IPSJ Transactions on Advanced Computing Systems Vol. 2 No. 3 58–70 (Sep. 2009)

Regular Paper

A Quantitative Evaluation Methodology of Interconnection Networks

Takashi Yokota,[†1] Kanemitsu Ootsu[†1] and Takanobu Baba[†1]

This paper addresses the quantitative evaluation methodology of interconnection networks. In the conventional evaluation method, performance curves are drawn by a series of simulation runs, and network methods are discussed by comparing the shape of performance curves. We present the Ramp Load Method that does not require repetitive simulation runs and produces continuous performance curves. Based on the continuous curves, we give a formal definition of critical load ratio. Furthermore, we introduce a feature quantity to represent both throughput and average latency, and propose a new measure called Network Performance Measure. Through detailed evaluation and some application examples, the effectiveness of the proposed evaluation methodology is confirmed.

1. Introduction

An interconnection network is inevitable in parallel computers. As well as connecting computation nodes, an interconnection network dominates one of the most important parts in performance, i.e., communication. Thus, increasing demands on massively parallel supercomputers drive us to active and continuous discussions on interconnection networks.

This paper addresses an evaluation methodology of interconnection networks. Typically, when we think of a new method, we first implement the method in a simulator and evaluate its performance. Then, we discuss the effectiveness of the new method by comparing it with some existing methods. Through the comparison, we typically use two aspects of performance: throughput and latency, and we need both high throughput and low latency. In many cases, we draw performance curves and compare their shapes to discuss the effectiveness.

†1 Graduate School of Engineering, Utsunomiya University

One of the major drawbacks of this evaluation method is the difficulty in quantitative comparison. A performance curve does not show a quantitative measure but characteristics. For example, if the new method offers a lower latency but slightly poorer throughput than those of existing methods, it is difficult to determine which one is better. Furthermore, we have great difficulty in comparing many methods at the same time. This suggests to us the necessity of a quantitative representation of network features.

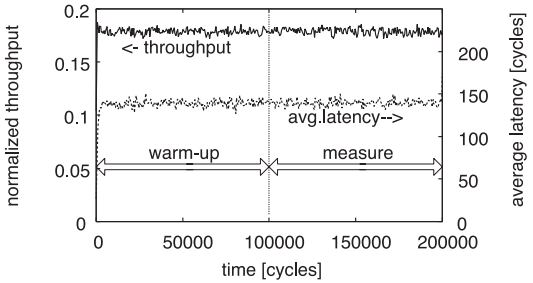
In this paper, we propose a quantitative methodology of network performance evaluation. The rest of this paper is organized as follows. We first review the conventional method and present fundamental problems in Section 2. Section 3 shows requirements, and Sections 4 and 5 propose an evaluation method and two quantitative measures, respectively. Section 6 shows detailed evaluation of the methodology and Section 7 shows practical application examples. Finally, Section 8 summarizes this paper.

2. Conventional Evaluation Method

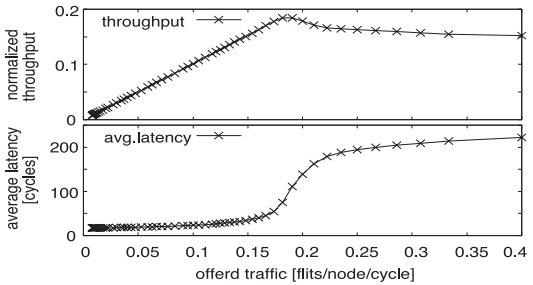
Network performance is discussed by means of throughput and latency. Both of them are deeply influenced by traffic pattern and traffic load. In most evaluations, only traffic load is changed as a parameter in a certain traffic pattern. Sometimes, random traffic, in which each node selects packet destinations randomly, is used as the traffic pattern.

Throughput and latency are implicitly measured as average values in a long-range of stable situation. Textbooks [1),2)] suggest drawing a performance curve by connecting a number of dots, where each dot corresponds to a simulation result of an individual parameter. In each simulation run, the parameter, i.e., offered traffic, is fixed. Furthermore, we need sufficient warm-up cycles before measurement so that a transient state is excluded. This evaluation method is widely accepted and many researches use the method to date [3)–6)].

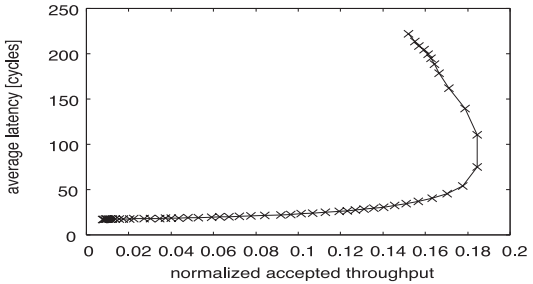
Figure 1 shows an example from Ref. 7). In each simulation run (Fig. 1 (a)), the traffic load is fixed to a certain level and we measure throughput as the number of received packets and average latency in the 'measurement' period in the figure. The measured values are plotted as a dot in the performance curve (Fig. 1 (b)). Figure 1 (c) is an alternative representation of performance, each



(a) A simulation run with a fixed parameter.



(b) Example of performance curves.



(c) Example of performance curve in BNF.

Fig. 1 A simulation result example.

dot represents the paired value of the measured throughput (x-axis) and average latency (y-axis). Reference 2) calls this form BNF. In any case, we need a large number of simulation runs to draw a smooth curve. Figure 1 has fifty points in each curve.

A major difficulty in the conventional evaluation method is the leap in the performance curve. The performance values drastically change in a narrow range of parameter. In practical terms, we need a rough performance plot prior drawing a smooth one.

Even after drawing a smooth performance curve, we have a further difficulty in evaluation. Since performance is represented in a two-dimensional graph, we should compare some graph curves when discussing networks methods. But, we do not have any established way of comparing two or more performance curves. For example, suppose that there are two network methods whose characteristics are different: one shows high throughput and high latency, and the other one shows low throughput and low latency. How can we discuss which one is superior? Thus, in many cases, it is very difficult to say which one is the best and how much is the benefit.

