Parallel Routing Algorithms for Nonblocking Electronic and Photonic Multistage Switching Networks

Enyue Lu and S. Q. Zheng
Department of Computer Science, University of Texas at Dallas, USA
{enyue, sizheng}@utdallas.edu

Abstract

Nonblocking multistage interconnection networks are favored to be used as switching networks whenever possible. Crosstalk-free requirement in photonic networks adds a new dimension of constraints for nonblockingness. Routing algorithms play a fundamental role in nonblocking networks, and any algorithm that requires more than linear time would be considered too slow for real-time applications. One remedy is to use multiple processors to route connections in parallel. In this paper, we study the connection capacity of a class of rearrangeable nonblocking and strictly nonblocking networks with/without crosstalk-free constraint, model their routing problems as weak or strong edge colorings of bipartite graphs, and propose efficient routing algorithms for these networks using parallel processing techniques.

1 Introduction

Interconnection networks have many different applications, including but not limited to, being used as interconnects for communications among processors and between processors and memory modules in a multiprocessor or multicomputer system, and as a switching network within a network router or switch. Roughly speaking interconnection networks are classified into two classes, direct (router-based) networks and indirect (switch-based) networks [3]. A typical indirect interconnection network is a multistage interconnection network (MIN). In this paper, we consider MINs in the context of their being used as switching networks. We investigate their ability of simultaneously realizing one-to-one I/O mappings in the form of permutations.

A switching network usually comprises a number of electronic or photonic switching elements (SEs) grouped into several stages interconnected by a set of wires or optical links. A photonic switching network can be built from 2×2 electro-optical SEs such as common lithium-niobate ($LiNbO_3$) SE (e.g. [4, 5]). Each SE is a directional coupler with two inputs and two outputs. Depending on the amount of voltage at the junction of two waveguides, optical signals carried on either of two inputs can be coupled to either of two outputs. An electronically controlled optical SE can have switching speed ranging from hundreds of picoseconds to tens of nanoseconds [15]. However, due to the nature of optical devices, optical switches hold their own challenges. One problem is *crosstalk*, which is caused by undesired coupling between signals carried in two waveguides so that two signal channels interfere with each other. Fig. 1 shows an example of crosstalk in an SE. Each SE has two logic states, namely, *straight* and *cross* (see Fig. 1 (a)). For the straight state, a small fraction of input signal injected at the upper input may be detected at the lower output (see Fig. 1 (b)). Crosstalk can also occur when an SE is in the cross state. Consequently, the input signal will be distorted at output due to crosstalk accumulated along connection path.

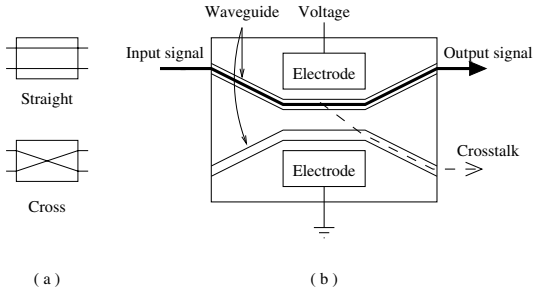

Figure 1. (a) States of an SE (b) Crosstalk in an electro-optical SE.

According to blocking properties, Switching networks are classified as *blocking* and *nonblocking*. In an SE, if two active inputs (resp. outputs) intend to be connected with the same output (resp. input), it causes *output link conflict* (resp. *input link conflict*). Crosstalk in photonic switching networks adds a new dimension of blocking, called *node conflict*, which happens when an SE has two active inputs/outputs. In order to reduce blocking effect, one approach, called *space dilation*, has been proposed. In space dilation approach, blocking can be eliminated by ensuring at most one connection passing through a link for electronic switching networks (in which there is no crosstalk-free constraint) or through both a link and an SE for photonic switching networks (in which there is the crosstalk-free constraint). More specifically, blocking can be avoided

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

by increasing the number of SEs in a switching network (e.g. [8, 9, 14, 16, 17, 18, 19]).

