Cabinet Layout Optimization of Supercomputer Topologies for Shorter Cable Length

Ikki Fujiwara, Michihiro Koibuchi
National Institute of Informatics / JST
2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, JAPAN 101-8430
{ikki, koibuchi}@nii.ac.jp

Henri Casanova
University of Hawai‘i at Manoa
1680 East-West Road, Honolulu, HI 96822
henric@hawaii.edu

Abstract—**As the scales of supercomputers increase total cable length becomes enormous, e.g., up to thousands of kilometers. Recent high-radix switches with dozens of ports make switch layout and system packaging more complex. In this study, we study the optimization of the physical layout of topologies of switches on a machine room floor with the goal of reducing cable length. For a given topology, using graph clustering algorithms, we group switches logically into cabinets so that the number of inter-cabinet cables is small. Then, we map the cabinets onto a physical floor space so as to minimize total cable length. This is done by modeling and optimizing the mapping problem as a facility location problem. Our evaluation results show that, when compared to standard clustering/mapping approaches and for popular network topologies, our clustering approach can reduce the number of inter-cabinet cables by up to 40.3% and our mapping approach can reduce the inter-rack cable length by up to 39.6%.**

Index Terms—**Topology, cabinet layout, interconnection networks, high performance computing, high-radix switches**

I. Introduction

As the scale of supercomputers increases, total cable length can reach enormous proportions. For example, the first generation Earth Simulator required over two thousand kilometers [1], while the K-computer requires one thousand kilometers [2]. Cable length directly affects the cabling medium. If the cable length for inter-cabinet connections is one or more orders of magnitude longer than the intra-cabinet connection, then inter-cabinet cables must be optical. In case of InfiniBand, typical maximum length of passive copper is 10m, that of active copper is 40m, connectorized copper is 30m, and embedded optical is 100m [3]. In addition, as the number of inter-cabinet cables increases the number of backup cables proportionally increases. These backup cables must be installed at deployment time since adding cables once the platform is deployed is costly.

Since traditional topologies used in supercomputers often exhibit both highly regular structures and low degree (e.g., k-ary n-cubes such as the 3D Torus in BlueGene/L [4]), they naturally fit into a simple physical cabinet layout. High-degree topologies, however, have become not only possible but also attractive, due to the availability of high-radix switches with dozens of ports (e.g., YARC routers for folded-Clos or Fat-tree networks [5]). With these topologies, optimal physical layouts are no longer intuitive and system designers are now faced with the difficult task of mapping switches to a physical layout so as to reduce total cable length. Furthermore, system designers need to carefully select each dimension of the network topology, as the impact on total cable length can be large. For example, in the 6D torus “Tofu” network in the K-computer, three dimensions are fixed at $2 \times 3 \times 2$, but the remaining three dimensions are scalable and must be chosen carefully [2]. Note that a drawback of almost all popular topologies is that the network size is strictly defined by topology dimensions (e.g., k^n vertices in a k-ary n-cube topology), even though the scale of a supercomputer should ideally be determined solely based on electrical power budget, surface area, and cost.

In this context, given a topology, we study the optimization of the physical cabinet layout in a view to minimizing the number and total length of inter-cabinet cables between switches. Note that intra-cabinet cables are usually short (e.g., 2m [6]) and constant regardless of the topology. Our approach consists in aggregating switches into groups that correspond to the cabinet size using graph clustering techniques so as to reduce the number of links between groups. We then map these groups onto a physical floorplan by framing the problem and solving it as a facility location problem. Our goal in this work is to reduce total inter-cabinet cable length not only for low-radix, layout-friendly topologies but also for more challenging high-radix topologies. Our main contributions and findings are as follows:

- Among several candidate methods, the Girvan-Newman clustering method [7] is effective for grouping switches and reduces the total number of inter-cabinet cables up to 40.3% when compared to the popular straightforward method used in conventional mesh, torus, and hypercube topologies. Incidentally, the Ward clustering method [8] leads to similar results.
- Clustering is particularly challenging for the recently proposed random ring topologies (consisting of a ring with additional random chordal shortcuts) because these topologies exhibit little regularity and locality.
- A scheme based on Simulated Annealing is effective

