Journal of Physics: Conference Series

PAPER • **OPEN ACCESS**

An enhanced simulated annealing routing algorithm for semi-diagonal torus network

To cite this article: Noraziah Adzhar and Shaharuddin Salleh 2017 *J. Phys.: Conf. Ser.* **890** 012082

IOP Conf. Series: Journal of Physics: Conf. Series **890** (2017) 012082 doi:10.1088/1742-6596/890/1/012082

An enhanced simulated annealing routing algorithm for semi-diagonal torus network

Noraziah Adzhar[1] and Shaharuddin Salleh[2]

[1]Faculty of Industrial Sciences & Technology, Universiti Malaysia Pahang, 26300 Gambang, Kuantan, Pahang, Malaysia.
[2]Department of Mathematical Sciences, Universiti Teknologi Malaysia, 81310 Skudai, Johor Bahru, Johor, Malaysia.

E-mail: noraziahadzhar@ump.edu.my

Abstract. Multiprocessor is another great technology that helps in advancing human civilization due to high demands for solving complex problems. A multiprocessing system can have a lot of replicated processor-memory pairs (henceforth regard as net) or also called as processing nodes. Each of these nodes is connected to each other through interconnection networks and passes message using a standard message passing mechanism. In this paper, we present a routing algorithm based on enhanced simulated annealing technique to provide the connection between nodes in a semi-diagonal torus (SD-Torus) network. This network is both symmetric and regular; thus, make it very beneficial in the implementation process. The main objective is to maximize the number of established connection between nodes in this SD-Torus network. In order to achieve this objective, each node must be connected in its shortest way as possible. We start our algorithm by designing shortest path algorithm based on Dijkstra's method. While this algorithm guarantees to find the shortest path for each single net, if it exists, each routed net will form obstacle for later paths. This increases the complexity to route later nets and makes routing longer than optimal, or sometimes impossible to complete. The solution is further refined by re-routing all nets in different orders using simulated annealing method. Through simulation program, our proposed algorithm succeeded in performing complete routing up to 81 nodes with 40 nets in 9x9 SD-Torus network size.

1. Introduction

A two-dimensional rectangular mesh network [1-3] constitutes an important class of network interconnection [4] and is the most powerful candidate for general-purpose routing due to its simplicity and scalability. This attracts researchers to deploy rectangular mesh as their NoC topology for example aSOC [5], Cliché [6], and Hermes [7]. However, this network is not symmetric at the edges and causes uneven traffic distribution. Its limited number of links or channels also degrades connection performance as the network size increases. Therefore, several improved topologies have been proposed in the literature to overcome these drawbacks such as DMesh [8], Xtorus [9], SD-Torus [10] and DTorus which is also being discussed in [10]. Among these proposed network interconnections, the topological properties of SD-Torus lead to a better implementation due to their regularity and symmetricity.

SD-Torus network is a square network in architecture to ensure the existence of complete diagonal wraparound. For $p \times p$ SD-Torus network, $p \geq 2$ because if p is less than 2, then no diagonal wraparound exists. This network is regular and each node in the network has the same node degree of six. Each node connected to its six neighbour nodes in east, west, south, north, northeast, and southwest direction, respectively. From the standpoint of each node, the SD-Torus network looks alike, thus it is symmetric. These two properties make this network easily scalable. Figure 1 shows the SD-Torus network. All wraparound connections for the peripheral nodes are omitted for clarity.

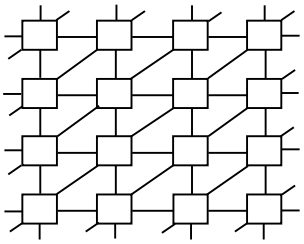

Figure 1. SD-Torus interconnection networks model.

In this paper, we would like to propose a routing algorithm which will determine a specific path optimally for a message to traverse from its source node to destination node. The rest of this paper is organized as follows. Section 2 is the problem statement. In Section 3, the infused Dijkstra's and simulated annealing routing algorithm are proposed. Section 4 presents the simulation part under random traffic, and the performance comparison is divided into two parts; comparison between topologies and comparison between our proposed method and Greedy Method. Finally, Section 5 concludes the paper.

