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

A Genetic Approach for Accelerating Communication Performance by Node Mapping

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

SUMMARY This paper intends to reduce duration times in typical collective communications. We introduce logical addressing system apart from the physical one and, by rearranging the logical node addresses properly, we intend to reduce communication overheads so that ideal communication is performed. One of the key issues is rearrangement of the logical addressing system. We introduce genetic algorithm (GA) as meta-heuristic solution as well as the random search strategy. Our GA-based method achieves at most 2.50 times speedup in three-traffic-pattern cases.

key words: *parallel computers, interconnection networks, collective communication, communication performance, topology mapping*

1. Introduction

Not a few parallel applications involve a set of collective communications, in which every node sends a certain amount of packets to a specified destination, and the computing performance is bound by the duration times of the collective communications. The essential problem comes from interferences between message packets that traverse at the same moment in the interconnection network. The problem remains open since we are not successful in finding a general and unique solution for eliminating unnecessary delays in communication to maximize the performance of parallel computing.

Packet interference occurs when two or more distinct packets share routing resources. To eliminate the interference, we have two options: temporal and spacial arrangement. The former option avoids simultaneous sharing of a resource by arrangement of timing. The latter one dissolves the shared status by arranging (physical) position.

We have discussed packet-scheduling by means of particle swarm optimization (PSO) [1]. This effort tries to find an (quasi-)optimal schedule that specifies the injection timing of every packet not to interfere with other packets until it reaches its destination node. This method is based on the temporal arrangement.

This paper discusses the spatio-temporal optimization of packet delivery in collective communication from a different angle, i.e., the spacial arrangement. We consider the optimization problem as a topology mapping problem. Every collective communication has its own topology in communication pattern such as transpose and perfect-shuffle. If the specified topology is embedded well on the physical structure of practical parallel machine, the application that involves the topology of collective communication runs without any communication overheads.

The fundamental point of this paper is to introduce logical addressing system apart from the physical one. By rearranging the logical node addresses to reduce communication overheads, the logical addresses show an optimal communication for a (set of) given traffic pattern(s). We further introduce genetic algorithm (GA) to find (quasi-)optimal solutions of logical addressing.

The rest of this paper is organized as follows. Section 2 formalizes the problem. Sections 3 and 4 discuss random and GA-based search methods, followed by our evaluation results in Sect. 5. Section 6 shows related work and Sect. 7 concludes this paper.

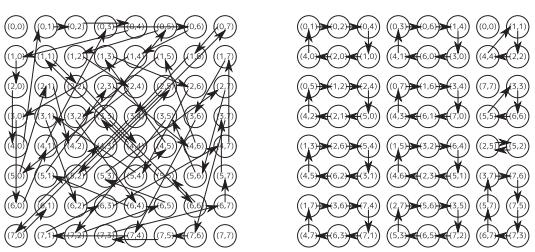
2. Topology Mapping

As the previous section briefly introduced, we discuss logical topology mapping of the applications' traffic patterns to the physical machine structure by separating logical node addresses from physical ones. For a specific pattern, we can obtain optimized topology mapping easily by hand, if the traffic consists of peer-to-peer communications. Figure 1 shows a typical example of perfect-shuffle traffic pattern.

Let a mapping function

$$\boldsymbol{x}^{\mathrm{L}} = \boldsymbol{\Gamma}(\boldsymbol{x}^{\mathrm{P}}) \tag{1}$$

map a logical address $\boldsymbol{x}^{\mathrm{L}}$ on a physical location (address) $\boldsymbol{x}^{\mathrm{P}}$. Collective communication is carried out on the basis of logical addressing. Assume that a source node $\boldsymbol{x}^{\mathrm{L}}_{\mathrm{S}}$ sends a packet to its destination $\boldsymbol{x}^{\mathrm{L}}_{\mathrm{D}}$. In this case, the packet actually starts from the physical node $\boldsymbol{x}^{\mathrm{P}}_{\mathrm{S}} = \boldsymbol{\Gamma}^{-1}(\boldsymbol{x}^{\mathrm{L}}_{\mathrm{S}})$ and it destines



(a) Physical traffic (b) Optimized logical traffic

Fig. 1 Topology mapping example of perfect-shuffle traffic

Manuscript received January 5, 2018.
Manuscript revised March 26, 2018.
Manuscript publicized September 18, 2018.

