Darkfiber Planning for Extensible HPC Network Design Under Uncertainties

Fabien Chaix*, Ikki Fujiwara*†, Michihiro Koibuchi*†‡
*National Institute of Informatics
†JST
‡The Graduate University for Advanced
Studies (SOKENDAI)
2-1-2 Hitotsubashi, Chiyoda-ku,
Tokyo, JAPAN 101-8430
Email: {fabien_chaix, ikki, koibuchi}@nii.ac.jp

***Abstract*—Cabling negatively affects not only the expandability of HPC systems, but also the reliability of their communications. In effect, the deployment of a supercomputer requires thousands of kilometers of cables, which are generally buried under the floor. Hence, moving or replacing these fibers is impossible once a supercomputer is deployed. In this study, we propose to exploit an efficient cabling method to enable multiple topologies in a system expanded incrementally. This approach reduces the cost of implementing an HPC system stage after stage, while requiring a limited knowledge about the future target applications and the final size of the system.**

***Index Terms*—Cabling, interconnection networks, network topology, high-performance computing**

I. Introduction

A large number of data-centers and supercomputers are incrementally expanded year after year, since precisely estimating the future demands of users and securing the whole HPC budget at once is often too complex. For example, many supercomputers of the Top 500 [1] have increased their computation power not only by optimizing the application software, but also by installing more computing nodes after the initial deployment. It is also described in [2] that a large number of data-centers are expanded incrementally. Figure 1 illustrates such an HPC implemented in 3 stages. Each cabinet consists of 4 switches, that are connected to several —not shown— compute nodes, and other switches through cables represented in yellow. At the interconnect level, cables are installed when required, following Figure 2.

Before the supercomputer is deployed, cables are usually installed under a floor (as in Figure 1) or on the cabinets side, for an increasing the packaging density. As a consequence, once cabinets have been assembled, the removal and/or addition of these cables becomes extremely tedious. Hence, cables connecting different cabinets can be seen as darkfibers, since operators cannot afford to manipulate them. Fortunately, a sufficient slack at both cable end lets administrators with the opportunity to select any switch within end cabinets. Accordingly, the connection of two switches belonging to the same cabinet requires little effort.

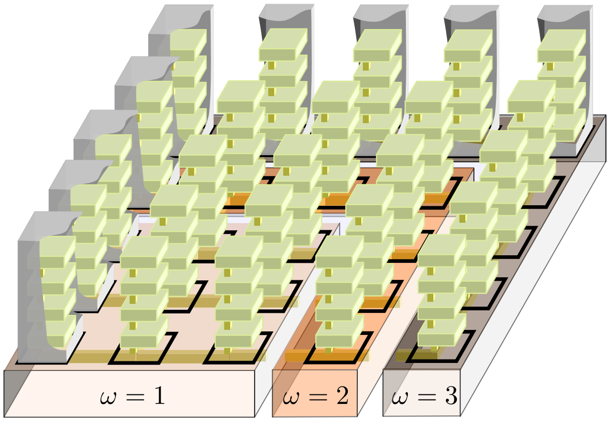


Figure 1: Representation of a HPC built in 3 stages ($\omega = 1$, $\omega = 2$ and $\omega = 3$). Each cabinet contains 4 switches (green) bound together by an underlying darkfiber network (yellow).

When the expansion of a supercomputer is considered, by adding compute nodes and/or storage, a primary concern is the possibility to re-design the supercomputer architecture, e.g. by using new GPUs. In this context, the upgrade of the interconnection network is essential, to fully exploit novel compute nodes. However, at present, the network is gracefully and conservatively updated without changing neither the topology nor the routing algorithm; because the preexistent cabling generally limits the re-design options regarding both topology and routing.

