PAPER *Special Section on Parallel and Distributed Computing and Networking*

Enhancing Entropy Throttling: New Classes of Injection Control in Interconnection Networks

Takashi YOKOTA[†a)], **Kanemitsu OOTSU**[†], ***and* Takeshi OHKAWA**[†], ***Members***

SUMMARY State-of-the-art parallel computers, which are growing in parallelism, require a lot of things in their interconnection networks. Although wide spectrum of efforts in research and development for effective and practical interconnection networks are reported, the problem is still open. One of the largest issues is congestion control that intends to maximize the network performance in terms of throughput and latency. Throttling, or injection limitation, is one of the center ideas of congestion control. We have proposed a new class of throttling method, Entropy Throttling, whose foundation is entropy concept of packets. The throttling method is successful in part, however, its potentials are not sufficiently discussed. This paper aims at exploiting capabilities of the Entropy Throttling method via comprehensive evaluation. Major contributions of this paper are to introduce two ideas of hysteresis function and guard time and also to clarify wide performance characteristics in steady and unsteady communication situations. By introducing the new ideas, we extend the Entropy throttling method. The extended methods improve communication performance at most 3.17 times in the best case and 1.47 times in average compared with non-throttling cases in collective communication, while the method can sustain steady communication performance.

key words: *parallel computers, interconnection networks, congestion control, throttling*

1. Introduction

Interconnection is one of the key technologies especially in massively parallel computers. Interconnection network methodologies have been actively discussed [1], [2], where topology and routing algorithms are their major issues. Topologies offer graph-theoretical but important parameters i.e., degree, diameter, and bisection bandwidth. Thus, topology discussions include a wide spectrum of issues in performance, implementation, fault-tolerance, and economy. Routing algorithms offer how each message packet can reach its destination node in a certain topology of network. Routing algorithms are classified into two major categories: deterministic routing offers fundamental usage of the corresponding topology and non-deterministic (adaptive) routing intends to enhance the performance.

In spite of even enhanced topology and routing algorithms, network performance is weak in congested situations. We empirically know that once heavy congestion arises, the network performance is drastically degraded and, to make things worse, the congested situation sustains for a long time in large-scale systems. Thus, mitigating heavy congestion is of importance in order to exploit full performance of the network.

The center idea of mitigation is flow control. Although the conventional flow control methods, which include transfer principles (store-and-forward, wormhole [3], and virtual cut-through [4]) and virtual channels [5], can achieve considerable improvement in network characteristics, they have no effects on controlling congestion.

The alternative idea of flow control is the stop-and-go principle. The core part of the idea is admission control, which is to reduce the number of packets to resolve the congested situation. As far as a routing algorithm guarantees deadlock-freedom, no packets are dropped or infinitely delayed. Thus, if any packet transfer to a congested area is completely suppressed, the congested situation should soon be resolved.

Admission control methods are categorized into two types. One is for existing packets not to enter the congested area, and the other one is to stop injecting new packets that will enter the congested area. The former may propagate or even expand the congested situation where we have no general solution. On the other hand, the latter one postpones injection of the new packets, where the postponed packets do not block any other packets and, thus, the method has no side effect.

Some research results are reported in the literature, which includes three major methods of scheduling, pacing, and throttling. The first two methods, scheduling and pacing, require synthetic or analytic discussions beforehand to obtain optimal solutions. Although the two methods optimally control the packet transfer not to fall into a congested situation, the applicable situations are limited due to prior settings. On the other hand, throttling methods require no prior discussions, thus, it is applicable to dynamic situations of the network. However, since a throttling method essentially acts as a feedback control system, it does not guarantee concrete effects on the practical network situations so far.

Challenges to optimal and practical throttling methods that can exploit full potentials of interconnection network are still open. Most of research efforts have paid attention to selection of appropriate network metrics for effective control of throttling. This paper focuses discussions on a global metric *packet entropy* that is proposed in [6], [7]. Literature reports that the packet entropy represents phase transition phenomena between free-running state to congested one [8], [9].

Entropy throttling is a new throttling method whose

Manuscript received January 6, 2016.
Manuscript publicized August 25, 2016.

[†]The authors are with Department of Information Systems Science, Graduate School of Engineering, Utsunomiya University, Utsunomiya-shi, 321–8585 Japan.
a) E-mail: yokota@is.utsunomiya-u.ac.jp

DOI: 10.1587/transinf.2016PAP0007

foundation is packet entropy. Its fundamental advantages in network performance are presented in [6], [7], but comprehensive evaluations are not clarified. Literature [10] unveils fundamental but comprehensive features of the throttling method and discusses an improvement method. Contributions of this paper include two major issues. The first one is to extend the Entropy Throttling method by introducing two new ideas; hysteresis function and guard time, where we reach via deep discussions of the fundamental behavior of the Entropy Throttling method. The other one is to clarify detailed behaviors where we assume four variants of the Entropy Throttling method. The detailed discussions lead us to new knowledge on dynamical congestion control.

The rest of this paper is organized as follows. Section 2 overviews past research results. Section 3 introduces the Entropy Throttling method. Then, we discuss extensions of the throttling method in Sect. 4. Section 5 shows the comprehensive evaluation results of the original Entropy Throttling method and its extensions. Furthermore, Sect. 6 deeply discusses the enhanced throttling methods as well as the originated one. Finally, Sect. 7 summarizes this paper.

2. Related Work

It is clear that unrestricted successive injection of packets results in heavy congestion. In massively parallel environments, it is difficult to keep the network situation well optimized in order to maximize the communication performance of the network. No routing algorithm can guarantee uncongested situations unless any combinations of packet paths do not share routing resources such as buffers, (virtual) channels, and physical links. Thus, as stated in the previous section, high levels of flow controls are promising to exploit full potentials of network performance. This section summarizes flow control issues from *admission control* point of view.

The first category of admission control guides existing packets not to enter the congested area. Many of adaptive routing algorithms intend to circumvent the congested area by taking alternative routes. Cross-Line [11] employs a set of bit-mapped representation of filled buffers in the packets' traveling directions. The algorithm evaluates the bit-mapped information to select appropriate transfer direction of the packet, to avoid entering into the congested area.

Ramp metering is an alternative idea in car traffic [12]. When the car density is high enough to raise a congested situation, newly entering cars are paused at the ramp-way to the thru-traffic. This method is known effective in the real world. But, application of the method to interconnection network is limited, since no specific thru-traffic appears in many cases.

The second category of admission control is to stop injecting new packets. This category of methods are further categorized into two types; scheduled and unscheduled. Morie et al. propose a task allocation method for MPI collective communication [13]. Major intention of the method is to reduce contention in MPI communication. The fundamental part of the method is based on a scenario and the scenario-based control is not always limited as predefined one but also dynamic one. Soga et al. propose dynamic selection of scenarios according to communication situation by measuring performance metrics [14].

The alternative admission strategy is based on feedback control. The major approach is to throttle new packet injection. Many research results are reported in the literature [15]–[19]. All of the proposed throttling methods introduce additional information to start and to stop throttling.

Baydal et al. have proposed a set of throttling methods, called U-Channels, ALO, and INC [19]. All of these methods employ local information within a router. U-Channel and ALO use the number of unblocked buffers. INC measures packet flows in a certain period of time to predict congested situations. Throttling in these methods is based on *prediction* results of current and future congestion in the network, while this paper tries to use *actual* congestion information. As the authors state in [19], confidence level of the prediction results is a problem. It is clear that the presented method in this paper detects congestion with a high level of confidence. Furthermore, their methods require appropriate tuning of control parameter(s) for optimal results. Furthermore, these methods assume adaptive routing methods that this paper does not require.

