On the Routing Algorithms for Optical Multi-$\log_2 N$ Networks

Yusuke Fukushima, Xiaohong Jiang, and Susumu Horiguchi

Graduate School of Information Sciences, Tohoku University, Japan
{yusuke,jiang,susumu}@ecei.tohoku.ac.jp

Abstract. Multi-$\log_2 N$ networks architecture is attractive for constructing optical switches, and the related routing algorithms are critical for the operation and efficiency of such switches. Although several routing algorithms have been proposed for multi-$\log_2 N$ networks, a full performance comparison among them is not available by now. Thus, this paper is committed to such a comparison in terms of blocking probability, time complexity, hardware cost and load balancing capability. Notice that the load balance is important for reducing the peak power requirement of a switch, so we also propose in this paper two new routing algorithms for optical multi-$\log_2 N$ networks to achieve a better load balance.

1 Introduction

It is expected that users of telecommunication services such as Internet, Web browsing, and tele-education will increase dramatically. This has greatly increased the demand for high-bandwidth and high capacity communication systems. All optical networks that work completely in the optical domain are expected to meet this demand. The optical switching networks (or switches), that can switch optical signals in optical domain with an ultra-high speed, will be the key supporting elements for the operation and efficiency of future high-capacity optical networks. The multi-$\log_2 N$ network architecture, which is based on the vertical stacking of multiple $\log_2 N$ networks [1], has been attractive for constructing the optical switching networks due to its small depth, absolute loss uniformity, etc.

Since a multi-$\log_2 N$ network consists of multiple copies (planes) of a $\log_2 N$ network, so for routing each request (e.g., a connection request between an input-output pair) we have to select a plane based on a specified strategy. We call such a plane selection strategy as the routing algorithm for multi-$\log_2 N$ networks. The routing algorithm is important for the operation and efficiency, since it directly affects the overall switch hardware cost and also the switching speed. By now, several routing algorithms have been proposed for multi-$\log_2 N$ networks, such as random routing [2,3,4], packing [2], save the unused [2], etc. It is notable that the available results on routing algorithms mainly focus on the nonblocking condition analysis when a specified routing algorithm is applied [1, 2, 5, 6, 7, 8]. The recent results in [2] indicate that although some routing algorithms are apparently very different, such as save the unused, packing, minimum index,

K. Li et al. (Eds.): NPC 2007, LNCS 4672, pp. 91–100, 2007.

etc., they actually require a same number of planes to guarantee the nonblocking property of multi-$\log_2 N$ networks. The results in [2] also imply that a very high hardware cost (in terms of number of planes) is required to guarantee the nonblocking property, which makes the nonblocking design of multi-$\log_2 N$ networks impractical for real applications. The blocking design of multi-$\log_2 N$ networks is a promising approach to significantly reducing the hardware cost [3]. However, little literature is available on the performance of the available routing algorithms when they are adopted in the blocking network design [3, 4]. In particular, no work is available on the detailed performance comparison among the available routing algorithms when they are applied to a blocking multi-$\log_2 N$ network (e.g., a multi-$\log_2 N$ network with a less number of planes required by its nonblocking condition). It is notable that the load-balancing capability of a routing algorithm is also important for the a multi-$\log_2 N$ network, since it directly affects the peak power requirement and power dispassion requirement (mainly determined by the maximum number of connections simultaneously supported by a plane). However, little available algorithms take into account the load-balancing issue in the routing process.

In this paper, to address the above two main issues, we will propose two routing algorithms which possess the load-balancing capability and also fully compare all routing algorithms in terms of blocking probability, hardware cost, complexity and load-balancing capability. The rest of this paper is organized as follows. Section Ⅱ illustrates the structure and the features of multi-$\log_2 N$ networks. Section Ⅲ introduces the available routing algorithms and our two routing algorithms. Section Ⅳ provides the comparison among the routing algorithms, and finally, the Section V concludes this paper.

2 Multi-$\log_2 N$ Networks

The multi-$\log_2 N$ network architecture was first proposed by Lea [1]. A multi-$\log_2 N$ network consists of multiple vertically stacked planes as shown in Fig. 1, where each plane of it is just a banyan class ($\log_2 N$) network [1, 2, 3, 4, 5, 6, 7, 8] illustrated in Fig. 2. For the convenience of explanations, we use the notation $\mathrm{Log}_2(N, m)$ to denote a multi-$\log_2 N$ network with m planes and we number its planes from the top to the bottom as $p_0, p_1 ..., p_{m-1}$ as shown in Fig. 1.