Another key design aspect is the topology implemented by the interconnection network. The family of fat-trees and k-ary n-tori/meshes is commonly used for existing supercomputers, thus various parallel algorithms are tuned to one topology of choice. Alternatively, random topologies recently received a fair attention for low-latency purposes [3]. These topologies generally perform better than regular topologies when the traffic patterns are unknown or unpredictable. Overall, an ideal solution is to support all the above topologies on a single supercomputer. For example, three networks could be installed in a supercomputer, with a threefold cabling cost though. Instead, our method attempts to reuse cables as much as possible, and limits the number of cables installed before a supercomputer is expanded, hence reducing unused cables if a topology is abandoned. Another interesting possibility is the choice of a high-radix topology for the first implementation stages, and the progressive diminution of radix as the interconnect grows, which avoids the explosion of the aggregated cable length.

In this context, we propose a cabling method to allow the change of the network topology after the supercomputer is deployed, while limiting the cabling budget. Our main contributions are:

- a formal description of the darkfiber planning process;
- the support of multiple topologies at once and its cost estimation;
- the reduction of backup cables cost.

Background information and related works are discussed in Section II. Our method is presented formally in Section III and some experimental results are discussed in Section IV. Section V concludes with a brief summary of our findings.

II. Related Work

A. Network Topology

A few topologies, e.g. k-ary n-cubes and fat trees, are traditionally used to interconnect compute nodes in most HPC systems. Each topology leads to a different tradeoff between degree and diameter; thus having different killer-applications. Stencil communications, which are a frequent pattern in fluid applications, fit well to k-ary n-cubes, whereas fat trees support well all-to-all exchanges and shuffle communications. All these topologies are *regular*, meaning that all switches have the same degree (i.e., each switch has an identical number of links to other switches).

Random topologies have better properties in terms of diameter and average shortest path length [4], which is crucial when every process needs to communicate to every other process at some point during the execution of the application. Recently, the advantage of random topologies has been reported for various communication patterns [5]. In the present study, we support the iterative implementation of above topologies, as the supercomputer expands.

B. Cabling

The layout of cabinets on a floor-plan is a major concern when designing large systems because it largely affects the installation and exploitation costs [6]–[8]. The features of floor layout include the cabinet footprint, the number of compute nodes and switches per cabinet, and spacing between cabinets. For instance, in the case of the Cray BlackWidow system, it is estimated that each cabinet has a 0.57m × 1.44m footprint, with 128 nodes per cabinet, and that the node density should be close to 75 nodes/m^2 [6]. A common way to view this problem is to come up with specifications for the widths of the aisles between cabinet rows. The ANSI/TIA/EIA-942 standard recommends site layouts with alternating cold and hot aisles, respectively with widths in excess of 4ft and 2ft. A similar specification can also be found in [9]. In this study, we assume that some 2-D physical layout of cabinets has been determined to comply with the power/heat constraints of the system to be deployed. Our approach can be readily parameterized to comply with the specific layout constraints.

The other assumptions are listed as follows:

- A Manhattan cabling is assumed.
- All cabinets contain the same number of switches.
- All cables between switches in a same cabinet have the same length.
- All cables below a certain length threshold are copper-made, all others are optic fibers.

C. Routing Update

Custom routing implementations in large-scale HPC systems that use non-random topologies can exploit topological regularity, such as dimension-order routing on k-ary n-cubes, to make routing logic simple and small. By contrast, supporting the topology change or randomness makes it impossible to rely on such schemes because the topology does not have a simple structure. In this case, it is necessary to use source routing or distributed routing that relies on routing tables. In practice, a large number of recent supercomputers posted in the Top500 list [1] use Ethernet or InfiniBand, both technologies relying on routing tables. Hence, for all these supercomputers, the size of routing tables is a scalability limitation regardless of the topology in use.

