A Fast Parallel Routing Algorithm for Strictly Nonblocking Switching Networks

Enyue Lu[†], S. Q. Zheng[‡], and Bing Yang[◇]
[†]Dept. of Mathematics and Computer Science, Salisbury University, Salisbury, MD 21801
[‡]Dept. of Computer Science, University of Texas at Dallas, Richardson, TX 75083
[◇]Cisco Systems, Inc., Richardson, TX 75082
[†]ealu@salisbury.edu, [‡]sizheng@utdallas.edu, [◇]binyang@cisco.com

Abstract - A class of strictly nonblocking (SNB) networks $B(N,p,\alpha)$ can be constructed from vertical stacking of multiple planes of Banyan networks. Fast routing algorithms are needed for finding available connection paths in $B(N,p,\alpha)$ networks. In this paper, by modeling the switching routing problem in SNB networks as strong edge coloring problem, we propose a simple and fast parallel routing algorithm for routing connections in SNB $B(N,p,\alpha)$ networks. The proposed algorithm can route connections in SNB $B(N,p,\alpha)$ networks in $O(\sqrt{N})$ time using a completely connected multiprocessor system of N processing elements. Our algorithm can be translated into algorithms with an $O(\lg N \lg\lg N)$ slowdown factor for the class of N-processor hypercubic networks, whose structures are no more complex than a single plane of $B(N,p,\alpha)$ networks.

Keywords: Banyan networks, crosstalk, strictly nonblocking networks, graph coloring, parallel algorithm.

1 Introduction

A switching network usually comprises a number of switching elements (SEs), grouped into several stages interconnected by a set of links. In an electrical switching network, links are wires and SEs are crossbar switches. In an optical switching network, links are implemented by optical waveguides and SEs can be implemented by electro-optical SEs such as common lithium-niobate ($LiNbO_3$) SEs (e.g. [5, 6, 18]). Without loss of generality, we assume that the size of an SE is 2×2, i.e. each SE has 2 inputs and 2 outputs. In a switching network, if two inputs (resp. outputs) of an SE intend to be connected with the same output (resp. input), *output link conflict* (resp. *input link conflict*) occurs.

An electronically controlled optical SE can have switching speed ranging from hundreds of picoseconds to tens of nanoseconds [17]. However, due to the nature of optical devices, optical switches introduce additional challenges. One is *crosstalk*[1] problem, which is caused by undesired coupling between signals with the same wavelength carried in two waveguides so that two signal channels interfere with each other within an SE. The crosstalk problem in photonic switching networks adds a new dimension of blocking, called *node conflict*, which happens when more than one connection with the same wavelength passes through the same SE at the same time. If an I/O connection path does not have any conflict with other connection paths, it is called a *conflict-free* path.

Nonblocking switching networks have been favored in switching systems because they can be used to set up any conflict-free one-to-one I/O connection paths. There are three types of nonblocking networks: *strictly nonblocking (SNB)*, *wide-sense nonblocking (WSNB)* and *rearrangeable nonblocking (RNB)* [2, 7]. In both SNB and WSNB networks, a connection can be established from any idle input to any idle output without disturbing existing connections. In SNB networks any of available conflict-free paths for a connection can be chosen and in WSNB networks, however, a rule must be followed to choose one. RNB networks can establish a conflict-free path for the connection from any idle input to any idle output if the rearrangement of existing connections is allowed.

Recently, a class of multistage nonblocking switching networks has been proposed. In this class each network, denoted by $B(N,x,p,\alpha)$, has relatively low hardware cost $O(N^{1.5} \lg N)$ and short connection diameter $O(\lg N)$ in terms of the number of SEs. A $B(N,x,p,\alpha)$, $\alpha \in \{0,1\}$, is constructed by horizontally concatenating $x(\leq n-1)$ extra stages to an $N \times N$ Banyan-type network, and then vertically stacking p copies of the extended Banyan[2]. $B(N,x,p,0)$ and $B(N,x,p,1)$ are similar in structure, but the latter does not allow any two connec-

[1] In this paper, the crosstalk is referred to the first-order non-filterable SE crosstalk [14, 15].

[2] In this paper, $N = 2^n$ ($n = \lg N$) and all logarithms are in base 2.

tions with the same wavelength passing through the same SE at the same time while the former does. $B(N, x, p, 0)$ and $B(N, x, p, 1)$ are suitable for electronic and optical implementation, respectively. It has been shown that $B(N, x, p, \alpha)$ can be SNB, WSNB and RNB with certain values of x and p for given N and α [8, 9, 15, 19, 20].

