(1,15) = 5$		
	5	N_1	1	$d(3,9) = 6$		

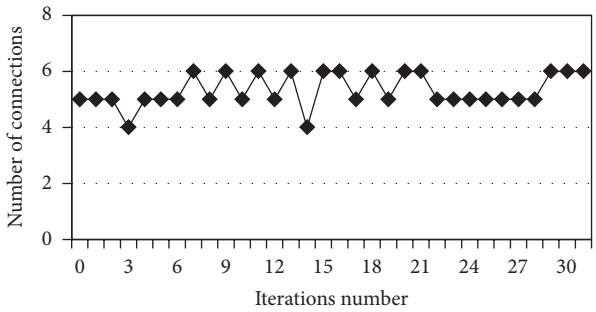

Figure 3: Number of connections at all iterations for $p = 4$.

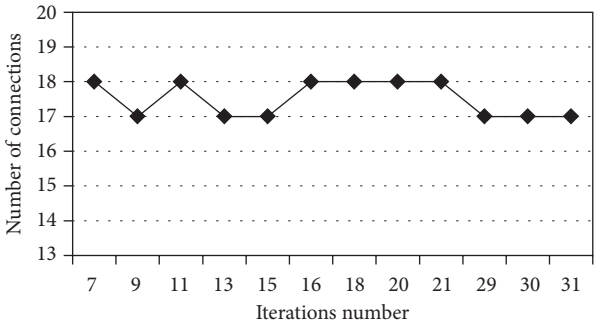

Figure 4: Energy level improvements among $R = 6$.

Notice that E reflects the total length of wire needed to perform wiring in PCB and the complexity/congestion in that routing region. Our proposed algorithm has been implemented by using Microsoft Visual C++ 2010 [16] and run on an Intel Core2 Duo CPU 2.00 GHz machine with 3 GB memory. We first tested our simulation program to problem size of 4^2 with 16 pins and random 8 nets to be routed. Graph in Figure 3 summarizes the process. Our algorithm improves the maximum number of R up to 6 nets out of 8 at as early as iteration number 6 and no best result was obtained after that. Therefore, for this problem, these six nets will be routed on the first layer and the remaining two nets on the second layer to achieve complete routing.

Figure 4 shows energy level improvement among all acceptable moves and the lowest energy level recoded for $R = 6$ was $E = 17$ which takes about only 70.83% from overall routing layout. By accepting some uphill moves based on Boltzman's probability, a less congested routing layout has been generated.

Further, our program has been tested to a bigger data set which has 64 pins and 32 nets on 8×8 network size. From the graph in Figure 5, the maximum number of R matching or connections was obtained only after iteration number 24 which is when $T = 29.20^{\circ}$. This is due to large number of nets and complicated routing requirements. Therefore, when N is large, the program is allowed to iterate further. Since our program takes less than a second to generate solution at each iteration, allowing the program to continue for a longer time does not really affect the computational time.

Recall, our objective is to maximize number of connections while minimizing the energy level. However, move that resulting to maximum connections is always accepted, neglecting its energy level. But if current value of E is not optimal, it will gradually decrease to an acceptable one as the program continues. As in Figure 6, the lowest energy level recoded for $N = 8$ at $R = 22$ is $E = 67$.

Then the annealing process is repeated to find maximum matching on second layer among remaining 10 nets. Figure 7 shows maximum number of R obtained at every iteration. As we can see, the algorithm is able to route all remaining 10 nets at 8th iteration. Thus, only two layers are needed to perform complete routing for this respective requirement for $p = 8$.

A clear result of accepting several worst moves with condition results to a lower energy level and this is summarized in Figure 8.

Figure 9 illustrates the whole wiring tracks for each layer. Both layers take less than 60% of the routing region. Therefore, the congestion level on the first and second floor is balanced. Therefore, the overhead cost is reduced and, from the electrical view, each layer holds quite the same amount of heat transfer and thus the performance is improved. This suggests that our algorithm is suitable for adoption into a real problem.

Our proposed algorithm manages to produce high number of connections in various network sizes with less congestion level throughout the region. Therefore, it suggests that this method is suitable for adoption into real problem.

