

AC 2008-227: HOT SPOT MINIMIZATION OF NOC USING ANT-NET DYNAMIC ROUTING ALGORITHM

Alireza Rahrooh, University of Central Florida

ALIREZA RAHROOH Alireza Rahrooh is a Professor of Electrical Engineering Technology at the University of Central Florida. He received the B.S., M.S., and Ph.D. degrees in Electrical Engineering from the Univ. of Akron, in 1979, 1986, and 1990, respectively. His research interests include digital simulation, nonlinear dynamics, chaos, control theory, system identification and adaptive control. He is a member of ASEE, IEEE, Eta Kappa Nu, and Tau Beta Pi.

Faramarz Mossayebi, Youngstown State University

FARAMARZ MOSSAYEBI Faramarz Mossayebi is an Associate Professor in the Department of Electrical and Computer Engineering at Youngstown State University teaching in the area of digital systems including computer architecture and embedded systems, digital signal processing and controls. His primary area of interest includes modeling and simulation of nonlinear dynamical systems, digital signal processing, and control.

Walter Buchanan, Texas A&M University

WALTER BUCHANAN Walter W. Buchanan is Professor and Head of Engineering Technology & Ind. Distribution Department at Texas A&M University, TAMU. He received his BSE and MSE from Purdue University, and his Ph.D. and J.D. from Indiana University. Walt is a P.E. in five states, and is Chair of ETD. He has written over 90 papers, and is a Member of TAC of ABET and Chair of IEEE's Committee for Technology Accreditation Activities.

Hot Spot minimization of NoC Using AntNet Dynamic Routing Algorithm

Abstract

In this paper, a routing model for minimizing hot spots in the network on chip (NoC) is presented. The model makes use of AntNet routing algorithm which is based on Ant colony. Using this algorithm, which we call AntNet routing algorithm, heavy packet traffics are distributed on the chip minimizing the occurrence of hot spots. To evaluate the efficiency of the scheme, the proposed algorithm was compared to the XY, Odd-Even, and DyAD routing models. The simulation results show that in realistic (Transpose) traffic as well as in heavy packet traffic, the proposed model has less average delay and peak power compared to the other routing models. In addition, the maximum temperature in the proposed algorithm is less than those of the other routing algorithms.

1. Introduction

The tile-based NoC architecture is known as a suitable solution for the communication problems in future VLSI circuits¹.

The routing algorithms could be classified as centralized versus distributed and static versus adaptive. In centralized algorithms, a central controller is responsible for updating the routing table for each node. The delay required for gathering the information regarding the network status and then broadcasting it to all nodes for updating their tables make the application of this type limited. Except for small size networks this method is only used in special cases. In distributed algorithms, the determination of the network status is distributed among the nodes which exchange information with each other. In static routing models, the path between the source and the destination of a packet is determined by the source and the destination themselves and the current traffic status of the network is not considered. In adaptive algorithms, however, the path between the source and the destination is determined node by node depending on the network status as the packet moves toward the destination. For example, DyAD² is an adaptive routing algorithm and XY³ is a static routing algorithm in NOCs. In this work, we propose an adaptive distributed algorithm which distributes the packet traffic to minimize hot spots in the network. The algorithm which is inspired by ant colony is based on the AntNet routing algorithm⁴. The router is a short path adaptive router which selects the shortest path with the least traffic for sending the packet forward. The shortest paths which have the minimum number of hops⁵ form the sets of the minimum paths, and the router select the set with the minimum traffic to minimize hot spots which are nodes with high traffics. The paper is organized as follows. In Section 2, we briefly describe the AntNet algorithm while the proposed architecture is discussed in Section 3. The results are discussed in Section 4. Finally, the summery and conclusion are given in Section 5.

2. AntNet Routing Algorithm

The routing model presented in this work is based on the AntNet algorithm⁴ which is for a network of computers. For the case of the hardware implementation for NoC, the algorithm should be modified. The control packets (ants) are used for updating the routing table based on the traffic status of the network. These control packets (ants)

simulates the behaviors of worker ants which leave footprints when quest for food (target). The stronger the footprints are the higher is the likelihood of finding the food if this path is paved.

In AntNet⁴, the routing table for node k, denoted T_k , is used to make a probabilistic routing decision.

Table1. Original AntNet routing table at node k.