DRIL [15] and CLIC [16] methods also employ local information within each router. Thottethodi et al. [18] use population of packets that are transferred to their destinations. Their major idea is to determine the optimal number of in-flight packets for the practical traffic situations. They introduced *meta-packet* to collect and distribute the number of in-flight packets which is global information. The meta-packet mechanism performs reduction and broadcast functions. They adopt the acquired information to throttle injection according to a certain threshold that is determined by their *hill-climbing* method. The basic idea of the meta-packet is close to our measurement circuit that will be described in Sect. 3.2. Our method assumes an ideal condition that the circuit runs independent of packet transfer.

With respect to the admission control, *packet pacing* [20] is an idea close to throttling. Obviously, the method does not essentially employ feedback properties.

State Propagation Throttling (SPTh) is given by monitoring quasi-global congestion information [21]. The method is derived from space-time chart that is frequently used in self-driven particle research in physics. The SPTh method predicts future congestion by observing distant congestion in packets' traveling directions. In this method, each router monitors congestion situations and the information is not summarized. Major difference in this paper is to use global information on congestion, i.e., packet entropy.

3. Entropy Throttling

3.1 Packet Entropy and Its Approximation

Entropy throttling is proposed under the fundamental observation results on congestion [6], [7]. As stated in Sect. 1, once congested situation arises, the congested area grows rapidly and the situation sustains for a long time. Congested and uncongested situations are clearly distinctive. This phenomenon is represented as *phase transition*.

We are successful in representing the phase transition behavior by introducing packet entropy. By substituting a molecule in thermodynamics to a packet in a network, entropy is discussed based on mean free path of the molecule (packet). The difference is that a collision of molecules is elastic and the molecules are assumed that they continuously travel even after the collision. We define free path of a packet as the number of hops of the packet in a specific period of time.

Suppose that a packet is transferred by $\varDelta P$ hops in a $\varDelta t$ period of time. The information quantity of the packet is represented as

$$e = -\log_2 \frac{1}{\varDelta P} = \log_2 \varDelta P. \tag{1}$$

Note that $\varDelta P \leq \varDelta t$ since a packet moves at most one hop in one clock cycle. The packet entropy E is defined as the average of (1):

$$E = \frac{1}{N_p} \sum_{i}^{N_p} \log_2 \varDelta P_i \tag{2}$$

where N_p is the number of packets.

We further approximate the packet entropy for practical control of the network. The originated packet entropy requires a certain period of time to measure and precision of the entropy depends on the length of the period. Short measurement period causes low precision of entropy. However, long measurement period causes too long latency to dynamically control the packet flow, whereas reasonable precision of entropy is given.

We make use of the inherent difference between molecules and packets, that is, packet collision is inelastic. When two or more packets conflict, only one of them is transferred and the other ones are blocked. Thus, a packet is likely to be blocked in a congested situation and the probability of the packet block increases as the congestion level increases.

Any packet goes through a packet buffer at every input port and packet conflict occurs among the head packets in the buffers within the same router. Thus, a router can recognize which local buffer holds packets (valid buffers) and furthermore which one transfers the head packet (active buffers) at every cycle. Each router counts the numbers of both valid and active buffers, which are represented as Nv_i and Na_i, respectively for the i-th router. By summing up the two numbers, the total numbers of valid and active buffers are obtained, i.e.,

$$N\mathrm{v} = \sum_{i}^{N_r} Nv_i, \quad N\mathrm{a} = \sum_{i}^{N_r} Na_i \tag{3}$$

where N_r is the number of routers. We can obtain average of packet mobility that is the ratio of active buffers over valid ones, i.e.,

$$R\mathrm{a} = N\mathrm{a}/N\mathrm{v}. \tag{4}$$

We call the ratio as *mobility ratio*. The ratio highly correlates with the packet entropy [7], and furthermore the ratio essentially requires no delay time, thus we use the ratio as approximation of the entropy.

3.2 Entropy Circuit

To measure the mobility ratio Ra, the numbers of active and valid buffers, Na and Nv, are required. To obtain these numbers, reduction operation over the whole network, which is costly, is required. This paper assumes a dedicated circuitry as depicted in Fig. 1, which is introduced in [6], [7].

The measurement circuit consists of five registers and four adders in each router. One of the registers, called *profile register*, holds Nv_i and Na_i. The other four registers correspond to the input/output port directions. Horizontal direction registers summarize horizontal sum of Nv_i and Na_i. Light green registers in Fig. 1 do reduction operation and dark green registers broadcast the sum in the horizontal direction. Vertical registers in pink color summarize the horizontal sum, and red registers broadcast the total sum. All the reduction and broadcast operations are executed in parallel and each register is updated in an asynchronous manner. Note that no reduction and broadcasting operation interferes with any other operations.

Each router reads the resulting Na and Nv values from its own red register and calculates Ra. The Nv value is also used for comparing with the release parameter Rn as stated in Sect. 3.3.

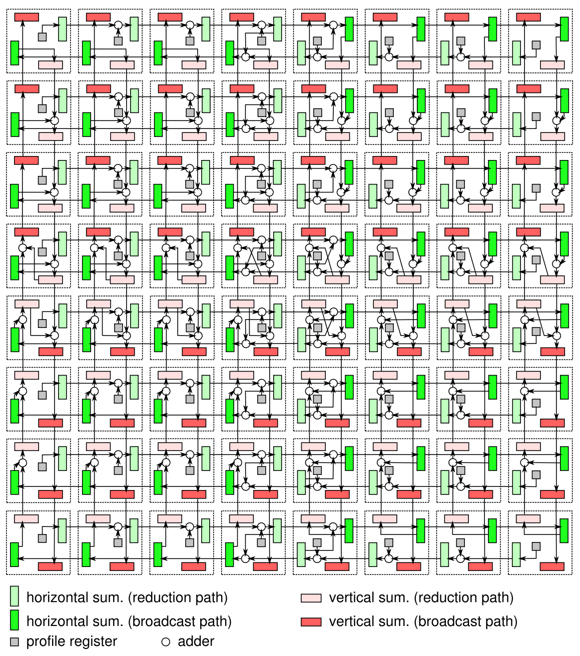

horizontal sum. (reduction path)
vertical sum. (reduction path)
horizontal sum. (broadcast path)
vertical sum. (broadcast path)
profile register
adder

Fig. 1 Entropy measurement circuits embedded in 2D-torus network.

```
if( Nv < Rn * Nr ) {
    throttle_OFF();
} else if( Ra < Rth ) {
    throttle_ON();
} else {
    throttle_OFF();
}
```

Fig. 2 Basic throttling algorithm.

```
if( Nv < Rn * Nr ) {
    throttle_OFF();
} else if( is_throttle_ON() ) {
    if( Ra < R_ON  ) throttle_ON();
} else {
    if( Ra > R_OFF ) throttle_OFF();
}
```

Fig. 3 Throttling algorithm with hysteresis feature.

3.3 Fundamental Throttling Algorithm

The mobility ratio $R\mathrm{a}$ represents the average degree of congestion. Low $R\mathrm{a}$ shows that the network is in a congested situation, and high $R\mathrm{a}$ shows that packets run freely. After calculating the $R\mathrm{a}$ value by $N\mathrm{a}$ and $N\mathrm{v}$, the simplest algorithm suppresses packet injection when $R\mathrm{a}$ is less than a certain threshold $R\mathrm{th}$.

