An Analytical Model for Hypercube Network-On-Chip Systems with Wormhole Switching and Fully Adaptive Routing

Jin Liu[1], Xiaofeng Wang[1], Hongmin Ren[1], Jin Wang[2] and Jeong-Uk Kim[3]

[1] *College of Information Engineering, Shanghai Maritime University, Shanghai, China*

[2] *Computer and Software School, Nanjing University of Information Science & Technology, Nanjing, China Department of Energy Grid*

[3] *Sangmyung University, Seoul 110-743, Korea*

{jinliu, xfwang, hmren}@shmtu.edu.cn; wangjin@nuist.edu.cn; jukim@smu.ac.kr

Abstract

In this paper, we present a novel performance analysis model for predicting average message latency of hypercube network-on-chip (NoC) systems that employ wormhole switching and fully adaptive routing method with uniformly distributed traffic. Unlike previous works, our model calculates the service rate that is provided by a physical link for the incoming traffic at a particular communicating node by obtaining the reversal service rate that is provided by the downstream nodes, when the network gets saturated. This model has simple closed-form calculations, and can produce accurate analytic results when the NoC system is operating in stable state. The proposed model is validated by comparing the results that are calculated by the model with the results that are obtained through simulations with different network configurations and traffic loads.

Keywords: *Model; NoC; Hypercube; Adaptive Routing*

1. Introduction

The number of components that are interacting to compute a solution in System-on-Chip keeps increasing with the rapid development of VLSI. Having dedicated connections between any given modules could be extremely complex as the number of modules increases. A viable alternative could be an interconnection network within the chip [17]. There are a number of network designs with different topology for network on chip reported in the literatures. [1-4, 15] As parallel architectures become larger and faster, packaging and physical constrains are assuming more important roles, among different designs, hypercube-based NoC systems have the desirable features such as symmetry, regularity and can reduce network's bisection width, thus they are receiving increasing attentions. A typical hypercube system is shown in Figure 1. NoC systems employ a hierarchical network architecture which is similar as usual computing network. Thus the switching and routing technology are vital for the performance of the networked system. At the link layer, wormhole switching [5] and virtual channel mechanism have been widely used in practical network on chip [6]. The former is used to achieve pipelined data transmission to reduce average message latency. The latter is to make better utilization of physical channel while using minimal local buffer and avoiding deadlock. At the network layer, routing algorithms can be implemented as either deterministic or adaptive. Deterministic routing protocols choose the path based on the message's source and

destination. When using deterministic routing, a packet will be delayed if any channel along the path is busy with other packets, or even worse, if a channel along the path is faulty then