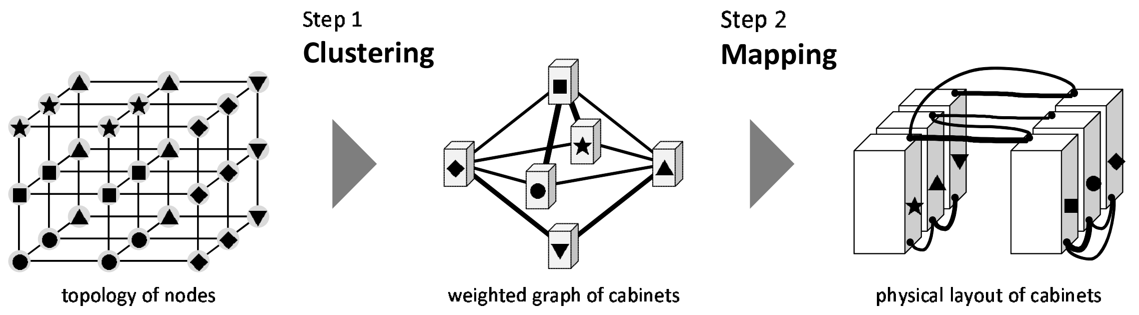

Figure 1. Approach for shortening total cable length. The thickness of a line indicates the number of individual network cables.

for mapping each switch cabinet to a physical floorplan, improving total inter-cabinet cable length by up for 39.6% when compared to traditional approaches for popular topologies.
- Both our clustering and mapping approaches scale up to at least the size of the largest existing supercomputers [2], maintaining their relative advantage over traditional approaches as the network size increases.

The rest of this paper is organized as follows. Related work is discussed in Section II. Preliminaries, including our assumption, target topologies, and an overview of our approach, are given in Section III. Our clustering approach to group switches into cabinets is described in Section IV, and our approach for mapping these cabinets onto a physical floorplan is described in Section V. Finally, a summary of our findings and of their impact is provided in Section VI.

II. Related Work

In this section we review high-radix supercomputer topologies, graph analysis techniques, and facility location problems.

A. High-radix Topologies

A few topologies are traditionally used to interconnect compute nodes in most HPC systems, and these topologies can be used to interconnect high-radix switches. In *direct topologies*, each switch connects directly to compute nodes as well as to other switches. Popular direct topologies include k-ary n-cubes, with a degree of $2n$, leading to tori, meshes, and hypercubes (including several variations of the hypercube such as folded hypercubes and twisted hypercubes). These topologies achieves well-known trade-offs between degree and diameter. Recently, [9] has proposed the use of random topologies (namely a ring with randomly generated chordal links), which achieves good such trade-offs. All these topologies are *regular*, meaning that all switches have the same degree as each switch is connected to the same number of other switches.

Indirect topologies, i.e., topologies in which some switches are connected only to switches, have also been proposed. They have low diameter at the expense of larger numbers of switches when compared to direct topologies. The best known indirect topologies are Fat trees, Clos network and related multi-stage interconnection networks such as the Omega and the Butterfly networks. A popular option is (p, q, r) Fat trees, with a degree of $p + q$, where p is the number of upward connections, q is the number of downward connections, and r is the number of tree levels. For large-scale HPC systems, the network layout has a high impact on network cost, especially because longer wires must be optical if high bandwidth is required [3]. Consequently, several variations of these topologies, such as the flattened butterfly [6], have been proposed as a way to improve cost effectiveness. More recently, the Dragonfly topology [10] uses a group of routers as a virtual router to reduce the wire length in the context of high-radix topologies, which is done by distinguishing inter-cabinet and intra-group networks.

B. Graph Analysis

Graph analysis techniques, which extract valuable knowledge out of a large graph's structure, have become increasingly used in conjunction with the popularization of social networking services [11]. These techniques are applied to analyze not only human networks but also computer networks like the Internet and the Web. In this work we attempt to apply such techniques [12], [13], [8], [7] because recently proposed high-radix topologies are not amenable to intuitive grouping of switches. To the best of our knowledge, this research is the first attempt to apply graph analysis techniques to cabinet-level design of the HPC systems.