[†]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.2018PAL0002

the logical node $\boldsymbol{x}^{\mathrm{L}}_{\mathrm{D}}$ at $\boldsymbol{x}^{\mathrm{P}}_{\mathrm{D}} = \boldsymbol{\Gamma}^{-1}(\boldsymbol{x}^{\mathrm{L}}_{\mathrm{D}})$.

For simplicity in discussion, we assume two dimensional torus topology for the physical addressing system. An actual mapping function can be represented as a two-dimensional matrix:

$$\boldsymbol{\Gamma} = \begin{pmatrix} \boldsymbol{x}^{\mathrm{L}}_{0,0} & \boldsymbol{x}^{\mathrm{L}}_{0,1} & \cdots & \boldsymbol{x}^{\mathrm{L}}_{0,\mathrm{n-1}} \\ \boldsymbol{x}^{\mathrm{L}}_{1,0} & \boldsymbol{x}^{\mathrm{L}}_{1,1} & \cdots & \boldsymbol{x}^{\mathrm{L}}_{1,\mathrm{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}^{\mathrm{L}}_{\mathrm{n-1},0} & \boldsymbol{x}^{\mathrm{L}}_{\mathrm{n-1},1} & \cdots & \boldsymbol{x}^{\mathrm{L}}_{\mathrm{n-1},\mathrm{n-1}} \end{pmatrix} \tag{2}$$

where $\boldsymbol{x}^{\mathrm{L}}_{\mathrm{i,j}} = (x_{i,j}, y_{i,j})$ and $\boldsymbol{x}^{\mathrm{L}}_{\mathrm{i,j}} \neq \boldsymbol{x}^{\mathrm{L}}_{\mathrm{l,m}}$ $(i \neq l \text{ or } j \neq m)$.

3. Random Search

According to the discussions in the previous section, the problem in this paper is formalized to find an optimal (or near-optimal) mapping function Eq. (1). The mapping function can be denoted in a matrix form as Eq. (2) shows. By generating mapping matrices randomly, we can expect that some of the generated matrices perform preferable communications.

We should discuss the search space in the problem before showing evaluation results in Sect. 5. As shown in Eq. (2), in an $n \times n$ system, the size of search space is possible number of permutation of n^2 items, i.e., $O(n^2!)$ that approximates $O((\frac{n^2}{e})^{n^2})$. For example, an 8×8 system should search a solution in $64! \approx 1.27\text{e+}89$ possible combinations.

4. Genetic Approach

To overcome the difficulty in large search space, we introduce Genetic Algorithm (GA). In this paper, we simply represent a mapping matrix as the representation of a gene. GA, in general, has variety of gene operations that include crossover and mutation for wide *divergence* in the search space. However, we omit crossover operations by two or more genes in this paper, since no duplicated members are not allowed in any mapping matrices and the crossover operation is not natural in our gene representation. Thus, we basically use self-reproduction operations (mutation).

In this paper, we discuss the GA operations from two orthogonal angles: mutation methods and surviving methods.

4.1 Mutation Methods

Due to the strong restriction of duplicated entries, the mutation operation is based on swapping of two members in the mapping matrix. We introduce the following variants of swapping operations.

Random swap selects two distinct members in physical address and swap their logical addresses.

Line swap selects two lines of members in a specified length and swaps member-by-member.

Box swap selects two box-shaped regions in the same aspects and swaps their members.