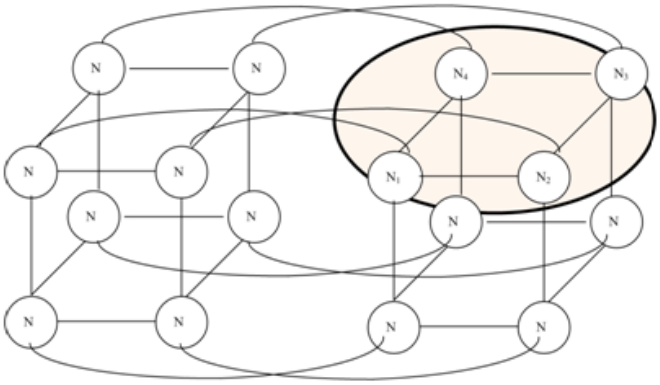

Figure 1. The Hypercube-based NoC

the packet cannot be delivered. Nevertheless, deterministic routing still has been widely used due to its simplicity [6]. And its analytical model has been widely reported in the literature [7-9]. Adaptive routing protocols which provide alternative paths for communicating nodes have been proposed to make more efficient use of bandwidth and to improve fault tolerance of interconnection network [16]. Several adaptive routing algorithms have been proposed, showing that message blocking can be considerably reduced, thus strongly improving network throughput. Among them, routing algorithms based on Duato's design methodology [10] are extensively used. These routing algorithms split each physical channel into two virtual channel sets, the adaptive and the deterministic channels. When the paths of adaptive channels are blocked, a message uses an escape channel at the congested node. If there is any free adaptive channel available at subsequent nodes, the message can go back to the adaptive channels. Adaptive routing algorithms can be further categorized to progressive and backtracking algorithms. Progressive routing algorithms move the message header forward by reserving a new channel. In our analytical model, we assume the routing algorithm is progressive, no backtracking is allowed. Several analytical models for wormhole switching network using fully adaptive routing protocols were reported in the literature, but their calculation process is complicated and the presented results only hold in relatively small network state region [11-13].

Unlike previous works, this paper presents a novel performance analysis model for hypercube based network on chip systems which are using wormhole routing, virtual channels and fully adaptive routing. The model has relatively simple calculation process and yields satisfactory predictions in the network's steady state region. The rest of this paper has been organized as follows. In Section 2, we present the model in detail. In Section 3, validation of the proposed model is made by comparing its prediction with simulation results. Some concluding remarks are included in Section 4.

2. The Performance Model

The detail of our model is presented in this section. First we describe the network configuration and some assumptions for the model; then we give a few notations used by the model, and then describe how to obtain the average message latency of the system by the model step by step in detail.

2.1 System Description

A hypercube network as shown in Figure 1 is used to illustrate our model. Each node consists of a processing element (PE) and a switch. PE is responsible for generating messages and consuming messages from other nodes. Each switch has 5 input and output channels. PE is connected to switch by local injection/ejection channel. A node

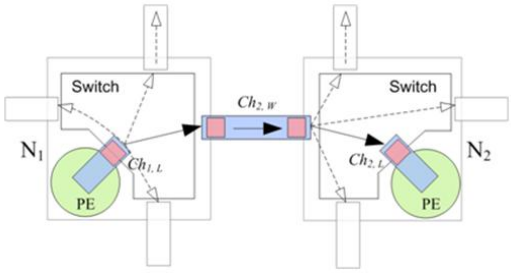

Figure 2. The shortest traveling path for a message

is connected to 4 adjacent neighboring nodes by bi-directional network channels. A message which is generated from a node's PE will first be transmitted to the switch by local injection channel. Then it will be routed toward the destination. At the destination node, the message is transmitted to PE through local ejection channel. Thus a message has to travel through at least 3 links from the source to destination. For instance, as shown in Figure 2, the message generated in node N1 has to traverse channels Ch1,L, Ch2,W and Ch2,L to arrive at its destination PE in N2. The model is also based on the following commonly accepted assumptions [7-13].

1) Each node generates traffic independently with traffic following a Poisson process on a mean rate of M_{gen} messages/cycle.

2) Message destinations are uniformly distributed across the network nodes. Although in an real application, if node A sends a message to B it's highly possible that B will send back a message to A.

Table 1. Parameter Notations Used by the Model

$Ch_{n,dir}$	Physical channel in *dir* direction of a message's nth hop node.
dir	Direction. Each nodes has ports on 5 directions, i.e. East, South, West, North and Local
D	Average path length for all the delivered messages
L	Average message length
M_{gen}	Average message generation rate at each node
N_q	Average number of nodes along the path
o	In/output ports number of a node (not including local channel port), in our network, it's 4.
P'_Q	The blocking probability for a message at a physical channel without contention.
P_Q	The blocking probability for a message at a physical channel
S	The service time for a message at a channel w/o contention.
SR	The service rate for a message at a channel w/o contention
T_{msg}	The mean latency for all the delivered messages
T_s	Routing (Switching) delay across a node
T_w	Propagation delay across the physical channel
v	The number of virtual channels multiplexing a transmission direction of physical channel
W_q	Average waiting time for a message at each intermediate nodes
W_{ej}	Mean waiting time for a message on ejection node channel
λ	The average message arrival rate at a channel
μ	The service rate for a message at a physical channel
μ_r^{dir}	The service rate that a channel observes from the immediate downstream channel on dir direction.
μ_r	The service rate that a channel observes from all its immediate downstream channels at nodes on the path.