C. Facility Location Problem

It is important in various industries for a corporation to decide the locations of factories and other facilities so as to minimize transportation costs between them. This "facility location problem" has been studied in operations research since the 1960s [14]. Since it is is NP-hard, metaheuristics-based techniques have been developed and used to compute good solutions in practice. Solutions to facility location problems have been used in the computer industry for computer chip design. To the best of our knowledge, this work is the first to apply the facility location problem to the design of physical network layouts for HPC systems.

III. Preliminaries

A. Assumption

We focus on a supercomputer that is to be newly deployed on a site (e.g., in a room, building). Upgrade of an existing system is out of scope of this paper, but our work should be adaptable to that scenario. The time to compute a layout should be at most a few hours so that our approach can be used repeatedly while designing the system, with system sizes ranging up to several thousands of nodes. In our context, a "node" comprises a switch and several compute nodes (e.g., a 128-port switch with 100 attached compute nodes and 28 ports left to connect to other switches). In all that follows, we omit these details and simply treat a node as an indivisible unit. We can now define the following terminology: A *link* is a connection between two nodes; a *cable* is the physical realization of a link; the *degree* of a node is the number of links at that node; and a *cabinet size* is the maximum number of nodes that can fit inside a physical cabinet.

B. Network Topologies

Our goal is to develop an approach that is effective not only for low-radix, layout-friendly topologies but also for high-radix topologies for which there is no clear or intuitive layout. We consider four direct network topologies: k-ary n-mesh, k-ary n-torus, n-dimensional hypercube, and n-degree Random Ring. The Random Ring, which consist of a ring with $n - 2$ additional random shortcut links from/to each node, was shown in [9] to achieve low diameter and low latency. However, among our four candidate topology, it is the least layout-friendly.

C. Approach Overview

Our approach consists of two optimization steps: (1) switch clustering and (2) cabinet mapping. Figure 1 depicts an overview of our approach. An instance of a mesh topology of 27 nodes is shown on the left. In the first step we cluster the nodes to obtain a weighted graph of cabinets, as shown in the center. The weights indicate the number of links between the cabinets. In the second step, we compute the physical layout of the cabinets on the floorplan, as shown on the right. The following two sections give the details of these two steps.

IV. Clustering

In this section we describe how we convert a topology of nodes into a weighted graph of cabinets.

A. Modeling

A topology is an unweighted undirected simple graph in which vertices represent nodes. Grouping nodes together in a cabinet is equivalent to contracting the vertices — in other words, converting the graph into a weighted undirected simple graph in which vertices represent cabinets — where loop edges are removed and multiedges are converted to weighted simple edges. Our approach attempts to group densely-connected nodes together in the same cabinet so that the number of inter-cabinet cables is minimized. We call this operation "clustering."

B. Methods

We develop hierarchical clustering methods based on those used for data mining, modifying them so that the resulting cluster size does not exceed the specified cabinet size. Hierarchical clustering methods can be classified into aggregative methods and divisive methods. The former start from a state in which each cluster contains only one node and recursively aggregates pairs of clusters. The latter, instead, start from a single cluster that contains all the nodes and recursively divides each cluster into two distinct clusters. We use the two following clustering methods: [1]

- Ward method [8] – an aggregative method that merges two clusters so that the increase of the variance of the distance between each vertex and the cluster center is minimized.
- Girvan-Newman method [7] – a divisive method that divides a cluster by iteratively removing the edge with the highest "betweenness", i.e., the highest number of shortest paths between all pairs of vertices that traverse the edge.

We also define a sequential method as a baseline. It groups every r or $r - 1$ nodes (so that the grouping is as even as possible) in order of generation, where r denotes the cabinet size. For example, given a 4-ary 4-mesh topology ($4 \times 4 \times 4 \times 4$ nodes) and $r = 16$, the sequential method assigns the first 4×4 nodes to the first cabinet, the next 4×4 nodes to the second cabinet, etc. As a result, in this example, the first two dimensions remain within each cabinet and the last two dimensions exit the cabinets. Note that this case can be considered a lucky occurrence; in general, the sequential method does not create topology-friendly clusters.