3. Requirements for Quantitative Measures

In general, interconnection networks are evaluated from many aspects of features; for example, random traffic performance, practical application performance, transient behavior, hardware cost, fault-tolerance, power consumption, and so on. In this paper, we focus on random traffic performance, which is widely used for performance comparison. In this section, we clarify what aspects are required for our purpose, quantitative evaluation.

In random traffic performance comparison, major points of interests are throughput and average latency. Throughput is measured by counting the number of received packets. Average latency is measured by counting the number of clock cycles for each packet to reach its destination.

We empirically know that, once the traffic load exceeds a certain threshold, the network becomes heavily congested. Thus, it is important to present how much offered traffic load the network tolerates. Thus, we reached the first criteria: *critical traffic load*.

Obviously, the critical traffic load is not enough to represent required network features by itself, because we have to discuss both performance and average latency. Thus, the second requirement should include both of them. Our major objective is a quantitative comparison of network methods. Some methods

may achieve high throughput but high latency, while other ones show moderate throughput with low latency. If we can represent the benefit of each method quantitatively, the measure is suitable for our purpose. Thus, the second criteria is *feature quantity* that represents both throughput and average latency.

4. Ramp Load Method

Before we discuss quantitative measures, we need a fundamental review of evaluation methodology.

There are many papers that mention critical load in networks [8)–13)]. But, their definitions are unclear. For example, many of them define critical load as "traffic load where average latency (throughput) is drastically increased (degraded)". Such a definition does not match our purpose and we need some formal definition based on some mathematical foundation.

Basically, difficulty in determining critical load comes from the repetitive method shown in Section 2. Network performance is measured by means of repetitive simulation runs, and in spite of great efforts, results do not show enough accuracy on critical load, because they are roughly quantized at each repetition.

Our idea is a continuous change of traffic load to solve the problem. We propose the *Ramp Load Method*. In this method, traffic load is gradually increased with the simulation time, as Eq. (1) shows.

$$r(t) = t/sl + r_0, \tag{1}$$

where t is the simulation time, sl is *slew rate* that shows the inverse of the gradient of the traffic load, and r_0 is the initial value. Typically, we use $r_0 = 0$. With sufficiently large slew rate, the evaluation method approximates the performance curve sufficiently.

This method enables us to evaluate any method with just *one* simulation run. And, obviously, no warm-up cycle is required, if the simulation is started with zero load ($r_0 = 0$). We should take care about the slew rate, because the network may react with some time-constant. We discuss the appropriate slew rate value, and the accuracy of results in Section 6.

The Ramp Load Method offers a large benefit in evaluating large-scale systems. The method can reduce simulation time drastically. As Refs. 14), 15) report that step response time is $O(N)$ in $N \times N$ two-dimensional torus network, warm-up cycles proportional to N are required and simulation time for one clock-cycle is also proportional to N. Thus, roughly N^2 times of simulation time is required in a repetitive evaluation method. The Ramp Load Method can reduce most of the warm-up cycles, thus, it can accelerate evaluation.

5. Quantitative Measures for Network Performance

The Ramp Load Method does not only offer a smooth performance curve in one simulation run, but also drives us to a mathematical discussion on network performance evaluation by approximating a continuous system. This section proposes two quantitative measures that are derived from the continuous nature of the Ramp Load Method.

5.1 Critical Load Ratio

Section 3 suggests the basic idea of critical load ratio, and Section 4 approximates a continuous system to make mathematical discussion on quantitative evaluation. In this section, we discuss its formal definition.

The critical load ratio should be defined mathematically from the simulation results by the Ramp Load Method, so that the critical load ratio is uniquely determined for a given network method. Furthermore, it should also be appropriate in engineering for practical use.

Firstly, we should discuss the general characteristics of network performance. Throughput is simply proportional to the offered traffic, when the traffic is not heavy. Roughly speaking, frequency of packet conflicts is proportional to packet density in the network, and the traffic load offers the density. Packet latency is increased by the conflicts, thus, the latency is increased as the traffic load increases.

The behavior of a network drastically changes at a certain condition because of the non-linear nature of the network. Research results, shown in Refs. 16), 17), show that the growing speed of a congestion is $O(N)$ and the reduction speed is $O(N^2)$ in two-dimensional torus networks. This nature causes a chained response of growing congestion when the packet density meets a certain condition [18),19)]. The critical load ratio should show the boundary condition between congested and uncongested (or free-running) situations.

We define the critical load ratio as the *inflection point* of the throughput

curve [14),15),20)]. Only throughput is proportional to the offered traffic in uncongested situations. Other possible measures are not proportional; for example, average latency, number of in-flight packets in the network, and entropy [16),17),19)].

Throughput can be represented by a function of the offered traffic r;

$$\textit{throughput} = f(r). \tag{2}$$

The inflection point is determined by the gradient of the throughput curve, and the gradient is given by differentiation with respect to the traffic load, i.e.,

$$\textit{gradient} = g(r) = \frac{df(r)}{dr} = f'(r). \tag{3}$$

As discussed above, the gradient is constant whenever the throughput is proportional to the offered traffic, and the gradient begins to decrease at the inflection point.

At this stage, we discuss what the inflection point means. Up to the inflection point, normalized throughput should be the same as the offered traffic. Generated packets are smoothly injected into the network. However, after the inflection point, not so many packets are delivered as generated ones. This means that the injection of new packets is frequently blocked, and it also means that the network is crowded so that it reaches the critical condition of serious congestion. Thus, our definition of the critical load ratio represents the upper-bound of the traffic load where the network is not congested.

Our definition of the critical load ratio is based on differentiation. In general, differentiation is sensitive to noise contents. Thus, we should further discuss the practical definition of the critical load ratio for engineering purposes.

Figure 2 shows a typical example of a throughput curve. The graph also shows the gradient of the throughput curve. As this figure shows, the gradient has a considerable level of fluctuations. Thus, it is not practical to identify the inflection point, at which the gradient begins to decrease.

We can observe that the gradient curve rapidly decreases to negative value when the offered traffic is around 0.15. We define the *practical* critical load ratio by using the falling edge of the gradient curve. In this paper, we use the threshold factor value $\theta_g = 0.50$ [⋆1].