For a $\mathrm{Log}_2(N, m)$ network, we define a request as an one-to-one (unicast) connection request and denote a request between input x and output y as $\langle x, y \rangle$. We further define a request frame as the set of all requests to the network and denote it as $\begin{pmatrix} x_0\ x_1 \cdots x_{k-1} \\ y_0\ y_1 \cdots y_{k-1} \end{pmatrix}$, where $0 < k \leq N$. An example of request frame in given in Example 1.

Example 1. *A request frame for Log$_2$(16, m)*

$$\begin{pmatrix} 0 & 1 & 5 & 7 & 12 & 15 \\ 1 & 13 & 10 & 2 & 8 & 0 \end{pmatrix}$$

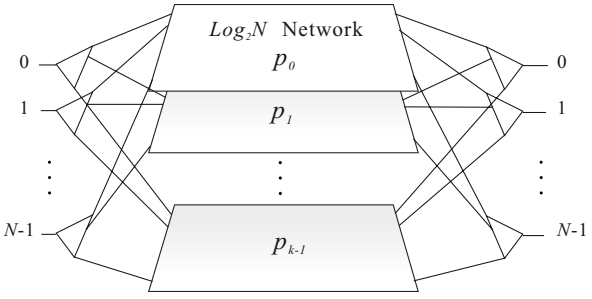

Fig. 1. A multi-$\log_2 N$ network with m-planes. Each request will be routed in a plane selected from p_0, p_1, ..., p_{m-1}.

The multi-$\log_2 N$ networks have several attractive properties, such as a small network depth and the absolute loss uniformity, which make them attractive for building the directional coupler (DC)-based optical switches, see, for example, [1, 3, 6]. Although the DC technology can support nano-second order switching speed [13] and can switch multiple wavelengths simultaneously, it may suffer from the crosstalk problem when two optical signals pass through a common DC at the same time. A simple and cost effective approach to guaranteeing a low crosstalk in DC-based optical multi-$\log_2 N$ networks is to apply the node-disjoint (or DC-disjoint) constraint to all the connections in this network [3, 4, 8]. Thus, this paper focuses on the optical multi-$\log_2 N$ networks with the node-disjoint constraint. It is notable that the above node-disjoint constraint will cause the node-blocking problem, which happens when two optical signals go through a common node (DC) at the same time and one of them will be blocked. Since link-blocking between two signals will definitely cause node-blocking between them, but the reverse may not be true. Thus, we only need consider the node-blocking issue in the routing process of optical multi-$\log_2 N$ networks. For the requests of the request frame in Example 1, the node-blocking (blocking for short) scenario among them is illustrated in Fig. 2, where the blocking happens between $\langle 0, 1\rangle$ and $\langle 1, 13\rangle$, between $\langle 0, 1\rangle$ and $\langle 7, 2\rangle$ and between $\langle 0, 1\rangle$ and $\langle 15, 0\rangle$.

Notice that a plane of the multi-$\log_2 N$ networks offers only one unique path to each connection, so the blocking will happen between two connections in the plane if their paths share a common node and the blocked one must be set up through another plane. For a set of requests to be set up, how to choose a plane for each request under the node-disjoint constraint requires a routing algorithm, as explained in the next section.

3 Routing Algorithms

In this section, we first introduce the seven available routing algorithms for a multi-$\log_2 N$ network and then propose two new ones with a better load-balancing capability. For a given request frame and its set of requests, a routing

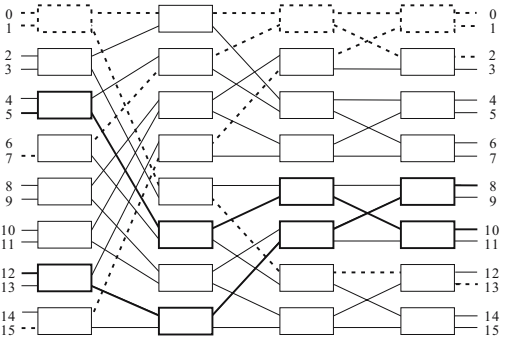

Fig. 2. A 16×16 banyan network with crosstalk set (even number of stages)

algorithm will try to route these requests one by one sequentially based on both the node-blocking constraint and a specified strategy (e.g., least occupied plane first, most occupied plane first, save the unused, etc.).