3) Messages are of constant length L flits. A message is long enough so that its data flits span from source to destination nodes.

4) Each flit requires one cycle to be transmitted from one node to the next over physical link between them. Two cycles are needed for a flit to cross a node, i.e. from an input buffer to an output port in absence of blocking.

5) The local queue at the injection channel in the source node has infinite capacity. Messages at the destination node are transferred to the local PE one at a time through the ejection channel.

6) A set of virtual channels are multiplexed across a physical channel. Virtual channels share physical channel's buffer and each has its own single flit space.

7) Physical channels between any two adjacent nodes are duplex. More than two virtual channels are used for each direction of a physical channel. If there are adaptive virtual channels available, a message can use a random one.

2.2 Calculation of Average Message Latency

There are a few notations used in the derivation of our model, a brief summary of them can be found in the Table 1. The average message latency T_{msg} comprises of the message transmission delay across the network channel t_w, the intra-router delay t_s, [5] the average contention delay W_q at the network channels and the average delay W_{ej} at nodes' local ejection channel. It can be computed as follows:

$$T_{msg} = (D + L) \cdot T_{tr} + (N_q - 1) \cdot W_q + W_{ej} \quad (1)$$

$$T_{tr} = \max(T_s, T_w) \quad (2)$$

Eq.2 demonstrates the nature of pipelined flits transmission of wormhole switching in absence of contention. $(D+L)T_{tr}$ denotes the mean time that a message's header flit and data flits need to travel from source to destination. $(N_q - 1)W_q$ shows the waiting time that header flit experienced at the $N_q - 1$ channels of intermediate nodes. As the minimum link number that a message travels is 3, which is shown in Figure 2, the average hops that messages take, D can be obtained by:

$$D = \sum_{i=3}^{k} p_i \cdot i \quad (3)$$

Where k is the diameter of the network, and p_i denotes the probability that a message's travel path is i links long. Under the uniform traffic pattern, the average traffic arrival rate λ for each channel is determined by the message generating rate M_{gen}, average routing hops D and output channels number of each node o.[4]

$$\lambda = \frac{M_{gen} \cdot D}{o} \quad (4)$$

In order to receive service from a link, the message's header flit will acquire a virtual channel. A VC keeps serving the message until all the data flits flow across this node. When the traffic rate is light, there is no congestion; the service time *S* and service rate *SR* at each channel can be defined respectively as follows:

$$S = t_s \cdot (L+1) \tag{5}$$

$$SR = \frac{1}{S} \tag{6}$$

As traffic rate keeps increasing, congestion appears in the network and waiting queues build up at bottleneck links. In this case, the service that one channel can provide to the incoming messages is not only determined by its own service capacity, but also by the blocking state of its immediate downstream channels. In a hypercube network, the traffic arrival rates for input channels of a node are equal to each other due to its symmetry. Without loss of generality, suppose the node that we analyze is at *n*th hop of a messages routing path. We analyze queuing system model of channel $Ch_{n,W}$ at west input port. The waiting queue $Q_{n,W}$ at channel $Ch_{n,w}$ is treated as two distinct queues Q_c and Q_d as shown in Figure. 3. Q_c is the result of contentions at $Ch_{n,w}$,which is determined by the traffic arrival rate λ and the router self's service rate *SR*. Q_d is due to the contention that a message experiences when it's to be accepted by a downstream input channel. An *M/M/m* queuing system is used to model Q_c. The probability that an arrival message will find *v* virtual channels are busy and will be forced to wait in queue can be obtained by the following equation: [14]

$$P_Q^{'} = \frac{p_0 \cdot (v \cdot \rho)^v}{v! \cdot (1-\rho)} \tag{7}$$

Where p_0 is given by:

$$p_0 = \left(\sum_{n=0}^{v-1} \frac{(v \cdot \rho)^n}{n!} + \frac{(v \cdot \rho)^v}{v! \cdot (1-\rho)} \right)^{-1} \tag{8}$$