Thus, assuming that the gradient is $g(r)$ and $g(0) = g_0$, the practical definition is

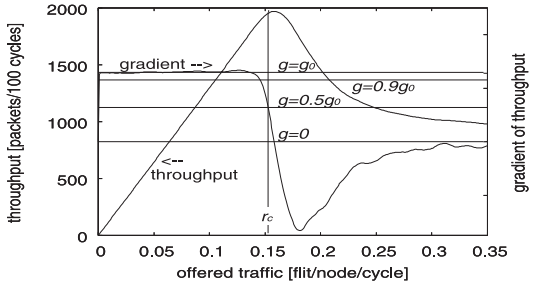


Fig. 2 Definition of the critical load ratio.

$$g(r_c) = \theta_g g_0. \tag{4}$$

The initial value g_0 is easily given by

$$g_0 = N^2 t_w / l_p, \tag{5}$$

where t_w is time window length, l_p is packet length, and $N \times N$ two-dimensional network is assumed.

5.2 Network Performance Measure

The critical load ratio shows the critical point of the offered traffic at which the network is at the boundary point between free-running and congested states. But, this quantitative measure does not show the performance feature of the network by itself. As described in Section 3, we need a feature quantity that can represent both throughput and latency. It should be a quantitative measure, instead of two-dimensional graph curves.

We discuss the figure of merit of an interconnection network method to reach the feature quantity. Suppose a method A performs twice as much in throughput, the method is worth twice. Similarly, suppose another method B reduces the average latency by half, the method is also worth twice. Thus, we introduce a feature quantity as throughput divided by average latency.

Each of the factors, throughput and average latency, is represented by a function of r. Thus, the feature quantity fq is also a function of r;

$$fq(r) = N_r(r)/l_a(r), \tag{6}$$

⋆1 In some of our prior papers, $\theta_g = 0.90$ is used. We use 0.50 for more stable results than those of 0.90.

where N_r shows the throughput as the number of received packets, and l_a is average latency.

In general, we cannot control or predict the offered traffic in practical systems, and an interconnection network should work properly over a certain range of traffic load. The average of the feature quantity over the assumed range of the traffic load is meaningful. Thus, we introduce a quantitative measure by integration of the feature quantity, and we call the new measure *Network Performance Measure* (NPM);

$$NPM = \int_{r_{\min}}^{r_{\max}} \frac{N_r(r)}{l_a(r)} dr. \tag{7}$$

In this equation, $r_{\min}$ and $r_{\max}$ are minimal and maximal traffic load ratios, respectively, and they offer the integral interval.

In principle, the actual values of $r_{\min}$ and $r_{\max}$ cannot be determined uniquely, since they are dependent on the design policy of the system. If they are smaller than the critical load ratio, NPM summarizes the feature in free-running situations. On the other hand, if the system emphasizes heavy-traffic situations, $r_{\max}$ should be beyond the critical load ratio (r_c).

To make NPM a general measure of interconnection networks, we should further discuss the standard values of $r_{\min}$ and $r_{\max}$ as a guideline. Usually, $r_{\min}$ should be zero, because the traffic is sometimes sparse. For fair comparison of network methods, $r_{\max}$ should be independent of methods. We introduce theoretical critical load ratio R_c that is derived by analysis results. For example, Ref. 14) shows the theoretical critical load ratio is

$$R_c = 8/N, \tag{8}$$

where N is the network size (in $N \times N$ two-dimensional torus networks). Thus, $r_{\max} = R_c$ is appropriate for general use of NPM.

5.3 Validity of the Methodology

The two proposed measures are defined on evaluation results of throughput and average latency, and the measures represent two-dimensional graphs of throughput and average latency as two numerical values. In other words, the proposed methodology offers projections of the two-dimensional graphs to (one-dimensional) values, conceptually.

Although the original two-dimensional graphs have more information than the projected measures, the measures are still useful for quantitative comparison, while we have much difficulty in comparing two-dimensional graphs.

What we should keep in mind is the validity of the proposed measures. The proposed measures are not universal in evaluation of interconnection networks. We have so many aspects of evaluation, such as dynamic behavior, costs and power consumption, but the proposed measures do not cover a wide spectrum of *performance*, since the measures represent only throughput and average latency.

Furthermore, we should be aware of the application range of the measures. We cannot use the measures for comparing different sizes of networks. Different size networks have different gradients of throughput in non-congested situations (i.e., g_0 in Fig. 2). Thus, comparison of critical load ratio and NPM between two different size networks has no meaning.

The proposed measures possibly illustrate distinctive natures in simulation results. Critical load ratio shows peak performance (throughput) in many cases, and we can estimate the average latency characteristic in heavily congested situations. Section 7 shows some examples.

This paper uses a random traffic pattern so that we can discuss the essential features of the proposed measures. But, the proposed methodology, i.e., Ramp Load Method and two quantitative measures, does not depend on a random traffic pattern. Thus, in principle, the methodology is applicable to any traffic patterns, iff we can control offered traffic load in simulation.

The conventional repetitive evaluation method can approximate smooth performance curves, if we have many simulation runs. Thus, the conventional method can derive NPM with certain level of accuracy. But, it is difficult for the conventional method to derive the critical load ratio accurately when the network shows very steep performance curves and a large-scale network sometimes shows such steep behaviors [14),17)].

Some references such as Refs. 2), 21)–23) introduce probability models to represent *analytical* performance as a formula. These approaches are useful for comparison in terms of a formula, but wide abstraction buries detailed discussion on actual routing algorithms. Our proposed measures are in the opposite direction, i.e., the measures *project* simulation results to (one-dimensional) numerical values.