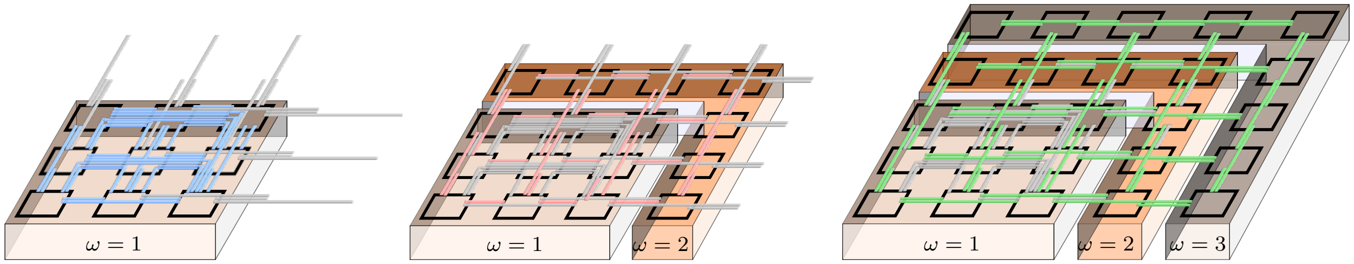
III. Method

In this section, the proposed method is described generically. Our method consists in four steps. First, the layout of topologies of interest and the expansion plan are defined in Section III-A. Then, nodes of the considered topologies are regrouped in clusters fitting accurately the resources of individual cabinets in Section III-B. Third, clusters are allocated to cabinets with respect to darkfiber constraints in Section III-C. The planning of cabling is then detailed in Section III-D.

A. Topologies and cabinets layout

The proposed approach divides the implementation of the complete supercomputer and its underlying network into ω_{max} stages. At each stage ω, additional cabinets —each containing multiple switches— are appended to the network, following Figure 1. At the interconnect level, cables are installed at each stage, following Figure 2. In the proposed example, a different topology is utilized at each stage, in order to mitigate the rapid increase of cabling costs when the network grows.

The computation of the solution depends on the current stage of implementation of the supercomputer, namely ω_{now}. In effect, the clusters allocated to cabinets during previous stages cannot be displaced, for pre-existing cables may not suffice anymore. Moreover, the computation of cable cost is different whether cable must be installed immediately or if its installation could be delayed to later stages, as we shall see in Section III-D.



(a) Stage 1: Full network (b) Stage 2: 3D-Mesh (c) Stage 3: 2D-Torus

Figure 2: Stage-by-stage implementation of an interconnect featuring different topologies. Cables exploited by the current topology are colored, while gray cables were installed for supporting earlier or future topologies.

In this work, cabinets (as known as racks) will be denoted r and switches s. The relation between switches and cabinets is expressed by the membership relation proposed in Definition 1.

Definition 1 (Switch membership): For each switch s, function $M(s)$ gives the containing rack (as known as cabinet) r. Inversely, $M^{-1}(r)$ returns the **ordered** set of switches belonging to cabinet r.

The installation stage of each rack r can be obtained through the stage function $\Omega(r)$ defined in Definition 2, and its physical position is denoted (x_r, y_r).

Definition 2 (Implementation stage): The stage ω when r is implemented is given by function $\Omega(r)$. Inversely, the set of cabinets installed during stage ω is $\Omega^{-1}(\omega)$.

Additionally, the corner rack function defined in Definition 3 retrieves the *extreme* cabinet for each stage. Finally, we denote R_ω the total number of cabinets installed at stage ω.

Definition 3 (Corner rack): For each implementation stage ω, $r(\omega)$ denotes the rack which has the highest coordinates, i.e. $\forall r \in \Omega^{-1}(\omega),\ x_{r(\omega)} \geq x_r \wedge y_{r(\omega)} \geq y_r$.

Based on the previous definition, several assumptions are made regarding the expansion of the supercomputer. During each stage, the complete set of cabinets is placed in a rectangular —if not square— fashion, alike Figure 1 and according to state-of-art practices discussed in Section II-B. Newer cabinets are placed in increasing coordinates, respecting Equation 1.

$$\forall r_1, \forall r_2,\ \Omega\left(r_1\right) < \Omega\left(r_2\right) \Rightarrow x_{r_1} < x_{r_2} \vee y_{r_1} < y_{r_2} \tag{1}$$

This assumption allows us to guarantee that cables between two cabinets implemented in later stages could be routed later with a minimal length, following Figure 2.