For Q_d, the service rate, μ_r, is the service rate that offered by all the immediate downstream channels to $Ch_{n,w}$. To obtain μ_r, we can further divide Q_d to four queues, each of which is associated with an immediate downstream channel in one of the possible downstream directions, *i.e*. east, south, north in this case and local injection, as shown in Figure 3. Consider the waiting queue at west downstream channel $Ch_{n+1,W}$. we can get average waiting time for a message by: [14]

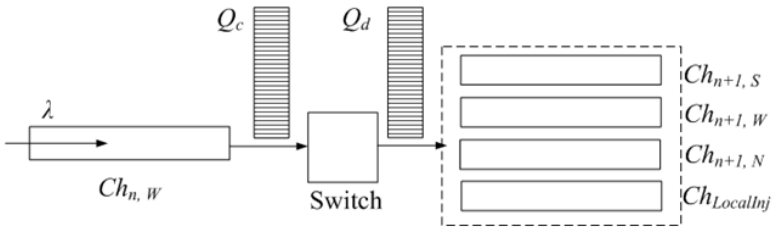

Figure. 3. Queuing model for an input channel

$$W_{q_d} = \frac{P_Q^{'} \cdot \rho}{\lambda \cdot (1-\rho)} \quad \textbf{(9)}$$

Thus the average time that a message spends at the queuing system of $Ch_{n+1,W}$ is $W_{qd} + S$. In addition, there are v virtual channels in $Ch_{n+1,W}$ and $Ch_{n+1,W}$ has o possible inputs(including local injection channel), therefore we can get μ_r^w as :

$$\mu_r^w = \frac{v}{o \cdot (S + W_{qd})} \quad (\textbf{10})$$

Hence, we get μ_r as:

$$\mu_r = p_W \mu_r^W + p_S \mu_r^S + p_N \mu_r^N + p_L \mu_r^L \quad (\textbf{11})$$

Due to symmetry of hypercube, we know that $p_W=p_S=p_N$, so $\mu_r=3\ p_W \mu_r^w + P_L \mu_r^L$. Using Little's Theorem, the average number of messages in the whole queuing system at $Ch_{n,W}$ can be derived as:

$$N = \lambda \cdot T = \frac{\lambda}{\mu} + \frac{\lambda \cdot P_Q}{v \cdot \mu - \lambda} \quad (\textbf{12})$$

Also by Little's Theorem, the average number of messages in Q_c and Q_d are:

$$N_d = \frac{\lambda}{\mu_r - \lambda} \quad (\textbf{13})$$

$$N_c = \frac{\lambda}{SR} + \frac{\lambda \cdot P_Q}{v \cdot SR - \lambda} \quad (\textbf{14})$$

As $Q_{n,E}$ comprises of Q_c and Q_d, combine Eq. (12), (13) and (14), we get:

$$\frac{\lambda}{\mu} + \frac{\lambda \cdot P_Q}{v \cdot \mu - \lambda} = \frac{\lambda}{SR} + \frac{\lambda \cdot P_Q}{v \cdot SR - \lambda} + \frac{\lambda}{\mu_r - \lambda} \quad (\textbf{15})$$

Based on Eq. (7) and (8), replace P_Q in Eq. (15), solve the resulting nonlinear equation, we can get service rate μ at channel $Ch_{n,W}$. Then, by Little's Theorem again, we can obtain the average time W_q that a message has to wait in queuing system at the intermediate nodes of a message's traveling path:

$$W_q = \frac{P_Q \cdot \rho}{\lambda \cdot (1-\rho)} \quad (\textbf{16})$$

Finally, as we mentioned that the node's local ejection channel is treated as independent *M/M/m* queuing system; average waiting time on ejection node channel can be obtained by:

$$W_{ej} = \frac{P_{Q_{ej}} \cdot \rho_{ej}}{(\lambda + \lambda') \cdot (1-\rho_{ej})}, \quad \rho_{ej} = \frac{\lambda + \lambda'}{v \cdot SR} \quad (\textbf{17})$$

At this point, the message delay T_{msg} defined in Eq. 1 can now be calculated out as all the unknown variables at the right hand side are all obtained.