Nonblocking networks have been favored in switching systems since they can set up any one-to-one I/O mapping. There are three types of nonblocking networks: *strictly nonblocking (SNB)*, *wide-sense nonblocking (WSNB)* and *rearrangeable nonblocking (RNB)* [1, 6]. 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. The high degree of connection capability in SNB and WSNB networks is at a high hardware cost. RNB networks, usually constructed with lower hardware cost, 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. A network is *self-routing* if any connection is established only by the addresses of its source and destination regardless of other connections. A self-routing network can be either blocking such as a Banyan network or nonblocking such as a crossbar.

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 contentions, 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.

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 and short connection diameter, $O(N^{1.5} \lg N)$ and $O(\lg N)$ respectively, in terms of the number of SEs[1]. A $B(N, x, p, \alpha)$, $\alpha \in \{0, 1\}$, is constructed by horizontally concatenating $x (\leq \lg N - 1)$ extra stages to an $N \times N$ Banyan-type network and vertically stacking p copies of the extended Banyan. Networks $B(N, x, p, 0)$ and $B(N, x, p, 1)$ are similar in structure, but the latter does not allow any two connection paths through the same SE 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, WNB and RNB with certain values of x and p for given N and α [8, 9, 12, 17, 18]. Routing K connections sequentially in $B(N, x, p, \alpha)$ needs $\Omega(K \lg N)$ time. When the number of connection requests becomes larger, the routing time complexity is greater than $O(N)$. To the best of our knowledge, except for some special cases such as Banyan network (i.e., $B(N, 0, 1, \alpha)$) and Benes network (i.e., $B(N, \lg N - 1, 1, \alpha)$), no effort of investigating faster routing for the whole class of these networks has been reported in the literature.

In this paper, by examining the connection capacity of $B(N, x, p, \alpha)$, we first model the routing problems for this class of networks as weak and strong edge colorings of bipartite graphs. Basing on our model, we propose fast routing algorithms for $B(N, x, p, \alpha)$ using parallel processing techniques. We show that the presented parallel routing algorithms can route K connections in $O((x + \lg p) \lg K + \lg N)$ time for an RNB $B(N, x, p, \alpha)$ and in $O(\lg p^* \lg K + p^* \lg p^*)$ time for an SNB $B(N, 0, p^*, \alpha)$. Since $p, p^* = O(\sqrt{N})$ and $x < \lg N$, the proposed algorithms can always set up $O(N)$ connections in $O(\lg^2 N)$ time for RNB $B(N, x, p, \alpha)$ and in $O(\sqrt{N} \lg N)$ time for SNB $B(N, x, p^*, \alpha)$.

2 Nonblocking Networks Based on Banyan Networks

A class of multistage self-routing networks, *Banyan-type* networks, has received considerable attention. A network belonging to this class has properties of short connection diameter, unique connection path, and uniform modularity, which are very attractive for constructing switching networks. Several well-known networks, such as *Banyan*, *Omega*, *Shuffle*, and *Baseline*, belong to this class. It has been shown that these networks are topologically equivalent. 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 $\log N$ 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. Each SE has two inputs, each named *upper input* or *lower input* if it is above or under the other, and two outputs, each named *upper output* or *lower output* similarly. The upper and lower outputs of each SE in stage i are connected with two $BL(N/2^{i+1})$'s, named *upper subnetwork* and *lower subnetwork*, respectively. 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 label of each stage, link and SE is represented by a binary number. Let $a_l a_{l-1} \cdots a_1 a_0$ be the binary representation of a. We use $\bar{a}$ to denote the integer that has the binary representation $a_l a_{l-1} \cdots a_1 (1 - a_0)$. An example is shown in Fig. 2.