But, this algorithm is too simple to adopt some unbalanced communication patterns with deterministic routing algorithms. In such situations, packets are blocked in very limited area and other portions allow packets freely-running. To adopt such situations, we have introduced the second parameter, *forced release ratio* $R\mathrm{n}$, that represents the minimum ratio of the number of valid buffers to the total number of routers. When

$$R\mathrm{n} > N\mathrm{v}/N_r, \tag{5}$$

throttling control is released even when $R\mathrm{a}$ is low and shows congested situations ($N\mathrm{v}$ is given by (3) in the previous section and N_r is the number of routers). Thus, the resulting throttling algorithm is given as shown in Fig. 2.

4. Extension of Entropy Throttling

4.1 Introducing Hysteresis Function

Although Entropy Throttling shows desirable features, the method has inherent oscillation characteristic. Detailed discussion on the effects of the throttling method is given in Sect. 6.1. Figure 8 shows the temporal behavior of the number of the in-flight packets and it is a representative behavior in collective communication.

The mechanism of the oscillatory behavior can be described as follows. (1) Each node detects congested situation via the entropy circuit, and (2) each node starts suppression of packet injection. Then, (3) the congested situation is gradually resolved and (4) each node releases packet injection. Although the detection mechanism works asynchronously, almost all of the nodes start new packet injection within a short period of time. Thus, (5) a large number of injected packets rush into the network in unison, (6) which results in new congestion.

In this paper, we assume that the throttling control principle is packet-by-packet basis. Each node checks its own throttling mode (ON or OFF) when it injects a new packet. If the mode is ON, the node suspends the new packet injection until the throttling mode returns to OFF. On the other hand, if the mode is OFF, the node can inject the new packet. Packet injection should not be suspended even when the network falls into a congested situation, since the network should guarantee deadlock freedom.

If the OFF-to-ON threshold is low, the network keeps a sparse situation that has sufficient room for performance improvement. When the ON-to-OFF threshold is high, this means that packet injection is released in a crowded situation, thus, packet injection quickly saturates network communication that will lead to severe congestion. Furthermore, the behaviors at the onset and release of congestion are not symmetrical. As literature has reported [8], [9], congestion grows rapidly in the order of $O(1)$, while the release speed is $O(N^2)$ in $N \times N$ 2D-torus networks.

This observation suggests that the throttling algorithm requires two threshold values for OFF-to-ON and ON-to-OFF, which correspond to a hysteresis function. The basic method of Entropy Throttling inherently assumes the same threshold for both situations, which spoils the potentials of the method. We can expect improved performance if the two threshold values are properly determined. By introducing the hysteresis idea, we split the threshold parameter $R\mathrm{th}$ to R_{ON} and R_{OFF} and the resulting throttling algorithm is shown in Fig. 3.

4.2 Introducing Guard Time

We further discuss observation delay of the entropy value. To update the mobility ratio (i.e., the approximated entropy), the entropy circuit (shown in Fig. 1) requires reduction and broadcast operations in two dimensions. Thus, observation delay time is essential to appropriately control the network.

When a node injects a new packet, the node cannot instantly detect whether the injected packet causes new congested situation or not, since the node can only recognize the effect after a certain delay. This implies that a node should suppress new packet injection until the entropy circuit reflects the entropy value. Thus, we introduce guard time to suppress new injection. This paper assumes two kinds of guard time: fixed and randomized, which we discuss in Sect. 5.1.3.

5. Evaluation

5.1 Evaluation Environment

In this paper, we use our interconnection network simulator.

Table 1 Traffic patterns used.

abbreviation	description
trns	transpose. $(X, Y) \longrightarrow (Y, X)$
shfl	perfect shuffle. $w_{2n-1}w_{2n-2}\cdots w_0 \longrightarrow w_{2n-2}\cdots w_0 w_{2n-1}$
bcmp	bit-complement. $w_{2n-1}w_{2n-2}\cdots w_0 \longrightarrow \overline{w_{2n-1}}\,\overline{w_{2n-2}}\cdots\overline{w_0}$
brev	bit-reverse. $w_{2n-1}w_{2n-2}\cdots w_0 \longrightarrow w_0\cdots w_{2n-2}w_{2n-1}$
brot	bit-rotation. $w_{2n-1}\cdots w_1 w_0 \longrightarrow w_0 w_{2n-1}\cdots w_1$
torn	tornado. $W \longrightarrow \mathrm{mod}(W + N/2, N^2)$
rand	random. Destination node is randomly selected.
rpar	random pair.

The simulator supposes a simple one-phase transmission of packets, thus a packet goes to the adjacent router in one cycle when it is not blocked. The simulator also employs virtual control channel features [22] that is extended to support the entropy measurement circuit illustrated in Fig. 1. This paper assumes that register values are transferred in every clock cycle to discuss the proposed method in an ideal situation. The simulator is also extended to follow the evaluation methodologies, unsteady (collective) and steady communication performance. We use eight representative traffic patterns, matrix transpose (trns), perfect shuffle (shfl), bit complement (bcmp), bit reverse (brev), bit rotation (brot), random (rand), and random pair (rpar). Table 1 gives brief descriptions.

This paper assumes 32×32 two-dimensional torus network with the deterministic dimension-order routing algorithm with three virtual channels. A packet is injected in the channel-0 and, every time it goes across a date-line, the assigned virtual channel number is increased by one. This paper assumes two date-lines in each of x and y dimension, i.e., $x = 0$ and $x = N/2$ positions where N is the number of nodes in x and y directions. Each input port for packet routing has three packet buffers that correspond to virtual channels. Capacity of each buffer is 15 flits. This paper uses fixed length packets of eight flits.

Throttling control is also implemented by extending the simulator. Each node has an additional throttled-mode register that is controlled by the corresponding router. Each router decides the node's throttling state by the collected values of Na and Nv computed by the entropy circuit (Fig. 1) at every cycle. Each node suppresses new packet injection when it is in a throttled mode. After the node starts injecting a new packet, the packet injection is not interrupted until the packet is fully injected.

5.1.1 Unsteady Communication Performance

We also assume unsteady (collective) communication situations to evaluate transient and practical performance. The degree of congestion caused by collective communication depends on the number of packets that are transferred in the communication. Traffic load with one or few packet injection from each node is absorbed in the packet buffers, and sometimes the network does not show severe congestion. Thus, we use ten packets per node per communication. The ten packet parameter is given by our preliminary evaluation.

The major concern in collective communications is duration time. The simulator is extended to count the number of collective communication packets in every simulation cycle, and it records the start and end times of collective communication, thus, we can measure duration time accurately.

5.1.2 Steady Communication Performance

Ordinal interconnection network research uses an iterative evaluation method to draw a performance curve. The method repetitively runs a series of simulation processes in which fixed traffic load is given as a parameter and measures throughput and average latency after the packet flow is stabilized. But, this method offers only sparse measured points or it requires many simulation runs to draw a smooth curve.

In general, major discussions points in interconnection networks performance are throughput and (average) latency. Ordinal research compares shapes of performance curves. A network method that can tolerate high traffic load is preferable. Furthermore, low average latency is welcomed. This evaluation methodology is used in common in interconnection network research, however, it cannot compare multiple interconnection network methods at a time. Thus, instead of comparing shapes of performance curves, we have introduced some metrics to represent interconnection network performance.

One of the major concerns is saturation point of throughput. We define a critical load ratio as the inflection point of performance curve. In a low-traffic situations, throughput is proportional to the given traffic load. But, as the traffic load exceeds a certain threshold, throughput is suddenly saturated. We measure the saturation point as decreasing point of the gradient of the performance curve. In this paper, we define the critical load ratio as 10 percent degradation point of the performance curve.

