Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

A general communication performance evaluation model based on routing path decomposition*

Ai-lian CHENG[†1], Yun PAN[†‡1,2], Xiao-lang YAN[2], Ruo-hong HUAN[3]

([1]*Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China*)

([2]*Institute of VLSI Design, Zhejiang University, Hangzhou 310027, China*)

([3]*College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China*)

[†]E-mail: {chengal, panyun}@vlsi.zju.edu.cn

Received Aug. 9, 2010; Revision accepted Mar. 28, 2011; Crosschecked May 31, 2011

Abstract: The network-on-chip (NoC) architecture is a main factor affecting the system performance of complicated multiprocessor systems-on-chips (MPSoCs). To evaluate the effects of the NoC architectures on communication efficiency, several kinds of techniques have been developed, including various simulators and analytical models. The simulators are accurate but time consuming, especially in large space explorations of diverse network configurations; in contrast, the analytical models are fast and flexible, providing alternative methods for performance evaluation. In this paper, we propose a general analytical model to estimate the communication performance for arbitrary NoCs with wormhole routing and virtual channel flow control. To resolve the inherent dependency of successive links occupied by one packet in wormhole routing, we propose the routing path decomposition approach to generating a series of ordered link categories. Then we use the traditional queuing system to derive the fine-grained transmission latency for each network component. According to our experiments, the proposed analytical model provides a good approximation of the average packet latency to the simulation results, and estimates the network throughput precisely under various NoC configurations and workloads. Also, the analytical model runs about 10^5 times faster than the cycle-accurate NoC simulator. Practical applications of the model including bottleneck detection and virtual channel allocation are also presented.

Key words: Analytical model, Communication performance, Network-on-chip (NoC), Queuing system, Routing path decomposition, Virtual channel

doi:10.1631/jzus.C1000281 **Document code:** A **CLC number:** TP302.7

1 Introduction

As a type of communication infrastructure, the network has been widely used in various systems and domains, such as biological systems, computer engineering, and electronics systems. The network structures are particularly popular in large systems. This may be because these architectures could keep design complexity extremely well under control and facilitate modular reuse (Marculescu and Bogdan, 2009). The network architectures have the inherent superiority to implement divide-and-conquer strategies. Also, many network problems can be abstracted and resolved by mathematical analysis, such as graph theory, queuing systems, and stochastic modeling. The theory provides reliability in analyzing the communication prototypes and network behaviors.

As more and more systems-on-chips (SoCs) adopt network communication paradigms, many design and evaluation methods in macro networks are gradually transferred to on-chip networks. However, the models developed for macro architectures cannot be applied in SoCs directly, since on-chip interconnects cannot be scaled down simply. The NoC fabric not only has to satisfy the requirements of transmission delay, but also is strictly limited by resource and energy consumption. Therefore, it is necessary to

‡ Corresponding author

* Project supported by the National High-Tech Research and Development Program (863) of China (No. 2009AA011706) and the Fundamental Research Funds for the Central Universities, China

create a specific performance evaluation model for an on-chip network according to system features and traffic patterns. The separation of communication and computation components in the design process (Keutzer *et al.*, 2000) further enhances the necessity of an individual model to evaluate the exact communication efficiency.

Many approaches, targeting various types of network architectures and traffic patterns, have been proposed for NoC performance estimation (Marculescu *et al.*, 2009). The wormhole routing technique (Ni and McKinley, 1993) has the merits of minimal communication delay, low buffer requirement, and simple hardware implementation; hence, it is the most popular in modern NoCs. In wormhole routing, a packet (i.e., a worm) is divided into a sequence of flits, which traverse along the path in a pipeline manner. As opposed to store-and-forward switching, the basic unit transmitted and buffered is a flit, and a long worm may stretch across several routers and links. The successive links occupied by the same packet have a certain relationship in terms of transmission latency. For example, when a packet is blocked, the waiting time of all links it occupies will increase. In the modeling process, the link dependency has to be decoupled before each link is analyzed. Moreover, the problem becomes even more complicated when virtual channel flow control (Dally, 1992) is considered. To mitigate the head-of-line blocking and increase channel utilization, virtual channels are usually implemented on top of a wormhole routing mechanism. The virtual channels associated with one physical channel are allocated independently to different packets, but they compete with each other for link bandwidth. It is not appropriate to apply merely a single queuing model or multiple-input multiple-output (MIMO) model to emulate the router's behavior. To solve this problem, some researchers resorted to the iterative methods based on routing priority in different dimensions (Draper and Ghosh, 1994) or topological sorting to obtain a computation order of links (Hu and Kleinrock, 1997). However, these approaches are complicated and do not explain how to handle the situations when multiple virtual channels are used.

In this paper, a novel simple method of routing path decomposition is proposed to deal with the link dependency exhibited in wormhole switching. Here, the concept of 'routing path' means the only available path between the specific source and the specific destination in a deterministic routing algorithm. All source-to-destination paths are composed of an ordered list of unidirectional links in the network. First, we assume that no virtual channels are used, and decompose the routing paths to obtain the physical links labeled with routing orders. According to this order, all links are classified into several categories. It is easy to eliminate the link dependency by choosing a proper computation order of link categories. Meanwhile, each link delivers traffic along different paths to different destinations. Considering the overlap of multiple routing paths on one physical link, we derive the modeling parameters from the weighted average over these of every path. Each link is modeled using the traditional queuing systems to compute the one-hop forwarding delay and blocking delay. Afterwards, the model is extended to consider virtual channel flow control. Since the modeling method is not limited to certain topologies or traffic patterns, a more general analytical model can be provided for assorted network architectures. The flexibility in configuration makes the model much more appealing in various situations, such as iterative optimization and large design space exploration.

2 Related works

Given an initial NoC design specification, a performance evaluation model is often used to identify reasonable configurations, congestion points, or redundant resources. The model also needs to be tractable for acceptable accuracy and fast evaluation speed.

One type of performance model widely used is the simulator, including Worm_sim (http://www.ece.cmu.edu/~sld/software/worm_sim.php), PopNet (Li *et al.*, 2003), NIRGAM (http://nirgam.ecs.soton.ac.uk/), and NoCSim (http://research.cs.tamu.edu/codesign/nocsim/). These simulators are usually implemented in high level modeling languages, such as C, C++, and SystemC. To obtain the desired precision, these models are mainly cycle accurate, and their simulation processes are time and resource consuming. Simulation models are usually developed for specific hardware structures and applications; thus,

most of them are poorly scalable in topologies and traffic patterns. Since they provide very accurate estimations, the simulators typically engage in the late design phases when exploration space has been lowered to only a few choices.