Destination Node Neighbor Link	1	2	...	d
1	P _{1,1}	P _{1,2}	...	P _{1,N}
2	P _{2,1}	P _{2,2}	...	P _{2,N}
3	P _{3,1}	P _{3,2}	...	P _{3,N}
4	P _{4,1}	P _{4,2}	...	P _{4,N}

A sample table is shown in Table 1 which has L rows corresponding to L neighboring nodes/links. The probability of sending the packet to the destination d via the link connected to node, i, is denoted as P_{id} . The AntNet algorithm can be summarized as follow:

1. New forward ants, F_{sd} , from the source (s) to the destination (d), are created periodically.
3. The next link (node j) of a forward ant is selected stochastically using $P'(j, d)$ which is a function of the routing table probabilities and the length of the corresponding output queue as given by⁴

$$P'(j, d) = \frac{P(j, d) + \alpha L_j}{1 + \alpha (N_k - 1)} \quad (1)$$

Here, $P(j, d)$ is the probability of selecting node j as the next hop, α which is a number between 0.2 and 0.5 weights the significance given to the local queue length (L_j) versus the global routing information, L_j is proportional to the inverse of the queue length at node j and N_k is the number of links from the current node.

3. When the next node is determined and this node is different from the destination, the forward ant checks the buffer to see if this node has been visited before. If this is the case, it shows that the ant is entering a cycle and, hence, must die otherwise ant saves the previously visited node identifier (ID) and time stamp at which the ant was serviced by the current node in a buffer. When the current node is the destination ($k = d$), then the forward ant should become a backward ant (B_{sd}). The information recorded in the buffer of the forward ant should be used to retrace the route passed by the forward ant.

4. When the node is visited by the backward ant the probabilities in the routing table of each node is updated using the following rules:

If (node was in the path of the ant)
then $P(i) = P(i) + r \cdot (1 - P(i))$
else $P(i) = P(i) - r \cdot P(i)$

Here, r which is a number between 0 and 1 is the reinforcement factor indicating the quality (length), the congestion, and the underlying network dynamics⁶.

3. AntNet Routing Scheme for NOC

The proposed routing scheme makes use of a regular router structure where each router is connected to the neighboring routers and a processing element (PE) at the node through five communication channels using the mesh configuration. To minimize the delay and the required resources, the wormhole technique for the routing of the data packets is used. The structure of the proposed router is shown in Figure 2. In the input port of the router, there are two separate queues for data and control packets. The priority of the queue for the control packet which includes both forward and backward ants is higher than data packet queue and the arbiter unit based on round robin scheme. The routing block consists of three units of the data packet routing, the forward ant routing, and the backward ant routing.

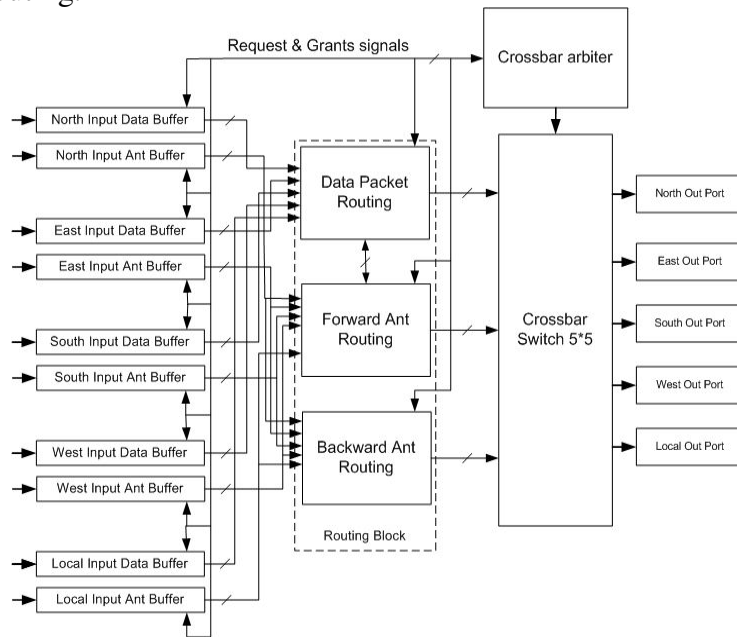


Figure 2. AntNet Routing Structure.

3.1 Data Packet Routing Unit

This unit determines the output port for the data packet based on its destination using a $4 \times (N - 1)$ routing table. Where, N is the total number of network nodes and 4 is the number of links for each node. For each destination d ($d = 1:N$), four probabilities which add up to 1 are stored. To make the hardware implementation simple, we use binary numbers between: 0 to 4 instead of 0 to 1. To minimize the delay, we use a minimal routing algorithm which is deadlock free⁵. The router is initialized by selecting one or two directions (links) for each node. As observed from Figure 3, if the desired destination node is in the same row or column as the current node (d_1 and L), there is only one direction for sending the data packet to the destination. Consequently, in column d_1 of the routing table for node L , there is only one non-zero probability which is four ($p = 1$). If the destination node is not on the same column or row as of the current node (e.g., L and d_2), there are two directions for sending the data packet to the destination. Therefore,

there two non-zero probability which are two ($p = 0.5$) in the column d2 of the routing table.