In a switching network, when more than one input requests to be connected with the same output, output contention occurs. Output contentions can be resolved by switch scheduling. For a set of connection requests without output contention, the process of establishing conflict-free connection paths to satisfy these requests is called *switch routing*. A switch routing (or simply, routing) algorithm is needed to find these paths. Once a set of conflict-free paths is found, the SEs on these paths can be properly set up. Routing algorithms play a more fundamental role in WSNB and RNB networks since the nonblockingness depends on them. For SNB networks, routing algorithms tend to be overlooked since a conflict-free path is always guaranteed for the connection from any idle input to any idle output without rerouting the existing connections. An efficient routing algorithm, however, is still needed to find such a conflict-free path for each connection request. Any routing algorithm requiring more than linear time would be considered too slow. Thus, finding efficient algorithms to speed up routing process is crucial for high-speed switching networks.

The focus of this paper is studying the control aspect of the class $B(N, 0, p, \alpha)$ networks, simply as $B(N, p, \alpha)$, in the context of being used as electrical and optical switching networks. In particular, our objective is to speed up routing process in SNB $B(N, p, \alpha)$ networks using parallel processing techniques. By examining the connection capacity of $B(N, p, \alpha)$, we reduce the routing problems for this class of networks to strong edge-colorings of bipartite graphs. Basing on our model, we propose a fast routing algorithm for $B(N, p, \alpha)$ using parallel processing techniques. We show that the presented parallel routing algorithm can route any set of $O(N)$ connections in SNB $B(N, p, \alpha)$ networks in $O(\sqrt{N})$ time, which improves the best known algorithm with time complexity $O(\lg N\sqrt{N})$in [12].

The remainder of this paper is organized as follows. In Section 2, we discuss the topology of $B(N, p, \alpha)$. In Section 3, we model routing in $B(N, p, \alpha)$ as strong edge coloring problems of an I/O mapping graph $G(N, K, g)$. In Section 4, we present a fast parallel routing algorithm for SNB $B(N, p, \alpha)$ networks. We conclude our paper in Section 5.

2 Nonblocking Networks Based on Banyan Networks

2.1 Banyan-type Networks

A class of multistage self-routing networks, *Banyan-type* networks, has received considerable attention. A network belonging to this class satisfies the properties of short connection diameter, unique connection path, uniform modularity, etc. Banyan-type networks are very attractive for constructing switching networks. Several well-known networks, such as *Banyan*, *Omega*, and *Baseline*, belong to this class. It has been shown that these networks are topologically equivalent [1, 21]. In this paper, we use Baseline network as the representative of Banyan-type networks.

An $N \times N$ Baseline network, denoted by $BL(N)$, is constructed recursively. A $BL(2)$ is a 2×2 SE. A $BL(N)$ consists of a switching stage of $N/2$ SEs, and a shuffle connection, followed by a stack of two $BL(N/2)$s. Thus a $BL(N)$ has n stages labeled by $0, \cdots, n-1$ from left to right, and each stage has $N/2$ SEs labeled by $0, \cdots, N/2-1$ from top to bottom. The upper and lower outputs of each SE in stage i are connected with two $BL(N/2^{i+1})$s. The N links interconnecting two adjacent stages i and $i+1$ are called *output links* of stage i and *input links* of stage $i+1$. The input (resp. output) links in the first (resp. last) stage of $BL(N)$ are connected with N inputs (resp. outputs) of $BL(N)$. To facilitate our discussions, the labels of stages, links, SEs, inputs and outputs are all represented by binary numbers. An example is shown in Fig. 1.

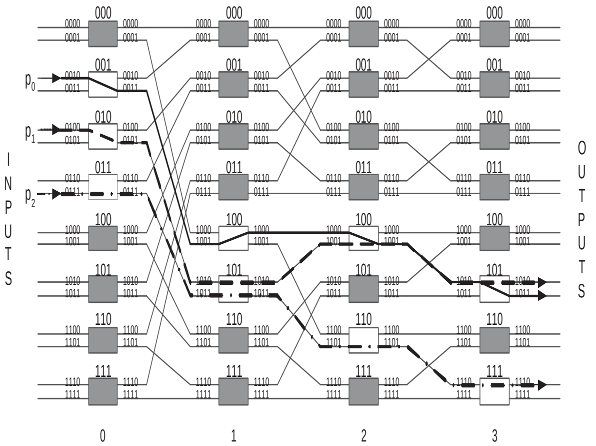


Figure 1: $BL(16)$.

$BL(N)$ is self-routing networks. The self-routing in $BL(N)$ is decided by the destination, $d_{n-1}d_{n-2}\cdots d_0$, of each connection. If $d_{n-i-1} = 0$, the input of the SE on the connection path in stage i is connected to the SE's upper output, and to the lower output otherwise (i.e., $d_{n-i-1} = 1$). As shown in Fig. 1, connection paths P_0, P_1, and P_2 are set up by self-routing in $BL(16)$. By this self-routing property, we have the following simple fact:

Lemma 1 *Given any $O(N)$ one-to-one distinct input/output pairs, the unique paths in $BL(N)$ for these pairs can be computed in $O(\lg N)$ time using N processing elements (PEs) if each PE is assigned to $O(1)$ pairs.*

2.2 Structure of $B(N, x, p, \alpha)$ Networks

If Baseline network is used for photonic switching, it is a blocking network since two connections may pass through the same SE, which causes node conflict. Even if Baseline network is used for electronic switching, it is still a blocking network since two connections may try to pass through the same input (resp. output) link, which causes input (resp. output) link conflict. Fig. 1 shows three connection paths P_0, P_1, and P_2. P_0 and P_1 have link and node conflicts in stages 2 and 3. P_1 and P_2 have node conflict in stage 1.