Conversely, the analytical models have the competitive advantages of good flexibility and high efficiency. They are based on mathematical formulae and theories, and expose the impacts on performance of different structural parameters from the statistical aspect. Due to much faster evaluation speed, analytical models are often used in the early stages for design optimization and design space reduction.

Classical and valuable work about analysis techniques for wormhole routing comes mainly from traditional macro-network studies (Ciciani *et al*., 1997; Ould-Khaoua and Sarbazi-Azad, 2001; Sarbazi-Azad *et al*., 2001). Much research in NoC field refers to methods widely used in macro networks and parallel computing domains. Moadeli *et al*. (2007a; 2007b) used the methodology proposed in Draper and Ghosh (1994) to generate the analytical models for mesh and Spidergon architectures, respectively. The mesh model does not support virtual-channel flow control. The Spidergon routing scheme uses two virtual channels only to avoid deadlock, and the model neglects the effect of multiple virtual channels. Hu *et al*. (2006) presented a performance model based on queuing theory to solve the buffer allocation problem. However, it is applicable only to store-and-forward or virtual cut-through switching mechanism without virtual channels. Ogras and Marculescu (2007) proposed a general router model for arbitrary topologies and provided three useful performance metrics for design and optimization purposes. Regrettably, the virtual-channel configuration was not included in their model. Moraveji *et al*. (2008; 2009) used Duato's methodology to build an exhaustive mathematical model for fully adaptive routing in irregular networks, and evaluated the system performance at the message level. Duato's methodology requires two classes of virtual channels separately for the implementation of fully adaptive routing and the deadlock-free routing function, and the model massively calculates the probability that each virtual channel is traversed by the given source-to-destination message. Another mathematical model of deterministic wormhole routing (Sarbazi-Azad, 2003) also considers the effect of virtual channel multiplexing on network performance for the hypercube structure.

The major contribution of this work is to demonstrate a cost-effective technique of performance evaluation for NoCs with wormhole switching and virtual channels, and the method of routing path decomposition is proposed to clarify the link dependency. An analytical model for any type of topology is constructed conveniently based on the traditional queuing systems. Useful information about network throughput and average packet latency is provided for designers.

3 Link modeling by queuing systems

In this section, the analytical method for performance estimation is presented, and the modeling flow is demonstrated extensively. We first apply the approach of routing path decomposition to decouple the link dependency under wormhole routing mechanism, and then build the performance evaluation model based on the traditional queuing system. Finally, a simple extension of the model is presented to include the virtual-channel configuration.

3.1 Basic assumptions and notations

The system under consideration is composed of unidirectional links, routers, and processing elements (PEs). Routers are interconnected by links, and PEs are attached to routers, sending data to and receiving data from the network. The local buffers in PEs are commonly regarded as infinite, while the routers have finite-size input buffers. The latter are typically very small to reduce silicon area and energy leakage, especially for NoCs using wormhole routing. Deterministic routing and no cycle of link dependency are assumed in our modeling analysis. Cycle of link dependency is as defined in Hu and Kleinrock (1997), and it may induce deadlock situations. For example, a set of messages have reserved a cycle of links, and each is requesting a channel held by another message in the set. As none of these links is available on request, the messages reach a deadlock. No cycle of link dependency ensures deadlock-free routing. More importantly, this unidirectional feature helps us decompose the routing paths. Buffer organization of the virtual channels is not included at first. Later, the

model is extended to support multiple virtual channels.

In terms of traffic characterization, a 'Poisson' distribution of the packet arrival rate at each input port is assumed to employ the Markov process, similar to most previous works (Guz *et al*., 2006; Bahn and Bagherzadeh, 2008; Arjomand and Sarbazi-Azad, 2009; Krimer *et al*., 2009). For simplicity, we assume that all packets have a fixed size, namely L. The packet length and buffer depth are both measured by flits to facilitate the computation.

The notations used throughout this paper are listed in Table 1. The detailed descriptions are presented in Section 3.3 and Fig. 3.

Table 1 Notations used in this paper

Symbol	Description
B	The buffer depth in terms of flits
κ	The port number of a router
L	The packet size in terms of flits
$l_{i,p}$	Link i of path p; $1\le i\le d_p$, where d_p is the number of hops in path p
$q_{i,p}$	The occupancy time of link $l_{i,p}$ traversed by a packet, during all its flits' transmissions
$\lambda_{i,p}$	The packet arrival rate of link $l_{i,p}$
$f_{i,p}$	The one-hop forwarding delay for a header flit through link $l_{i,p}$
$\eta_{i,p}$	The propagation delay for a flit to pass link $l_{i,p}$
$b_{i,p}$	The blocking delay for a header flit to wait in the buffer of link $l_{i,p}$
$w_{i,p}$	The waiting time of the queuing model based on link $l_{i,p}$
$s_{i,p}$	The service time of the queuing model based on link $l_{i,p}$
T_p	The average packet latency of path p from source PE to destination PE

3.2 Routing path decomposition

As mentioned previously, in wormhole routing mechanism, a packet stretching over multiple links has established a certain relationship between the links it occupies. Apparently, the number of links a packet can spread is determined by the ratio of packet length to buffer depth, that is, L/B. From another point of view, the specified link is occupied by a packet until the header flit reaches the next $\lceil L/B\rceil$th link. However, if the packet is absorbed by the destination PE not far away, links behind that PE will not be counted in this range. More precisely, assuming a packet has just obtained access to link i of path p, there are still $N_{i,p}$ links it must traverse before link i can be thoroughly released, and $N_{i,p}$ is given by

$$N_{i,p}=\min(\lceil L/B\rceil, d_p-i), \tag{1}$$

where d_p is the total number of links in path p. Eq. (1) implies that if link i locates too close to the destination so that buffers in the residual path are not large enough to accommodate a whole packet, the number of links to be considered is truncated.

Consequently, we compute the occupancy time of link i as follows:

$$q_{i,p}=\begin{cases}\eta_{i,p}+r_{i,p}, & N_{i,p}=0,\\ \sum_{j=i}^{i+N_{i,p}}(w_{j+1,p}+b_{j+1,p}+\eta_{j,p})+r_{i+N_{i,p},p}, & N_{i,p}\ge 1,\end{cases} \tag{2}$$

where $r_{i+N_{i,p},p}$ represents the residual time during which the packet traverses the last link involved, either accepted by the subsequent buffer or sunk into the destination PE. Since the residual time is very small compared with the link occupancy time, we neglect it for simplicity.

Eq. (2) means that the occupancy time of link i is equal to the time interval that the packet spends in the routers and channels along the path, before it reaches link $i+N_{i,p}$.