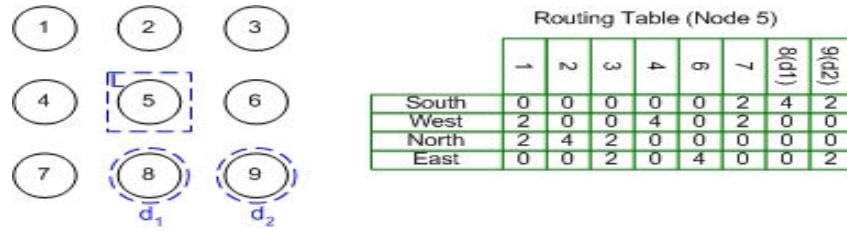


Figure 3. The routing table for node 5.

3.2 Forward Ant Routing Unit

This unit determines the output port for the forward ant F_{sd} . To determine the destination node (d) for this forward ant we use the concept of “popularity” based on which the destination of the forward ant is determined as the node to which most of the data packets are sent⁶. To determine the popular node for the source node (s), the structure shown in Figure 4 is used. Here, the shift registers with the length of H (length of desired history) are utilized to store the data packet destination node number. Each shift register stores one bit of the destination node. The number of shift registers is determined by the number of bits required to show the node numbers of the network. Using Block A (Figure 4) the number of zeros are compared to the number of ones at that bit position and whichever is more is stored in the corresponding bit of the popular node.

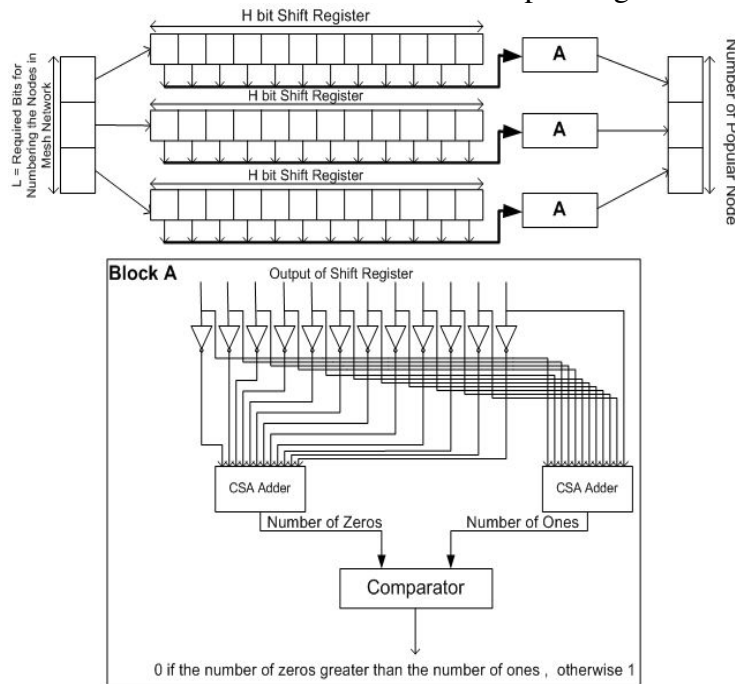


Figure 4. The circuit which determines the popular node.

For routing the forward ant, if the destination resides on the same row or column on the mesh, as the current node, the forward ant moves on the same row or column. If the

current and destination nodes are not on the same row or column, then two nodes, which we denote them by b and a may be selected. The probability of selecting the next node j ($j \in \{a, b\}$) may be obtained from

$$P'(j) = \frac{3 \times P(j) + L_n(j)}{4} \quad (2)$$

where $P(j)$ is the probability of selecting node j as the next node and $L_n(j)$ is the parameter indicating the instantaneous status of the ant queue for the corresponding input port at node j. This relation is the same relation as (1) when $N_k = 2$ and $\alpha = 1/3$. To obtain the value for $L_n(j)$, we make use of Tables 2 which is based on the warning signal of “buffer full” of each link (w_full). Each buffer of the router has a status signal which is w_full . When the number of empty cells of the buffer is less than a certain value, w_full (warning for the full status) is activated which warns that most of the buffer cells are full. When the forward ant enters the router of a node, the number (ID) and congestion information (CS) of the node are stored (pushed) in the body of the forward ant. If the forward ant reaches to its destination, the destination router converts the forward ant to a backward ant ($B_{d \rightarrow s}$) with the destination as the source of the forward ant. The backward ant passes through the same routers (in reverse order).