(a) Select candidate cell (b) Swap the selected cell

Fig. 2 Neighboring swap

Table 1 Variants in the mutation operations

sym-bol	percentage of mutation operations: random swap	line swap	box swap	neighboring swap	random-ize	% survivors	note
S00	37.5	25.0	12.5	0.0	25.0	25.0	
S01	12.5	12.5	12.5	37.5	25.0	25.0	
S02	20.0	20.0	20.0	20.0	20.0	25.0	
S03	0.0	0.0	0.0	75.0	25.0	25.0	$r = N_p$
S04	0.0	0.0	0.0	75.0	25.0	25.0	r : randomly selected
S05	100.0	0.0	0.0	0.0	0.0	25.0	
S10	0.0	0.0	0.0	100.0	0.0	25.0	$r = 2$
S11	0.0	0.0	0.0	100.0	0.0	50.0	$r = 2$
S12	0.0	0.0	0.0	100.0	0.0	12.5	$r = 2$
S20	0.0	0.0	0.0	100.0	0.0	25.0	$r = N_p$
S21	0.0	0.0	0.0	100.0	0.0	50.0	$r = N_p$
S22	0.0	0.0	0.0	100.0	0.0	12.5	$r = N_p$
S99	0.0	0.0	0.0	0.0	100.0	0.0	random search

Neighboring swap selects a source-destination pair that has longer distance than a specified radius r and it further selects near member within the radius r. The method swaps the longer destination with the near (i.e., neighboring) member. Figure 2 illustrates the neighboring swap method.

The optimization process intends to eliminate interferences of packets and it works to minimize (physical) distances between source-destination node pairs. Thus, during an optimization process, some node pairs are placed adjacently, where they are locally optimal. Line and box swap operations intend further optimization, maintaining the locally optimal structure. These operations allow 90-degree rotation and mirror image in swapping operation.

Table 1 summarizes the mutation operations used in this paper. In this table, N_p shows the number of traffic patterns. We use 13 variants of mutation operations in this paper, which are expressed as S00 to S99. Operations from S00 to S05 intend to evaluate the effects of swap operations, where one-fourth of genes survive. Operations from S10 to S22 shows the variants of the neighboring swap. S1* and S2* differ on the radius r that is represented in Fig. 2. Operations S*0, S*1, and S*2 differ on the surviving strategy (i.e, ratio of survivors). We can expect that the small survivor ratio will show a steep characteristic in search, but it will fall into local-minima. Operation S99 refers to the non-GA method.

We can expect that the logical node arrangement for a single traffic pattern is not a difficult task. Thus, we basically assume multiple traffic patterns for a single arrangement configuration, as well as a single traffic pattern. We denote *two(three)-traffic-pattern* when two (three) traffic patterns are applied during the optimization process. Even when the optimization process find an optimal solution for a specific traffic pattern, the solution (i.e., logical arrangement) is

not always optimal in other traffic patterns. Thus, although the neighboring swap operation forces arrangement toward a specific traffic pattern, the operation is not always successful.

4.2 Surviving Methods

A surviving method specifies how the surviving genes are selected. In the ordinary GA application, highly-ranked genes survive and generate their descendant(s). Our method also follows the principle, however, we should discuss selection method of *ranking*.

We use duration time to quantitatively represent performance in collective communication. Networks have strong non-linear characteristics and, thus, the duration time does not necessarily shows the performance linearly.

For simple discussion, here we assume to source-destination pairs. Each of pairs has its own routing paths. If the paths share routing resources (physical links, for example), exclusive use of the shared resource delays the corresponding packet transfer and results in performance degradation. As a simple assumption, when the two packets fully interfere, the resulting duration time becomes twice. Furthermore, since the duration time is measured as the worst-case communication time, the enlarged duration time does not directly show the number of interferences in the network. In other words, the duration time is not proportional to the number of interferences.

If any combinations of source-destination pairs do not share routing resources, communications are performed at full speed without any interferences. For general conditions, we do not find appropriate metrics to represent the level of interferences, however, we use average number of hops (avg.hop) in all of the possible combinations of source-destination pairs. Small avg.hop suggests that we can expect small possibilities of interferences. If avg.hop equals to 1, it means that all source-destination pairs are placed at adjacent addresses.

In this paper, we use the following surviving methods.

Duration first. (sc) Genes are sorted by the duration-time order. If the duration is same, avg.hop is used.

Avg.hop first. (ah) Genes are sorted by the avg.hop order. If the avg.hop is same, duration is used.

Multiplication of duration and avg.hops. (ml) Genes are sorted by multiplied value of duration time and avg.hop.