More importantly, from Eq. (2), we observe that, to calculate the occupancy time of link i, the transmission time on the $N_{i,p}$ subsequent hops should first be obtained. It is regarded that link i is unidirectionally dependent on the finite $N_{i,p}$ links downstream, and the nearer it is to the destination, the fewer links it relies on. This is a simple instance in only one of the routing paths. In the real network, the paths overlap and interlace each other to share the physical links, further aggravating the difficulty in decoupling the link dependency.

To solve this problem, we analyze the routing paths and label each link with a routing order. Then it is possible to find the specific links that locate at the tail of all paths through them. Keeping tracks, the last-but-one link and similar others can be easily picked out in sequence. Note that the original routing paths have been decomposed and the physical links are classified into a series of categories in a reverse order of routing. That is, the last links are assigned to

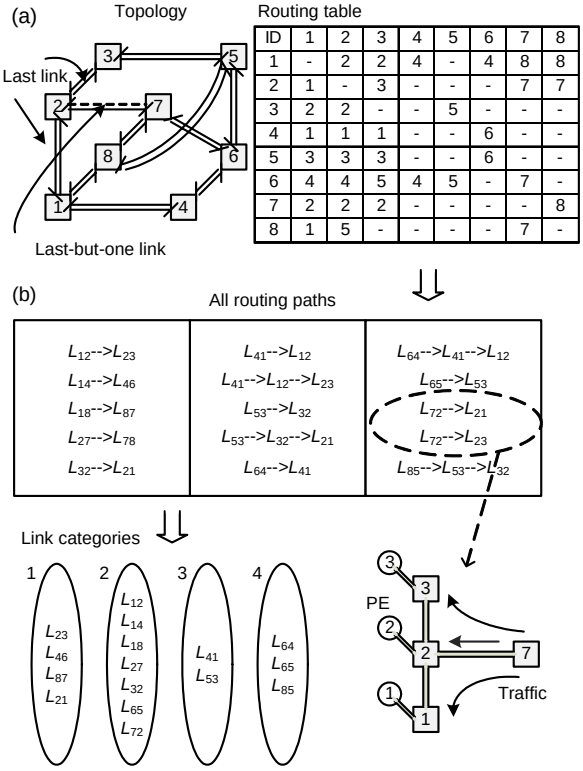
Category 1, and so on. Any link in a class certainly relies on at least one link in the previous classes, and this leads to a direct relationship between the successive categories. In the modeling process, links inside the same category have no dependency and can be computed in parallel, while those from different categories should be resolved in a numbered order if recursive calls are not desired.

In the sorted link categories, one link may depend on multiple links in the previous categories. This is because the physical link is shared by several routing paths, and the packets through it are forwarded to different branches in a given probability. We use the weighted average method to calculate the transmission time in the physical link. In other words, the modeling parameters are derived for each relevant path, and weighted by the traffic forwarding probability respectively. Then the results are summed up. The ultimate value is used to evaluate the overall latency of the physical link.

An example of the decomposition steps is illustrated in Fig. 1. An arbitrary network topology (Ogras and Marculescu, 2007) is selected to apply the decomposition process. The routing table for the specific topology is shown at the right side in Fig. 1a, in which ID of the first column gives the source node, and ID of the first row gives the destination node. The table represents the subsequent router to be traversed from the specified source to the specified destination, with '-' meaning that no path exists between the two IDs. We derive all the routing paths from the routing table and the corresponding four sorted link categories through the decomposition process. In particular, two of the last links and a last-but-one link L_{72} are depicted in this figure, and their original routing paths are situated in the dashed ellipse. To resolve link L_{72} in Category 2, the unidirectional last links of L_{23} and L_{21} should be computed in precedence. The traffic in L_{72} can be split into three parts, namely packets to PE1, PE2, and PE3. Then the modeling parameters are computed separately and finally averaged.

3.3 Modeling flow based on the queuing systems

The complete modeling flow for NoCs without virtual channels is illustrated in this subsection. The block diagram of the modeling method is shown in Fig. 2.



ID	1	2	3	4	5	6	7	8
1	-	2	2	4	-	4	8	8
2	1	-	3	-	-	-	7	7
3	2	2	-	-	5	-	-	-
4	1	1	1	-	-	6	-	-
5	3	3	3	-	-	6	-	-
6	4	4	5	4	5	-	7	-
7	2	2	2	-	-	-	-	8
8	1	5	-	-	-	-	7	-

Fig. 1 An example of applying the routing path decomposition method

(a) An arbitrary network topology and the routing table; (b) The decomposition and classification process of the routing paths

Given a specific NoC and its application mapping, we obtain inputs for the modeling algorithm including architecture and application parameters. The former contain the routing function and propagation delay in a router, and the latter contain traffic injection rates of PEs and forwarding probabilities to different branches. With the help of the decomposition method in Section 3.2, we successfully decouple the link dependency from the routing paths (Step 1 in Fig. 2), and then the queuing model could be applied to the separate physical links. As links from different categories have a computational order, it is necessary to use two loops for the iteration of all links, that is, Loop 1 for different categories and Loop 2 for links in the same category.

Next, we use the Markov models of M/M/1/K and M/M/1 systems to calculate the packet waiting time of each link. There are three key parameters to determine in the queuing models, namely the traffic arrival rate, the system service rate, and the waiting

buffer size. The system service rate is derived as the average of all branches' service time (Step 2 in Fig. 2). The non-blocking transmission delay is resolved in Step 3, and the blocking delay is calculated by an M/M/1 system (Step 4). These two delays are two parts of the total waiting time for a single hop. Finally, after links in all categories are resolved, the average packet latency of network is computed for each source-to-destination path (Step 5). The highlighted blocks represent the intermediate results derived from the preceding steps. Steps 2–4 are all associated with the computation of queuing models.