6. Evaluation

6.1 Evaluation Environment

We implemented the Ramp Load Method in our interconnection network simulator [14),15),20)]. Each node generates packets at the load ratio r that is given by Eq. (1). A packet consists of eight flits and a node generates at most one flit at every clock cycle. We further assume that flits of a packet are continuously generated. Thus, assuming that a packet consists of l_p flits and that the inter-packet gap is t_g on average, the load ratio is

$$r = l_p/(l_p + t_g). \tag{9}$$

In our simulator, packet generation follows the Poisson process. Each node starts to generate a packet stochastically when it is in an idle state. Once the node starts generating a packet, it continuously generates packet contents, one flit at each clock cycle, until it reaches the end of the fixed-length packet. Practically, each node generates a random number within the range [0:1), and if the random number is less than a certain fraction p, it starts generating of a new packet. The average inter-packet gap can be given by the sum of the geometric series;

$$t_g = (1 - p)/p. \tag{10}$$

The packet generation probability p is derived from Eqs. (9) and (10) as follows.

$$p = \frac{r}{l_p(1 - r) + r}. \tag{11}$$

In this paper, we use the following parameter values and settings. The packet length (l_p) is eight flits. The time window (t_w) is 100 cycles. Evaluation values are measured in every time window; throughput (the number of received packets), average and maximal latency, the number of in-flight packets, and entropy [16),17),19)]. A 32×32 ($N = 32$) two-dimensional torus network is used. Routing algorithms are dimension-order (do), simple adaptive routing (sa), and Cross-Line (cl). Details of the routing algorithms are described in Refs. 2), 24).

6.2 Sufficient Slew Rate

The first evaluation is about the appropriate value of the slew rate (sl) in Eq. (1). As we discussed in Section 4, a small sl probably shows inaccurate results, but, a large sl needs long simulation times.

We measured the critical load ratio for variations of sl; from 10,000 (10 K) to 10,000,000 (10 M) [cycles]. **Figure 3** shows the throughput and gradient curves of 100 K, 1 M, 2 M, and 5 M of sl.

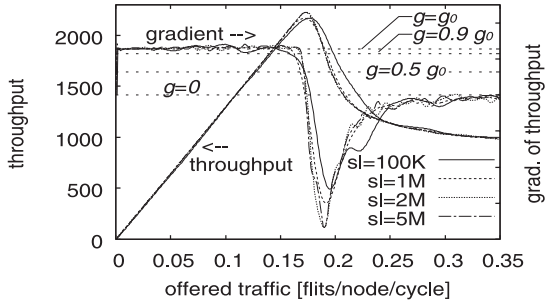


Fig. 3 Performance curves of various slew rate values.

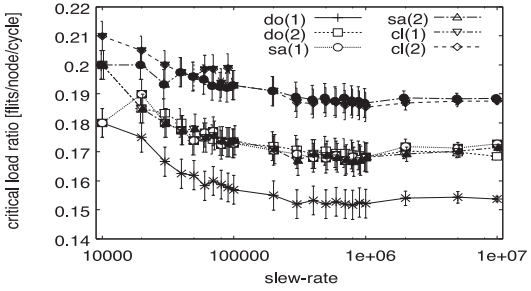
In Fig. 3, throughput curves of $sl = 2$M and 5M are almost overlapped. A small sl, e.g., $sl = 100$ K, shows a gradual change near the critical load ratio, while a large sl shows a steep change. This illustrates that a small sl does not follow the time-constant nature of the network response in throughput. This feature of sl reflects the gradient curve in Fig. 3.

Figure 4 shows the measured values of the critical load ratio (Fig. 4 (a)) and Network Performance Measure (Fig. 4 (b)). In this figure, do, sa, and cl mean dimension-order, simple adaptive, and Cross-Line routing algorithms, respectively, and (1) and (2) show variations of virtual channel assignment algorithms [⋆1]. Figure 4 (a) show maximum possible errors as vertical error-bars, as well as measured values. Both graphs show that the measured values decrease until sl is around 1 million, and that measured values are stable when sl exceeds 1 million. This shows that $sl = 1$M is sufficient for this paper's evaluation environment.

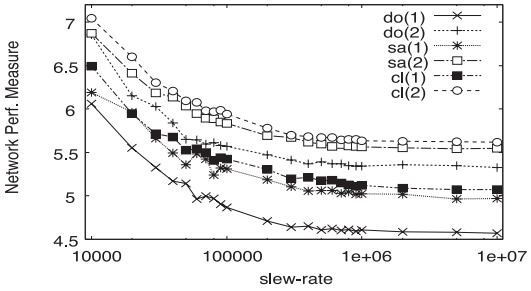
6.3 Fluctuation and Denoising

In many cases of evaluations by simulation, detailed behaviors in each router are determined stochastically; for example, packets are generated stochastically.

⋆1 This paper does not discuss routing algorithms. Thus, do, sa, cl and their variations are for reference.

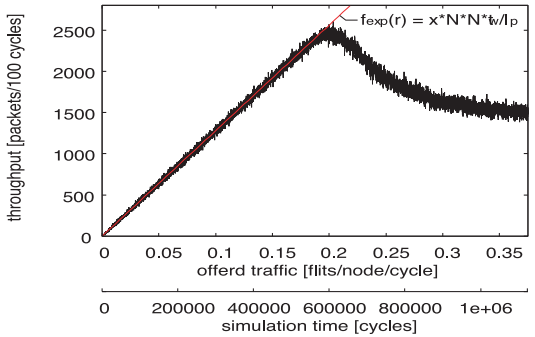


(a) Critical load ratio.

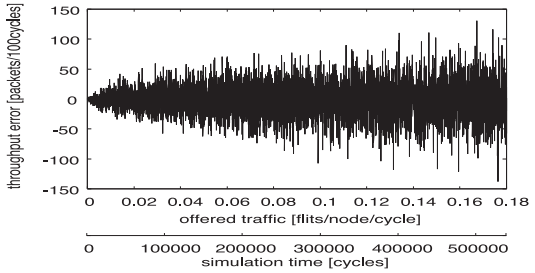


(b) Network Performance Measure.

Fig. 4 Slew rate and measured values.



(a) Raw results of simulation.



(b) Fluctuations.

Fig. 5 Fluctuations of throughput in raw simulation results.

In this paper, we use the Poisson process to generate new packets as described in Section 6.1, and we observe large fluctuations in measured values. These fluctuations cannot be ignored because they are related to the accuracy and reliability of evaluation results.

For example, **Fig. 5** (a) plots the raw data of throughput, where $sl = 3 \times 10^6$ [cycles]. This graph also includes the expected throughput curve, shown as $f_{exp}(r) = (N^2 t_w / l_p) r$ in the figure. Figure 5 (b) shows net fluctuations of the throughput, given by $f(r) - f_{exp}(r)$. These fluctuations are critical to measuring the critical load ratio, because the measure is based on differentiation of the throughput curve.