Table 2. $L_n(a)$ and $L_n(b)$ as a function of $w_full(a)$ and $w_full(b)$.

$w_full(a)$	$w_full(b)$	$L_n(a)$	$L_n(b)$
0	0	2	2
0	1	4	0
1	0	0	4
1	1	2	2

The CS register contains a binary number between 0 to 4 which is the sum of four Congestion Flags (CF's). Each input port has a congestion signal called Congestion Flag (CF) through which it informs its adjacent router about the type of its congestion (congested or non-congested) status.

3.3 Backward Ant Routing Unit

For routing the backward ant, the node number (router ID) is popped from the body of the ant. This way, the backward ant moves along the same path as the original forward ant but in the reverse direction. When the backward ant reaches its destination the ant is killed. As the second task, when the backward ant enters router k via link f, the probabilities of the routing table of the node are updated using:

$$P_{fd} = P_{fd} + \frac{(4 - CS) \times (4 - P_{fd})}{4} \quad (3)$$

$$P_{nd} = P_{nd} - \frac{(4 - CS) \times P_{nd}}{4} \quad (4)$$

where, P_{fd} (P_{nd}) is the probability of selecting link f for routing from node k to node d (destination). The probability of selecting link f increases while for the other links it decreases. Since the increase is dependent on the node congestion (CS).

4. Result and Discussion

To assess the efficiency of the AntNet routing algorithm, we implemented three other routing algorithms. These algorithms include DyAD², XY³, and Odd-Even⁵. We have developed an NoC simulator written in C++ which can calculate the average delay and the power consumption for the packet transmission. We have used the mesh configuration for the NoC. To calculate the power consumption, we have made use of Orion library functions⁷.

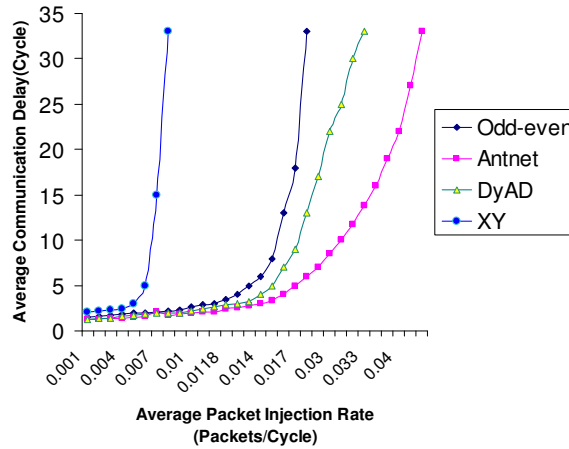


Figure 5. Performance of different algorithms under the transpose traffic.

4.1 First Transpose Traffic Profiles

In this traffic profile, for a $n \times n$ mesh network, a PE at position (i, j) ($i, j \in [0, n - 1]$) only sends a data packet to other PE at position $(n-1-i, n-j-1)$. This traffic pattern which is a more realistic one^{2,5} is similar to the concept of transposing a matrix. This traffic profile leads to a non-uniform traffic distribution with heavy traffics for the central nodes of the mesh. Therefore, hot spots close to the center of the network may be created. As shown in Figure 5, if the data packet injection rate is very low, the hot spots are not created and hence DyAD routing algorithm leads to a smaller delay. As the injection rate increases, the AntNet algorithm leads to smaller average delay. For high injection rates where hot spots (heavy traffic nodes) may be created if adaptive algorithms such as the AntNet algorithm are not used. In general the adaptive algorithms normally behave much better in realistic traffic profiles such as the first transpose case.

4.2 Power Consumption

We have calculated the dynamic power consumption of the first transpose traffic profile for different routing algorithms. As shown in figure 6 the average power consumption of the AntNet routing algorithm is 14% and 10% more than those of XY and DyAD algorithms, respectively. The average peak power of the proposed algorithm,

however, is 26% and 25% less than those of the XY and DyAD algorithms, respectively. This shows that the AntNet algorithm manages to minimize the hot spots on the chip and distribute the traffic more.

5. Summary and Conclusion

In this paper, a routing algorithm based on AntNet was presented. The technique made use of shortest route obtained from the Ant Colony. For the hardware implementation of the technique for NoC's, some modifications to the AntNet algorithm were made. The results of the simulations for this algorithm, XY, DyAD, and Odd-Even routing algorithms showed a lower average delay and less hot spots for the first transpose traffic which is a more realistic traffic profile.