Each considered network topology t is fully defined by its symmetric connectivity matrix A_t between each pair of network nodes n_1 and n_2, following Equation 2.

$$A_t = \left(a_{tn_1n_2}\right)_{n_1,n_2},\ a_{tn_1n_2} \in \{0,1\} \tag{2}$$

For each stage ω, the set T_ω contains all topologies that are to be utilized for this stage. Of course, for each topology $t \in T_\omega$, the number of nodes $card(t)$ shall be equal to R_ω, which is the total number of installed switches at stage ω. Though, if topologies of inferior sizes should be considered, disconnected nodes $n_\varnothing$ could be appended (i.e. $\forall n_1, n_2, a_{tn_\varnothing n_2} = a_{tn_1n_\varnothing} = 0$). Additionally, an utilization ratio θ_t is associated to each topology t, following Definition 4. Its interpretation is twofold. θ_t both represents the probability to actually implement t when $\omega > \omega_{\text{now}}$ and the expected time share when the supercomputer will utilize the topology t. Hence, the relation $\sum\limits_{t\in T_\omega} \theta_t = 1$ shall hold for all stages ω.

Definition 4 (Topology utilization ratio): In our approach, a ratio $\theta_t \in]0,1]$ is associated to each topology t implemented at stage ω. This ratio enables designers to quantify the relative impact of each topology on the system cost and power, following Equations 6, 7 and 10.

B. Topologies clustering

For each topology t, the goal of our method is to assign each node n to a different switch s. However, the complexity of the considered problem demands to reduce the size of input data (i.e. the size of considered topologies). Since cables connected to switches belonging to the same cabinet could be exchanged with little effort, a common approach is to group the nodes into clusters that each fit accurately a cabinet. For instance, if the supercomputer presented in Figure 1 is being designed, the nodes of each topology will be grouped into clusters of 4 nodes, since each cabinet contains 4 switches. The clustering function is presented in Definition 5.

Definition 5 (Node clustering): For each node n of a topology t, the clustering function $\Gamma(n)$ returns the containing cluster c. Inversely, $\Gamma^{-1}(c)$ returns the **ordered** set of enclosed nodes. The number of nodes in any cluster c must be **strictly equal** to the number of switches per cabinets, which greatly simplifies the cluster allocation.

At the cluster level, the connectivity of topology t is given by the aggregated function of Definition 6.

Definition 6 (Aggregated connectivity): For each topology t, the aggregated connectivity $\alpha_t(c_1, c_2)$ between cluster c_1 and cluster c_2 is defined as the sum of connectivity between nodes belonging to each cluster, following Equation 3.

$$\alpha_t(c_1, c_2) = \frac{1}{2} \sum_{\substack{n_1 \in \Gamma^{-1}(c_1) \\ n_2 \in \Gamma^{-1}(c_2)}} a_{tn_1n_2} \tag{3}$$

Hence, the clustering step consists in grouping nodes into clusters of fixed size. An optimal clustering would minimize the aggregated connectivity following Equation 4, while respecting the cluster size constraint.

$$\min \sum_{c_1,c_2} \alpha_t\left(c_1, c_2\right) \tag{4}$$

In [10], several heuristic approaches were tested, such as the Ward and the Girvan-Newman methods. Empirically however, a topology-specific heuristic approach is leading to a satisfying solution for the most common cases. For example, toruses and meshes usually have a dimension matching the number of switches in a cabinet (e.g. 4 in Figure 1), and nodes are clustered following this direction.

C. Clusters allocation

The third step of our method consists in allocating each cluster c to a unique rack r, while reducing the cost and power induced by darkfibers. Since there is a one-to-one relation between clusters and racks, the solution can be modeled as a set of cluster permutations Φ_t following Definition 7.

Definition 7 (Cabinet permutation): The allocation of each cluster c of a topology t to a rack r of the network is represented by a bijective function $\Phi_t(r)$ that returns the assigned cluster c. Inversely, $\Phi_t^{-1}(c)$ returns the allocated rack r.