C. Adjusting the Cluster Size

The aggregative/divisive methods described in the previous section produce a tree structure called dendrogram, whose branches represent the aggregations/divisions of the cluster(s). For a data mining purpose, the dendrogram is cut horizontally and the resulting cluster size, i.e., its number of vertices, is unpredictable. For our purpose, however, the cluster size must not exceed the cabinet size. Consequently, we develop our own cutting method as follows:

1) Pick two clusters from the leaves furthest from the root.
2) Merge these clusters if the aggregate size does not exceed the cabinet size; otherwise leave them alone and cut the larger one off the dendrogram.
3) Repeat steps 1 and 2 until the root is reached.

Table 1. Topology details

	256 nodes	4,096 nodes
Mesh	4-ary 4-mesh	4-ary 6-mesh
Torus	4-ary 4-torus	4-ary 6-torus
Hypercube	8-dimensional	12-dimensional
Random Ring	8-degree	12-degree

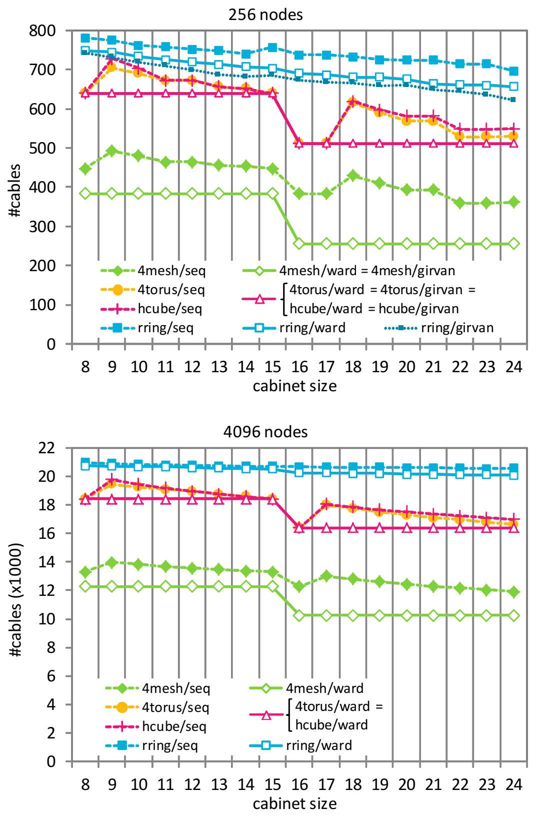

Figure 2. Number of inter-cabinet cables after clustering.

D. *Evaluation*

We evaluate the topologies listed in Table 1. Figure 2 shows the number of inter-cabinet cables produced by the Ward (ward) and Girvan-Newman (girvan) clustering methods compared to those by the sequential (seq) method for topologies of 256 (top chart) and 4,096 nodes (bottom chart), versus the cabinet size. The Girvan-Newman method was not tested on 4,096 nodes since its computation is $O(n^3)$, thus requiring several hours to complete. Recall that we omit the intra-cabinet cables in our counts, since their number is constant regardless of the clustering result.

These results show that the Girvan-Newman method leads to the best result regardless of the cabinet size. The same goes for the Ward method except for the Random Ring topology. The sequential method, in contrast, causes an unneeded increase in number of cables for some of the odd cabinet sizes (e.g., 9 and 18), but works well for even cabinet sizes (e.g., 8 and 16). By contrast, using our clustering methods leads to monotonically decreasing numbers of cables as the cabinet size increases. In these results, comparing to the baseline, the number of cables is reduced by clustering up to 40.3%, 16.7%, 17.3% and 10.9% for mesh, torus, hypercube and Random Ring topologies, respectively. We see that the Random Ring topology proves challenging due to its randomness and lack of locality.

Beyond the reduction in number of cables, these results also show that our automatic clustering approach can handle odd-sized cabinets for which there is no intuitive way to assign nodes to cabinets. Our approach thus makes it possible for a system designer to make hardware decisions (cabinet size, component sizes) without worrying about unexpected negative impact on the network topology layout, and thus on cabling cost.

V. Mapping

We now describe how we map our weighted graph of cabinets to a physical layout on a floorplan.

A. *Modeling*

Input to our approach, provided by the system designer, is a floorplan that indicates the possible locations for the cabinets. Our scheme then assigns each cabinet to a location so that the inter-cabinet total cable length is minimized. This process can be modeled as a facility location problem and formulated as a quadratic assignment problem (QAP).