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

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

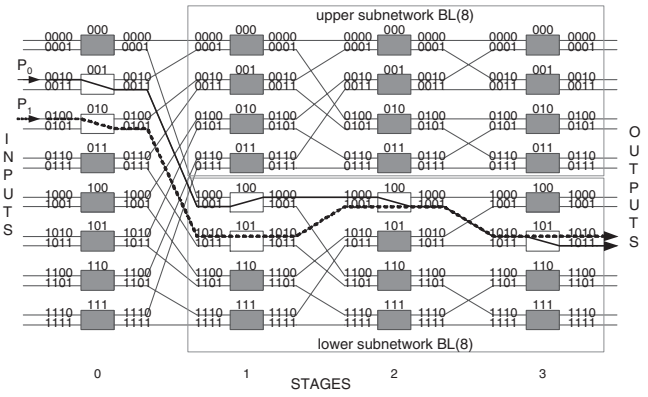

Figure 2. Self-routing connection paths P_0 and P_1 in $BL(16)$.

Self-routing in $BL(N)$ is decided by the destination of each connection. If the $(n-i)$-th bit, d_{n-i-1}, of the destination equals to 0, the input of the SE that the connection path enters in stage i is connected to the SE's upper output, and otherwise (i.e., $d_{n-i-1}=1$) to the lower output. Since two adjacent stages are connected by shuffle connection, the unique path for each connection can be derived.

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. 2 shows two connection paths P_0 from 0010 to 1011 and P_1 from 0100 to 1010. P_0 and P_1 have the output link conflict in stage 2 and input link conflict in stage 3 because both two active inputs of SE 4 in stage 2 intend to be connected with its lower output and both active outputs of SE 5 in stage 3 intend to be connected with its upper input; they have node conflicts at SEs 4 and 5 in stages 2 and 3, respectively.

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, 17, 18]. 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 $BL(N)$, then vertically making p copies of 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*. That is, $\alpha=0$ if the network has no crosstalk-free constraint and $\alpha=1$ if the network has crosstalk-free constraint. Clearly, $B(N,0,1,\alpha)$ is a Baseline network and $B(N,n-1,1,\alpha)$ is a Benes network [1]. In $B(N,x,1,\alpha)$, a *subnetwork*, denoted by $B(N,x,1/2^l,\alpha)$ $(0\le l\le n-1)$ is defined as a $B(N/2^l,\max\{x-l,0\},1,\alpha)$ from stage l to stage $n+\max\{x-l,0\}-1$. Fig. 3 shows an example of $B(16,2,3,\alpha)$, which contains three planes of $B(16,2,1,\alpha)$, and each $B(16,2,1,\alpha)$ contains two extra stages.

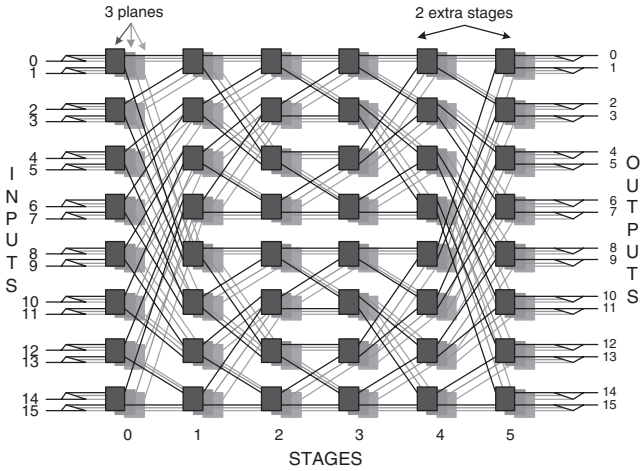

Figure 3. A network $B(16,2,3,\alpha)$.

3 Graph Model

3.1 I/O Mapping Graphs

Let I and O be the sets of N inputs, denoted by $I_0,\cdots,I_{N-1}$, and N outputs, denoted by $O_0,\cdots,O_{N-1}$, of $B(N,x,p,\alpha)$ respectively. For $B(N,x,p,\alpha)$, a set of g inputs (resp. outputs) is called the k-th *modulo-g input group* (resp. *modulo-g output group*) if the inputs (resp. outputs) in the set are congruent to $k-1$ when the modulus is $g(=2^i, 1\le i\le n)$. 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.