Although a Baseline network is blocking, a nonblocking network can be built by extending it in three ways: horizontal concatenation of extra stages to the back of a Baseline network, vertical stacking of multiple copies of a Baseline network, and the combination of both horizontal concatenation and vertical stacking [8, 9, 19, 20]. In the general approach, a network is constructed by concatenating the mirror image of the first $x (< n)$ stages of $BL(N)$ to the back of a $BL(N)$ to obtain $BL(N, x)$, then vertically making p copies of $BL(N, x)$, the extended $BL(N)$ (each copy is called a *plane*), and finally connecting the inputs (resp. outputs) in the first (resp. last) stage to N $1 \times p$ splitters (resp. $p \times 1$ combiners). Specifically, the i-th input (resp. output) of the j-th plane is connected with the j-th output (resp. input) of the i-th $1 \times p$ splitter (resp. $p \times 1$ combiner), which is connected with the i-th input (resp. output) of this network. We denote a network constructed in this way by $B(N, x, p, \alpha)$, where α is *crosstalk factor*: $\alpha = 0$ if the network has no crosstalk-free constraint (i.e. the network has only link conflict-free constraint) and $\alpha = 1$ if the network has crosstalk-free constraint (i.e. the network has node conflict-free constraint). If $x = 0$, $B(N, x, p, \alpha)$ becomes $B(N, p, \alpha)$. In this paper, we focus on designing fast routing algorithm for a class of SNB $B(N, p, \alpha)$ networks. Fig. 2 shows the structure of $B(8, 3, \alpha)$.

For $B(N, p, \alpha)$, let I be a set of N inputs, $I_0, \cdots, I_{N-1}$, and O be a set of N outputs, $O_0, \cdots, O_{N-1}$. Let $g = 2^i$, $0 \leq i \leq n$. The k-th *modulo-g input group* comprises inputs $I_{(k-1)g}, I_{(k-1)g+1}, \cdots, I_{kg-1}$, and the k-th *modulo-g output group* comprises outputs $O_{(k-1)g}, O_{(k-1)g+1}, \cdots, O_{kg-1}$, where $1 \leq k \leq N/g$. Let $\pi : I \longmapsto O$ be an I/O *mapping* that indicates connections from I to O. If there is a connection from I_i to O_j, then set $\pi(i) = j$ and $\pi^{-1}(j) = i$; otherwise set $\pi(i) = -1$. If $j \neq \pi(i)$ for any I_i, then set $\pi^{-1}(j) = -1$. We say that an input (resp. output, link, SE) is *active* if it is on a connection path, and *idle* otherwise. An I/O mapping from I to O is *one-to-one* if each I_i is mapped to at most one O_j and $\pi(i) \neq \pi(j)$ for any $i \neq j$. In this paper, all I/O mappings are one-to-one and all connections belong to a one-to-one I/O mapping.

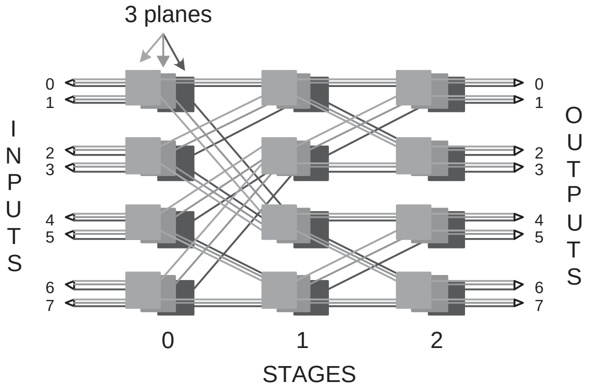


Figure 2: A network $B(8, 3, \alpha)$.

2.3 Designing Parallel Switch Routing Algorithms

A trivial lower bound on the time for routing K $(0 \leq K \leq N)$ connections sequentially in $B(N, p, \alpha)$ is $\Omega(K \lg N)$. This lower bound is obtained by Lemma 1 and assuming that for any connection it takes $O(1)$ time to correctly guess which plane to use without causing conflict. Clearly, when the number of connection requests is large, the routing time complexity is greater than $O(N)$. Parallel processing techniques should be used to meet the stringent timing requirement [7]. In [12], we proposed a parallel routing algorithm with time complexity $(\lg N \sqrt{N})$ for $B(N, p, \alpha)$ on a completely connected multiprocessor system.