3. Validation

A number of simulation experiments which are based on various network configurations have been carried out to validate the proposed model. The simulator we used is based on flexsim1.2 [7]. The results presented here are average message latency obtained from both model and simulations on 64, 256 and 1024 nodes hypercubes. The comparisons can be seen in Figure 4, 5 and 6 respectively. In both calculations and simulations, 4 virtual channels are multiplexing a physical channel and the message size is set to 32 flits. As we can observe, in these figures, the simulation results and predictions of our model match well from a very light usage of a network channel to about 50% average channels utilization after which point the network gets into saturation state.

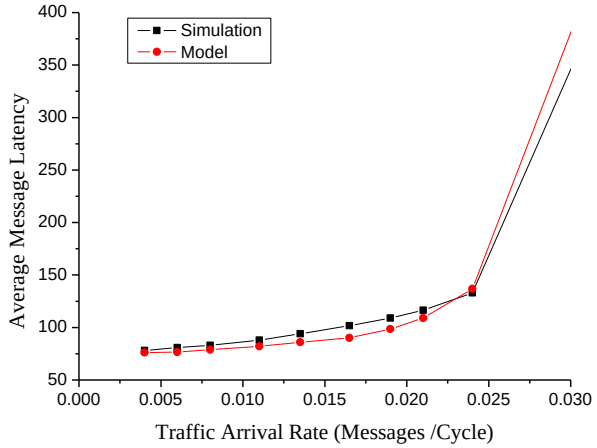

Figure 4. Model vs. Simulation Results in 64 Nodes Hypercube Network

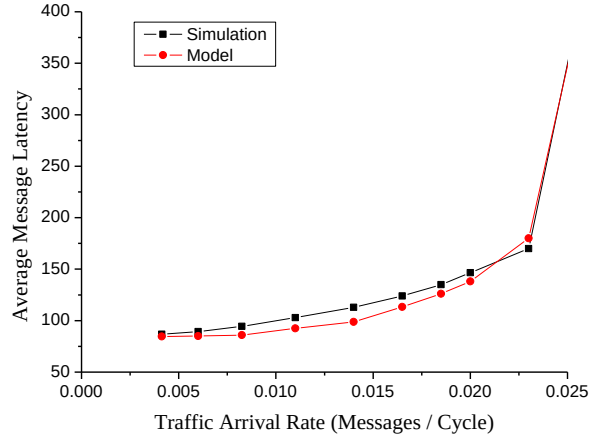

Figure 5. Model vs. Simulation Results in 256 Nodes Hypercube Network

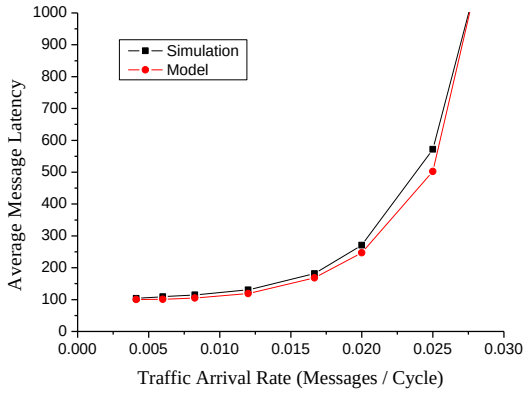

Figure 6. Model vs. Simulation Results in 1024 Nodes Hypercube Network

4. Conclusions and Future Work

We have presented a performance analysis model for predicting average message latency and average link waiting time in hypercube based NoC systems which are adopting wormhole switching, virtual channel and adaptive routing. Unlike previously proposed models, this model is based on queuing theory and probability analysis. It's simple and yields very accurate predictions in network's steady state region. By applying its close-form calculations, one can correctly predict performance of high-radix hypercube NoCs in steady state regions with a short code snippet. It can be concluded that our model provides an effective and practical evaluation tool. In addition, since this model obtains message waiting time at each queuing system at channels of a routing path, it can be easily adapted for networks with other topologies.