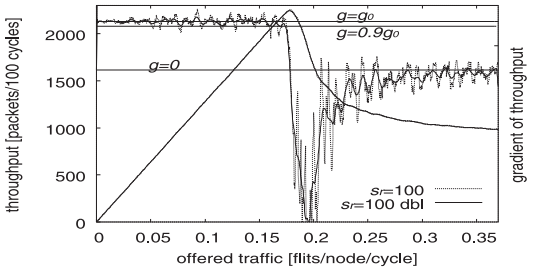
To reduce the fluctuation, we should apply a low-pass filter function to the sequence of measured values. We used moving average as the low-pass filter; the filter calculates the average value of $2s_r + 1$ samples that include before and after s_r consecutive samples. This moving average is equivalent to low-pass FIR (finite impulse response) filter. So, selecting proper s_r value is the next problem.

The average of $|f(r) - f_{exp}(r)|$ is about $\xi = 23.43$ and its standard deviation is $\sigma = 29.66$ $(0 < r < 0.18)$. We can estimate the maximal errors of the gradient in the filtering period $2s_r t_w / sl$;

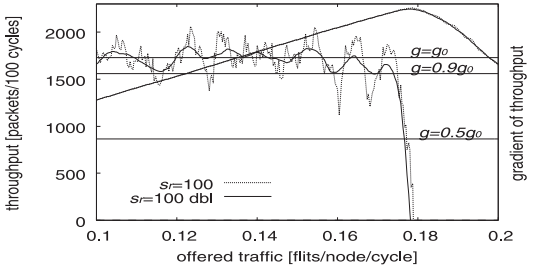
$$g_{err} = \frac{\xi}{s_r t_w / sl}. \tag{12}$$

In Fig. 5 case, by applying $sl = 3 \times 10^6$ [cycles] and $t_w = 100$ [cycles] to Eq. (12), the maximal error is derived as $g_{err} = 234300/s_r$. The expected gradient in free-running situations is given by Eq. (5). In this case, $N = 32$ and $l_p = 8$, thus, $g_0 = 12800$. If the maximal gradient error is required within 10 percent of g_0, i.e., $g_{err} < 0.1 g_0$, we can obtain the condition $s_r > 183$.

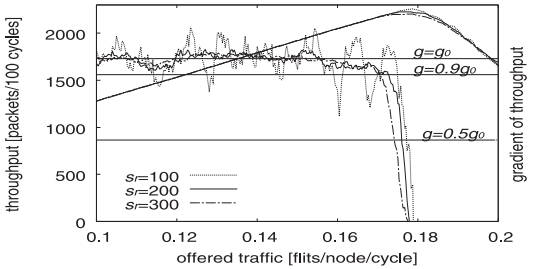
The filtering effect can be enhanced by applying the filter twice. **Figure 6** shows the effects of some filters. The throughput and its gradient curves are drawn in the figure. Figure 6 (a) shows the effect of the double filtering. In this figure, s_r shows the size of the moving average and dbl shows double filtering. Figure 6 (b) shows a magnification of Fig. 6 (a) at around the falling edge of the gradient curve. Figure 6 (c) shows filtering effects with respect to s_r.



(a) Single and multi-stage filter.



(b) Magnification of (a).



(c) Filters with various range.

Fig. 6 Effects of filters.

As discussed above, the fluctuations of filtered data of $s_r = 100$ are too large to determine the proper critical load ratio. But, the double filtering s_r=100 dbl in Figs. 6 (a) and (b) reduces the fluctuation within 10 percents of g_0. Larger s_r shows smaller fluctuations as shown in Fig. 6 (c), but, the falling edge of the gradient curve is inaccurate.

6.4 Accuracy of Result

There are two aspects that affect the accuracy of results; one is the methodological error and the other one is the statistical error.

The methodological error comes from the evaluation method. In our method, the range of moving average (s_r) is a major factor. As discussed in the previous subsection and observed in Fig. 6 (c), the wide range of moving average (s_r) stabilizes the performance curve, but, it affects the accuracy of results.

A critical load ratio is defined by a falling edge of gradient curve of throughput, and a large s_r value shifts the falling edge as shown in Fig. 6 (c). Thus, a large s_r shows a small critical load ratio value. As a rough approximation, the maximal error caused by s_r is represented by $s_r \cdot t_w/sl$. The vertical bar at each measured point in Fig. 4 (a) shows the maximal error.

Statistical errors are determined empirically. We measured the critical load ratio and NPM for 100 times for each of six routing algorithms.

Figure 7 shows the distribution of measured values. The horizontal axis shows

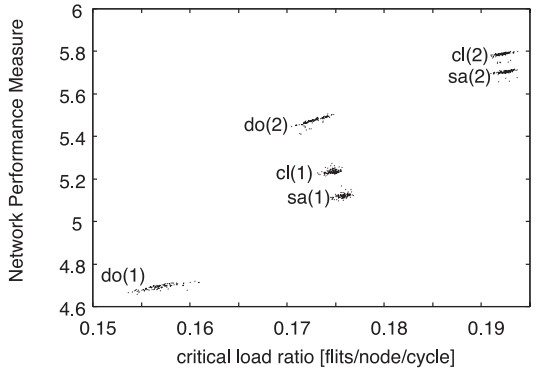


Fig. 7 Distribution of measured r_c and NPM values.

the critical load ratio and the vertical axis shows NPM. Each dot in this graph corresponds to one evaluation result.

The measured values scatter in a relatively small range, when they follow the same algorithm. Each algorithm is distinctive in Fig. 7. **Table 1** shows the average and standard deviation of critical load ratio and NPM.

6.5 Appropriateness of the Critical Load Ratio

In Section 5.1, we discussed the meaning of the critical load ratio that is defined as the inflection point of the throughput curve. This subsection probes the appropriateness of the definition.

Figure 8 shows the performance curves of throughput, average latency, maximal latency, and the number of in-flight packets in the same graph. These are the cl(2) results. The critical load ratio is illustrated as a vertical bar near the center of the graph.