For a request, the random routing (R) algorithm [2,4] selects a plane randomly among all the available planes for this request, if any.

Based on the packing (P) routing algorithm [3, 9, 10, 11], we always try to route a request through the busiest plane first, the second busiest plane second, and so on untill the first available one emerges. The P algorithm has been well-known as an effective algorithm to reduce the request blocking probability, but it may introduce a very heavy traffic load (in terms of number of requests) to a plane.

The minimum index (MI) algorithm [2] always try to route a connection through the first plane p_0 first, second plane p_1 second and so on untill the the first available plane appears.

To route a request based on the save the unused (STU) algorithm [2], we do not select the empty plane(s) unless we can not find an occupied plane that can route the request.

The cyclic static (CS) algorithm [2] always keep a pointer to last used plane [2]. To route a new request, it checks the pointed plane first, then follows the same manner as that of the MI algorithm. The difference between the MI and CS is that starting point of latter is not fixed and it depends on the last used plane.

The cyclic dynamic (CD) algorithm [2] is almost the same as CS algorithm, and the only difference is that the CD algorithm always check the next plane of the pointed one first.

It is interesting to notice that although the above six routing algorithms are apparently different, a recent study in [2] revealed that they actually require a same number of planes to guarantee the nonblocking property of a multi-$\log_2 N$ network. The results in [2] also imply that the nonblocking design of multi-$\log_2 N$ networks is not very practical since it requires a very high hardware cost (in terms of number of required planes).

Danilewicz *et, al* (D) [6] recently proposed an novel routing algorithm for multi-$\log_2 N$ networks to guarantee the nonblocking property with a reduced number of planes. The main idea of this algorithm is to select plane that new connection will block the fewest number of future requests of all planes. It is notable, however, the time complexity of this algorithm is significantly higher than the above six routing algorithms.

Notice that the blocking design of multi-$\log_2 N$ networks is a promising approach to dramatically reducing their hardware cost [3] without introducing a significant blocking probability. However,the available study on the above seven algorithms mainly focus on their corresponding nonblocking conditions analysis and little literature is available on the performance of these routing algorithms when they are adopted in the blocking network design [3, 4]. It is an interesting question that although the nonblocking conditions of the first six routing algorithms are the same, whether their performance is still the same when that they are applied to a blocking multi-$\log_2 N$ network (e.g., a multi-$\log_2 N$ network with a less number of planes required by its nonblocking condition.) To answer this question, in the next section we will conduct a detailed performance comparison among the available routing algorithms in terms of blocking probability, hardware cost, complexity and load-balancing capability.

It is notable that the load-balancing capability of a routing algorithm is also important for the multi-$\log_2 N$ network, since it directly affects the peak power requirement and power dispassion requirement (mainly determined by the maximum number of connections simultaneously supported by a plane). We will see in the next section that although the most available algorithms for multi-$\log_2 N$ networks can achieve a low blocking probability, they usually result in a very uneven load distribution among all planes. To provide a better load balance among all planes of a multi-$\log_2 N$ network, we propose here two new routing algorithms, namely the load sharing (LS) algorithm and low-load minimum index (LMI) algorithm as follows.

The main idea of the LS algorithm, as contrasted to the P algorithm, is to route request in the least occupied plane first, the second least occupied plane and so on until the first available one emerges.

The LMI algorithm will route a request in the least occupied plane number first, the second least occupied plane second and so on until the first available one emerges. The purpose of the LMI algorithm is also the same as the LS algorithm, however, the LMI algorithm do not need the sorting operation of the connection loading table.

4 Experimental Results

In this section, we will conduct an extensive simulation study to compare the performance of above nine routing algorithms in terms of their blocking probability, time complexity, hardware cost and load balancing capability. Our simulation program consists of two main modules: a request frame generator and a request router. The request frame generator generates request frames based on the

occupancy probability r of each input or output port (e.g.,r is the probability that an input/output port is busy) and the order of all requests in a frame is determined randomly. For a request frame, the request router module will apply a specified routing algorithm to route the requests in the frame one by one sequentially according their order. Our simulation program is implemented in C on a cluster workstation - Opteron 2.0 GHz cluster and all together 10^7 request frames are generated in our simulation.