In this paper, we try to improve the time complexity to $O(\sqrt{N})$ using graph coloring approach. We choose to present our parallel algorithm on a completely connected multiprocessor system. A completely connected multiprocessor system of size N consists of N processing elements (PEs), PE_i, $0 \leq i \leq N-1$, connected in such a way that there is a connection between every pair of PEs. We assume that each PE can communicate with at most one PE during a communication step. The time complexity of an algorithm on such a multiprocessor system is measured in terms of the total number of parallel computation and communication steps required by the algorithm. Such a multiprocessor system is by no means to be practical, but used as a general abstract model to derive parallel algorithms. Efficient algorithms on more realistic models, such as the class of hypercubic parallel computers, whose architectural complexity is the same as that of a single plane of $B(N,p,\alpha)$, can be easily obtained from our algorithms.

3 Graph Model

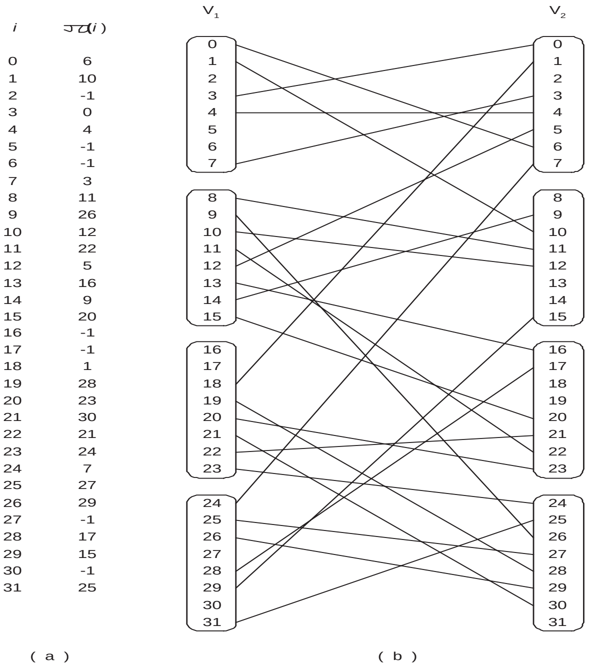
3.1 I/O Mapping Graphs

Given any I/O mapping with K connections for $B(N,p,\alpha)$, we construct a graph $G(N,K,g)$, named *I/O mapping graph*, as follows. The vertex set consists of two parts, $V_1 = \{v'_1, v'_2, \cdots, v'_{N/g}\}$ and $V_2 = \{v''_1, v''_2, \cdots, v''_{N/g}\}$. Each modulo-$g$ input (resp. output) group is represented by a vertex in V_1 (resp. V_2). There is an edge between vertex $\lfloor i/g \rfloor$ in V_1 and vertex $\lfloor j/g \rfloor$ in V_2 if $j = \pi(i)$. Thus, $G(N,K,g)$ is a bipartite graph with N/g vertices in each of V_1 and V_2 and K edges, where at most g edges are incident at any vertex. Clearly, the *degree* of $G(N,K,g)$, the maximum number of edges incident at a vertex, is no larger than g. Since there may be more than one connection from a modulo-g input group to the same modulo-g output group, $G(N,K,g)$ may have parallel edges between two vertices and it may be a multigraph. Fig. 3 (a) shows an I/O mapping with 32 inputs, 25 of which are active. Fig. 3 (b) shows the I/O mapping graph $G(32,25,8)$ of Fig. 3 (a), where V_1 (resp. V_2) of $G(32,25,8)$ has 4 vertices and each vertex in V_1 (resp. V_2) includes 8 inputs (resp. outputs) belonging to the same modulo-8 input (resp. output) group.

3.2 Graph Coloring and Nonblockingness

We say that two connections *share* a modulo-g input (resp. output) group if their sources (resp. destinations) are in the same modulo-g input (resp. output) group.

Lemma 2 *For any connection set C of $B(N,1,\alpha)$, if no two connections in C share any modulo-g input (resp. output) group, then the connection paths for C satisfy the following conditions:*
(i) they are node conflict-free in the first (resp. last) $\lg g$ stages;
(ii) they are input link conflict-free in the first $\lg g + 1$ (resp. last $\lg g$) stages and output link conflict-free in the first $\lg g$ (resp. last $\lg g + 1$) stages.



i	$\pi(i)$
0	6
1	10
2	-1
3	0
4	4
5	-1
6	-1
7	3
8	11
9	26
10	12
11	22
12	5
13	16
14	9
15	20
16	-1
17	-1
18	1
19	28
20	23
21	30
22	21
23	24
24	7
25	27
26	29
27	-1
28	17
29	15
30	-1
31	25

Figure 3: (a) An I/O mapping π; (b) An I/O mapping graph $G(32,25,8)$.