To precisely measure the critical load ratio, we have introduced a consecutive evaluation method where the traffic load is gradually increased from zero. This evaluation method, *ramp load method* [23], does not require multiple simulation runs, but requires long simulation cycles to obtain precise results. In this paper, we conservatively increase the traffic load by 0.1 flits/cycle in every 1,000,000 cycles. Throughput and average latency values are measured in every 100 cycles. The measured values are in some levels of variety since packet injection intervals are randomized while satisfying the given traffic load. Thus, to measure accurate critical load ratio, we use moving average in 400 samples of the measured data.

5.1.3 Evaluation Methodology

In this paper, we assume four Entropy Throttling models in addition to non-throttling case for baseline of comparison.

noth is a baseline model where No throttling function is applied. This model is used for comparison and no parameter is assumed.

base— **base(Rth, Rn)** is the fundamental throttling method whose algorithm is given in Fig. 2. This model is represented by Rth and Rn parameters.

hyst— **hyst(R_{ON}, R_{OFF}, Rn)** is the Entropy Throttling that employs the hysteresis function, whose algorithm is shown in Fig. 3. This mode is represented by three parameters R_{ON}, R_{OFF}, and Rn.

gt_a— **gt_a(t_G, R_{ON}, R_{OFF}, Rn)** is the Entropy Throttling that employs the hysteresis function and guard time. This model is represented by four parameters R_{ON}, R_{OFF}, Rn, and average guard time t_G. In this model, guard time is given as an average value; actual guard time is determined by uniform random number between $[0{:}2t_G]$.

gt_x— **gt_x(t_G, R_{ON}, R_{OFF}, Rn)** is also the Entropy Throttling that employs the hysteresis function and guard time and four parameters R_{ON}, R_{OFF}, Rn, and fixed guard time t_G. Although this model is similar to gt_a, the length of guard time t_G is fixed, not randomized.

As simulation time of collective communication is short, we have exhaustive evaluation for collective communication. Actual parameter values are

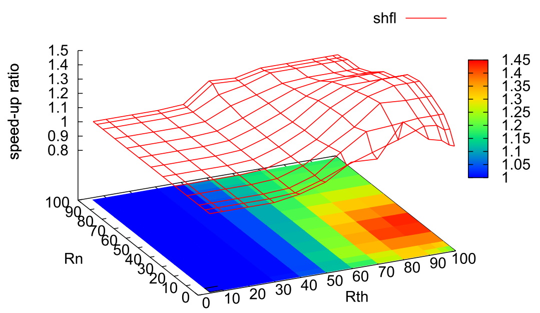
$$\begin{aligned} R\mathrm{th}, R_{\mathrm{ON}}, R_{\mathrm{OFF}} \in \{ & 5, 10, 20, 30, 40, 50, 60, 70, 80,\\ & 90, 95, 98, 100\},\\ R\mathrm{n} \in \{ & 0, 1, 5, 10, 20, 30, 40, 50, 60, 70, 80,\\ & 90, 95, 98\}. \end{aligned}$$

For example, in the hysteresis model hyst(R_{ON}, R_{OFF}, Rn), all of the possible combination of R_{ON}, R_{OFF} and Rn were evaluated. Evaluations for deterministic conditions are executed once. Non-deterministic cases, i.e., rand and rpar traffic patterns and the gt_a model, are executed 100 times and average and standard deviation are recorded.

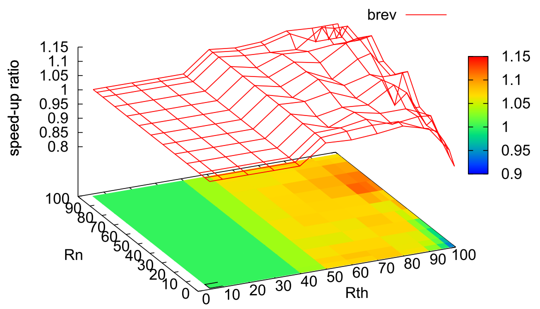
Short simulation time of collective communication allows exhaustive evaluation, however evaluation of steady communication performance requires long time. Our focus on the steady communication is critical load ratio where throughput performance just starts degraded. Thus, we executed steady communication evaluations for a selected set of parameters that achieves good performance in collective communication situations. Details of the selection are stated in Sect. 5.

5.2 Selecting Representative Cases

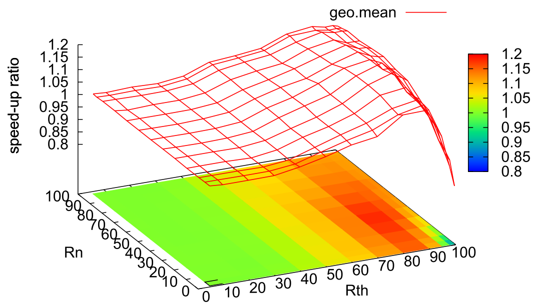
If a user has prior knowledge on the traffic pattern in his application program, he can select the best combination of parameters. Thus, it is worth to determine the best combination for each traffic pattern. On the other hand, when the user's application uses multiple kinds of traffic patterns, or when the user cannot recognize traffic patterns in advance, the optimal parameter-set is not strictly determined. Thus, we need the best combination of parameters that leads to the best performance in average. This discussion suggests two kind of representative parameter-sets for different classes of applications.



(a) shfl traffic results



(b) brev traffic results



(c) geometric mean for each parameter combination.

Fig. 4 Distribution of speed-up ratio.

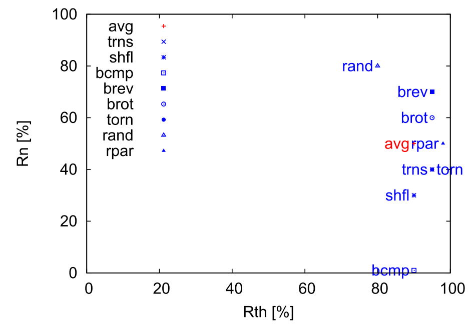
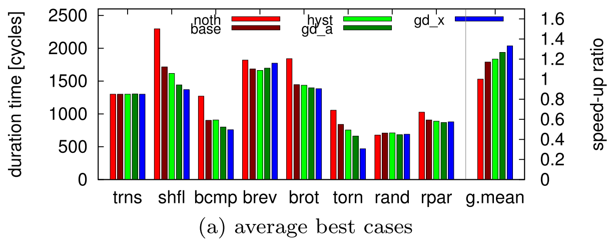


Fig. 5 Distribution of the best parameter combinations.

Figure 4 shows distribution of performance in terms of speed-up ratio for possible combinations of parameters. In this figure, base model is used. Figures 4 (a) and (b) show the shfl and brev results, respectively. As the figures show, the best parameter-sets are different in different traffic patterns. Figure 5 shows the distribution of the best parameter-set for each traffic pattern. We define an *individual best* case


(a) average best cases

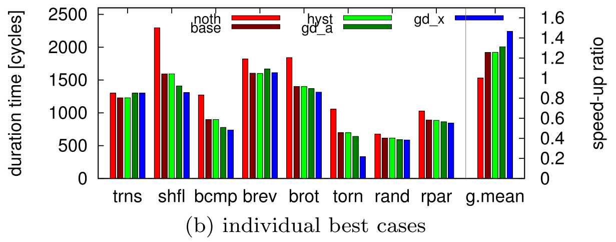
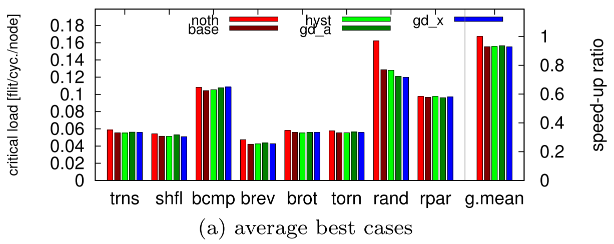

(b) individual best cases

Fig. 6 Collective communication performance.