Following the tagged-path based analysis as that of the studies in [3,4,7], we will focus on a tagged-request in the blocking probability simulation. For each request frame, we define the tagged-request as the last request in the frame (e.g., for the frame in the Example 1, the tagged request is $\langle 15, 0\rangle$).

4.1 Network Size Versus Blocking Probability

To compare the blocking probability of various routing algorithms, we simulated two network configurations, $\mathrm{Log}_2(128, m)$ and $\mathrm{Log}_2(256, m)$, for different m and workload (r). For reference, we also included in our simulation the upper bound and the lower bound on the blocking probability established in [3]. The corresponding results are summarized in the Fig. 3 and Fig. 4, respectively.

As mentioned earlier, we focus on the tagged request of each frame in the simulation of blocking probability, where the blocking probability (BP) is calculated as follows.

$$BP = \frac{\text{blocked times of the tagged request}}{\text{iteration times}} \tag{1}$$

The results in both Fig. 3 and Fig. 4 indicate that all routing algorithms could be roughly divided into three groups based on their blocking probability. The group of algorithms with high BP, say g_h, consists of R, LS, LMI, CS and CD algorithms. The group of algorithms with middle BP, say g_m, consists of STU algorithm. The group of algorithm with low BP, say g_l, consists of MI, P and D algorithms. It is interesting to see from the above two figures that although the nonblocking conditions of the first six routing algorithms is the same, their blocking probability are very different. E.g., the BP of MI and P algorithms in g_l are very close to that of the D algorithm, which results in the lowest blocking probability among all algorithms, but the algorithms in g_m have a significantly higher BP than that of the g_l. It is notable, however, all the nine routing algorithms have a much lower BP than the upper bound established in [3].

4.2 Hardware Cost Versus Blocking Probability

When the upper limit on blocking probability is set as 10^{-6}, the minimum number of planes required by different algorithms are summarized in Fig. 5. For comparison, we also include in Fig. 5 the number of planes required by the nonblocking conditions [7]. It is interesting to notice the required hardware

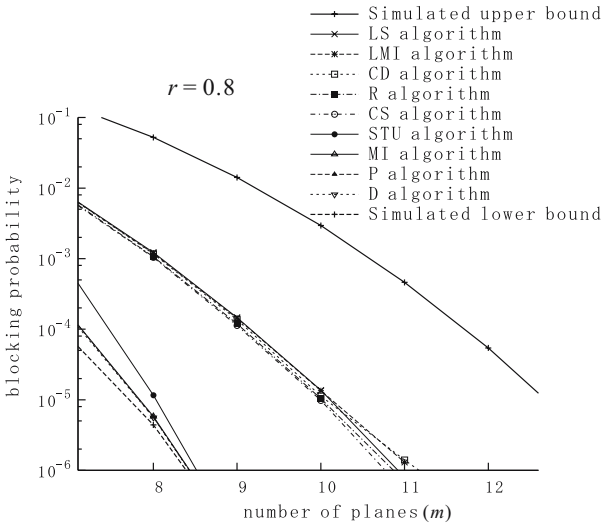

Fig. 3. Blocking probability of each algorithm on $\text{Log}_2(128, m)$ networks (odd number of stages)

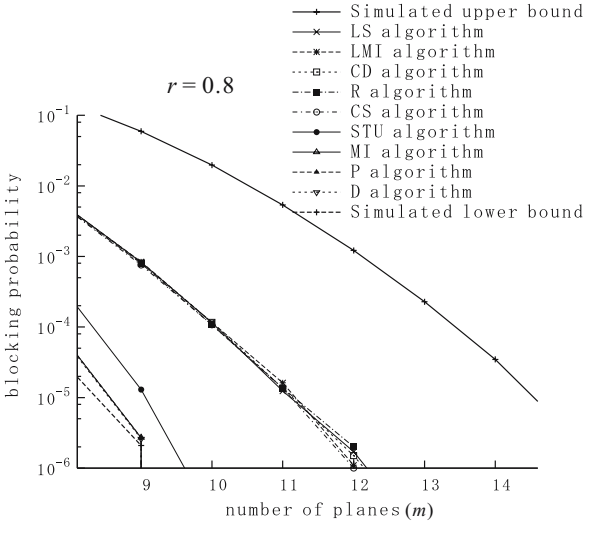

Fig. 4. Blocking probability of each algorithm on $\text{Log}_2(256, m)$ networks (even number of stages)

cost (number of planes) of all routing algorithms is much less than that of the nonblocking condition even when a high requirement on BP is applied (BP is less than 10^{-6} here). We can also observe from the Fig. 5 the hardware cost of routing algorithms in g_m and in g_l are all similar and close to the lower bound.