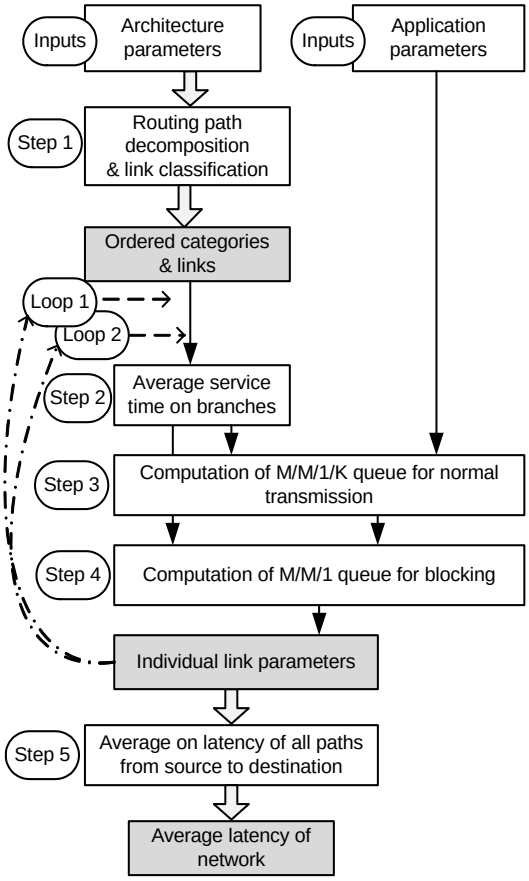


Fig. 2 Block diagram of the analytical modeling flow

In the modeling process, we choose the physical link as the modeling target, other than the router. It is because the router locates at the cross point of several transmission paths and the packet rate changes along any direction. Therefore, we refer to the method presented in Hu and Kleinrock (1997) and develop the model in abstraction of elements around a link, including the upstream crossbar and the downstream buffer (Fig. 3). Also, the packet, instead of the flit, is regarded as a customer in the queuing system, since a packet's transmission cannot be interrupted even though it is divided into units of flit.

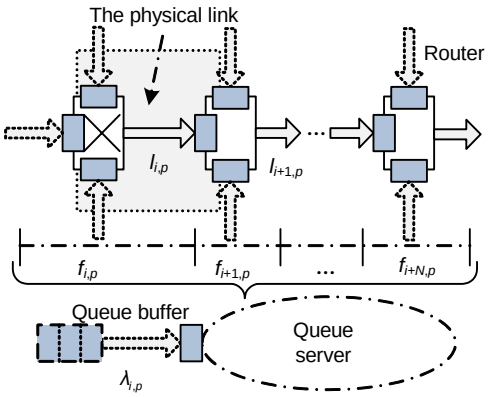


Fig. 3 Notations used in the link model and the corresponding delay parameters

To calculate the non-blocking transmission delay, we employ an M/M/1/K system to analyze the link model shown in Fig. 3. It has three input parameters, namely the traffic arrival rate, the system service rate, and the buffer size. The traffic arrival rate $\lambda_{i,p}$ of each link is calculated from the amount of data exchange in the application and its mapping. As assumed, the distribution of the packet arrival rate follows a Poisson process.

The second parameter is the reciprocal of the service time, i.e., the duration in which a packet is transmitting along the specific link. It is similar to the link occupancy time mentioned previously, and here we use a simpler expression to approximate it. For simplicity, we assume the packet service time follows an exponential distribution. More precisely, it is defined as follows:

$$s_{i,p} = \sum_{j=i+1}^{N_{i,p}} f_{j,p}, \tag{3}$$

where $f_{j,p}$ represents the one-hop transmission time in the subsequent link j.

If the physical link has multiple subsequent branches, the service time is calculated for each branch. Assuming branch z is traversed by a packet at a given forwarding probability $P_{i,z}^{\mathrm{f}}$, the average service time is weighted by the probability and summed up eventually; that is,

$$s_{i,p}^{\text{avg}} = \sum_{z} \left(P_{i,z}^{\text{f}} \cdot \sum_{j=i+1}^{N_{i,p}} f_{j,z} \right). \tag{4}$$

As far as the buffer depth is concerned, the queuing model has a different capacity from the actual buffer in the router. As shown in Fig. 3, all packets before the crossbars have an influence on the waiting time of the physical link considered. That is, packets requesting this physical link are all regarded as customers waiting for the queuing server. Therefore, the queue buffer is extended to $(\kappa-1)+\lceil B/L \rceil$ in size of packets, where κ is the port number of a single router. The crossbar has κ inputs and κ outputs respectively, but packets from a certain input will not turn back to the output in the same direction, so only $\kappa-1$ inputs in crossbar can request for the unidirectional link.

For an M/M/1/K system with the three variables mentioned above, the average waiting time is computed by (Lu, 2009)

$$w_{i,p} = \frac{\rho_{i,p}}{\mu_{i,p}(1-\rho_{i,p})} - \frac{K\rho_{i,p}^{K}}{\mu_{i,p}(1-\rho_{i,p}^{K})}, \tag{5}$$

where $\rho_{i,p}=\lambda_{i,p}/\mu_{i,p}$ reflects the traffic load of the queuing system. It is notable that Eq. (5) is applicable only in the situation where the network is not overloaded ($\rho_{i,p}<1$, $\forall l_{i,p}$). $\mu_{i,p}=1/s_{i,p}$ is the average service rate.

This M/M/1/K queue has another important parameter, the blocking probability, which is given by

$$P_{i,p}^{\text{b}} = \frac{(1-\rho_{i,p})\rho_{i,p}^{K}}{1-\rho_{i,p}^{K+1}}. \tag{6}$$

In an ideal queuing system, when an arriving customer finds the buffer full, the customer is lost and the system is blocked. However, in many real networks, packets are not rashly discarded but keep on waiting in the upstream routers until the next buffers are available. Therefore, when the network is severely congested, the average waiting time will increase sharply. To emulate this phenomenon, we use an M/M/1 queue to calculate the blocking delay. We name it the blocking model, and it has the same packet arrival rate and service time as the corresponding link model. The average waiting time of the infinite queue is achieved as (Lu, 2009)

$$w_{i,p}^{\text{inf}} = \frac{\rho_{i,p}}{\lambda_{i,p}(1-\rho_{i,p})}. \tag{7}$$

The blocking delay $b_{i,p}$ is induced by the situation where one of the downstream buffers is full. Its value is small when the subsequent $N_{i,p}$ links are not congested, and increases as the blocking probability becomes high. Thus, the blocking delay is approximated as

$$b_{i,p} = \left(P_{i,p}^{\text{b}} + \sum_{j=i+1}^{N_{i,p}} (P_{i,j}^{\text{f}} \cdot P_{j,p}^{\text{b}}) \right) w_{i,p}^{\text{inf}}, \tag{8}$$

where $P_{i,j}^{\text{f}}$ represents the packet forwarding probability from link i to link j.

In the above computations, the packet transmission time $w_{i,p}$ and the blocking delay $b_{i,p}$ are derived separately from solutions of the two queuing models. We observe that the single-hop forwarding time between two adjacent routers is composed of three parts: the non-blocking transmission time, the propagation time, and the blocking delay. Hence, it has the following relationship:

$$f_{i,p}=\eta_{i,p}+w_{i,p}+b_{i,p}. \tag{9}$$

This parameter is used to analyze the links dependent on it, such as in Eq. (3).