2. Problem Statement

This routing graph can be modeled as $G(N,E)$, where graph N represents a set of processing nodes, and the element of graph E is the link of communication between nodes. Supposed we were given a set of routing requirements, which consists of pairs of processing nodes (henceforth regards as nets), $N = \{N_1, N_2, ..., N_n\}$ where $N_1 = (S_1, T_1), N_2 = (S_2, T_2), ..., N_n = (S_n, T_n)$ with S and T as the source and target nodes, respectively. The ultimate goal of this routing problem was to minimize the number of layers needed to perform a complete routing. In order to do this, we first sought the maximum number of interconnecting structure for a set of nets, N while minimizing the level of congestion throughout the region. The objective function for this problem was defined as:

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

where $q_{ij} = \begin{cases} 1 & \text{non blocking} \\ 0 & \text{blocking} \end{cases}$, $a_{ij} = m \times m$ matrix, representing order and pair, $m =$ total number of nets, $i =$ order, and $j =$ nets.

A complete connection on this SD-Torus network should obey the design rules and satisfy the necessary conditions as follows:

(i) $N_i \cap N_j = \varnothing, i \neq j$

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

(iii) Each path 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.

3. Routing Algorithm

Routing algorithm will manage network traffic and determine the best route for sending message from the source node to the destination node. Once a message is delivered, the path it takes will be blocked and cannot be used to route other pairs of nodes. This will increase network traffic and makes later route longer than optimal and sometimes impossible to complete, thus, increase the energy level throughout the network as well as the cost (mainly refers to total link numbers). Several routing methods have been proposed in the literature to avoid deadlock [11] so as to provide high throughput on adversarial traffic patterns, matching, or exceeding fully randomized routing [12].

A shortest path algorithm is crucial for an interconnection topology. It was important to have all nodes connected in the shortest way to reduce energy level and network cost. In addition, we could provide larger routing space to route the remaining nets, thus maximizing the number of successive nets in each layer. In this paper, Dijkstra's algorithm is used as a tool to provide best path since all the weights in the graph is positive values. Current sequential routing often applies heuristic methods to further refine the solution. Through this process, the connections for some nets will be swapped and re-routed in a different order to improve the routing quality. In this research, a probabilistic method simulated by Metropolis et al. and Kirkpatrick et al. [13,14] called simulated annealing is applied to produce better sequence. During the simulation, temperature was lowered gradually until the system 'froze' and no further change occurred. At each temperature, the simulation must proceed long enough for the system to reach a steady state or thermal equilibrium. This method avoids being trapped in local minima by accepting uphill movement sometimes. The acceptance was determined using Boltzmann probability:

$$P(\Delta E) = e^{\frac{-\Delta E}{T_i}},$$

where ΔE is the cost difference between the current and previous solutions, while T_i is the current temperature. For a given annealing schedule of temperature, $T = \{t_1, t_2, ...\}$, our proposed algorithm are as follows:

1. Determine an initial sequence for all to be routed nets, called L_0. Set $L = L_0$.
2. For each S_i, assign to every node a tentative distance value: set it to zero for our initial node and to infinity for all other nodes.
3. Mark all nodes 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 visited, then compute $d(S_i, T_i)$. If T_i is not reached, therefore a blockage has occur. Abandon the net and continue with the next net in L. Repeat the process for every nets in L.
6. Compute initial energy, E_0 using equation:

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

   where i is the sequence (i can be referred as number of iterations too). 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 $P(\Delta E) > \varepsilon$, go to Step 13. Otherwise, reject the move and repeat Step 7.
13. Accept the candidate sequence as a current solution, and set $L = L', E(L) = E(L')$ and $T = T_k$. Update the temperature counters and parameters. Set $k = k+1, T_{k+1} = \alpha T_k$ and repeat Step 7. Stop until no changes occur.

4. Simulation Results

4.1. Comparison between Mesh, Torus and SD-Torus.

The simulation program is tested on Mesh, Torus and SD-Torus network of size 4×4 for a few data sets consist of random pairs and 100% traffic load (maximum number of possible pairs). The results are summarized in table 1.

Table 1. Simulation results for 4×4 network size with random data.

Data	**Mesh**		**Torus**		**SD-Torus**	
	R	*E*	*R*	*E*	*R*	*E*
Exp-1	6	17	8	20	8	15
Exp-2	5	15	8	16	8	15
Exp-3	6	18	8	20	8	15

From table 1, Torus and SD-Torus network succeed in perform connection for all pairs in the data set. On the contrary, Mesh was unable to route all nets in time O(1) even on a 4×4 problem size. Extra layers of routing region will be needed to route the remaining nets. As total routed nets, R, increases, the energy level throughout the network also increases. However, SD-Torus is having much lower energy level compared to Torus network since it has p^2 more communication links. Therefore, a net can be routed in a shorter way in SD-Torus network. With 6×6 and 8×8 network size, the performance of Torus network starts to degrade. The network starts to have blockages and unable to perform complete routing in time O(1). By using our proposed algorithm, as the network size increases, SD-Torus is better at maximizing *R* while minimizing energy level at the same time when compared to Mesh and Torus. Figure 2 illustrates the results.

