Exploring Time-space Trade-off for Application Mapping onto 3-D Torus NoCs

Yao Hu, Michihiro Koibuchi
National Institute of Informatics
2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, Japan 101-8430
{huyao, koibuchi}@nii.ac.jp

***Abstract*—One application usually has many parallel tasks running on multiple processing cores which communicate with each other on a many-core chip. Traditionally, the tasks are mapped onto a regular topology of network-on-chip (NoC) with nearby processing cores to reduce the network distances. In this case, fragmentation of unused processing cores may occur when receiving a new incoming application on a chip. In this study, we assume that each application has to be executed on a pre-fixed network topology on a many-core chip with 3-D torus NoC. To improve the system utilization, i.e. reducing a number of unused processing cores, we allow to use non-adjacent processing cores for an application mapping, which form a pre-fixed network topology. We evaluate the time-space trade-off during node allocation with different mapping dilations for the purpose of improving job scheduling abilities. Evaluation results show that, for a large compound workload of NAS Parallel Benchmarks (NPB) applications, the proposed mapping can reduce up to 6% of turnaround time when compared with the regular topology mapping on a large 3-D torus NoC.**

***Index Terms*—Network-on-Chip (NoC), topology embedding, interconnection network, job mapping**

I. Introduction

A number of highly parallel applications simultaneously run on multiple processing cores which communicate with each other on a many-core chip. The number of processing cores on a chip and the scale of applications have been growing exponentially in the last decade. Mapping virtual inter-*process* topologies to physical inter-*processor* topologies is one of the important issues for NoCs. Traditionally, each application is allocated to an unused processing-core set connected by a regular *guest* network topology. By dispatching communicating tasks to nearby processing cores, the communication latency and network contention could be minimized.

One disadvantage of the traditional topology mapping on regular topologies is that, when the instantaneous workload becomes large, the system utilization of processing cores would decrease. In this case, a number of applications have to wait for the release of occupied adjacent processing cores, because they cannot be mapped on non-adjacent processing cores even when they are available on a chip. In other words, there are many cases where the system utilization is low while some available processing cores are not allowable for user jobs since they are disjoint. It is a time-space trade-off that the regular topology mapping tends to minimize the execution time but wasting the space of non-adjacent processing cores.

In this study, we investigate the job scheduling performance with different mapping dilations on a chip. Here, the *dilation* in the topology embedding refers to the length (in number of path hops) of the shortest path between embedded vertices of an edge. Concretely, we investigate a new mapping algorithm with incrementally increased dilations of topology embedding. Considering the heavy node fragmentation due to a large dilation, the first priority is always given to regular (dilation-1) mappings to make the best use of adjacent available processing cores on the system. We also take into account the topology mapping on non-adjacent processing cores whose embedding dilation is larger than one. In this way, instead of waiting in the queue, user jobs can be still immediately dispatched to the 1-chip system even though the available processing cores cannot form regular topologies to accelerate the execution times. Therefore, the fragmentation of unused processing cores caused by previous allocation can be used for constructing a random topology for next incoming application so that they can be simultaneously executed on the system. Generally, when the *dilation* increases, its topology mapping becomes easier while imposing larger communication latency between processing cores.

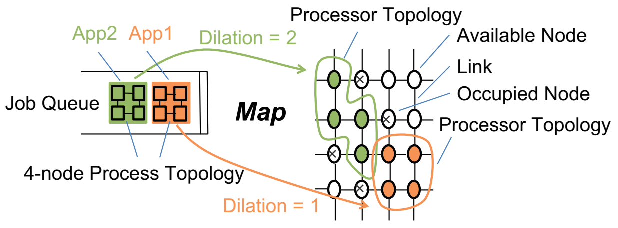

Figure 1. The job mapping with different embedding dilations.

Figure 1 depicts an example of topology mapping with different embedding dilations. For the regular topology mapping (dilation-1), the application can be dispatched to the adjacent available processing cores that form a regular topology, e.g., 2-D mesh, based on its process topology. For the topology mapping with dilation-2, the application can be still mapped on the available processing cores even though they do not form a regular 2-D mesh. It seems that, compared the regular topology mapping (dilation-1), the dilation-n (n>1) topology mapping can simultaneously accommodate a larger number of applications. In this work, our main objective is to show that a workload of diverse applications can be better supported with a certain time-space trade-off rather than regular topology mapping on a many-core chip interconnected with 3-D torus NoC.

II. Related Work

There are a large number of researches on application-specific NoC design [1] that optimizes network topology, routing, and floor layout for target traffic patterns. The objective is typically to improve the energy efficiency or the application execution-time. By contrast, this study assumes to use 3-D torus NoC, and we consider the case where applications are mapped onto 3-D torus.

There are some researches on mapping and scheduling on NoCs for heat dissipation problem and power efficiency [2]. By contrast, in this study, our objective is to minimize the makespan of each job on a many-core chip under the condition that the network topology of processes on every job is pre-fixed on NoC.

III. Large Dilation Topology Mapping

Assume a mapping F from the vertices of a graph G to the vertices of a (larger) graph H. Given an edge in G between two

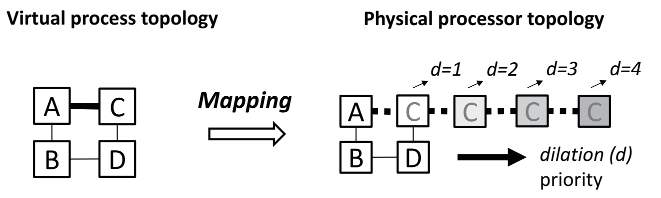

Figure 2. A new mapping algorithm with incrementally increased dilation of topology mapping.