Assuming n cabinets and n locations, the QAP solution is represented by a permutation $\Phi = \phi(1), \ldots, \phi(n)$ such that

$$\text{Minimize } \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} d_{\phi(i)\phi(j)} \qquad (1)$$

where d_{ij} denotes the distance between locations i and j, w_{ij} denotes the number of cables between locations i and j, and $\phi(i)$ denotes the location where cabinet i is assigned.

B. *Methods*

We employ Simulated Annealing (SA) [15], a well-known metaheuristics successfully applied to QAPs[2].

We defined the following baseline method for comparison. Assuming the locations are aligned on a 2-D grid, we consider two schemes: (1) a “wrap-around” method, which assigns the locations from left to right in every row, and (2) a “zigzag” method, which assigns the locations from left to right in the first row, from right to left in the second row, etc. The former can produce reasonable results for mesh/torus topologies whereas the latter is more reasonable for ring-based topologies. Our baseline method uses both schemes and returns the best of the two obtained results.

[1]We also tried Average, Single and Complete hierarchical methods as well as Walktrap [12] and Newman [13] methods, but omit them here since they produce either similar or poorer results.

[2]We also tried Robust Taboo Search[16], GRASP [17] and Fant [18], but omit them here since they produce almost the same results as SA.

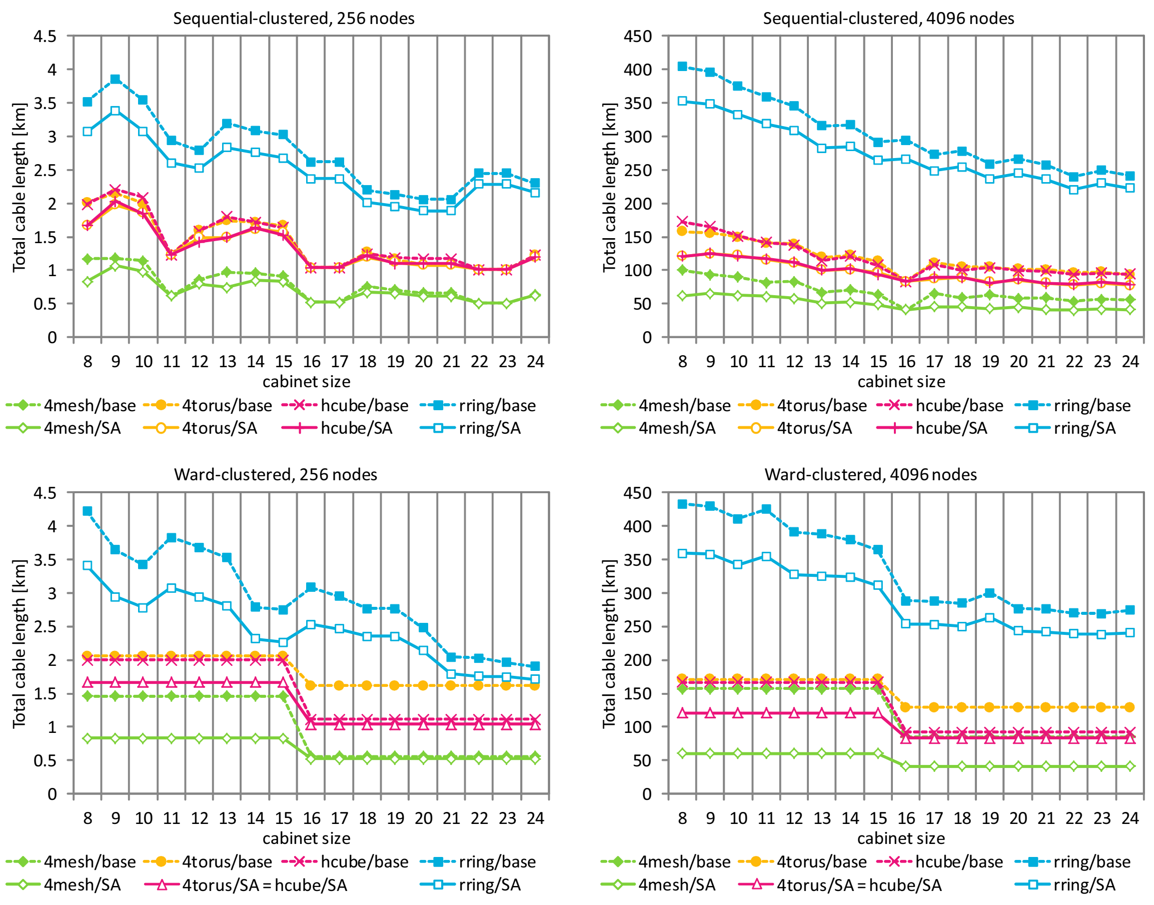

Figure 3. Total inter-cabinet cable length after mapping.

C. Map Generation

Our approach is applicable to an arbitrary map since the model is not built based on coordinates but solely on distances between locations. We assume that the system designer provides a floorplan that is sufficiently large to align all cabinets on a grid. Formally, assuming n cabinets, the number of row is $q = \lceil \sqrt{n}\ \rceil$ and the number of cabinets per row is $p = \lceil n/q \rceil$.

D. Evaluation