Two curves of average latency and the number of in-flight packets are steeply rising near the critical load. The maximal latency curve is rising at the critical load ratio. These observations show the appropriateness of the definition of the critical load ratio; before the critical load ratio, the network is not congested, and the ratio shows a boundary between free-running and congested situations.

Table 1 Distribution of measured values.

algo-	critical load ratio		NPM	
rithm	avg.	σ	avg.	σ
do(1)	0.1569	0.00148	4.694	0.0125
do(2)	0.1727	0.00099	5.475	0.0180
sa(1)	0.1758	0.00051	5.122	0.0103
sa(2)	0.1925	0.00062	5.704	0.0094
cl(1)	0.1746	0.00053	5.237	0.0102
cl(2)	0.1921	0.00055	5.787	0.0106

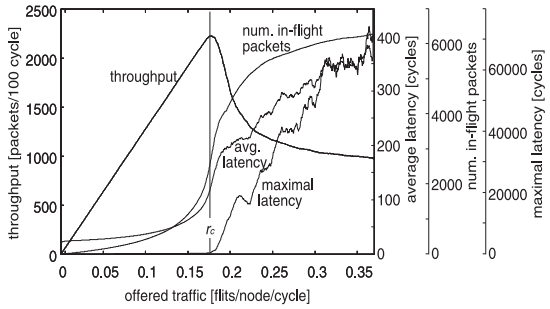


Fig. 8 Critical load ratio and various network metrics.

7. Applications of the Methodology

This section shows some practical application of the proposed evaluation methodology to demonstrate the effectiveness.

7.1 Buffer Size Study

A router has a packet buffer for each of the virtual channels in each input port. The size of the buffer strongly affects the network performance. A large buffer tolerates congestion, but once it falls into a congested situation, since each buffer has a lot of packet data, packet latency is largely increased. Thus, there is a trade-off between buffer size and performance. We discuss the trade-off point by applying the proposed quantitative methodology.

Figure 9 shows the critical load ratio and NPM values (in y-axis) for various buffer sizes (in x-axis). In this evaluation $r_{\max} = R_c$ (Eq. (8)), and $sl = 3 \times 10^6$ [cycles]. The critical load ratio monotonically increases with the buffer size. But, NPM saturates near $b = 10$ [flits], and gradually decreases.

We discuss what the result means. **Figure 10** shows performance curves of $b = 16$ and $b = 96$ cases, where b represents the buffer size in a flit unit. The critical load ratio values are illustrated as vertical lines in the figure. In both cases, throughput is around 2,000 [packets/100 cycle]. But, the average latency

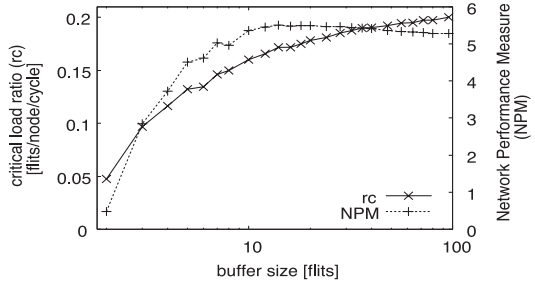


Fig. 9 Critical load ratio and NPM for buffer sizes.

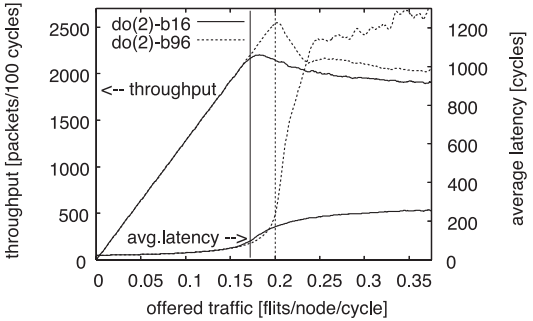


Fig. 10 Performance curves of $b = 16$ and $b = 96$.

of $b = 96$ is much larger than that of $b = 16$. This shows why NPM of $b = 16$ is better.

As illustrated above, we can guess the performance feature by comparing the paired values of the critical load ratio and NPM. A high critical load ratio means high peak throughput, and actually $b = 96$ has high peak throughput in Fig. 10. If the two methods have comparable NPMs whereas their critical load ratios differ, we can guess one of them has low latency with a low critical load ratio. Our proposed quantitative measures, critical load ratio and NPM, properly illustrate the feature quantitatively. On the other hand, the performance curves (illustrated in Fig. 10) do not offer such quantitative discussions.

7.2 Routing Algorithm Study

The second example is to discuss routing algorithms. First we look at the conventional evaluation method. **Figure 11** shows the performance curves of throughput and average latency. In this figure, do, sa and cl represent the same routing algorithms as those given in Section 6.1, and an additional parameter th shows an allowance level for a 'congested' situation; a routing algorithm senses that a packet buffer is 'congested' if the buffer has less room than th [flits] for incoming packets, th0 means that only a full situation is detected.

From Fig. 11, we can recognize that (1) do(2) has a low peak throughput, but its average latency is also low, (2) sa(2) has a slightly better peak throughput than cl(2)-th0, but its average latency seems comparable to cl(2)-th0, (3) cl(2)-th0 is more preferable than cl(2)-th8 in both throughput and average latency.

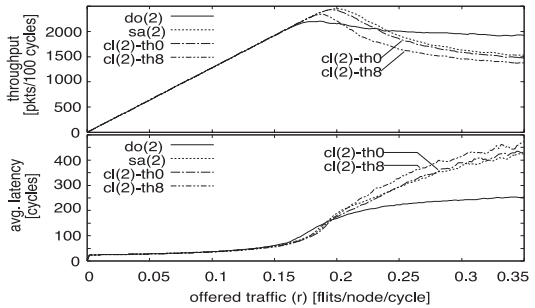


Fig. 11 Performance curves example.

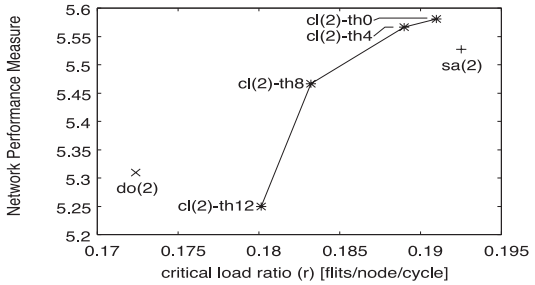


Fig. 12 Map of critical load ratio and NPM.