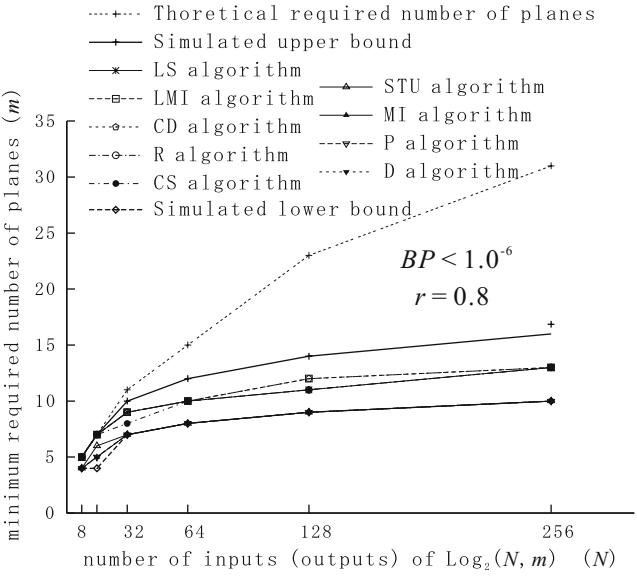

Fig. 5. Minimum number of planes for different size networks with blocking probability $< 1.0^{-6}$ on $r = 1.0$

In contrast, the hardware cost for algorithms in g_h is not very high. It is also notable that the required hardware cost of P and MI algorithms in g_l and STU algorithm in g_m is the same as that of the D algorithm and also the lower bound given by [3].

4.3 Complexity of Routing Algorithm

The complexities of all routing algorithm are summarized in Table 1. Since the worst case scenario of plane selecting in $\mathrm{Log}_2(N, m)$ is to try all m planes for a request, thus, the complexity of plane selecting is $O(m)$. We need also $O(\log_2 N)$ time to check the path availability for each request, the overall time complexity of all algorithms, except Danilewicz's algorithm (D) [6], is just $O(m{\cdot}\log_2 N)$. For the D-algorithm, its complexity is as high as $O(m \cdot N^2 log_2 N)$ since it needs to calculate many complex matrices to achieve its low BP feature .

Table 1. Complexity of routing algorithm for $\mathrm{Log}_2(N, m)$

Algorithm	Complexity of Routing Algorithm
Danilewicz's algorithm (D)	$O(m \cdot N^2 log_2 N)$
Otherwise	$O(m \cdot log_2 N)$

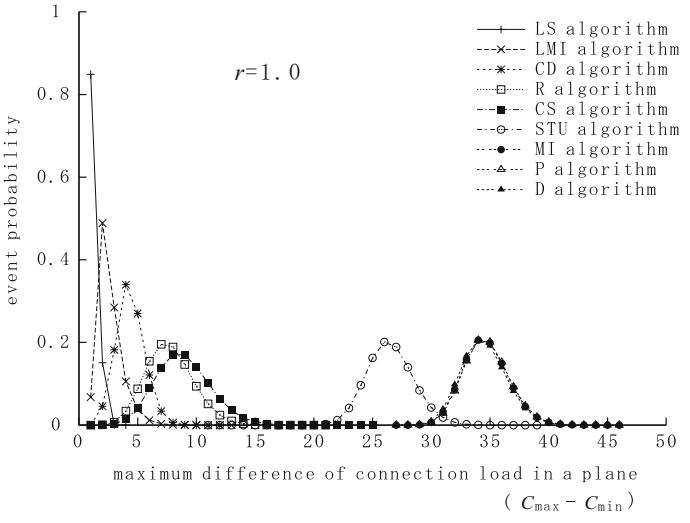

Fig. 6. Distribution of connection load on 128 × 128 multi-$\log_2 N$ network with 10 planes, $r = 1.0$