After selecting the survivors, our methods generate new gene(s) from each of the survivors according to the mutation strategy (Table 1). For example, S02 strategy selects the top-25 percents of genes for survivals. The survivors remain alive in the next generation and the 75-percent non-survivors are substituted by the mutations of the survivors. In this case (S02), each survivor generates three mutants in the next generation.

5. Evaluation

5.1 Evaluation Environment and Method

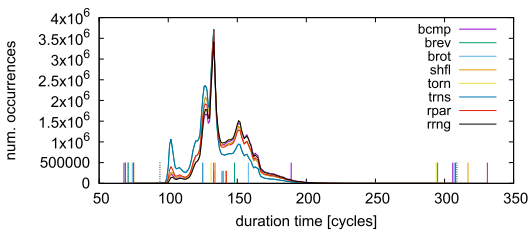
We implemented an evaluation platform for node mapping, which is extended from our interconnection network simulator that achieves considerable speed up by the cellular automata principle [2]. Although the evaluation platform is based on a fast simulator, GA operations are time consuming and we use a small-size (8×8) 2D-torus network.

Packet length is 8 [flits] and four packets are transferred in each collective communication session. Routing algorithm is deterministic dimension order and three virtual channels are used. Duration time is measured from the beginning of the communication session to its completion. In the GA methods, we use 1,000 genes and run 1,000 generations. Random search runs $1{,}000 \times 1{,}000$ cases. We use eight traffic patterns: bit-complement (bcmp), bit-reversal (brev), bit-rotation (brot), perfect shuffle (shfl), tornado (torn), transpose (trns), random pair (rpar), and random ring (rrng). In the tornado traffic, each node sends packets to the node whose distance is $(n/2)$. Random pair selects two nodes randomly. Random ring forms an $n \times n$-node unidirectional ring in which all of the nodes are employed in the ring. GA operations are applied to all possible combinations of traffic patterns, and each combination has ten runs and average values are used.

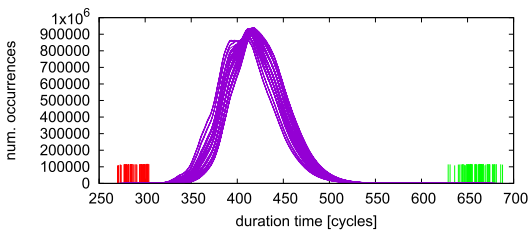
5.2 Random Search Results

Figure 3 shows random search results. In this figure, vertical lines show the minimum and maximum duration times and, in Fig. 3 (a), vertical short lines at the intermittent part show the average duration times whereas long lines show minimal and maximal duration times in physical addressing.

According to the evaluation conditions, the minimal



(a) Single-traffic-pattern results



(b) Three-traffic-pattern results

Fig. 3 Random search results

Table 2 Best duration results (two-traffic-pattern)

traffic pattern	best-case dur.	best-case gen.	phys. dur.	top mutation-sorting combination
bcmp-brev	68.2	70.0	283	S20-ah
bcmp-brot	98.5	613.3	242	S21-sc
bcmp-shfl	100.9	491.2	252	S10-ah
bcmp-torn	76.3	386.8	228	S22-ah
bcmp-trns	68.0	170.7	225	S20-ah, S03-ah
brev-brot	69.0	110.4	337	S20-ah, S22-sc
brev-shfl	69.0	204.2	347	S21-ah, S22-ah
brev-torn	93.3	548.9	323	S10-ah
brev-trns	68.0	222.7	320	S11-ah, S03-ah
brot-shfl	70.0	14.9	306	S22-ah, S22-ml
brot-torn	129.0	624.4	282	S12-ah
brot-trns	83.4	619.8	279	S11-ml
shfl-torn	129.8	672.1	292	S03-ah
shfl-trns	82.6	427.7	289	S12-ah
torn-trns	102.5	485.8	265	S12-ah

duration time is 34 [cycles] that includes injection and reception time (one cycle for each) and 8 [flits/packet] × 4 [packets] = 32 [flits=cycles] transfer time. As Fig. 3 (a) shows, performance of the random search is far from the theoretical best. Furthermore, curves in Fig. 3 (a) shows some local peaks that corresponds to the packet interferences as discussed in Sect. 4.2.