(a) average best cases

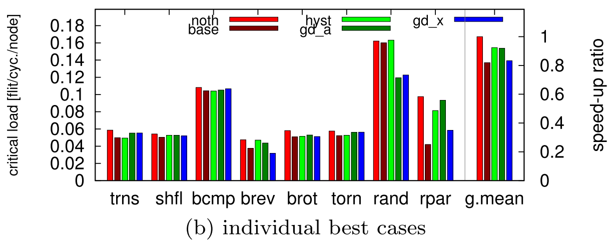

(b) individual best cases

Fig. 7 Steady communication performance.

in which the best performance is achieved in a certain traffic pattern.

On the other hand, Fig. 4 (c) show the distribution of geometric mean of speed-up ratios in all eight traffic patterns. We define an *average best* case in which the geometric mean value is the largest. As we discussed above, when the user knows the traffic pattern precisely, the individual best case is applicable. The average best case is useful if the user has no prior knowledge in traffic patterns in his application.

5.3 Collective Communication Performance

Figure 6 summarizes the evaluation results of collective communication, where the average best and individual best cases are shown in the separated figures. Each figure consists of two parts; the left side of the figure shows duration times for each traffic pattern, and the right shows geometric mean (g.mean) of the speed-up ratio. As defined in Sect. 5.2, the average best case uses the same parameter-set for any traffic patterns, while the individual best cases may use different parameters for different traffic patterns.

gt_x shows the best performance except in a few traffic patterns, trns and bcmp. The average speed-up ratios are 1.331 in the average best case and 1.465 in the individual best case. gt_x with $t_G = 16$ marks the best performance improvement of 3.17 times in the torn traffic (where the duration time is reduced from 1056 to 333 cycles).

5.4 Steady Communication Performance

Evaluation results of steady communication situations are summarized in Fig. 7. Likely to Fig. 6, each figure consists of two parts, but critical load ratios are shown in the figure.

This figure illustrates that the proposed Entropy Throttling models do not show significant improvements in critical load ratio. As Fig. 7 (a) shows, the average best case sustains steady communication performance except in the rand case. On the other hand, individual best cases show some drastic degradation in the steady communication performance. Sections 6.4 and 6.5 discuss about the phenomena.

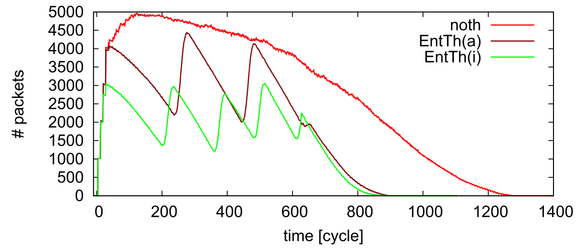


Fig. 8 Temporal behavior of Entropy Throttling and no-throttling cases, traffic pattern is bcmp.

6. Discussions

6.1 Effects of Throttling Function

Figure 8 shows the temporal behaviors of the number of packets that are transferred in the interconnection network, which we call *in-flight* packets. In this figure, noth shows no throttling case. EntTh(a) means the best average and EntTh(p) shows the individual best cases in the base model.

The figure compares Entropy Throttling effects with that of the non-throttling case. When no throttling is applied, the number of packets increases rapidly at the beginning, resulting in severe saturation of the network. After the saturation, the number of packets gradually decreases and finally completes the collective communication.

On the other hand, Entropy Throttling shows unstable and oscillating behavior as depicted in Fig. 8. This instability comes from the throttling mechanism. We further need detailed discussions. Figure 9 shows detailed behavior in the individual best case of parameters in bcmp traffic.

In Fig. 9, EntTh(i) curve shows the number of in-flight packets in the upper-half of the figure. In the lower-half of the figure, %ba shows N_V/N_r where N_r is the number of

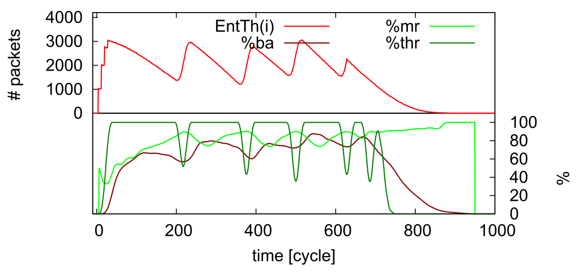
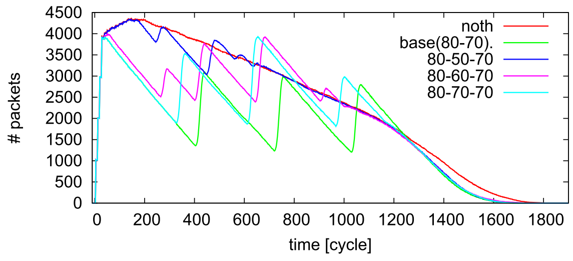
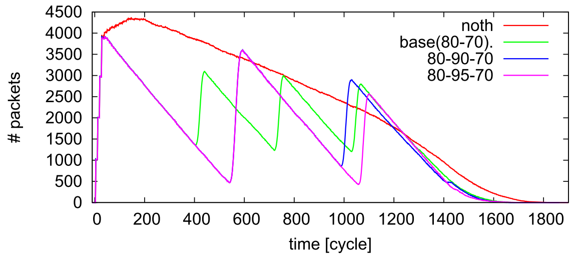


Fig. 9 Detailed behavior of Fig. 8, bcmp.

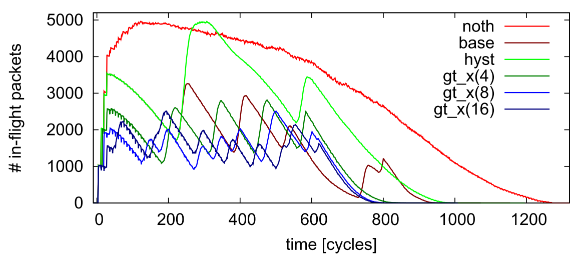


(a) R_{OFF} is low.

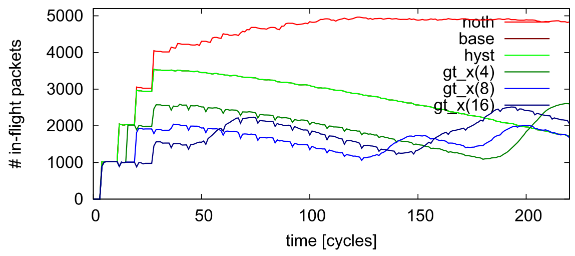


(b) R_{OFF} is high.

Fig. 10 Effects of the hysteresis parameter.



(a) whole results



(b) initial part of (a)

Fig. 11 Temporal behaviors of Entropy Throttling models, shfl traffic.

nodes and in this case $N_r = 32 \times 32 = 1{,}024$. %thr shows the percentage of the nodes that suspend new packet injection. Furthermore, %mr represents Ra in percentage unit.

As this figure shows, when the %thr value is high, i.e., the throttling mode is ON, the number of in-flight packets gradually decreases and the mobility ratio increases. When the mobility ratio exceeds the threshold, the throttling mode turns OFF and the number of in-flight packets rapidly increases. Thus, the mobility ratio keeps around 80% with some oscillatory behaviors.