4.4 Connection Load Distribution Versus Event Probability

Let C_{max} (C_{min}) denotes the maximum (minimum) number of connections in one plane, then the smaller the difference between C_{max} and C_{min} for a algorithm, the better load-balancing capability the algorithm has. Fig. 6 illustrates the load distribution ($C_{max} - C_{min}$) for each routing algorithm in a $\text{Log}_2(128, 10)$ network with $r = 1.0$. We can see from the Fig. 6 that the LS and the LMI algorithms in g_h can achieve much better load-balancing capability than others. In contrast, the algorithms in g_l (e.g, P and D algorithm) may suffer from very heavy connection load.

5 Conclusion and Remarks

The blocking design of multi-$\log_2 N$ networks is an attractive to reduce their hardware cost. In this paper, we fully compared the seven available routing algorithms and also two newly proposed algorithms for multi-$\log_2 N$ networks in terms of blocking probability, complexity and load-balancing capability. We found that the routing algorithms which try to pack requests into few number of planes usually achieve a lower blocking probability, but such algorithms require high peak power supply and also advanced power dispassion. In contrast, our two newly proposed routing algorithms provide a good load-balancing capability but may result in a higher blocking probability. Thus, how to design a routing algorithm to achieve a nice trade-off between a low blocking probability and a good load-balancing capability is an interesting future work.

Acknowledgement

This research is supported partially by Scientic-Aid grant No.17300010, JSPS.

References

1. Lea, C.T.: Multi-$\log_2 N$ Networks and Their Applications in High-Speed Electronic and Photonic Switching Systems. IEEE Trans. Commun. 38(10) (October 1990)
2. Chang, F.H., Guo, J.Y., Hwang, F.K.: Wide-sense nonblocking for multi-$log_d N$ networks under various routing strategies. Theoretical Computer Science (October 2005)
3. Jiang, X., Shen, H., Khandker, M.R., Horiguchi, S.: Blocking Behaviors of Crosstalk-Free Optical Banyan Networks on Vertical Stacking. IEEE/ACM Trans. Networking 11(6) (Decmeber 2003)
4. Jiang, X., Ho, P.H., Horiguchi, S.: Performance Modeling for All-Optical Photonic Switches Based on the Vertical Stacking of Banyan Network Structures. IEEE JSAC 23(8) (August 2005)
5. Lea, C.T., Shyy, D.J.: Tradeoff of Horizontal Decomposition Versus Vertical Stacking in Rearrangeable Nonblocking Networks. IEEE Trans. Commun. 39(6) (June 1991)
6. Danilewicz, G., Kabacinski, W., Michalski, M., Zal, M.: Wide-Sense Nonblocking Multiplane Photonic Banyan-Type Switching Fabrics With Zero Crosstalk. In: Proc. IEEE ICC 2006, June 2006, Istanbul, Turkey, pp. 11–15 (2006)
7. Vaez, M.M., Lea, C.: Strictly Nonblocking Directional-Coupler-Based Switching Networks Under Crosstalk Constraint. IEEE Trans. Commun. 48(2) (February 2000)
8. Maier, G., Pattavina, A.: Design of Photonic Rearrangeable Networks with Zero First-Order Switching-Element-Crosstalk. IEEE Trans. Commun. 49(7) (July 2001)
9. Ackroyd, M.H.: Call Repacking in Connecting Networks. IEEE Trans. Commun. com-27(3) (March 1979)
10. Jajszczyk, A., Jekel, G.: A New Concept - Repackable Networks. IEEE Trans. Commun. 41(8) (August 1993)
11. Mun, Y., Tang, Y., Devarajan, V.: Analysis of Call Packing and Rearrangement in a Multi Stage Switch. IEEE Trans. Commun. 42(2/3/4) (1994)
12. Yang, Y., Wang, J.: Wide-Sense Nonblocking Clos Networks under Packing Strategy. IEEE Trans. Computers 48(3) (March 1999)
13. Papadimitriou, G.I., Papazoglou, C., Pomportsis, A.S.: Optical Switching: Switch Fabrics, Techniques, and Architectures. Journal of lightwave technology 21(2) (February 2003)