In the three-traffic-pattern case, average duration time of the three patterns is 426.3 [cycles] in the physical addressing. Random search achieves average of 287.6 [cycles] (i.e., 1.48 times speedup) after 1 million random trials.

5.3 GA Results

Table 2 shows the average values of the best duration times and the achieved generations in two-traffic-pattern cases. In this table, duration time in physical addressing is also shown (denoted as "phys. dur."). Our GA-based method achieves considerable speed-up, at most 5.03 times in the brev-shfl traffic. Degree of speed-up depends on the topological characteristics in the traffic patterns: topologies that contain cycles have long duration time. For example, bcmp-brev and bcmp-trns patterns do not contain cycles and they achieve shortest (i.e., fully optimized) duration times. On the other hand, our method fails the shortest duration time in bcmp-brot and bcmp-shfl cases, since brot and shfl topologies contain cycles.

Table 3 shows the results for mutation operations (in Table 1) in three-traffic-pattern cases. This table shows average values with respect to the sorting order. As discussed in Sect. 4.2, duration time based sorting does not necessarily achieve good performance due to the strong non-linearity in the network performance characteristics. Average hop based sorting sometimes drops hopeful genes that have good duration time scores, however, the sorting method achieves reasonable performance. Multiplied value based sorting achieves the best performance, since the sorting method can distinguish promising genes that have good duration time or average number of hops. The average duration time in physical addressing is 426.3 [cycles], thus, our GA-based method achieves at most 2.50 times speed-up.

Table 3 GA results of three-traffic-pattern

mutation	sorting order [cycles] duration	avg.hop	mult
S00	224.1	211.9	207.5
S01	210.2	191.4	189.5
S02	209.5	189.2	186.7
S03	193.6	175.2	170.2
S04	230.5	221.6	217.3
S05	225.7	216.0	211.6
S10	199.3	180.9	177.8
S11	200.8	180.2	178.0
S12	188.7	174.2	172.4
S20	183.5	174.7	171.4
S21	185.7	174.8	172.0
S22	184.8	177.4	171.1
S99	301.4	301.6	301.3

5.4 Qualitative Comparison with Temporal Arrangement

As described in Sect. 1, we have discussed temporal arrangement in our prior work [1] and proposed an optimal packet-scheduling method based on PSO. The temporal optimization does not handle logical addressing and it only arranges packet injection timing at every node. Thus, when two or more packets share the same resource, the method cannot reduce the duration time less than the multiplicity degree of packets.

On the other hand, we can expect that the proposed spacial optimization method performs better than the temporal one, since the logical addressing eliminates the sharing states of packets. Theoretically minimal duration times are multiple of 34 [cycles] by the number of traffic patterns in the evaluation condition in this paper. Actually, as shown in Table 2, many of traffic patterns nearly achieve the theoretical minimum (in bcmp-brev, bcmp-trns, brev-brot, brev-shfl, brev-trns, and brot-shfl patterns).

6. Related Work

Parallel processing research has the old problem in *topology embedding*, for example, embedding tree structure in a hypercube topology [3]. This formalizes the problem as mapping a logical structure to a physical one. If the mapping is successful and communications on the logical structure are effective, applications on the specific logical structure run smoothly and achieve high performance. Topology mapping shares the core idea with the embedding approach.

We can find similar approach in *application mapping* as literature [4]–[9] shows. A typical example is embedding a specific application, such as encoding process of moving pictures (MPEG), on a multicore/many-core architecture with a specific NoC. This also shares the idea of topology mapping, however, our method assumes one or more collective communication traffic patterns.

As a different viewpoint from mapping (or embedding), randomization offers an alternative approach to improve communication performance under a specific traffic pattern. For example, transpose traffic has a quite regular traffic pattern that leads the traffic to a heavily concentrated situation, since many communication paths share some limited portions in the system. Such concentrated communica-

tion causes severe congestion that drastically degrades the network performance.