6.2 Effects of Hysteresis Function

Results in Fig. 6 show that the hysteresis feature does not contribute in many traffic pattern except rand. However, the additional parameters should affect the communication behavior, where we do not yet clarify detailed behaviors of the hysteresis method.

Figure 10 shows temporal behaviors of some combinations of R_{ON}, R_{OFF}, and Rn parameters with the non-throttling and original Entropy Throttling results. In this figure, we assume $R_{\text{ON}} = R$th and the same Rn value is used. Figure 10 (a) shows $R_{\text{OFF}} < R$th cases. In these cases, a throttled situation will soon be released. When a router detects the throttling condition $R\text{a} < R_{\text{ON}}$ and the router suppresses packet injection, at that moment, the release condition is also met, i.e., $R\text{a} > R_{\text{OFF}}$. Although the two conditions of trigger and release seem to conflict to each other, they are independently evaluated. We should notice that the mobility ratio at that moment is in a transient situation and that the ratio will not stay at the same value at the next moment. Thus, when the mobility ratio rapidly decreases, the low-mobility situation will sustain for a certain period of time. When the release parameter R_{OFF} is sufficiently low, the communication behavior is closed to the non-throttling case. As R_{OFF} increases, the behavior is closed to that of the original Entropy Throttling method. When $R_{\text{ON}} = R_{\text{OFF}} = R$a, the hysteresis method is equivalent to the original method (where $R\text{a} = R_{\text{ON}} = R_{\text{OFF}}$).

On the other hand, when the release parameter R_{OFF} is higher than the trigger parameter R_{ON}, throttled situation sustains as shown in Fig. 10 (b). This results in deep valleys in the performance curves, while the duration times are not remarkably different.

6.3 Effects of Guard Time

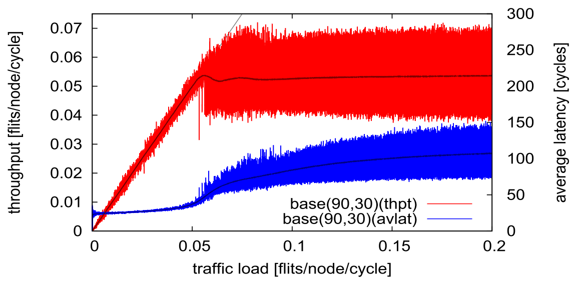
Figure 11 shows time sequence chart of the number of in-flight packets in various Entropy Throttling models and non-throttling case. Traffic pattern is bcmp in this figure. Figure 11 (a) shows full sequence from the beginning to completion and Fig. 11 (b) shows the initial behavior during 220 clock cycles. In the figure, gt_x(n) represents the length of guard time (n).

At the initial stage, the number of in-flight packet increases in a staircase pattern in any cases in the figure. At the initial step, any router has no packet and every input buffer is empty. Every node starts packet injection in unison. Then, the number of in-flight packets increases by the total number of node (that is $32 \times 32 = 1024$) at every eight cycles (that is the length of packet in flit unit). On the other hand, the entropy circuit performs reduction and broadcasting operations in parallel with the packet injection, This operation requires $2N$ clock cycles in $N \times N$ networks in the worst case, however, even intermittent results of the entropy circuit is meaningful to control the throttling state, since the entropy registers (shown in Fig. 1) are updated asynchronously. The initial behaviors shown in Fig. 11 (b) clearly represents the effects of the guard time. In the figure, base and hyst run four steps at the initial stage, whereas gt_x(4) runs three steps and gt_x(8) and gt_x(16) run two steps. These behaviors show the effect of the guard time that pauses new packet injection until the control variable (i.e., the mobility ratio) is reflected for feedback control.

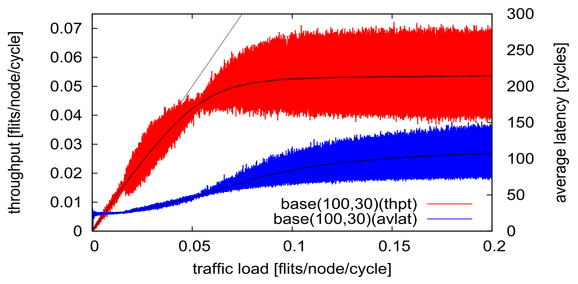
With respect to the stability in the throttling control, we should consider capacity of buffers, which might affect the effectiveness of the proposed method. This paper assumes 15-flit buffers in each of virtual channel in each input port. A large buffer capacity makes the network tolerant to temporal fluctuation of traffic flows, which also allows the network traffic sustain uncongested. From a different point of view, the buffer capacity will cause *time constant* characteristic in the network behavior. Large buffer may tolerate transient fluctuations, however, once a congested situation arises, the situation sustains long. Discussions on the buffer capacity are our future work.

6.4 Throttling Effects on Steady Communication

We can find two contrast cases at the shortest duration time; one achieves the shortest duration time and relatively high critical load ratio, and the other one fails to achieve high critical load. Parameters of the two cases are base(90,30) and base(100,30) (Ra,Rn), respectively. Figure 12 illustrates the performance curves of steady communication situations.



(a) Rth = 90, Rn = 30 (percent) case



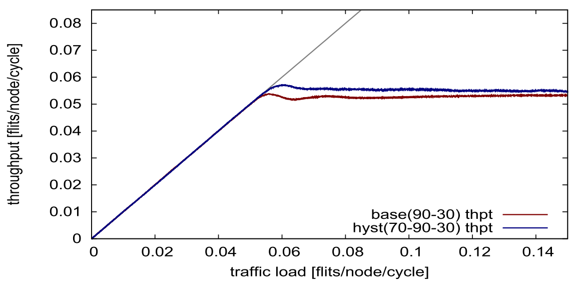
(b) Rth = 100, Rn = 30 (percent) case

Fig. 12 Comparison of steady communication performance of two parameter sets that mark the same collective communication performance.

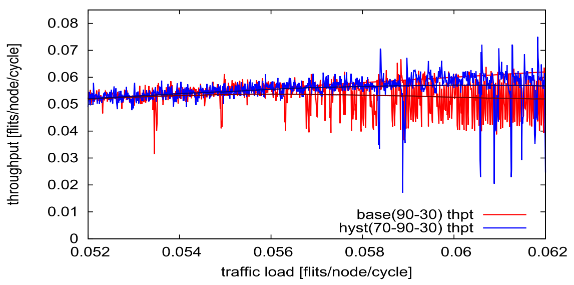
thpt shows throughput in the left y axis, and avlat shows the average latency in the right y axis. The base(90,30) case sustains throughput until the traffic load is near 0.06 [flits/node/cycle], whereas the base(100,30) case saturates at the early stage of the traffic load. This indicates that we should select appropriate values of control parameters. Some set of parameters is not optimal for steady communication even if the parameter-set performs best in collective communication.

Figure 13 compares throughput performance curves of the original Entropy Throttling method, base, and the hysteresis method hyst. Both parameters are selected by the average best cases; base(90,30), i.e., Rth = 90 percent and Rn = 30 percent, and hyst(70,90,30), i.e., R_{ON} = 70 percent, R_{OFF} = 90 percent, and Rn = 30 percent. The goal of this comparison is to clarify the effectiveness of the hysteresis method. Figure 13 (a) shows smoothed curves calculated by moving average of 400 samples. As shown in the figure, the critical load ratio is improved by introducing the hysteresis feature. Actual critical load ratios are 0.0505 and 0.0532 [flits/node/cycle].