Figure 12 illustrates a performance *map* of the algorithms. Each dot represents the critical load ratio and NPM of its corresponding routing algorithm. The figure includes additional results (sl(2)-th4 and -th12) that are omitted in Fig. 11. We can easily recognize characteristics of routing algorithms in Fig. 12. We can obtain similar results from Fig. 11, however, the results are not straightforward and we need careful consideration. Furthermore, we cannot discuss many performance curves at one time in the conventional way, while the proposed methodology can deal with many cases and can show them clearly. sa(2) is the best in a critical load ratio and it also scores well in NPM. cl(2)-th0 is the best in cl(2) variants, cl(2)-th0 has better NPM than sa(2), but the critical load ratio is slightly lower.

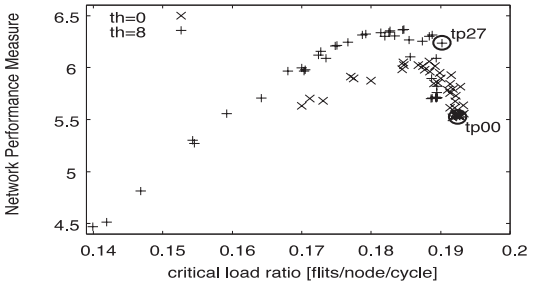


Fig. 13 Paired values of critical load ratio and Network Performance Measure.

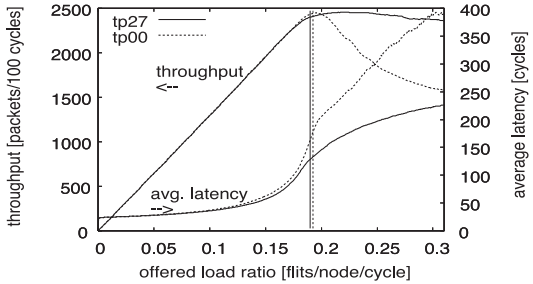


Fig. 14 Performance curves of tp00 and tp27.

7.3 Screening Test of Routing Algorithms

The third application example is the screening test of possible algorithms. In Ref. 20), we discussed possible combinations of output selection functions and throttling, and we evaluated 82 combinations. It was difficult to predict network performance in a proper way. Thus, we should select possible methods by some screening method. We applied the proposed methodology to screen the candidate methods.

Figure 13 shows the map of paired values of the critical load ratio (x-axis) and NPM (y-axis). In this figure, th is the same allowance parameter as in Section 7.2. Each distinct point corresponds to a candidate method that has a distinct evaluation function. tp00 shows the original Cross-Line algorithm and does not employ throttling functions, and tpNN ($NN \neq 00$) employs a throttling function with distinct evaluation function of global congestion information derived from the Cross-Line mechanism [20].

This graph provides an overview of characteristics of the candidate methods. Needless to say, both high NPM and high critical load ratio are preferable. Thus, we can select the appropriate one that is nearest to the upper-right corner of the graph.

We can find that a certain method tp27 is a promising candidate. For example in Fig. 13, we discuss the appropriateness of tp27 by comparing it with tp00. Both methods are comparable in the critical load ratio, and tp27 has a far better NPM than that of tp00.

Figure 14 shows their performance curves. Their critical load ratios are very close to each other, and their performances are comparable when the offered traffic is less than the critical load ratio. But, they have a distinct behavior when the offered traffic exceeds the critical load ratio. The throughput of tp00 is steeply decreased while tp27 keeps the throughput at a high level. Both methods increase the average latency, but tp00 shows a steep increment. Their difference in NPM reveals these facts.

8. Conclusions

In the conventional evaluation method, we need many simulation runs to draw a smooth performance curve, and we discuss network methods by comparing the shape of performance curves. The major problem was that we had no established way of comparing network methods properly and quantitatively.

This paper presented a new evaluation methodology that includes a new simulation method and two quantitative measures.

The new simulation method, named Ramp Load Method (RLM), is the method that gradually increases traffic ratio with the simulation time. The method does not require repetitive simulation runs.

The newly proposed two quantitative measures are critical load ratio and Network Performance Measure. We gave a formal definition of the critical load ratio by using a continuous nature of RLM results after considering the general characteristics of interconnection networks. Network Performance Measure (NPM) represents both throughput and average latency.

Appropriateness of the proposed methodology is shown via detailed evaluation.

Critical load ratio and NPM allow us quantitative evaluation and comparison of network methods.

Acknowledgments This research was supported in part by Grant-in-Aid for Scientific Research ((C)21500050, (C)21500049, (C)20500047) of Japan Society for the Promotion of Science (JSPS), and by Eminent Research Selected at Utsunomiya University.