After links in all categories are resolved, we re-organize them into the source-to-destination routing paths for the computation of end-to-end packet latency. This latency is the time interval seen by packets originating from the source nodes to their individual destinations. For a routing path, the time span is naturally cut into several one-hop forwarding times by the routers. Thus, the average latency for path p is expressed as

$$T_p = \sum_{i=1}^{d_p} f_{i,p} + (L-1), \tag{10}$$

where d_p is the number of links in path p, and the last item in Eq. (10) denotes that each body flit takes one time unit to sink into the destination PE.

Finally, the overall packet latency T_G for the specific network is averaged over all routing paths:

$$T_G = \frac{1}{N_G}\sum_{p\in G} T_p, \tag{11}$$

where G represents the set of routing paths in the network, and N_G gives the number of elements in this set.

3.4 Extension for multiple virtual channels

In the previous analysis, we assume that no virtual channels are used and that each input port connects to only one physical channel of the router. In this subsection, the analytical model is extended to support multiple virtual channels, as the buffer organization of virtual channels is widely used with wormhole routing mechanism to improve the communication performance. Typically, for routers with virtual channels, incoming packets are independently allocated to different channels. When one virtual channel is blocked, packets in another channel may still traverse to their outputs. Conversely, in routers without virtual channels, the packet blocked at the head of channel will hinder all the transmissions from this single queue, leaving the physical link idle. Also, the virtual channels associated with one physical channel compete with each other for physical bandwidth. To summarize, this organization does not change the channel bandwidth, but helps to increase the utilization of the physical link.

Consider the effects of virtual channel multiplexing on the five steps of the modeling approach presented in Fig. 2. As we use virtual channels for performance reasons other than avoidance of deadlock routing, the routing strategy and the decomposition method in Step 1 are not affected. For Step 2, computation of the service time is dependent on the parameters of possible subsequent links. The effects of virtual channel multiplexing are limited in a single link, and thus Step 2 stays unchanged. Similarly, virtual channel multiplexing also has no influence on Step 5.

For the link model in Step 3, we still take the physical link as a single queue as in the original model. Limited by the physical bandwidth, only one packet can be transmitted at a time unless it encounters blocking. Thus, the physical link serves packets from different channels in a time-multiplexed manner. When no blocking occurs, the integrity of all virtual channels is approximate to one physical buffer, except for the first-come-first-served (FCFS) rule. Assuming that the packets are allocated independently to each virtual channel and that the round-robin arbitration is applied in the crossbar, the effect of non-FCFS situations is negligible.

Fig. 4 shows the block diagram of virtual channel modeling. Assuming V_c and B_v denote the number and depth of virtual channels in each input port respectively, we have

$$B = B_v \cdot V_c. \tag{12}$$

Also, the number of dependent links given by Eq. (1) changes to

$$\begin{aligned} N_{i,p}^{v} &= \min\left(\lceil L / B_v \rceil,\ d_p - i\right) \\ &= \min\left(\lceil V_c \cdot L / B \rceil,\ d_p - i\right). \end{aligned} \tag{13}$$

Moreover, because of virtual channel multiplexing, the port number of crossbar increases, and the buffer size of the queuing system changes to

$$K = V_c \cdot \left\lceil (\kappa - 1) + \lceil B_v / L \rceil \right\rceil. \tag{14}$$

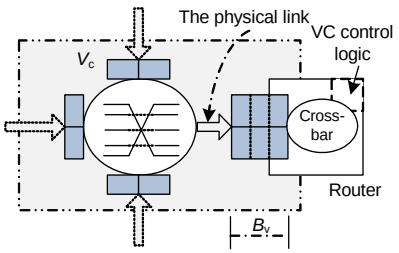


Fig. 4 Block diagram of virtual channel modeling

With these modifications, the extended link model has the same parameters as the original one. The traffic arrival rate remains the same as $\lambda_{i,p}$, and the service time is given in Step 2. Again, Eq. (5) is applied to compute the average waiting time $w_{i,p}$ in this physical link.

From the above analysis, we notice that the buffer depth of the M/M/1/K system increases by V_c times, and that the blocking probability given by Eq. (6) reduces consequently. Therefore, the blocking delay estimated by the M/M/1 system also decreases for the same packet arrival rate. Virtual channel flow control improves utilization of the physical channels. In other words, when a virtual channel is blocked, other virtual channels associated with the same

physical channel can establish another transmission. The probability that all virtual channels happen to be blocked is greatly reduced. It is reported that adding virtual channels to a network has little effect on latency at low traffic rates, but greatly improves the overall throughput (Dally, 1992). It is easy to understand this fact, since virtual channels have a strong effect on the system performance under blocking conditions.

At heavy traffic load, the multiplexing transmission is limited by the bandwidth capacities of links, and the blocking probability increases rapidly. The M/M/1 queuing system is still suitable to estimate the waiting time when the network becomes congested. Thus, the blocking model in Step 4 is adopted directly in the extended model.

In addition, extra control logic of virtual channels is added and the packet transmission delay in one hop slightly increases.

Note that the extension of the analytical model is compatible with the original one, in which the number of virtual channels is equal to one. Also, the analytical method of virtual channels is modeled at a fine granularity of the router level, and configurations of virtual channels can be different from router to router. Hence, the extended model can explore the virtual channel allocations in heterogeneous architectures.

4 Experimental results

In this section, we report the accuracy and runtime of the analytical model, and present two case studies of performance evaluation used in the NoC design process. The model was implemented in C language. For comparison, the cycle-accurate simulator Worm_sim was employed to generate simulation results under the same configurations and workloads. All experiments were tested on a DELL 2950 server running Linux OS.

For configurations of the network tested in the experiments, the link bandwidth was 64 bits/cycle. Packets from PEs were divided into 64-bit flits before being injected into the network. In the absence of contention, the router service time was four cycles. Each simulation experiment was run for 2×10^5 cycles, and statistics was not gathered for the first 2×10^4 cycles to avoid distortions due to start-up transient. Besides, multiple simulations with different seeds were undertaken to collect relevant averages.

4.1 Average latency and throughput

First, the common tests of uniform traffic in a regular network were undertaken with our model and with the simulator Worm_sim operating at the flit level. For configurations, 2D mesh topology and *XY* routing were used. Each PE injected packets at the same rate and the message destinations were randomly selected to ensure uniform traffic. The network parameters were comprehensively regulated to equal values for fair comparison.

To obtain the simulation results at the specified injection rate, the Worm_sim simulator was run 50 times with different seeds; the results were averaged such that the measured value is within one standard deviation of the actual latency with 95% confidence. For different injection rates, the number of packets gathered ranged from 5×10^4 to 2×10^5. Also, we can find the network throughput, which is a packet injection rate at which the average latency increases sharply. The traffic load was normalized relative to the derived throughput, and the relative error between analysis and simulation results was calculated for different load percentages.