The objective of cluster allocation is to minimize the power and cost of inter-cabinet cables (i.e. darkfibers), while including the cost of backup cables, as shown in Equation 5.

$$\min\ \beta_{\text{pwr}} \sum_{r_1,r_2} pwr_{r_1 r_2} \quad + \quad \beta_{\text{cost}} \left\| \sum_{r_1,r_2} cost_{r_1 r_2} \right\| \tag{5}$$

The quantity to minimize is a linear aggregation of these two objectives with respective weights β_{pwr} and β_{cost}, based on the quantity $pwr_{r_1 r_2}$ defined in Equation 6 and the vector $cost_{r_1 r_2} = \left(cost_{r_1 r_2 \omega}\right)_\omega$ described in detail later in Section III-D. It worth noting that the cost vector contains one dimension per implementation stage, hence providing a yearly estimate of the cost of interconnect cables.

The power consumed by cables between racks r_1 and r_2 is expressed in Equation 6, based on the vendor-specific $Power(.)$ function that gives the power consumed by one cable of the given length. In this work, we consider a stepwise linear function to account for difference between optical and copper cables. The cable length function $\Lambda(.)$ represents the actual length of cables, and will be later defined in Equation 8.

$$pwr_{r_1 r_2} = Power\left(\Lambda(r_1, r_2)\right) \sum_t \theta_t\ \alpha_t(\Phi_t(r_1)\ ,\ \Phi_t(r_2)) \tag{6}$$

Similarly to [10], the optimization of clusters allocation is obtained by simulated annealing. At each step, two clusters of a given topology t are swapped between racks r_1 and r_2. The probability $p_{\text{swap}}(t)$ that a topology t is chosen for swapping is described in Equation 7. This value depends on the number of clusters that remain to be allocated either during the current stage ω_{now} or later stages $\omega > \omega_{\text{now}}$, and the topology utilization ratio presented in Definition 4. Accordingly, only clusters that belong to cabinets not yet implemented can be swapped during allocation, i.e. $\Omega(r_1) \geq \omega_{\text{now}} \wedge \Omega(r_2) \geq \omega_{\text{now}}$.

$$p_{\text{swap}}(t) = \theta_t\ \max\left(\, 0\ ,\ card(t) - R_{\omega_{\text{now}}-1}\,\right) \tag{7}$$

D. Cabling planning

The planning of cables installation is a complex step, since practical choices change significantly the cabling operation. First, the actual length of cables requires some tuning compared to the distance between end-racks. Second, the number of cables must be increased to include backup cables, and satisfy reliability constraints. At last, the switch-to-switch cabling method is described, finalizing our approach.

The actual length of cables is affected by the Manhattan cabling scheme and the predefined cable slacks. Indeed, cables are routed following perpendicular aisles (Manhattan scheme), which reduces the complexity of cable installation and allow for cables to be added in latter stages of the implementation, akin to Figure 2. The length of cables connecting switches within a given rack is set to a constant value $\lambda_{\text{intra-rack}}$, while inter-rack cables are augmented with a constant slack $\lambda_{\text{inter-rack}}$ at each end.

$$\Lambda\left(r_1 r_2\right) = \begin{cases} \lambda_{\text{intra-rack}} & \text{if } r_1 = r_2 \\ 2\lambda_{\text{inter-rack}} + \left\| r_1, r_2 \right\| & \text{else} \end{cases} \tag{8}$$

The real number of cables $\nu_{\text{real}}(r_1, r_2)$ installed between racks r_1 and r_2 is given by Equation 9, as the maximum amongst all topologies. The prospective number of cables $\nu_{\text{prosp}}(r_1, r_2)$ given in Equation 10 is highly similar but the requirement of each topology t is weighted by its utilization ratio θ_t, following Definition 4. The generalized cable number $\nu_\omega(r_1, r_2)$ is then proposed in Equation 11.

$$\nu_{\text{real}}(r_1, r_2) = \max_{t \in \bigcup T_\omega} \alpha_t(\Phi_t(r_1)\ ,\ \Phi_t(r_2)) \tag{9}$$

$$\nu_{\text{prosp}}(r_1, r_2) = \sum_{t \in \bigcup T_\omega} \theta_t \alpha_t(\Phi_t(r_1)\ ,\ \Phi_t(r_2)) \tag{10}$$

$$\nu_\omega(r_1, r_2) = \begin{cases} \nu_{\text{real}}(r_1, r_2) & \text{if } \omega \leq \omega_{\text{now}} \\ \nu_{\text{prosp}}(r_1, r_2) & \text{else} \end{cases} \tag{11}$$