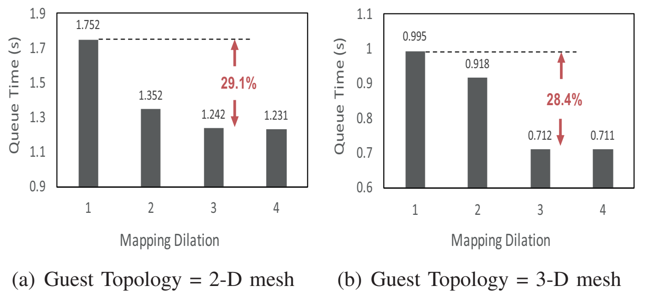

Figure 3. The average queue time of dispatched applications over 3-D torus topology.

vertices v and w, its *dilation* is the length of the shortest path between $F(v)$ and $F(w)$ in number of hops in H. The *dilation* for a topology mapping is the maximum value of all the edge dilations. A dilation-1 mapping does not expand any edge of G in H, and a dilation-n mapping leads to the maximum n hops for any edge of G in H [3].

As shown in Fig. 2, a dilation-1 topology embedding is first considered if such processing cores are not occupied by another application. This step is equivalent to a regular topology mapping. If such a dilation-1 topology mapping cannot be found on a many-core chip, then we consider a dilation-2 topology mapping for application execution. If a dilation-2 topology mapping is still not found, we consider a dilation-n (n >2) topology mapping for application execution.

IV. Evaluation

We developed an HPC simulator written in Python 2.7 in a machine with Intel i7-6500U (2.50GHz) CPU and 16GiB Memory to model job scheduling. We use a common approach to model parallel "rigid" applications, which refer to ones that use a fixed number of processing cores during runtime. We evaluate the performance of a large-scale NoC with a workload composed of various NPB applications [4] including FT (Fast Fourier), IS (Integer Sort), CG (Conjugate Gradient), BT (Block Tri), SP (Scalar Penta) and MG (Multi-Grid), and MM (Matrix Multiplication). To acquire execution times (simulation cycles) for the NPB applications running on 2-D and 3-D guest topologies, we first got a series of simulation results by the event driven simulator SimGrid [5].

The whole network contains 3072 processing cores on a chip which are connected by a 3-D torus ($16 \times 16 \times 12$) topology. We generate $a = 2000$ applications with random arrival timings for a *Poisson* process with $\lambda = \frac{a}{1000}$. For each application, we use regular mesh (dilation-1) mapping as first priority, and set the maximum mapping dilation. If the topology mapping with the maximum dilation is still not found, the application has to wait for the release of occupied processing cores until a required guest topology is found where the mapping dilation is less than or equal to the maximum dilation.

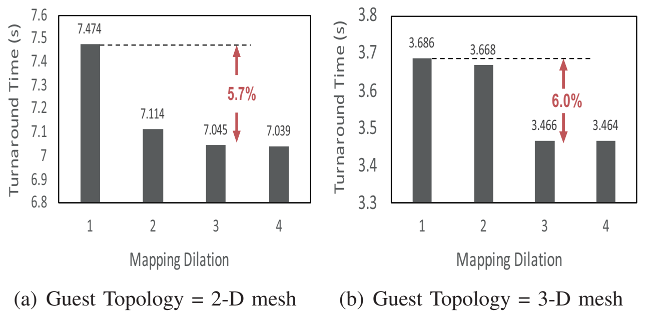

Figure 4. The average turnaround time of dispatched applications over 3-D torus topology.

As shown in Fig. 3 and Fig. 4, the dilation-n (n>1) topology mapping performs better job scheduling performance than the regular (dilation-1) topology mapping on 2-D or 3-D mesh. For instance, when the guest topology is 3-D mesh, the dilation-3 topology mapping saves 28.4% of queue time, and still decreases 6.0% of turnaround time. Considering the algorithm complexity and execution cost, the dilation-3 topology mapping comparatively is an appropriate choice because it presents better performance than the dilation-2 topology mapping while shows very marginal inferiority to the dilation-4 topology mapping.

V. Conclusion

Efficient mapping of application to network topology gains importance as the number of processing cores on a chip grows to petascale and beyond.

In this study, we assume that each application has to be executed on a pre-fixed network topology on a many-core chip with 3-D torus NoC. To improve the system utilization, i.e. reducing a number of unused processing cores, we consider the case for allowing to use the dilation-n processor topologies for applications on a many-core chip. We evaluate the time-space trade-off during topology mapping so that applications can be rapidly dispatched to processing cores that do not form a regular topology while not sacrificing much execution time.

Evaluation results showed that the dilation-n (n>1) topology mapping performs better job scheduling performance than the regular (dilation-1) topology mapping over 3-D torus. Especially, the dilation-3 topology mapping is recommended for its high cost-performance ratio. This avoids optimization of given applications to specified network topologies, and shows a great long-term potential for supporting parallel applications in job scheduling abilities.

Acknowledgments

This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

[1] W. H. Ho and T. M. Pinkston, "A design methodology for efficient application-specific on-chip interconnects," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 2, pp. 174–190, 2006.
[2] C. Addo-Quaye, "Thermal-aware mapping and placement for 3-d noc designs," in *Proceedings 2005 IEEE International SOC Conference*, Sep. 2005, pp. 25–28.
[3] I. Fujiwara, M. Koibuchi, T. Ozaki, H. Matsutani, and H. Casanova, "Augmenting low-latency hpc network with free-space optical links," in *21st International Conference on High-Performance Computer Architecture (HPCA)*, Feb. 2015, pp. 390–401.
[4] The NAS Parallel Benchmarks, http://www.nas.nasa.gov/Software/NPB/.
[5] "Simgrid: Versatile simulation of distributed systems," http://simgrid.gforge.inria.fr/.