It is easy to verify that Lemma 2 is true according to the topology of $BL(N)$ (refer to [13] for formal proof).

We say that a set C of I/O connections is *feasible* for $B(N,p,0)$ (resp. $B(N,p,1)$) if they can be routed without any link (resp. node) conflict. Using the above lemma, the following claim can be easily derived from the results of [15].

Lemma 3 *Given a connection set C of $B(N,1,\alpha)$, if any two connections in C do not share any modulo-$2^{\lfloor \frac{n+\alpha}{2} \rfloor}$ input group and also do not share any modulo-$2^{\lfloor \frac{n+\alpha}{2} \rfloor}$ output group, then C is feasible for $B(N,1,\alpha)$.*

By Lemma 3, if we assign the connections of $B(N,p,\alpha)$ with sources (resp. destinations) passing through the same modulo-g input (resp. output) group to different planes, then we can route connections in $B(N,p,\alpha)$ without conflict. Thus, in order to route conflict-free connections in $B(N,p,\alpha)$, we only need to determine which plane to be used for each connection. To achieve this goal, we decompose a set of connections into disjoint subsets, and route each subset in one plane of $B(N,p,\alpha)$ so that each subset is feasible for its assigned plane. By constructing an I/O mapping graph $G(N,K,g)$ with $g = 2^{\lfloor \frac{n+\alpha}{2} \rfloor}$, we can reduce the problem of routing K connections in $B(N,p,\alpha)$ to the following strong edge graph coloring problem:

Strong Edge Coloring Problem (SEC problem): Given an I/O mapping graph $G(N,K,g)$ with $K_0(< K)$ colored edges, color $K - K_0$ uncolored edges with a set of colors such that no two edges with the same color are incident at the same vertex of $G(N, K, g)$ without changing the colors of the K_0 colored edges. If we can find a strong edge-coloring of $G(N, K, g)$ using at most c different colors, we call this coloring *a strong* c*-edge coloring* of $G(N, K, g)$.

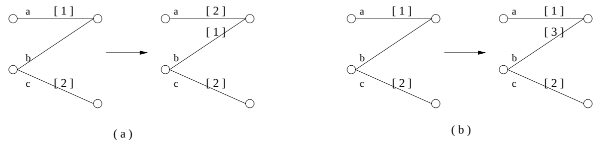


Figure 4: (a) A edge-coloring (b) A strong edge-coloring.

If we consider the colored (resp. uncolored) edges in $G(N, K, g)$ as the existing (resp. new) connections in $B(N, p, \alpha)$, a solution to the SEC problem is a plane assignment for routing in an SNB network since rerouting existing connections is prohibited. In Fig. 4, we show a simple example. There are three edges labeled a, b, c, respectively. Edges a and b have already been colored using colors 1 and 2, respectively. An edge coloring solution is given in (a), and an SEC solution is given in (b). Note that, in (b), an additional color is needed for edge b because the colors of existing colored edges a and c cannot be changed.

4 Routing in Strictly Nonblocking Networks

4.1 Strict Nonblockingness

The following lemma can be easily derived from the results of [20].

Lemma 4 *If*

$$p \geq \begin{cases} 2^{\frac{n}{2}}(\frac{3}{2} + \frac{1}{2}\alpha) - 1, & \textit{for even } n \\ 2^{\frac{n+1}{2}}(1 + \frac{1}{2}\alpha) - 1, & \textit{for odd } n \end{cases}$$

then $B(N, p, \alpha)$ *is strictly nonblocking.*

For an SNB network, we can route new connections (as long as these connections form an I/O mapping from idle inputs to idle outputs) without disturbing the existing ones; however, this routing problem is harder than that in an RNB network when we need to route the new connections simultaneously. Based on the discussions in Section 3, we know that the routing problem for an SNB $B(N, p, \alpha)$ can be solved by finding a strong edge-coloring of the I/O mapping graph $G(N, K, g)$.

We consider a subclass of SNB networks, $B(N, p^*, \alpha)$ with $p^* = 2^{\lfloor \frac{n+\alpha}{2} \rfloor + 1} - 1$. By Lemma 4, we know that $B(N, p^*, \alpha)$ is an SNB network. Since each plane of $B(N, p^*, \alpha)$ is a Baseline network, the routing of connections in any plane can be done by self-routing. Thus, the problem of routing connections in $B(N, p^*, \alpha)$ is reduced to finding a plane for each new connection so that all connections, including existing ones, are conflict-free.

Lemma 5 *Any multigraph G has a strong* $(2\Delta - 1)$*-edge coloring, where* Δ *is the degree of* G*.*

By Lemmas 3 and 5 (proved in [12]), this can be done by finding a strong $(2g-1)$-edge coloring for $G(N, K, g)$ of $B(N, p^*, \alpha)$ with K_0 existing connections and $K - K_0$ new connections, where $g = 2^{\lfloor \frac{n+\alpha}{2} \rfloor} = \frac{p^*+1}{2}$. In the next subsection, we present a parallel algorithm to find a strong $(2g - 1)$-edge coloring of $G(N, K, g)$ using strong edge coloring approach.