In our method, the installation process follows Figure 2. Assuming that $\Omega(r_1) \leq \Omega(r_2)$, the complete cable is installed at stage $\Omega(r_1)$, generating the stage cabling cost presented in Equation 12. The vendor-specific $Cost(.)$ function represents the cost of a single cable of given length. Similarly to the $Power(.)$ function, $Cost(.)$ is a stepwise linear function.

$$cost_{r_1 r_2, \omega} = \begin{cases} Cost\left(\Lambda(r_1, r_2)\right) \nu_\omega(r_1, r_2) & \text{if } \omega = \Omega(r_1) \\ 0 & \text{else} \end{cases} \tag{12}$$

```
Require: $A_t$: The connectivity matrix of topology $t$
Require: $\Phi_t(.)$: The cluster permutation of topology $t$
for all $r_1 \in R_{\omega_{\text{now}}}$ do
    for all $r_2 \in R_{\omega_{\text{now}}}$ do
        $i \leftarrow 0$
        for all $(s_1, n_1) \in M^{-1}(r_1), \Gamma^{-1}(\Phi_t(r_1))$ do
            for all $(s_2, n_2) \in M^{-1}(r_2), \Gamma^{-1}(\Phi_t(r_2))$ do
                if $a_{tn_1n_2} = 1$ then
                    Connect switch $s_1$ to cable $i$ for rack $r_2$
                    $i \leftarrow i + 1$
                end if
            end for
        end for
    end for
end for
```

Figure 3: Procedure for switch-to-switch cabling of a chosen topology t

Once cables and cabinets are installed, the final set-up for the desired topology t **only** requires to connect each required cable to the adequate end-switches, amongst switches from the end-racks. The key issue is to connect each switch rigorously with other switches according to the connectivity matrix A_t, and not mixing cable ends. Assuming that for each pair of racks (r_1, r_2), connecting cables are labeled with a unique integer ranging from 0, a topology could be set up by following the approach described in Figure 3.

IV. Evaluation of topologies cost

The proposed method is evaluated with a set of programs developed in the laboratory. We consider the implementation of an HPC containing up to 256 cabinets of 4 switches each, totalling 1024 nodes. The considered topologies are tori of different degrees (xD-Tor.), a random ring with random shortcuts of node degree 6 (Rand.), and a sole ring (Ring).

The cost of implementing a network topology varies largely with the nodes degree. In Figure 4, the cabling cost of several topologies is displayed as a function of the number of nodes. Topologies with high degrees, such as the 6D-torus, see the cost raising rapidly, while topologies with lower degree, such as the ring, remains linear. Hence, it seems more economical to consider high-degree topologies for smaller interconnects, and use topologies with decreasing degree as the network increases.

Second, Figure 4 enables the comparison between naive installation and the proposed method for all considered topologies. While both are clearly more expensive than individual topologies, it is worth noting that our method is significantly less costly than a naive approach, especially for small topologies (up to 33%). Improvement could be even more important for use-cases that include only a few disparate topologies with similar node degrees, while the presented configuration features many topologies with different node degrees, which tend to hide cabling reductions achieved on lower degree topologies.

Table 1: Notations.

Notation	Description
ω	stage
ω_{now}	current stage
ω_{max}	stages total
n	node
c	cluster
s	switch
r	rack
R	stage rack total
t	topology
A	topology connectivity
θ	topology utilization ratio
T	stage topology set
(x, y)	physical position
pwr	inter-rack cable power
$cost$	inter-rack cable cost
$\Omega(.)$	implementation stage
$M(.)$	switch membership
$\Gamma(.)$	node clustering
$\alpha(.)$	aggregated connectivity
$p_{\text{swap}}(.)$	topology swapping probability
$\Phi(.)$	cluster permutation
$\nu(.)$	inter-rack cable number
$Cost(.)$	unitary cable cost
$Power(.)$	unitary cable power
β_{pwr}	objective power factor
β_{cost}	objective cost factor
$\lambda_{\text{intra-rack}}$	intra-rack cable slack
$\lambda_{\text{inter-rack}}$	inter-rack cable slack