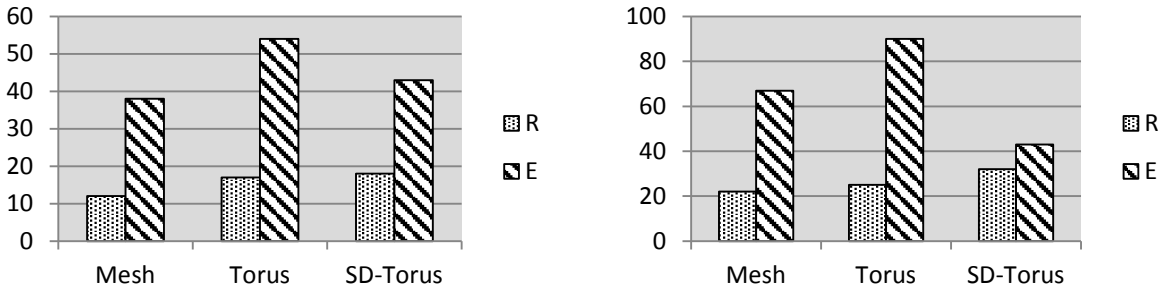

Figure 2. Simulation results for 6×6 (left) and for 8×8 network size (right).

4.2. Comparison between our proposed algorithm and Greedy Method.
From the simulation results in *Section 4.1*, it has been proven that SD-Torus network topology is the best topology compared to Mesh and Torus network. In this subsection, we compare the performance of our proposed algorithm (SA) and Greedy Method (GM) by using SD-Torus as the routing platform. Figure 3 illustrates the result obtained.

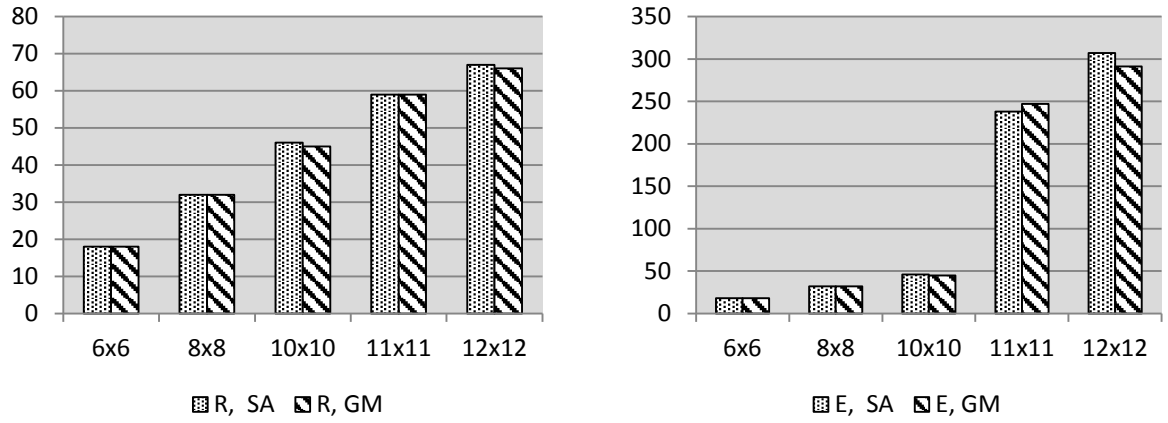

Figure 3. Total of *R* and E over various SD-Torus network sizes.

From the graphs in figure 3, both methods produce the same result for 6×6 and 8×8 network size. However, greedy method converges to the result faster than our proposed algorithm due to the process of accepting some uphill moves according to Boltzmann's probability function in simulated annealing. As the program continues, for 10×10 network size, the performance of greedy method starts to degrade. It freezes at $R=45$ while simulated annealing able to routes up to $R=46$ with the same nets requirement. Larger number of routed nets using our proposed algorithm yields to slightly higher E, but it is tolerable and acceptable because our main objective is to route as many nets as possible in time O(1). When we perform further simulation with 50 and 60 nets over 11×11 and 12×12 network size respectively, we notice that simulated annealing always produces better result compared to greedy method. From the simulation results, it can be conclude that our proposed algorithm improves greedy method by providing either larger number of R or same number of R with smaller energy level especially for larger SD-Torus network size and bigger data size.