As we can see from the calculation process, blocking probability at a transmitting channel determines the waiting time for a message given a particular service rate. A higher blocking probability will results in an exponentially larger waiting time. Thus if we can reduce the blocking probability for a message travelling across the network, the network will have better performance. At given message generation rate and link service rate, one can alter the message blocking probability by adopting different routing algorithm, network topology, or increasing the buffer size at each node. The first two are major factors in determining network performance while the buffer size has small impact [5]. Because buffer of one physical channel at a given node is usually not deep enough to store the whole blocked message, the upstream channels that occupied by the blocked message cannot be freed for other contending messages. This is especially the case for NoCs where buffer needs to be carefully designed to be both compact and efficient to reduce hardware cost while keeping network performance. Therefore, we will keep researching on how to predict the subtle impact of buffer size on the network performance with more sophisticate modeling and queuing system analysis.

Acknowledgements

This work was supported by Shanghai Municipal Baiyulan Sci.&Tech Talent Fund (11BA1404700), Shanghai Science Foundation for The Excellent Youth Scholars (No.shs10033), Shanghai Municipal Education Commission Sci&Tech Innovation Project (12ZZ157), the Natural Science Foundation of Jiangsu Province (No.

BK2012461) and a grant (07-HUDP-A01) from High-tech Urban Development Program funded by Ministry of Land, Transport and Maritime Affairs of Korean government. Prof. Jin Wang is the corresponding author.

References

[1] P. Guerrier and A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnections", Proceedings of Design, Automation and Test in Europe (DATE), **(2000)** Mar 14-18; Grenoble, France.
[2] S. Kumar, "A Network on Chip Architecture and Design Methodology", Proceedings of IEEE Computer Society Annual Symposium on VLSI, **(2002)** April 25-26; Pennsylvania, USA.
[3] W. J. Dally and B. Towles, Editors, "Route Packets, Not Wires: On-Chip Interconnection Networks", Proceedings of the 38th Design Automation Conf. (DAC), **(2001)** June 18-22, Las Vegas, NV, USA.
[4] F. Karim, "An Interconnect Architecture for Networking Systems on Chips", IEEE Micro, vol. 22, no. 5, **(2002)**.
[5] J. Duato, S. Yalmanchili and L. Ni, "Interconnection Networks: An Engineering Approach", Morgan Kaufmann Publishers, San Francisco, **(2003)**.
[6] P. P. Pande, C. Grecu, M. Jones, A. Ivanov and R. Saleh, "Performance Evaluation and Design Trade-offs for Network on Chip Interconnect Architectures", IEEE Transactions on Computers, vol. 54, no. 8 **(2005)**.
[7] J. T. Draper and J. Ghosh, "A Comprehensive Analytical Model for Wormhole Routing in Multicomputer Systems", Journal of Parallel and Distributed Computing, vol. 32, no. 2, **(1994)**.
[8] R. Greenberg and L. Guan, "Modeling and Comparison of Wormhole Routed Mesh and Torus Networks", Proceedings of the 9th IASTED International Conference on Parallel & Distributed Computing and Systems, **(1997)** October, Washington DC,USA.
[9] W. J. Guan, W. K. Tsai and D. Blough, "An Analytical Model for Wormhole Routing in Multicomputer Interconnection Networks", Proceedings of the 7th International Conference Parallel Processing, **(1993)** April 13-16, Irvine, CA, USA.
[10] J. Duato, "A New Theory of Deadlock-Free Adaptive Routing in Wormhole Routing Networks", IEEE trans. Parallel and Distributed Systems, vol. 4, no. 12, **(1993)**.
[11] Y. Boura, C. R. Das and T. M. Jacob, "A Performance Model for Adaptive Routing in Hypercubes", Proceedings of International Workshop on Parallel Processing, **(1994)** December 11-16; Potsdam, Germany.
[12] M. Ould-Khaoua, "A Performance Model for Duato's Fully Adaptive Routing Algorithm in k-Ary n-Cubes", IEEE Transactions on Computers, vol. 48, no. 12, **(1999)**.
[13] M. K. Ould and H. A. Sarbazi, "Analytical Model of Adaptive Wormhole Routing in Hypercubes in the Presence of Hot Spot Traffic", IEEE Transactions on parallel and distributed systems, vol. 12, no. 3, **(2001)**.
[14] D. Bertsekas and R. Gallager, "Data Networks, second edition", Prentice-Hall, NJ, **(1992)**.
[15] V. N. Nitnaware and S. S. Limaye, "Folded Architecture of Scheduler for Area Optimization in An On-chip Switch Fabric", International Journal of Hybrid Information Technology, vol. 4, no. 1, **(2011)**.
[16] E. Baydal, P. L´opez and J. Duato, "Increasing the Adaptivity of Routing Algorithms for k-ary n-cubes", Proceedings of the 10th Euromicro Workshop on Distributed and Network-based Processing, **(2002)** January 9-11; Canary Islands, Spain.
[17] L. Benini and G. De Micheli, "Networks on chips: a new SoC paradigm", IEEE Computer, vol. 35, no. 1, **(2002)**.