4.2 Algorithm for Strong $(2g-1)$-Edge Coloring of $G(N, K, g)$

Let $G(N, K - K_0, g)$ denote the graph obtained from $G(N, K, g)$ by removing the K_0 colored edges. Since $G(N, K, g)$ is a bipartite multigraph, $G(N, K - K_0, g)$ is also a bipartite multigraph. The edges between the same two vertices are called *parallel* edges. We say color c is *free* at vertex v if none of edges adjacent to v has color c. If color c is free at two ends of edge e, then c is *free* for e. One edge e is *conflict* with another edge f if e and f are adjacent to each other and they have the same color. Let $E_{i,j} = \{e_{i,j} = v'_i v''_j | e_{i,j} \in G(N, K - K_0, g)\}$. Thus, $E_{i,j}$ contains all uncolored parallel edges between nodes v'_i and v''_j. Clearly, each uncolored edge is in and only in one of such $E_{i,j}$s.

Our algorithm consists of $2g$ iterations. In each iteration, we try to color a set of non-parallel uncolored edges using one of colors in a set of $2g$ colors, $\{0, 1, \cdots, 2g-1\}$, so that no two edges with the same color adjacent to the same vertex. Then for each edge e with color $2g - 1$, we recolor it by a free color in $\{0, 1, \cdots, 2g - 2\}$.

In order to find a set of non-parallel uncolored edges in each iteration, we need a preprocessing step. For each vertex v'_i, we can sort all parallel edges in $E_{i,j}$ in nondecreasing order of is where is are the input labels corresponding to edges. The sorting for each $E_{i,j}$ can be done in $O(\lg^2 |E_{i,j}|)$ time using $|E_{i,j}|$ PEs. Thus, the preprocessing step can be done in $O(\lg^2 g)$ time using N PEs since $|E_{i,j}| \leq g$ and $\sum_{i,j} |E_{i,j}| = N$. After this preprocessing, the operation of finding uncolored non-parallel edges can be done in $O(1)$ time in each iteration. The outline of the algorithm is listed in Algorithm 1.

The correctness of this algorithm can be derived from the following facts.

Algorithm 1 A Strong Edge Coloring of an I/O Mapping Graph $G(N, K, g)$

```
for $l = 0$ to $2g - 1$ do
    for all $i, j \in \{1, 2, \cdots, N/g\}$ do
        $c_{i,j} := (i + j + l) \bmod 2g$;
        if there is an uncolored edge in $E_{i,j}$ and color $c_{i,j}$ is free at both $v'_i$ and $v''_j$ then
            assign color $c_{i,j}$ to this edge;
            update free colors at $v'_i$ and $v''_j$ and remove the colored edge from $E_{i,j}$;
        end if
    end for
end for
for all edges with color $2g - 1$ do
    color these edges with one of free colors in $\{0, 1, \cdots, 2g - 2\}$;
end for
```

(i) In iteration i, one uncolored edge, if any, in each $E_{i,j}$ is selected. Fact (i) is assured by preprocessing step.

(ii) In iteration i, if two edges, one in $E_{i,j}$ and one in $E_{p,q}$, are assigned the same color, i.e. $c_{i,j} = c_{p,q}$, then $i \neq p$ and $j \neq q$. Fact (ii) can be proved by contradiction as follows. Assume there are two pairs of (i, j) and (i, q) with $j \neq q$ and $c_{i,j} = c_{i,q}$. (For the case that there are two pairs of (i, j) and (p, j) with $i \neq p$ and $c_{i,j} = c_{p,j}$, the proof is similar). Thus, there is l so that $i+j+l \equiv i+q+l \bmod 2g$. Then $j - q = 2g \cdot x$ where x is a nonnegative iteger. Since $j, q \in \{1, 2, \cdots, N/g\}$ and $g = 2^{\lfloor \frac{n+\alpha}{2} \rfloor}$, we have $j - q < 2g$. Thus, $x = 0$ and $j = q$, which contradicts the assumption.

(iii) For uncolored edges in $G(N, K - K_0, g)$, all $2g$ possible colors are tried. Fact (iii) is obviously true from the algorithm.

(iv) After $2g$ iterations, none of adjacent edges is assigned the same color $2g$. By Fact (iii), it is clear for any non-parallel edges. By preprocessing, we know that any two parallel edges are colored in different iterations. Since there are total $2g$ iterations and in each iteration we assign different colors to the edges in $E_{i,j}$, fact (iv) is true.

(v) The edges with the same color $2g$ can be recolored concurrently using the colors in $\{0, 1, \cdots, 2g-2\}$ so that none of adjacent edges is assigned the same color. By Lemma 5, for any edge e with color $2g$, we know such a free color in $\{0, 1, \cdots, 2g - 2\}$ is available. Since all edges with original color $2g$ are not adjacent to each other by fact (ii), the recoloring will not result in any conflict colors.