Figure 3 shows the total length of inter-cabinet cables achieved by the Simulated Annealing (SA) method compared to those by the baseline (base) method for topologies of 256 and 4,096 nodes. Topology details, cabinet size and clustering methods are the same as in Section IV. We assume that each cabinet is 0.6m wide and 2.1m deep including space for the aisle [19]. The distance between the locations is the Manhattan distance. We run SA for 100 million iterations and pick the best solution out of five trials. Here again we do not consider intra-cabinet cabling, since its total length is constant regardless of the layout.

The results show that the cable lengths produced by SA are always shorter than or at least equal to those produced by the baseline method. More specifically, in these results the total length, compared to the baseline, is reduced by up to 39.6%, 24.8%, 29.7% and 28.6% for mesh, torus, hypercube and Random Ring topologies, respectively. The implication is that a system designer can use our approach to come up with physical layouts without worrying about topology concerns. Finally, recall that our approach can be applied to any floorplan, including floorplans that span separate rooms.

We now turn to evaluating the scalability of our approach. As mentioned in Section III, we must support up to several thousands of nodes. For instance, considering 8,192 nodes, each with 10 attached compute nodes, would lead to 81,920 compute nodes, which is around the number of compute nodes in the K-computer [2].

For the topologies detailed in Table 2, Figure 4 shows the reduction rate of the total length of inter-cabinet cables optimized by SA compared to the baseline as the number of nodes increases (cabinet size is set to 12). The chart indicates that our approach remains effective when going from small systems to large systems. For the hypercube topologies, for example, a 3–19% reduction in cable length is observed

Table 2. Topology details (scalability evaluation)

3mesh	3-ary 5,6,7,8-mesh
3torus	3-ary 5,6,7,8-torus
4mesh	4-ary 4,5,6-mesh
4torus	4-ary 4,5,6-torus
hypercube	7,8,9,10,11,12,13-dimension
random ring	7,8,9,10,11,12,13-degree

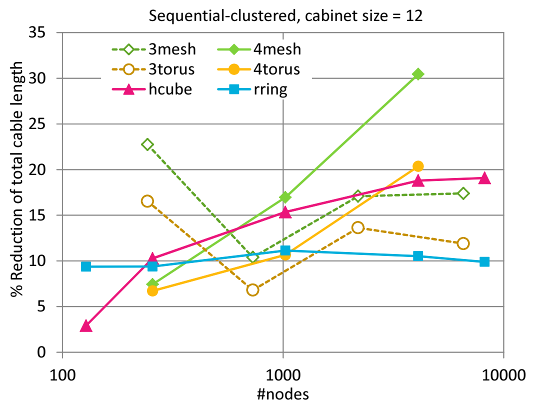

Figure 4. Percentage reduction in cable length with respect to the baseline versus the number of nodes.

between a 7-D hypercube with 128 nodes to 13-D hypercube with 8,192 nodes.