5. Conclusion
The simulation results showed that mesh interconnection network is the worse in its class but popular due to its simplicity. It is almost impossible to have all nets to be routed in one layer using mesh. Torus improves the weaknesses of mesh network by introducing a left-right and top-bottom wraparound. This modification increases the number of communication links, thus, increases the value of R. SD-Torus is the best network to our knowledge since it is both regular and symmetrical. Due to these properties, the router design for this network is highly modular. This will enhance the reusability of the network implementation and makes it easily scalable. The performance of SD-Torus is also much better when compared to torus in terms of energy level and R value. It succeeded in performing complete routing up to 81 nodes with 40 nets in 9x9 network size. However, the routing complexities still depends highly on the net requirement and net ordering. It also was shown that by accepting some uphill movements based on the Boltzmann's probability led to better results, and this was how the routing algorithm worked. By using this annealing technique, the energy level is kept minimized while maximizing number of R.

Then, when we compare simulated annealing and greedy method, we notice that for small problem size, both method produce the same result but with different running time. Greedy method converges to the result faster compared to simulated annealing. This shows us that greedy method has considerably high convergence rate. When we run the program using simulated annealing, a longer running time is needed due to its high acceptance rate especially for problem with larger number of local minima. However, as the network size increases, the performance of greedy method starts to degrade. Simulated annealing improves the result given by greedy method by providing high number of routed nets, R or same number of R with lower energy level.

Acknowledgments
The authors would like to thank Universiti Malaysia Pahang for supporting part of this research.

References

[1] Adzhar N and Salleh S 2014 *MESH ROUTING: Maximing number of connections using heuristic method*. *Proc. International Conference on the Analysis & Mathematical Applications in Engineering & Science* p 161
[2] Adzhar N and Salleh S 2014 *Simulated Annealing Technique For Routing In A Rectangular Mesh Network*. *Modelling and Simulation in Engineering*
[3] Adzhar N and Salleh S 2015 *Obstacle-Aware Routing Problem In A Rectangular Mesh Network*. *Applied Mathematical Sciences* **9** 14
[4] Daeho S, Akif A, Won-Taek L et al. 2005 *Near-Optimal Worst-case Throughput Routing for Two-Dimensional Mesh Networks*. *Technical Report Purdue University*
[5] Jian L, Sriram S and Russell T 2000 *ASOC: A Scalable, Single-Chip Communications Architechture*. *Proceeding of the IEEE International Conference on Parallel Architechtures and Compilation Techniques* p 37
[6] Kumar S et al. 2002 *A Network on Chip Architechture and Design Methodology*. *Proceeding of International Symposium VLSI* p 117
[7] Fernando M, Ney C, Aline M et al. 2004 *HERMES: An Infractructure For Low Area Overhead Packet-Switching Networks On Chip*. *Journal of VLSI Integration* **38** 69
[8] Tang K W and Padubidri S A 1994 *Diagonal And Toroidal Mesh Networks*. *IEEE Transactions on Computer*, 3 693
[9] Liu Y H, Zhu M F, Wang J, Xiao L M and Gong T 2012 *Xtorus: Anextended Torus Topology For On-Chip Massive Data Communication. IEEE International Parallel and Distribution Processing Sympossium Workshops & PhD Forum*
[10] Wang Y G, Du H M and Shen X B 2011 *Topological Properties And Routing Algorithm For Semi-Diagonal Torus Networks*. *The Journal of China Universities of Posts and Telecommunications*. **18** 64
[11] Krishnan M N, Raghunath S, Ajith Pravin Dhas D, Benny Raj A M and Pounambal M A 2012 *A Deadlock-Free Routing Algorithm For Torus Network*. *Network and Complex System* **2** 4
[12] Arjun S, William J D, Amit K G and Brian T 2003 *GOAL: A Load-Balanced Adaptive Routing Algorithm For Torus Network Proc. of The 30th Annual International Symposium on Computer Architechture (ISCA'03)*
[13] Metropolis N, Rosenbluth A W, Rosenbluth W N, Teller A H and Teller E *1953 Equation of State Calculations by Fast Computing Machine*. *J. Chem. Phys*. **21** 498
[14] Kirkpatrick S, Gellat C D and Vecchi M P 1983 *Optimization by Simulated Annealing. Science* **220** 671