5. Conclusion

This paper has described a basic routing model which usually uses rectangular grid array as routing layout. This routing

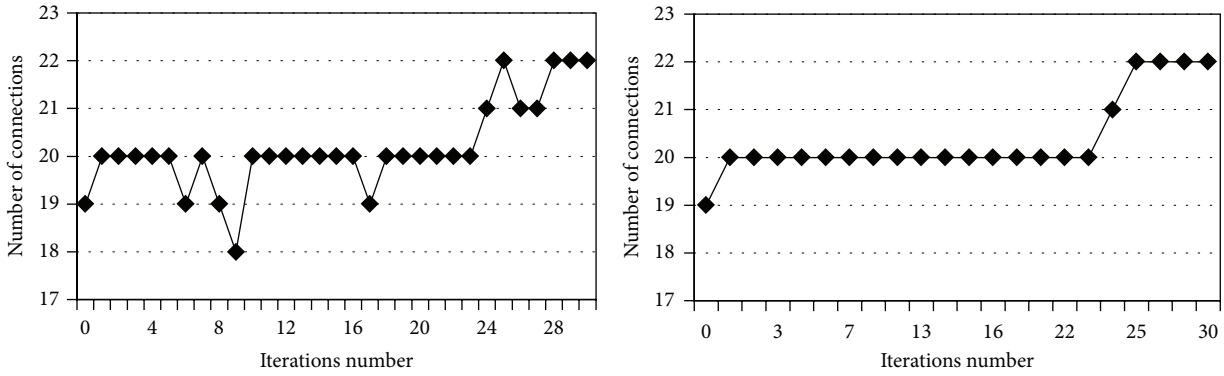

Figure 5: (a) Number of connections at all iterations. (b) Number of connections among acceptable moves.

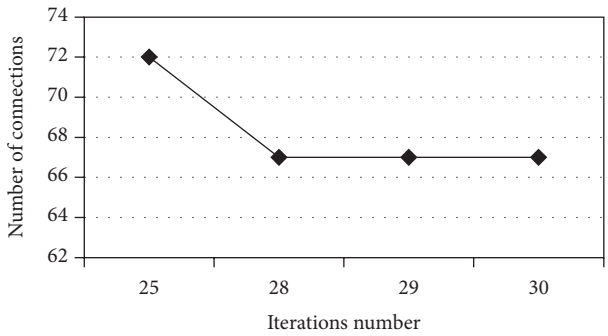

Figure 6: The algorithm gradually decreases the energy level among $R = 22$.

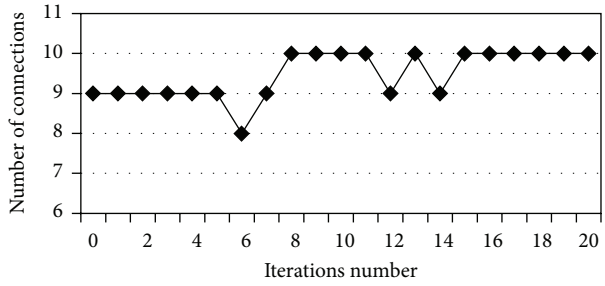

Figure 7: Number of connections at all iterations on second layer.

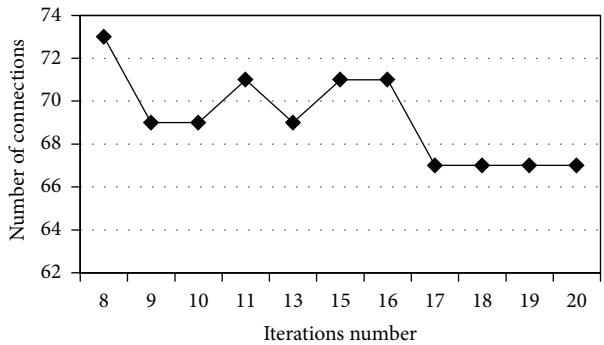

Figure 8: Minimization of energy level.