VI. Conclusion

In this paper we have studied the problem of reducing total inter-cabinet cable length in supercomputer physical layouts for various network topologies. For a given topology, we solve this problem by first grouping switches logically so that each group fits in a physical cabinet and the number of inter-cabinet links is low. This is accomplished by using graph clustering algorithms, and in particular the Girvan-Newman method [7] or the Ward method [8], which reduce the total number of links between cabinets by up to 40.3% compared to the straightforward method used in conventional topologies such as meshes, tori, and hypercubes. In the case of random ring topologies, the benefits of clustering are limited because the topology lacks locality. The obtained switch-cabinet set is then physically mapped to a floorplan in a way that reduces the total cable length between cabinets. This is accomplished by modeling the mapping problem as a facility location problem and solving this problem via Simulated Annealing. This approach is applicable to any type of floorplan and distance definition. Results show that SA is effective. For instance, it can reduce the total inter-cabinet cable length by up to 39.6% for mesh topologies when compared to a standard baseline approach. Finally, both clustering and mapping approaches scale up to at least the size of the largest existing supercomputer [2], still proving beneficial at these large scales.

As supercomputer scales increase, reducing cabling cost becomes increasingly crucial. The approach proposes in this work provides a solution that outperforms standard practice in terms of total cable length. Furthermore, one advantage of the approach is that it makes it possible to decouple layout concerns from topology concerns, while remaining effective even for high-degree topologies built from high-radix topologies, which are not amenable to intuitive physical layouts.

Acknowledgments

This work was partially supported by JST CREST.

References

[1] "Earth simulator project," http://www.jamstec.go.jp/es/en/index.html.

[2] Y. Ajima, S. Sumimoto, and T. Shimizu, "Tofu: A 6D Mesh/Torus Interconnect for Exascale Computers," *IEEE Computer*, vol. 42, pp. 36–40, 2009.

[3] InfiniBand Trade Association, Pluggable Interfaces Passive Copper, Active Copper and Optical Devices (White Paper), http://www.infinibandta.org/, 2007.

[4] P. Coteus and et. al., "Packaging the Blue Gene/L supercomputer," *IBM Journal of Research and Development*, vol. 49, no. 2/3, pp. 213–248, Mar/May 2005.

[5] S. Scott, D. Abts, J. Kim, and W. J. Dally, "The BlackWidow High-Radix Clos Network," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, 2006, pp. 16–28.

[6] J. Kim, W. J. Dally, and D. Abts, "Flattened Butterfly : A Cost-Efficient Topology for High-Radix Networks," in *ISCA '07 Proceedings of the 34th annual international symposium on Computer architecture*, vol. 35, no. 2, Jun. 2007, pp. 126–137.

[7] M. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, Feb. 2004.

[8] Joe H. Ward Jr., "Hierarchical Grouping to Optimize an Objective Function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

[9] M. Koibuchi, H. Matsutani, H. Amano, D. F. Hsu, and H. Casanova, "A Case for Random Shortcut Topologies for HPC Interconnects," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, 2012, pp. 177–188.

[10] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-Driven, Highly-Scalable Dragonfly Topology," in *ISCA '08 Proceedings of the 35th Annual International Symposium on Computer Architecture*, vol. 36, no. 3. IEEE, Jun. 2008, pp. 77–88.

[11] C. C. Aggarwal and H. Wang, *Managing and Mining Graph Data*. Springer, Feb. 2010.

[12] P. Pons and M. Latapy, "Computing communities in large networks using random walks," in *Computer and Information Sciences - ISCIS 2005*, 2005, pp. 284–293.

[13] A. Clauset, M. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, Dec. 2004.

[14] Pitu B. Mirchandani and Richard L. Francis, Eds., *Discrete Location Theory*. Wiley-Interscience, 1990.

[15] D. T. Connolly, "An improved annealing scheme for the QAP," *European Journal of Operational Research*, vol. 46, no. 1, pp. 93–100, May 1990.

[16] E. D. Taillard, "Robust taboo search for the quadratic assignment problem," *Parallel Computing*, vol. 17, no. 4-5, pp. 443–455, Jul. 1991.

[17] Y. Li, P. M. Pardalos, and M. G. Resende, "A greedy randomized adaptive search procedure for the quadratic assignment problem," *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 16, pp. 237–261, 1994.

[18] E. D. Taillard, "FANT: Fast ant system," in *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, Oct. 1998.

[19] HP, "Optimizing facility operation in high density data center environments, technoloogy brief," 2007.