Detailed results for the average latency predicted using the analytical model against those from the simulator, as well as the corresponding errors, are plotted in Fig. 5.

As can be seen in Figs. 5a and 5c, the proposed model presented a good approximation of the average network latency in a wide range of traffic loads, and estimated the saturation point accurately. The mean relative error between the analysis and simulation results was about 13%, with obvious deviations in the presence of heavy traffic. At the low traffic load, the analytical model worked very well when the blocking situations rarely occur. As the injection rate increased, the relative error between the analysis and simulation results increased slowly. At the high load, the difference further enlarged. The slight deviations between the analysis and simulation results may be induced by the approximations made in the analysis process, including the exponential distribution for service time and packet arrival rate. Also, the traditional queuing system is applicable only to stable systems with $\rho<1$, and congestions may cause some links to become

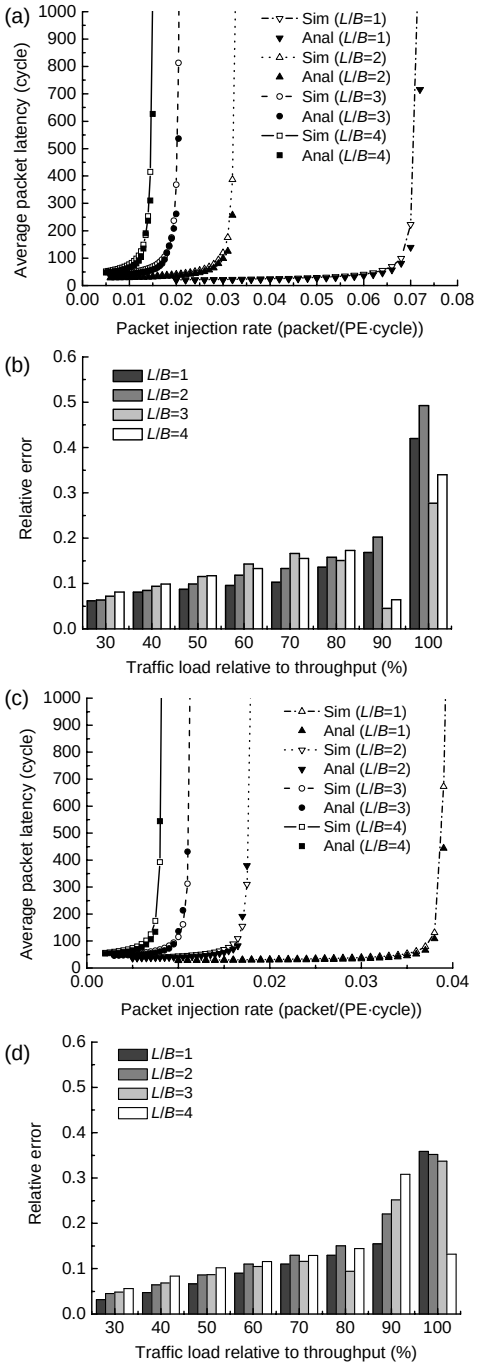


Fig. 5 Average packet latency as a function of the injection rate derived from analysis (Anal) and simulation (Sim) and the relative error

(a) Configuration: 4×4 mesh, buffer depth B=4 flits, packet length L=4, 8, 12, and 16 flits separately; (b) Relative error of analysis to simulation results from 30% to 100% load for different L/B ratios in 4×4 mesh; (c) Configuration: 8×6 mesh, buffer depth B=4 flits, packet length L=4, 8, 12, and 16 flits separately; (d) Relative error of analysis to simulation results from 30% to 100% load for different L/B ratios in 8×6 mesh

overloaded, leading to incorrect predictions. However, for some configurations, the analytical model still performed well with heavy traffic load. This may be attributed to the blocking model, which predicts the blocking delay accurately.

Figs. 5b and 5d show the relative error as a function of the traffic load for different configurations. The analytical model performed well in various configurations under light and medium traffic loads. The relative error grew as the load percentage increased and the model lost its accuracy at very high load. In spite of this, the error was acceptable for coarse-grained estimation and the results still revealed the increasing trend of latency as a function of workloads. Conversely, the analytical model took only 1.69 s to complete the experiments in 4×4 mesh and run by five orders of magnitude faster than the simulator, which cost 6 h to collect all the sample data. The analytical method with fast speed is especially helpful in exploring the large design space and searching for the optimal configuration.

4.2 Latency distribution in channel

Next, we apply the proposed model to hotspot traffic pattern and illustrate the fine-grained estimations when performance degradation and bottleneck may exist.

Hotspot traffic is very common in real embedded applications, and designers need to know the performance degradation induced by local congestion. Again, taking a 4×4 mesh network as an example, we chose routers (1, 0) and (2, 2) respectively as the hotspot, and compared their average latency with that of uniform traffic pattern. For the hot-spot node, the PE connected to it sent and attracted an extra proportion of the total traffic, and other PEs still received packets with an equal probability. For configurations, the buffer depth was four flits, and the packet length was eight flits; the hotspot proportion was set to 6.25%. The packet latency under three traffic patterns is shown in Fig. 6.

The hotspot traffic degraded network throughput slightly, and the position of hotspot also had an influence on system performance. However, this figure gives us little information on congestion in space and the performance bottlenecks. The latency distribution in different channels is desired in some situations. With the help of our model, the latency in four

channels around a router can be derived. Fig. 7 plots the latency distribution in all channels under the above three traffic patterns at the same traffic injection rate of 0.025 packets/(PE·cycle).

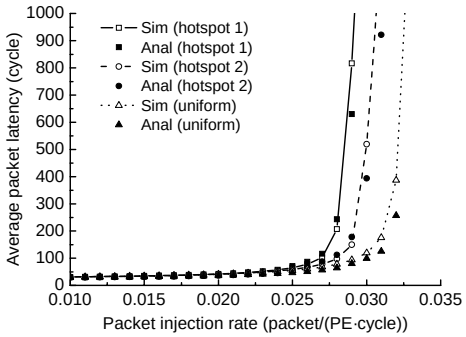


Fig. 6 Average packet latency as a function of the injection rate derived from analysis (Anal) and simulation (Sim) under uniform and hotspot traffic pattern
Configuration: 4×4 mesh, buffer depth=4 flits, packet length =8 flits, hotspot1=router (1, 0), hotspot2=router (2, 2), hotspot traffic=6.25%

For uniform traffic, the latency in channel was symmetrically distributed; for example, latency in the south channel of router (0, 0) was equal to that in the north channel of router (3, 0). Latency in the local buffers was much larger than that in the surrounding channels. This is because an infinite buffer is used in PE. When the traffic injection rate increases, the local buffer stores more and more packets, and its latency increases significantly. On the contrary, the traffic arrival rate of each link changes more slightly, and latency in these links is relatively small.