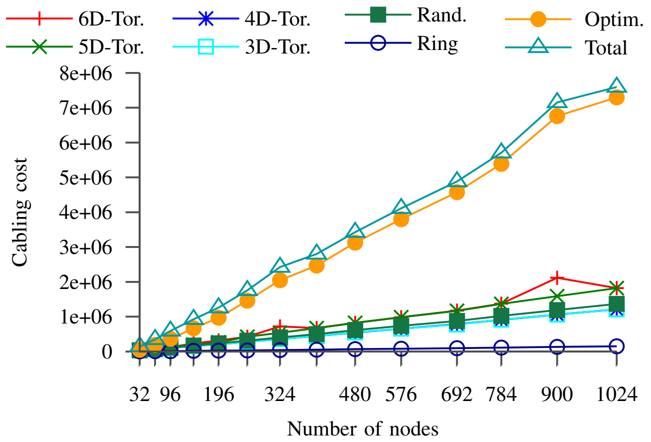


Figure 4: Cabling cost for different individual topologies, and for the implementation of all of them following our proposal or a naive approach

V. Conclusions

The high diversity of the scientific applications that are running in HPCs lead to many difficult design choices regarding the interconnect network. In particular, the best topology to adopt may be different depending on the traffic pattern of target applications. In this context, an approach for planning the installation of HPC interconnect darkfibers supporting multiple topologies seems appropriate. Starting from loose specifications on the interconnect topology, our method allows designers to accurately model the cabling expenses, including backup cables.

Beneath the support of a wide range of configurations, our approach attempts to optimize the amount of installed darkfibers through an heuristic cabinet allocation presented in Section III-C. A rapid evaluation shown in Figure 4 illustrates the impact of this optimization. While this technique achieves a reduction of 33% for small interconnects, its potential is limited for large interconnects. As a future work, we would like to introduce optimization techniques during later planning stages, which we believe could increase the cable savings and hence the interest of this method.

Acknowledgments

This work was partially supported by JST CREST and by KAKENHI #25280018 and #25280043.

References

[1] Top 500 Supercomputer Sites, http://www.top500.org/.

[2] A. Singla, C.-Y. Hong, L. Popa, and P. B. Godfrey, "Jellyfish: Networking Data Centers Randomly," in *Proc. of USENIX Symposium on Network Design and Implementation (NSDI)*, 2012.

[3] 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.

[4] B. Bollobás and F. R. K. Chung, "The Diameter of a Cycle Plus a Random Matching," *SIAM J. Discrete Math.*, vol. 1, no. 3, pp. 328–333, 1988.

[5] X. Yuan, S. Mahapatra, W. Nienaber, S. Pakin, and M. Lang, "A new routing scheme for jellyfish and its performance with hpc workloads," in *Proceedings of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '13, 2013, pp. 36:1–36:11.

[6] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: a cost-efficient topology for high-radix networks," in *Proc, of the International Symposium on Computer Architecture (ISCA)*, 2007, pp. 126–137.

[7] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-Driven, Highly-Scalable Dragonfly Topology," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, 2008, pp. 77–88.

[8] M. Koibuchi, I. Fujiwara, H. Matsutani, and H. Casanova, "Layout-conscious random topologies for hpc off-chip interconnects," in *19th International Conference on High-Performance Computer Architecture (HPCA)*, Feb. 2013, pp. 484–495.

[9] HP, "Optimizing facility operation in high density data center environments , technoloogy brief," 2007. [Online]. Available: http://h18004.www1.hp.com/products/servers/technology/whitepapers/datacenter.html

[10] I. Fujiwara, M. Koibuchi, and H. Casanova, "Cabinet Layout Optimization of Supercomputer Topologies for Shorter Cable Length," in *Proc. of the 13th International Conference on Parallel and Distributed Computing, Applications and Technologies*, Dec. 2012.