This suggests that introducing some levels of randomization (or introducing irregularity) relaxes the concentrated situation so that it can increase the network performance. Literature [10], [11] shows randomization effort in the network topology. Alternative idea is *oblivious* routing [12].

7. Conclusion

This paper aims at improving communication performance in collective communication. We introduced logical addressing system apart from the physical one. When the logical addresses are mapped appropriately for objective traffic patterns, collective communications are performed ideally without any overheads.

To obtain the appropriate mapping solutions, we introduced genetic algorithm (GA) as a hopeful meta-heuristics as well as the random search method. Evaluation results show that our GA-based method achieves at most 5.03 and 2.50 times speedups in two- and three-traffic-pattern cases, respectively, whereas the random search achieves 1.48 times speedup.

Acknowledgments

This work was partly supported by JSPS KAKENHI Grant Numbers 15K00068, 16K00068, and 17K00072.

References

[1] T. Yokota, K. Ootsu, and T. Ohkawa, "A static packet scheduling approach for fast collective communication by using PSO," IEICE Trans. Inf. & Syst., vol.E100-D, no.12, pp.2781–2795, Dec. 2017. DOI: 10.1587/transinf.2017PAP0015.

[2] T. Yokota, K. Ootsu, and T. Ohkawa, "Large-scale interconnection network simulation methods based on cellular automata," Proc. 5th International Symposium on Computing and Networking (CANDAR'17), pp.58–67, Nov. 2017. DOI: 10.1109/CANDAR.2017.52.

[3] A.Y. Wu, "Embedding of tree networks into hypercubes," Journal of Parallel and Distributed Computing, vol.2, no.3, pp.238–249, Aug. 1985. DOI: 10.1016/0743-7315(85)90026-7.

[4] J. Hu and R. Marculescu, "Energy- and performance-aware mapping for regular NoC architectures," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol.24, no.4, pp.551–562, April 2005. DOI: 10.1109/TCAD.2005.844106.

[5] L. Bononi, N. Concer, M. Grammatikakis, M. Coppola, and R. Locatelli, "NoC topologies exploration based on mapping and simulation models," Proc. 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007), pp.543–546, Aug. 2007. DOI: 10.1109/DSD.2007.4341521.

[6] N.F. Butt, M. Chowdhury, and R. Boutaba, "Topology-awareness and reoptimization mechanism for virtual network embedding," Lecture Notes in Computer Science, vol.6091, pp.27–39, 2010. DOI: 10.1007/978-3-642-12963-6_3.

[7] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang, "Virtual network embedding through topology-aware node ranking," ACM SIGCOMM Computer Communication Review, vol.41, no.2, pp.38–47, April 2011.

[8] T. Hoefler and M. Snir, "Generic topology mapping strategies for large-scale parallel architectures," Proceedings of the International Conference on Supercomputing, ICS '11, New York, NY, USA, pp.75–84, ACM, 2011. DOI: 10.1145/1995896.1995909.

[9] P.K. Sahu and S. Chattopadhyay, "A survey on application mapping strategies for network-on-chip design," Journal of Systems Architecture, vol.59, no.1, pp.60–76, Jan. 2013. DOI: 10.1016/j.sysarc.2012.10.004.

[10] M. Koibuchi, H. Matsutani, H. Amano, D.F. Hsu, and H. Casanova, "A case for random shortcut topologies for HPC interconnects," SIGARCH Comput. Archit. News, vol.40, no.3, pp.177–188, June 2012. DOI:10.1145/2366231.2337179.

[11] I. Fujiwara, M. Koibuchi, H. Matsutani, and H. Casanova, "Swap-and-randomize: A method for building low-latency HPC interconnects," IEEE Trans. Parallel Distrib. Syst., vol.26, no.7, pp.2051–2060, July 2015. DOI: 10.1109/TPDS.2014.2340863.

[12] A. Singh, W.J. Dally, A.K. Gupta, and B. Towles, "Goal: A load-balanced adaptive routing algorithm for torus networks," Proceedings of the 30th Annual International Symposium on Computer Architecture, ISCA '03, New York, NY, USA, pp.194–205, ACM, 2003. DOI: 10.1109/isca.2003.1207000.