Now, we show that this algorithm can be implemented in $O(g)$ time using a completely connected multiprocessor system of N PEs. By the previous discussion, we know that the preprocessing step takes $O(\lg^2 g)$ time using a completely connected multiprocessor system of N PEs. Then we show that, each of the $2g$ iterations takes $O(1)$ time. We associate a $2g$-bit binary array $C_v[0 \ldots 2g - 1]$ with each vertex v of $G(N, K, g)$ such that $C_v[c] = 1$ if and only if color c is free at vertex v, and assign $g/2$ PEs to v. Then the operations of finding out if a given color c is free at v and updating $C_v[c]$ can be carried out in $O(1)$ time. Finally, the recoloring of the edges with color $2g$ can be done in $O(\lg g)$ since the degree of $G(N, K, g)$ is g. In summary, we have the following result:

Theorem 1 *For any I/O mapping graph $G(N, K, g)$ with $K_0 (< K)$ colored edges, a strong $(2g - 1)$-edge coloring can be found in $O(g)$ time using a completely connected multiprocessor system of N PEs.*

4.3 Performance Analysis

Since $O(g) = O(\sqrt{N})$ in $G(N, K, g)$, by Lemma 1 and Theorem 1, we summarize the overall performance of our routing algorithm for SNB network $B(N, p^*, \alpha)$ by the following theorem.

Theorem 2 *For an SNB network $B(N, p, \alpha)$ with $p \geq p^* = 2^{\lfloor \frac{n+\alpha}{2} \rfloor + 1} - 1$, connections from any $K - K_0$ idle inputs to any $K - K_0$ idle outputs, with K_0 existing connections, can be correctly routed in $O(\sqrt{N})$ time using a completely connected multiprocessor system of N PEs.*

By Lemma 4, we can derive the minimum number of planes, $p_{\min}$, in $B(N, p, \alpha)$ as follows: If there is no crosstalk-free constraint (i.e., $\alpha = 0$), then $p_{\min} = \frac{3}{2}2^{\frac{n}{2}} - 1$ for even n and $p_{\min} = 2^{\frac{n+1}{2}} - 1$ for odd n. If there is a crosstalk-free constraint (i.e., $\alpha = 1$), then $p_{\min} = 2^{\frac{n}{2}+1} - 1$ for even n and $p_{\min} = \frac{3}{2}2^{\frac{n+1}{2}} - 1$ for odd n. Compared with $B(N, p_{\min}, \alpha)$, the hardware redundancy $p_{red} = p^* - p_{\min}$ of $B(N, p^*, \alpha)$ is: $p_{red} = 0$ if $\alpha = 0$ and n is odd or $\alpha = 1$ and n is even, $p_{red} = \sqrt{N}/2$ if $\alpha = 0$ and n is even, and $p_{red} = \sqrt{2N}/2$ if $\alpha = 1$ and n is odd. The hardware cost of $B(N, p^*, \alpha)$, in terms of the number of SEs, is higher than that of $B(N, p_{\min}, \alpha)$ in half of the cases, but both have the same hardware complexity of $\Theta(N^{1.5} \lg N)$. The time for routing $O(N)$ connections, however, is improved from $\Omega(N \lg N)$ to sublinear $O(\sqrt{N})$ in the worst case.

5 Conclusion

The major contribution of this paper is the design and analysis of parallel routing algorithms for a class of strictly nonblocking switching networks, $B(N, p, \alpha)$. Although the assumed parallel machine model is a completely connected multiprocessor system of N PEs, the proposed algorithms can be transformed to algorithms for more realistic parallel computing models. Let $S(N)$ be the time for sorting N elements on a parallel machine M with N processors, then our algorithms can be implemented with a slow-down factor $S(N)$ on M. It is known that sorting N numbers on the class of hypercubic networks takes $O(\lg N \lg\lg N)$ time [4, 10]. This class of networks include hypercube, cube-connected-cycles, butterfly networks, baseline networks, reverse baseline networks, Omega networks, flip networks, de Bruijin graphs, shuffle-exchange networks, banyan networks, delta networks, bidelta networks, k-ary Butterflies, and Benes networks [10]. Our algorithms can route connections in $B(N, p, \alpha)$ with a slow-down factor $O(\lg N \lg\lg N)$ on all these realistic parallel machine models, though some have topologies that are quite different from others, whose structural complexity is no larger than one plane of $B(N, p, \alpha)$. Compared with sequential algorithms, we consider that our algorithms on realistic parallel computers provide a significant speedup, making them potentially valid and useful for large switches.