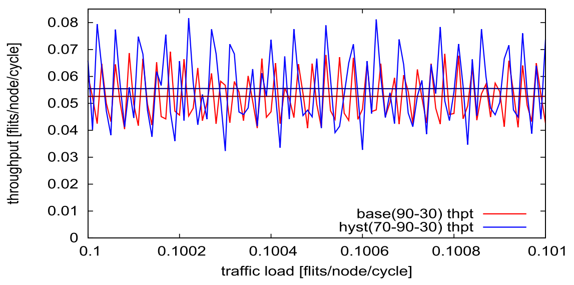
Raw evaluation data shows extremely large fluctuations in large traffic load conditions as shown in Fig. 12.



(a) smoothed performance curves, shfl traffic pattern.



(b) around critical load ratio.



(c) heavily congested situation.

Fig. 13 Effect of the hysteresis feature.

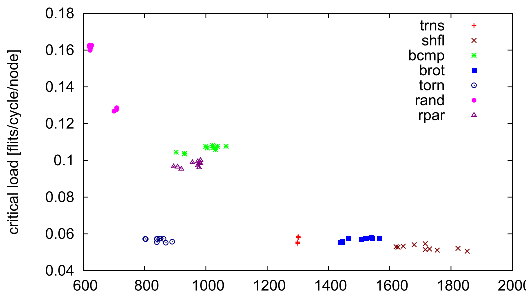
Here, we compare detailed behaviors of the original Entropy Throttling and hysteresis methods in Figs. 13 (b) and (c) that are enlarged and detailed part of Fig. 13 (a). Figure 13 (b) illustrates near-threshold situations and (c) shows heavily congested situations. In Fig. 13 (b), both cases show steeply oscillating behaviors, however, they are different in duration time. The original method shows frequent oscillation whereas the hysteresis method has certain intervals of time. But, as shown in Fig. 13 (c), in heavily congested situations, oscillation frequencies of both methods become similar. These behaviors obviously come from the control parameters and the observation results imply potential performance improvement in the hysteresis method.

6.5 Correlation between Collective and Steady Performance

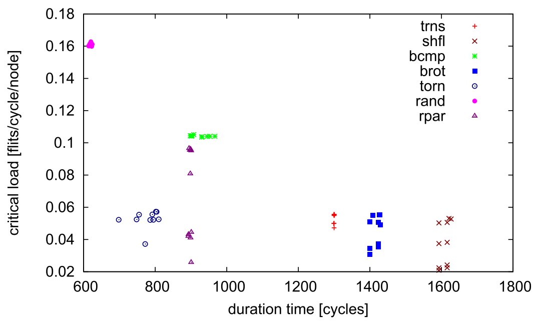
As discussed in Sect. 5.2, we intend the individual best cases where the traffic pattern is clarified in advance, whereas the average best cases are assumed for the situations whose traffic is unknown. Thus, the average best cases should be versatile in traffic pattern, as Sect. 5.3 and Fig. 6 show in collective communication performance.

We can expect the versatility in steady communication traffic as well as collective one. Figure 14 shows the correlation between duration time and critical load ratio in various traffic patterns. The Entropy Throttling model is base. Figures 14 (a) and (b) show the results of the average best and individual best cases, respectively. In each figure, top-10 combinations of parameter are depicted.

As illustrated in Fig. 14 (a), the average best cases show narrow distribution in critical load ratio except rand traffic, whereas the individual best cases show wide distributions in many traffic patterns. This observation suggests that the individual best cases may achieve insufficient performance when the actual traffic differs from their expected one. On the other hand, the average best cases are versatile although their performance is slightly lower than that of the individual best cases.



(a) average best cases



(b) individual best cases

Fig. 14 Correlation between duration time and critical load.

6.6 Variants in Randomness Settings

Table 2 shows average duration time and standard deviation in various traffic patterns. Throttling algorithm is gt_a and individual best cases are shown in this table. Average guard time in gt_a shows the quantity of randomness, thus, we expect that the standard deviation correlates with the length of guard time. Table 2 does not show any correlation that we firstly expected.

This fact suggests that small difference in timing may cause large fluctuation in performance. This implicates that variants in packet arrival timing in each router affect communication performance in duration time.

Ths amount of standard deviation also varies in traffic pattern. For example, transpose traffic (trns) shows almost zero deviation. This result comes from stiffness in communication traffic and routing algorithm (that is deterministic dimension-order routing in this paper). Actually, as Fig. 6 shows, any variants of the Entropy Throttling cannot achieve significant improvement in the trns traffic.

6.7 Performance under Practical Conditions

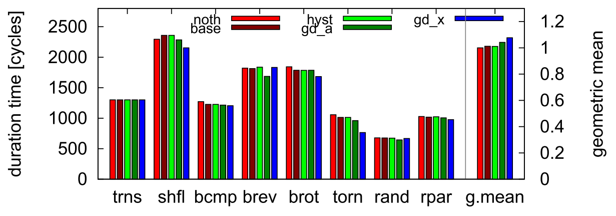
This paper assumes dedicated communication paths for the entropy circuit (shown in Fig. 1), although many of the systems cannot allow such costly implementation in the practical situations. Our interconnection network simulator supports the virtual control channel (VCC) mechanism that is proposed in the literature [22]. The virtual control channel shares the physical link with virtual channels (VCs) that transfer the contents of packets. We further extend VCCs to share physical links in a cycle-stealing manner, where transmission of VCC is permitted only when no VC uses the shared physical link.

Figure 15 shows the collective communication perfor-

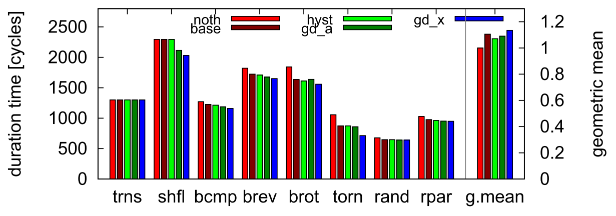
Table 2 Fluctuations by randomness in guard time.

traffic	guard time=4		guard time=8		guard time=16	
pattern	dur. time	std. dev.	dur. time	std. dev.	dur. time	std. dev.
trns	1300.8	0.37	1301.0	0.00	1301.0	0.00
shfl	1500.2	37.03	1455.9	34.46	1438.4	20.30
bcmp	844.0	15.49	815.8	16.26	796.5	14.40
brev	1670.8	21.24	1685.6	23.61	1674.6	18.79
brot	1393.5	18.54	1389.2	13.45	1379.0	14.08
torn	696.9	21.13	643.3	10.38	642.0	6.50
rand	611.1	24.58	603.8	22.97	594.2	20.56
rpar	880.4	51.86	867.6	46.49	865.0	61.27

(unit: clock cycles)



(a) average best cases



(b) individual best cases

Fig. 15 Collective communication performance under practical conditions.

mance results under that simulation condition. In this realistic situation, no VCC interferes with VC and thus communication performance is not degraded unnecessarily. But, on the other hand, performance of the entropy circuit is considerably degraded. Although the performance improvements are limited, as compared with Fig. 6, average best and individual best cases still mark 1.077 and 1.134 of speed-up ratios, respectively.

7. Conclusions

Once a congested situation arises in the network, communication performance is drastically degraded in both throughput and latency. This problem severely affects in massively parallel computing systems, since an interconnection network should not drop any packets even when severe congestion almost stops the network functions.

To solve this problem, this paper addresses admission control schemes based on a concept of feedback control. This paper focuses discussions on the Entropy Throttling method that the authors have proposed as a congestion control mechanism based on the packet entropy. The packet entropy is a global control variable that can represent congestion status of packets independent of communication situations such as traffic patterns.