Authors

Jin Liu

Dr. Jin Liu received the M.S.degree from University of Electrical Technology of China, Chengdu, China, and the Ph.D degree from Washington State University, Pullman, WA, USA in 2009, all in computer science. He is currently assistant professor at the College of Information Engineering, Shanghai Maritime University, Shanghai, China. He had held industrial R&D position at Windows Core Networking Division, Microsoft, WA, USA. His research interests include wireless sensor network, network on chip, distributed computing and semantic web.

Xiaofeng Wang

Professor Xiaofeng Wang received the M.S and Ph.D degree in Electronic Engineering from Shenyang Institute of Automation Chinese Academy of Sciences. He is currently a professor at the College of Information Engineering, Shanghai Maritime University, Shanghai, China. His main research interests are artificial intelligence and its application in traffic information and control engineering, data mining and knowledge discovery..

Hongmin Ren

Dr. Hongmin Ren received the M.S.degree from Dalian University of Technology, Dalian, China, the Ph.D degree from Fudan University, Shanghai, and conducted his post doctoral research at Fudan University and Changjiang Computer Group. He is currently associate professor at the College of Information Engineering, Shanghai Maritime University, Shanghai, China. His research interests include software architecture, software components technology and formalization methods..

Jin Wang

Dr. Jin Wang received the B.S. and M.S. degree in the Electronical Engineering from Nanjing University of Posts and Telecommunications, China in 2002 and 2005, respectively. He received Ph.D. degree in the Ubiquitous Computing laboratory from the Computer Engineering Department of Kyung Hee University Korea in 2010. Now, he is a professor in the Computer and Software Institute, Nanjing University of Information Science and technology. His research interests mainly include routing protocol and algorithm design, performance evaluation and optimization for wireless ad hoc and sensor networks. He is a member of the IEEE and ACM.

Jeong-Uk Kim

Dr. Jeong-UK Kim received his B.S. degree in Control and Instrumentation Engineering from Seoul National University in 1987, M.S. and Ph.D. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology in 1989, and 1993, respectively. He is a professor in SangMyung University in Seoul. His research interests include smart grid demand response, building automation system, and renewable energy.