The approach of applying edge-coloring techniques to investigate the capacity and routability of RNB switching networks has been widely used (refer to [3, 7, 11, 16]). We extended this approach to SNB networks by defining strong edge-coloring. For a class of SNB banyan-based switching networks we proposed a unified mathematical formulation, namely SEC problems, for designing parallel routing algorithms using this approach. Our algorithm can find the solutions for SEC problem in sublinear time. Finding faster parallel algorithms for the SEC problem, however, remains to be very challenging.

References

[1] D.P. Agrawal, "Graph Theoretical Analysis and Design of Multistage Interconnection Networks", *IEEE Transactions on Computers*, vol. C-32, no. 7, pp. 637-648, July 1983.

[2] V.E. Benes, *Mathematical Theory of Connecting Networks and Telephone Traffic*, Academic Press, New York, 1965.

[3] J. Carpinelli and A. Y. Oruc, "Applications of Matching and Edge-Coloring Algorithms to Routing in Clos Networks", *Networks*, vol. 24, pp. 319-326, Sep. 1994.

[4] R. Cypher and G. Plaxton, "Deterministic Sorting in Nearly Logarithmic Time on the Hypercube and Related Computers," *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pp. 193-203, 1990.

[5] H. Hinton, "A Non-Blocking Optical Interconnection Network Using Directional Couplers", *Proceedings of IEEE Global Telecommunications Conference*, pp. 885-889, Nov. 1984.

[6] D.K. Hunter, P.J. Legg, and I. Andonovic, "Architecture for Large Dilated Optical TDM Switching Networks", *IEE Proceedings on Optoelectronics*, vol. 140, no. 5, pp. 337-343, Oct. 1993.

[7] F.K. Hwang, *The Mathematical Theory of Nonblocking Switching Networks*, World Scientific, 1998.

[8] C.T. Lea, "Multi-log2N Networks and Their Applications in High-Speed Electronic and Photonic Switching Systems", *IEEE Transactions on Communications*, vol. 38, no. 10, pp. 1740-1749, Oct. 1990.

[9] C.T. Lea and D.J. Shyy, "Tradeoff of Horizontal Decomposition Versus Vertical Stacking in Rearrangeable Nonblocking Networks", *IEEE Transactions on Communications*, pp. 899-904, vol. 39, no. 6, June 1991.

[10] F.T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays · Trees · Hypercubes*, Morgan Kaufmann Publishers, 1992.

[11] G.F. Lev, N. Pippenger and L.G. Valiant, "A Fast Parallel Algorithm for Routing in Permutation Networks", *IEEE Transactions on Computers*, vol. 30, pp. 93-100, Feb. 1981.

[12] E. Lu and S. Q. Zheng, "Parallel Routing Algorithms for Nonblocking Electronic and Photonic Multistage Switching Networks", *Workshop on Advances in Parallel and Distributed Computing Models*, April, 2004.

[13] E.Lu, Mei Yang, Bing Yang and S. Q. Zheng, "A Class of Self-Routing Strictly Nonblocking Photonic Switching Networks", *Proceedings of IEEE Global Communications Conference*, Nov.-Dec., 2004.

[14] G. Maier, A. Pattavina, and S. G. Colombo, "Control of Non-filterable Crosstalk in Optical-Cross-Connect Banyan Architectures", *Proceedings of IEEE Global Telecommunications Conference GLOBECOM*, vol. 2, pp. 1228-1232, Nov.-Dec. 2000.

[15] G. Maier and A. Pattavina, "Design of Photonic Rearrangeable Networks with Zero First-Order Switching-Element-Crosstalk", *IEEE Transactions on Communications*, vol. 49, no. 7, pp. 1268-1279, Jul. 2001.

[16] N. Nassimi and S. Sahni, "Parallel Algorithms to Set Up the Benes Permutation Network", *IEEE Transactions on Computers*, vol. 31, no. 2, pp. 148-154, Feb. 1982.

[17] R. Ramaswami and K. Sivarajan, *Optical Networks: A Practical Perspective*, second edition, Morgan Kaufmann, 2001.

[18] G.H. Song and M. Goodman, "Asymmetrically-Dilated Cross-Connect Switches for Low-Crosstalk WDM Optical Networks", *Proceedings of IEEE 8th Annual Meeting Conference on Lasers and Electro-Optics Society Annual Meeting*, vol. 1, pp. 212-213, Oct. 1995.

[19] M. Vaez and C.T. Lea, "Wide-Sense Nonblocking Banyan-Type Switching Systems Based on Directional Couplers", *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 7, pp. 1327-1332, Sep. 1998.

[20] M. Vaez and C.T. Lea, "Strictly Nonblocking Directional-Coupler-Based Switching Networks under Crosstalk Constraint", *IEEE Transactions on Communications*, vol. 48, no. 2, pp. 316-323, Feb. 2000.

[21] C.L. Wu and T.Y. Feng, "On a Class of Multistage Interconnection Networks", *IEEE Transactions on Computers*, vol. C-29, no. 8, pp. 694-702, Aug. 1980.