References

1) Duato, J., Yalamanchili, S. and Ni, L.: *Interconnection Networks: An Engineering Approach*, Morgan Kaufmann Pub. (2003).
2) Dally, W.J. and Towles, B.: *Principles and Practices of Interconnection Networks*, Morgan Kaufmann Pub. (2004).
3) Park, D., Eachempati, S., Das, R., Mishra, A.K., Xie, Y., Vijaykrishnan, N. and Das, C.R.: MIRA: A Multi-layered On-Chip Interconnect Router Architecture, *Proc. 2008 ISCA*, pp.251–261 (2008).
4) Ferrer, J.-L., Baydal, E., Robles, A., López, P. and Duato, J.: On the Influence of the Packet Marking and Injection Control Schemes in Congestion Management for MINs, *Lecture Notes in Computer Science*, No.5168, pp.930–939 (2008).
5) Yoshinaga, T., Murakami, H. and Koibuchi, M.: Latency Reduction Utilizing Dynamic Communication Prediction in 2-D Tori, *IPSJ Trans. Advanced Computing Systems*, Vol.1, No.1, pp.28–39 (2008).
6) Koibuchi, M., Yoshinaga, T., Murakami, H., Matsutani, H. and Amano, H.: A Low-Latency Network-on-chip Using Predictive Routers, *IPSJ Trans. Advanced Computing Systems*, Vol.1, No.2, pp.59–69 (2008).
7) Yokota, T., Nishitani, M., Ootsu, K., Furukawa, F. and Baba, T.: Cross-Line — A Globally Adaptive Routing Method of Interconnection Network, *Lecture Note in Computer Science*, No.4759, pp.191–198 (2008).
8) Ohira, T. and Sawatari, R.: Phase Transition in a Computer Network Traffic Model, *Physical Review E*, Vol.58, No.1, pp.193–195 (1998).
9) Woolf, M., Arrowsmith, D.K., Mondragón-C, R.J. and Pitts, J.M.: Optimization and Phase Transitions in a Chaotic Model of Data Traffic, *Physical Review E*, Vol.66, p.046106 (2002).
10) de Moura, A.P.: Fermi-Dirac Statistics and Traffic in Complex Networks, *Physical Review E*, Vol.71, p.066114 (2005).
11) Echnique, P., Gómez-Gardeñes, J. and Moreno, Y.: Dynamics of Jamming Transitions in Complex Networks, *Europhysics Letters*, Vol.71, No.2, pp.325–331 (2005).
12) Lawniczak, A.T. and Tang, X.: Network Traffic Behavior near Phase Transition Point, *European Physical Journal B*, Vol.50, pp.231–236 (2006).
13) Lawniczak, A.T. and Tang, X.: Packet Traffic Dynamics near Onset of Congestion in Data Communication Network Model, *ACTA Physica Polonica B*, Vol.37, No.5, pp.1579–1604 (2006).
14) Yokota, T., Ootsu, K. and Baba, T.: Preliminary Discussions on Scaling Effects of Interconnection Networks, *IPSJ SIG Technical Report*, Vol.2008, No.75, pp.91–96 (2008).
15) Yokota, T., Ootsu, K. and Baba, T.: Are Uniform Networks Scalable?, *Proc. 9th International Conference on Parallel and Distributed Computing, Applications and Technologies* (*PDCAT 2008*), pp.137–140 (2008).
16) Yokota, T., Ootsu, K., Furukawa, F. and Baba, T.: A Cellular Automata Approach for Understanding Congestion in Interconnection Networks, *IPSJ Trans. Advanced Computing Systems*, Vol.47, No.SIG7 (ACS 14), pp.21–42 (2006).
17) Yokota, T., Ootsu, K., Furukawa, F. and Baba, T.: Phase Transition Phenomena in Interconnection Networks of Massively Parallel Computers, *Journal of Physical Society of Japan*, Vol.75, No.7, p.074801 (2006).
18) Yokota, T., Ootsu, K., Furukawa, F. and Baba, T.: Analysis of Intermittent Congestion in Large-Scale Interconnection Networks, *IPSJ SIG Technical Report*, Vol.2006, No.88, pp.91–96 (2006).
19) Yokota, T., Ootsu, K., Furukawa, F. and Baba, T.: Entropy Throttling for Maximizing Packet Mobility in Interconnection Networks, *IPSJ Trans. Advanced Computing Systems*, Vol.47, No.SIG12 (ACS 15), pp.1–11 (2006).
20) Yokota, T., Ootsu, K. and Baba, T.: Adaptation Level and Injection Control in a Quasi-Globally Adaptive Routing, *Proc. 20th IASTED International Conference on Parallel and Distributed Computing and Systems* (*PDCS 2008*), pp.112–117 (2008).
21) Dally, W.J.: Performance Analysis of k-ary n-cube Interconnection Networks, *IEEE Trans. Comput.*, Vol.39, No.6, pp.775–785 (1990).
22) Agarwal, A.: Limits on Interconnection Network Performance, *IEEE Trans. Parallel and Distributed Systems*, Vol.2, No.4, pp.398–412 (1991).
23) Ould-Khaoua, M.: A Performance Model for Duato's Fully Adaptive Routing Algorithm in k-Ary n-Cubes, *IEEE Trans. Comput.*, Vol.48, No.12, pp.1297–1304 (1999).
24) Yokota, T., Nishitani, M., Ootsu, K., Furukawa, F. and Baba, T.: Cross-Line: A Novel Routing Algorithm That Uses Global Information, *IPSJ Trans. Advanced Computing Systems*, Vol.46, No.SIG16 (ACS 12), pp.28–42 (2005).

(Received January 27, 2009)
(Accepted May 28, 2009)

Takashi Yokota received his B.E., M.E. and Ph.D. degrees from Keio University in 1983, 1985 and 1997, respectively. He joined Mitsubishi Electric Corp. in 1985, and he was engaged in several research projects in special-purpose, massively parallel and industrial computers. He was engaged in research and development of a massively parallel computer RWC–1 at Real World Computing Partnership as a senior researcher from 1993 to 1997. From 2001, he is associate professor at Utsunomiya University. His research interests include computer architecture, parallel processing, network architecture and design automation. He is a member of IPSJ, IEICE and IEEE Computer Society.

Kanemitsu Ootsu received his B.S. and M.S. degrees from the University of Tokyo in 1993 and 1995 respectively, and later he obtained his Ph.D. in Information Science and Technology from the University of Tokyo in Japan. From 1997 to 2009, he is a research associate and then an assistant professor at Utsunomiya University. Since 2009, he is an associate professor at Utsunomiya University. His research interests are in high-performance computer architecture, multi-core multithread processor architecture, binary translation, and run-time optimization. He is a member of IPSJ and ISCIE.

Takanobu Baba received the B.E., M.Eng., and Dr.Eng. degrees from Kyoto University in 1970, 1972, and 1978, respectively. He is a professor of Information Systems Science at Utsunomiya University. In 1982, he spent one year leave as a Visiting Professor at University of Maryland. His interests include computer architecture and parallel processing. He is author of the books, "Microprogrammable Parallel Computer" (The MIT Press, 1987), and "Computer Architecture, 2nd ed." (Ohmsha, Japan, 2000). He was elected an IPSJ Fellow in 2006 and an IEICE Fellow in 2008, respectively. He is a member of IPSJ, IEICE and IEEE Computer Society.