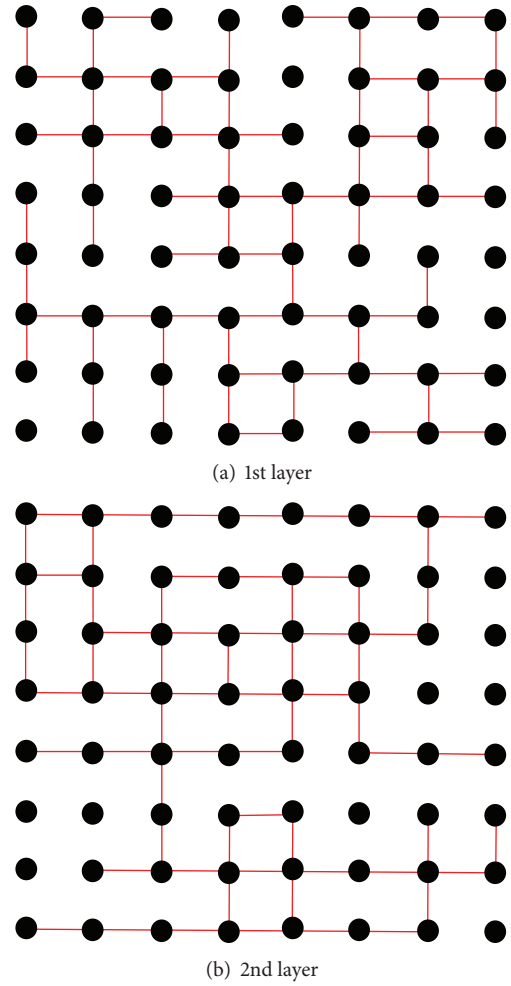

Figure 9: Final wiring tracks on each layer.

graph can be modeled as $G = \{V, E\}$, where V represents a node and (u, v) represents the grid edge or boundary between nodes u and v. In this paper, we propose a method to maximize the number of connections in $p \times p$ grid consisting

of an array of mesh connected processing elements (PEs) with $N/2$ number of nets. Notice that each net will be routed sequentially. Therefore, once a net is routed, it will block later path and this will add more complexities to route remaining nets. In order to overcome this, an intelligent heuristic based algorithm using annealing technique has been built to produce better net ordering to further refine the sequence. Together, we infuse the classical shortest path method into the algorithm as the tool to provide shortest route for each net. Our simulation results show that our proposed algorithm is able to minimize number of layers for a complete PCB routing by first maximizing number of connections at each layer. We also provide a low energy level throughout the routing layout by having all nets routed in shortest way. The results also show that accepting some uphill moves with Boltzman's probability leads to a better result and this is how our heuristic algorithm works. For further research, we would like to do some modifications and improvements on the net ordering.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to thank the Ministry of Education Malaysia for the scholarship given and Universiti Teknologi Malaysia for supporting this research under Vote number 04H43.

References

[1] X. Duan and J. Wu, "Deadlock-free routing scheme for irregular mesh topology NoCs with oversized regions," *Journal of Computers*, vol. 8, no. 1, pp. 27–32, 2013.

[2] J.-T. Yan and Z.-W. Chen, "Obstacle-aware length-matching bus routing," in *Proceedings of the International Symposium on Physical Design (ISPD '11)*, pp. 61–67, ACM, Santa Barbara, Calif, USA, March 2011.

[3] Y. Jin-Tai, J. Ming-Ching, and C. Zhi-Wei, "Obstacle-aware longest path using rectangular pattern detouring in routing grids," in *Proceedings of the 15th Asia and South Pacific Design Automation Conference (ASP-DAC '10)*, pp. 287–292, January 2010.

[4] C. Y. Lee, "An algorithm for path connections and its applications," *IRE Transactions on Electronic Computers*, vol. 10, pp. 346–365, 1961.

[5] N. Mani, "Heuristics in the routing algorithm for circuit layout design," *IEE Proceedings: Computers and Digital Techniques*, vol. 147, no. 2, pp. 59–64, 2000.

[6] Y.-L. Lin, Y.-C. Hsu, and F.-S. Tsai, "Hybrid routing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 9, no. 2, pp. 151–157, 1990.

[7] L. T. Wang, Y. W. Chang, and K. T. Cheng, *Electronic Design Automation: Synthesis, Verification, and Test*, Elsevier, Boston, Mass, USA, 2009.

[8] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[9] R. Bellman, "On a routing problem," *Quarterly of Applied Mathematics*, vol. 16, pp. 87–90, 1958.

[10] L. R. Ford Jr., "Network flow theory," Paper P-923, RAND Corporation, Santa Monica, Calif, USA, 1956.

[11] R. W. Floyd, "Algorithm 97: shortest path," *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.

[12] S. Warshall, "A theorem on boolean matrices," *Journal of the Association for Computing Machinery*, vol. 9, pp. 11–12, 1962.

[13] L. C. Abel, "On the ordering of connections for automatic wire routing," *IEEE Transactions on Computers*, vol. 21, no. 11, pp. 1227–1233, 1972.

[14] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[15] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[16] S. Salleh, A. Y. Zomaya, and S. A. Bakar, *Computing for Numerical Methods Using Visual C++*, Wiley-Interscience, Hoboken, NJ, USA, 2008.