Compared to uniform traffic, the hotspot led to asymmetrical latency distribution, and even local congestion. For the hotspot in router (1, 0), latency in the local buffer increased notably. This is because the extra traffic generated by the source PE induces contention at the local buffer. Also, contention propagates along the west channels and makes links far away from the hotspot also have high latency. For the hotspot in router (2, 2), channels with high latency were mainly the east channels, and fewer channels were congested. We conclude that the hotspot is prone to causing congestion in the local buffers. Moreover, the hotspot in the middle region had less effect on the overall throughput than that in the boundary. The reason may be that the hotspot in the boundary induces an even worse imbalance of the traffic distribution in the whole network.

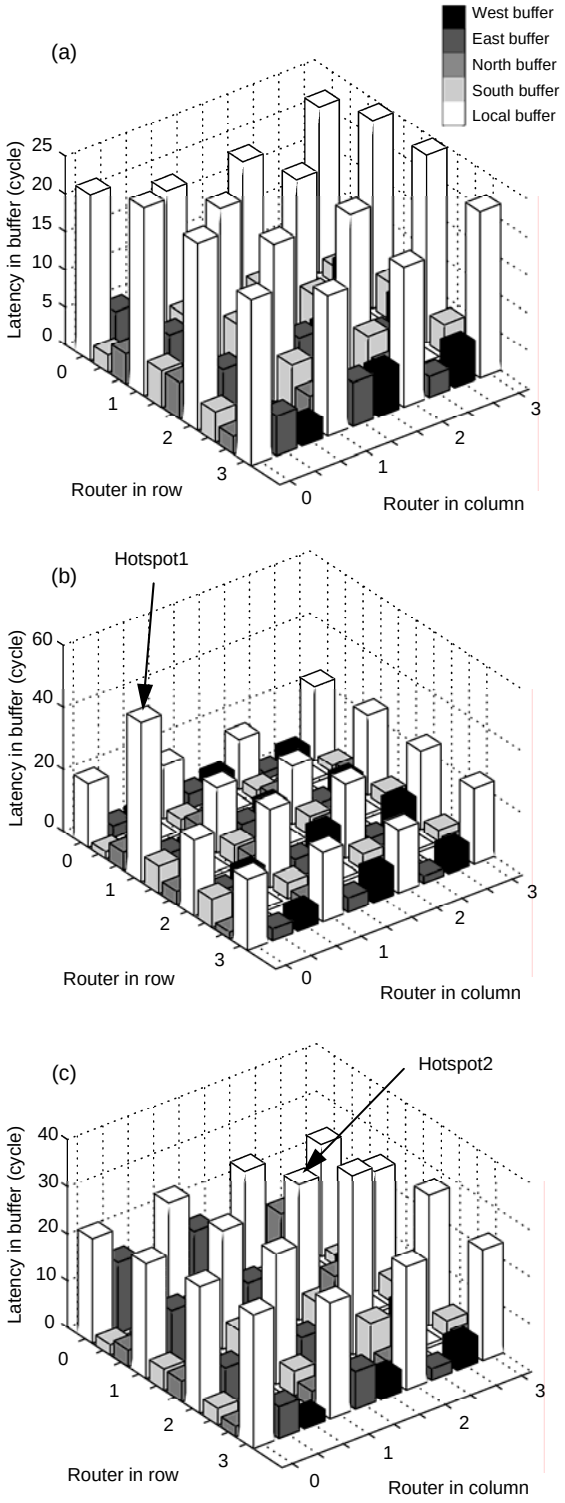


Fig. 7 Packet latency distribution in each channel
(a) Uniform traffic; (b) Hotspot1=router (1, 0), hotspot traffic=6.25%; (c) Hotspot2=router (2, 2), hotspot traffic=6.25%. Configuration: 4×4 mesh, buffer depth B=4 flits, packet length L=8 flits, injection rate=0.025 packets/(PE·cycle)

Additionally, the latency distribution reveals that congestion often occurs at the local buffer, while latency in other channels is relatively small. It is easy to understand since the infinite buffer at the source PE adjusts the traffic injection rate. At light traffic load, few links are blocked, and the arrival rate of each channel increases with the total injection rates. However, at heavy load, many links could not accept new packets, and congestion first occurs at the source buffer. That is, the arrival rate at the links is limited by the blocking phenomenon and is no longer proportional to the total injection rates.

4.3 Virtual channel allocation

Another case study of virtual channel allocation is shown herein. To mitigate the head-of-line blocking phenomenon in wormhole routing, the virtual channel technique is widely employed in NoCs. We evaluate the performance improvement of adding more virtual channels to the physical channels with the analytical method. In the previous experiment with the hotspot in router (1, 0), we observe that its overall latency is relatively high. In this experiment, we used two virtual channels for the whole network to reduce its latency. The buffer depth of each virtual channel was equal to four flits to keep the number of dependent links (given by Eq. (1)) constant. Moreover, packets had to select one of the virtual channels before entering the router; thus, the actual service time was assumed to increase by one time unit, that is, five cycles. Then we repeated the pervious experiment. Fig. 8 shows the latency distribution.

Compared to Fig. 7b, packet latency in all channels was reduced efficiently with the help of virtual channels. Also, we notice that the local buffer at the hotspot benefited much from the use of virtual channels, and local contention was alleviated.

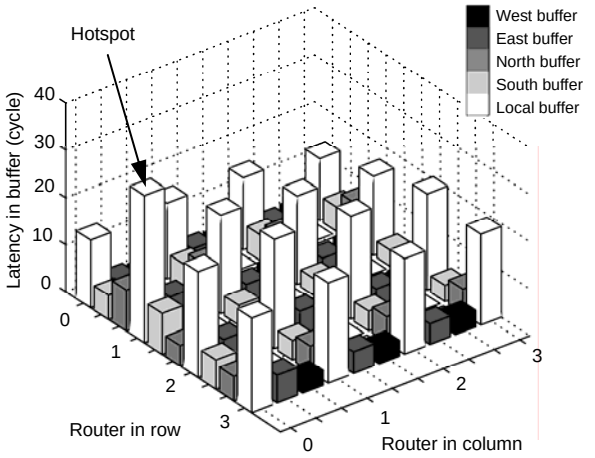


Fig. 8 Packet latency distribution with virtual channels in the specified channel

Configuration: 4×4 mesh, buffer depth=4 flits, packet length=8 flits, hotspot=router (1, 0), hotspot traffic=6.25%, injection rate=0.025 packets/cycle, VC number in the east channel of router (3, 1)=2

5 Conclusions

High-level modeling and simulation methods are necessary for the evaluations of various options in NoC architectures, in order to meet the stringent requirements of the increasing complicated future MPSoCs. An analytical solution is proposed in this paper, and a performance model based on the routing path decomposition and the traditional queuing system is presented in detail. The analytical model features comprehensive and elaborate performance estimation for a variety of networks under different configurations.