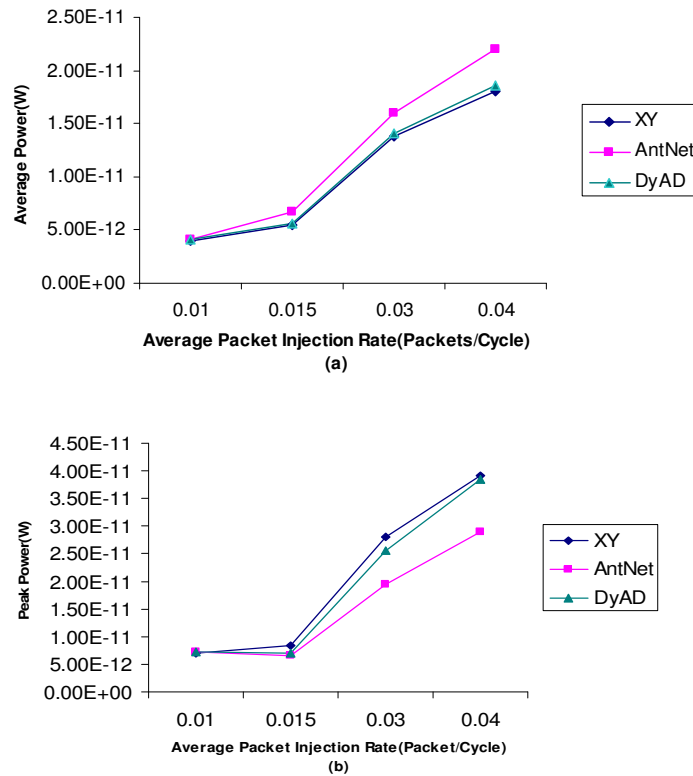


Figure 6. (a) The average power dissipation of the AntNet, DyAD and the XY algorithms. (b) The maximum power dissipation of the AntNet, DyAD and the XY algorithms.

References

1. W. J. Dally and B. Towles, "Route packets, not wires: on-chip interconnection Networks," in Proc. DAC, pp. 684-689, June 2001.
2. J. Hu and R. Marculescu, "DyAD-Smart Routing for Networks-on-Chip," DAC 2004, pp. 260 - 263, 2004, San Diego, California, USA.
3. J. Glass and L. M. Ni, "The Turn Model for Adaptive Routing," Proc, Symp, Computer Architecture, pp. 278-287, May 1992.
4. Di Caro G., Dorigo M., "AntNet: Distributed stigmergetic control for communications networks," Journal of Artificial Intelligence Research, 9, pp. 317-365, 1998.
5. G. Chiu, "The Odd-Even Turn Model for Adaptive Routing," IEEE Tran, On Parallel and Distributed System, pp: 729-738, July 2000.
6. H. Yun and A. N. Zincir-Heywood, "Intelligent Ants for Adaptive Network Routing," in Proc. CNSR, pp. 255-261, 2004.
7. Wang, X. Zhu, L. Peh, S. Malik, "Orion: A Power-Performance Simulator for Interconnection Network," in Proc. Hot Interconnection, Stanford, CA, pp. 294 – 305, August 2002.

ALIREZA RAHROOH

Alireza Rahrooh is a Professor of Electrical Engineering Technology at the University of Central Florida. He received the B.S., M.S., and Ph.D. degrees in Electrical Engineering from the Univ. of Akron, in 1979, 1986, and 1990, respectively. His research interests include digital simulation, nonlinear dynamics, chaos, control theory, system identification and adaptive control. He is a member of ASEE, IEEE, Eta Kappa Nu, and Tau Beta Pi.

FARAMARZ MOSSAYEBI

Faramarz Mossayebi is an Associate Professor in the Department of Electrical and Computer Engineering at Youngstown State University teaching in the area of digital systems including computer architecture and embedded systems, digital signal processing and controls. His primary area of interest includes modeling and simulation of nonlinear dynamical systems, digital signal processing, and control.

WALTER BUCHANAN

Walter W. Buchanan is Professor and Head of Engineering Technology & Ind. Distribution Department at Texas A&M University, TAMU. He received his BSE and MSE from Purdue University, and his Ph.D. and J.D. from Indiana University. Walt is a P.E. in five states, and is Chair of ETD. He has written over 90 papers, and is a Member of TAC of ABET and Chair of IEEE's Committee for Technology Accreditation Activities.