This paper firstly presents the packet entropy and the original Entropy Throttling method. Then, this paper discusses extensions of the method based on the detailed observations on the original method. Through the discussions, this paper introduces two major concepts for the extension; hysteresis function and guard time. The former corresponds to the asymmetric behavior at the onset and release of congestion, and the latter corresponds to the observation delay of the entropy.

The comprehensive and detailed evaluation results offer us new knowledge on dynamic congestion control based on the feedback concept by means of the packet entropy. We can expect that deep understanding of the throttling methods drives us for further improvement in congestion control of networks.

The remaining discussions that are our future work are as follows. One of the weak points of the Entropy Throttling methods is oscillatory behavior between congested and uncongested situations, which prevents the network's full ability. We need to continue discussions for stable control at the edge of uncongested situation where the best performance is expected.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers 24500055, 15K00068, and 16K00068.

References

[1] J. Duato, S. Yalamanchili, and L. Ni, Interconnection Networks: An Engineering Approach, Morgan Kaufmann, 2003.

[2] W.J. Dally and B. Towles, Principles and Practices of Interconnection Networks, Morgan Kaufmann, 2004.

[3] L.M. Ni and P.K. McKinley, "A survey of wormhole routing techniques in direct networks," Comput., vol.26, no.2, pp.62–76, Feb. 1993.

[4] P. Kermani and L. Kleinrock, "Virtual cut-through: A new computer communication switching technique," Computer Networks, vol.3, no.4, pp.267–286, 1979.

[5] W.J. Dally, "Virtual-channel flow control," IEEE Trans. Parallel Distrib. Syst., vol.3, no.2, pp.194–205, March 1992.

[6] T. Yokota, K. Ootsu, F. Furukawa, and T. Baba, "Entropy throttling: A physical approach for maximizing packet mobility in interconnection networks," Proc. 11th Asia-Pacific Computer Systems Architecture Conference (ACSAC 2006), Advances in Computer Systems Architecture, Lecture Notes in Computer Science, vol.4186, pp.309–322, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[7] T. Yokota, K. Ootsu, F. Furukawa, and T. Baba, "Entropy throttling for maximizing packet mobility in interconnection networks," IPSJ Trans. Advanced Computing Systems, vol.47, no.SIG 12 (ACS 15), pp.1–11, 2006 (in Japanese).

[8] T. Yokota, K. Ootsu, F. Furukawa, and T. Baba, "A cellular automata approach for understanding congestion in interconnection networks," IPSJ Trans. Advanced Computing Systems, vol.47, no.SIG 7 (ACS 14), pp.21–42, May 2006 (in Japanese).

[9] T. Yokota, K. Ootsu, F. Furukawa, and T. Baba, "Phase transition phenomena in interconnection networks of massively parallel computers," J. Phys. Soc. Jpn., vol.75, no.7, 074801 (7 pages), 2006.

[10] T. Yokota, K. Ootsu, and T. Ohkawa, "Entropy throttling: Towards global congestion control of interconnection networks," Proc. 3rd International Symposium on Computing and Networking (CANDAR '15), pp.40–49, Dec. 2015.

[11] T. Yokota, M. Nishitani, K. Ootsu, F. Furukawa, and T. Baba, "Cross-Line: A novel routing algorithm that uses global information," IPSJ Trans. Advanced Computing Systems, vol.46, no.SIG 16 (ACS 12), pp.28–42, 2005 (in Japanese).

[12] M. Papageorgiou and A. Kotsialos, "Freeway ramp metering: An overview," Proc. 2000 IEEE Intelligent Transportation Systems, pp.228–239, 2000.

[13] Y. Morie, N. Sueyasu, T. Matsumoto, T. Nanri, H. Ishihata, K. Inoue, and K. Murakami, "Optimization of MPI rank allocation considering communication timing for reducing contention," IPSJ Trans. Advanced Computing Systems, vol.48, no.SIG 13 (ACS 19), pp.192–202, Aug. 2007.

[14] T. Soga, K. Kurihara, T. Nanri, M. Kurokawa, and K. Murakami, "Dynamic optimization of load balance in MPI broadcast," IPSJ

Trans. Advanced Computing Systems, vol.1, no.3, pp.67–82, Dec. 2008 (in Japanese).

[15] P. López, J.M. Martínez, and J. Duato, "DRIL: Dynamically reduced message injection limitation mechanism for wormhole networks," Proc. 1998 International Conference on Parallel Processing, pp.535–542, 1998.

[16] M.S. Obaidat, Z.H. Al-Awwami, and M. Al-Mulhem, "A new injection limitation mechanism for wormhole networks," Comput. Commun., vol.25, no.11-12, pp.997–1008, 2002.

[17] M. Thottethodi, A.R. Lebeck, and S.S. Mukherjee, "Self-tuned congestion control for multiprocessor networks," Proc. 7th International Symposium on High-Performance Computer Architecture (HPCA '01), pp.107–118, 2001.

[18] M. Thottethodi, A.R. Lebeck, and S.S. Mukherjee, "Exploiting global knowledge to achieve self-tuned congestion control for *k*-ary *n*-cube networks," IEEE Trans. Parallel Distrib. Syst., vol.15, no.3, pp.257–272, 2004.

[19] E. Baydal, P. López, and J. Duato, "A family of mechanisms for congestion control in wormhole networks," IEEE Trans. Parallel Distrib. Syst., vol.16, no.9, pp.772–784, 2005.

[20] H. Shibamura, H. Miwa, R. Susukita, T. Hirao, Y. Ajima, I. Miyoshi, T. Shimizu, H. Ishihata, and K. Inoue, "Optimization and simulation evaluation of an all-to-all communication using packet pacing," IPSJ Trans. Advanced Computing Systems, vol.4, no.3, pp.56–65, 2011 (in Japanese).

[21] T. Yokota, K. Ootsu, and T. Ohkawa, "Relaxing heavy congestion by state propagation," Journal of Information Processing, vol.23, no.5, pp.730–743, 2015.

[22] T. Yokota, H. Matsuoka, K. Okamoto, H. Hirono, and S. Sakai, "System support functions of an interconnection network," IPSJ Journal, vol.38, no.4, pp.873–882, April 1997 (in Japanese).

[23] T. Yokota, K. Ootsu, and T. Baba, "A quantitative evaluation methodology of interconnection networks," IPSJ Trans. Advanced Computing Systems, vol.2, no.3, pp.58–70, 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 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 to 2009, he was an associate professor at Utsunomiya University. Since 2009, he has been a 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 the IEEE Computer Society.

Kanemitsu Ootsu received his B.S. and M.S. degrees from University of Tokyo in 1993 and 1995 respectively, and later he obtained his Ph.D. in Information Science and Technology from 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 include high-performance computer architecture, multi-core multithread processor architecture, mobile computing devices, binary translation, and dynamic optimization. He is a member of IPSJ, IEICE, and ISCIE.

Takeshi Ohkawa received his B.E. and M.E. and Ph.D. degrees in Electronics Engineering from Tohoku University in 1998, 2000 and 2003, respectively. He was a postdoctoral fellow at Tohoku University, where he engaged in the reconfigurable LSI and software/hardware co-design in 2003. In 2004, he joined National Institute for Advanced Industrial Science and Technology (AIST), Japan as a researcher, where he started working on distributed object technology and its application to FPGA. From 2009, he started work in start-up company TOPS systems, where he engaged in R&D of heterogeneous multi-core processor and its software platform. From 2011, he is an assistant professor in Utsunomiya University. His research interests include reconfigurable technology, software/hardware co-design, parallel and distributed architecture and component-based design technology. He is a member of ACM, IEICE, and IPSJ.