We validate the reliability of the proposed model through extensive experiments and demonstrate its usefulness in performance prediction, including computation of the overall packet latency for the network and the latency distribution in different channels. The analytical model benefits from the unique modeling method and presents a good approximation to the actual communication performance. With acceptable accuracy and fast evaluation speed, it is quite suitable for early-stage estimation and large space exploration.

In future research, we will improve the model in the presence of heavy workload and validate the reliability of the proposed model when multiple virtual channels are considered.

References

Arjomand, M., Sarbazi-Azad, H., 2009. A Comprehensive Power-Performance Model for NoCs with Multi-flit Channel Buffers. Proc. 23rd Int. Conf. on Supercomputing, p.470-478. [doi:10.1145/1542275.1542341]

Bahn, J.H., Bagherzadeh, N., 2008. Design of simulation and analytical models for a 2D-meshed asymmetric adaptive router. *IET Comput. Digit. Techn.*, **2**(1):63-73. [doi:10.1049/iet-cdt:20070043]

Ciciani, B., Colajanni, M., Paolucci, C., 1997. An Accurate

Model for the Performance Analysis of Deterministic Wormhole Routing. Proc. 11th Int. Parallel Processing Symp., p.353-359. [doi:10.1109/IPPS.1997.580926]

Dally, W.J., 1992. Virtual-channel flow control. *IEEE Trans. Parall. Distr. Syst.*, **3**(2):194-205. [doi:10.1109/71.127260]

Draper, J.T., Ghosh, J., 1994. A comprehensive analytical model for wormhole routing in multicomputer systems. *J. Parall. Distr. Comput.*, **23**(2):202-214. [doi:10.1006/jpdc.1994.1132]

Guz, Z., Walter, I., Bolotin, E., Cidon, I., Ginosar, R., Kolodny, A., 2006. Efficient Link Capacity and QoS Design for Network-on-Chip. Proc. Design, Automation and Test in Europe, p.1-6. [doi:10.1109/DATE.2006.243951]

Hu, J.C., Ogras, U.Y., Marculescu, R., 2006. System-level buffer allocation for application-specific networks-on-chip router design. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst.*, **25**(12):2919-2933. [doi:10.1109/TCAD.2006.882474]

Hu, P.C., Kleinrock, L., 1997. An Analytical Model for Wormhole Routing with Finite Size Input Buffers. Proc. 15th Int. Teletraffic Congress, p.549-560. [doi:10.1016/S1388-3437(97)80058-3]

Keutzer, K., Newton, A.R., Rabaey, J.M., Sangiovanni-Vincentelli, A., 2000. System-level design: orthogonalization of concerns and platform-based design. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst.*, **19**(12):1523-1543. [doi:10.1109/43.898830]

Krimer, E., Erez, M., Keslassy, I., Kolodny, A., Walter, I., 2009. Packet-Level Static Timing Analysis for NoCs. Proc. 3rd ACM/IEEE Int. Symp. on Networks-on-Chip, p.88. [doi:10.1109/NOCS.2009.5071451]

Li, S., Peh, L.S., Jha, N.K., 2003. Dynamic Voltage Scaling with Links for Power Optimization of Interconnection Networks. Proc. 9th Int. Symp. on High-Performance Computer Architecture, p.91-102. [doi:10.1109/HPCA.2003.1183527]

Lu, C.L., 2009. Queuing Theory (2nd Ed.). Beijing University of Posts and Telecommunications Press, Beijing, China, p.32-41 (in Chinese).

Marculescu, R., Bogdan, P., 2009. The chip is the network: toward a science of network-on-chip design. *Found. Trends Electron. Des. Automat.*, **2**(4):371-461. [doi:10.1561/1000000011]

Marculescu, R., Ogras, U.Y., Peh, L.S., Jerger, N.E., Hoskote, Y., 2009. Outstanding research problems in NoC design: system, microarchitecture, and circuit perspectives. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst.*, **28**(1):3-21. [doi:10.1109/TCAD.2008.2010691]

Moadeli, M., Shahrabi, A., Vanderbauwhede, W., 2007a. Analytical Modelling of Communication in the Rectangular Mesh NoC. Int. Conf. on Parallel and Distributed Systems, p.1-8. [doi:10.1109/ICPADS.2007.4447826]

Moadeli, M., Shahrabi, A., Vanderbauwhede, W., Ould-Khaoua, M., 2007b. An Analytical Performance Model for the Spidergon NoC. IEEE 21st Int. Conf. on Advanced Information Networking and Applications, p.1014-1021. [doi:10.1109/AINA.2007.31]

Moraveji, R., Moinzadeh, P., Sarbazi-Azad, H., 2008. A General Approach for Analytical Modeling of Irregular NoCs. IEEE Int. Symp. on Parallel and Distributed Processing with Applications, p.327-334. [doi:10.1109/ISPA.2008.109]

Moraveji, R., Moinzadeh, P., Sarbazi-Azad, H., 2009. A general mathematical performance model for wormhole-switched irregular networks. *Clust. Comput.*, **12**(3):285-297. [doi:10.1007/s10586-009-0084-0]

Ni, L.M., McKinley, P.K., 1993. A survey of wormhole routing techniques in direct networks. *Computer*, **26**(2):62-76. [doi:10.1109/2.191995]

Ogras, U.Y., Marculescu, R., 2007. Analytical Router Modeling for Networks-on-Chip Performance Analysis. Proc. Design, Automation and Test in Europe, p.1096-1101. [doi:10.1109/DATE.2007.364440]

Ould-Khaoua, M., Sarbazi-Azad, H., 2001. An analytical model of adaptive wormhole routing in hypercubes in the presence of hot spot traffic. *IEEE Trans. Parall. Distr. Syst.*, **12**(3):283-292. [doi:10.1109/71.914770]

Sarbazi-Azad, H., 2003. A mathematical model of deterministic wormhole routing in hypercube multicomputers using virtual channels. *Appl. Math. Model.*, **27**(12):943-953. [doi:10.1016/S0307-904X(03)00135-5]

Sarbazi-Azad, H., Ould-Khaoua, M., Mackenzie, L.M., 2001. An accurate analytical model of adaptive wormhole routing in *k*-ary *n*-cubes interconnection networks. *Perform. Eval.*, **43**(2-3):165-179. [doi:10.1016/S0166-5316(00)00049-3]