If a connection path does not have any link (resp. node) conflict with other connection paths, it is called a *link conflict-free* (resp. *node conflict-free*) path. Clearly node conflict-free path is also link conflict-free, but the converse is not true. If a set of connections can be set up by conflict-free paths in $B(N,x,1,\alpha)$, these connections are

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

called *feasible connections* of $B(N, x, 1, \alpha)$. Our goal is to quickly set up K link (resp. node) conflict-free paths for K connections of any I/O mapping in $B(N, x, p, 0)$ (resp. $B(N, x, p, 1)$). To achieve this goal, we usually decompose a set of connections into disjoint subsets, and route each subset in one plane of $B(N, x, p, \alpha)$ so that each subset is feasible for its assigned plane.

Given any I/O mapping with K connections for $B(N, x, p, \alpha)$, we construct a graph $G(N, K, g)$, named *I/O mapping graph*, as follows. The vertex set consists of two parts, V_1 and V_2. Each part has N/g vertices, i.e., 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. Thus, the *degree* of $G(N, K, g)$, the maximum number of edges incident at a vertex, equals to 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. However, there is a one-to-one correspondence between active inputs/outputs in the I/O mapping and edges in the I/O mapping graph, and thus, we can label each edge by its corresponding input. An edge i is called the *left edge* (resp. *right edge*) of edge j if $i = \bar{j}$ (resp. $\pi(i) = \overline{\pi(j)}$). Any edge has at most one left edge and at most one right edge in $G(N, K, g)$. Two edges i and j are called *neighboring edges* if i is the left or right edge of j. We define a *component* of $G(N, K, g)$ as follows: two edges e and f belong to the same component if and only if there is a sequence of edges $e = e_1, \cdots, e_j = f$ such that e_i and e_{i+1}, $1 \leq i \leq j-1$, are neighboring edges. If every edge in a component has two neighboring edges, it is called a *closed component*; otherwise it is called an *open component*. It is easy to verify that each edge is in exactly one component, and thus, components are edge disjoint in $G(N, K, g)$. In Fig. 4, (a) shows an I/O mapping with 32 inputs, 25 of which are active; (b) shows the I/O mapping graph $G(32, 25, 8)$ of (a), where each of V_1 and V_2 of $G(32, 25, 8)$ has 4 vertices and each vertex includes 8 inputs (resp. outputs) belonging to the same modulo-8 input (resp. output) group; (c) shows all components of $G(32, 25, 8)$ in (b).

3.2 Graph Coloring and Nonblockingness

If we set up connections in $B(N, x, p, \alpha)$ one by one by sequential algorithms, the time complexity for establishing K connections is $\Omega(K \cdot (\lg N + x))$ since it takes $\Omega(\lg N + x)$ time to set up one connection. For a large number of connections, the time required is more than $O(N)$, which is not acceptable for real-time applications. Parallel processing techniques can be used to speed up routing in $B(N, x, p, \alpha)$. 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. Let us study the connection capability of $B(N, x, p, \alpha)$ first.

Lemma 1 *For any connection set C of $B(N, 0, 1, \alpha)$, 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.*

Lemma 2 *For any pair of input and output in $B(N, x, 1, \alpha)$, there are 2^x paths connecting them.*

It is easy to verify that Lemmas 1 and 2 are true according to the topology of $BL(N)$ (refer to [12] for formal proofs). Using the above two lemmas, the following claim can be easily derived from the results of [12].

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

By Lemma 3, if we assign the connections of $B(N, x, p, \alpha)$ with sources (resp. destinations) passing through the same modulo-g input (resp. output) group to different planes, then we can route connections in $B(N, x, p, \alpha)$ without conflict. Thus, in order to set up conflict-free connections in $B(N, x, p, \alpha)$, we first need to determine which plane to be used for each connection. By constructing an I/O mapping graph $G(N, K, g)$ with $g = 2^{\lfloor \frac{n-x+\alpha}{2} \rfloor}$, we can reduce the problem of routing K connections in $B(N, x, p, \alpha)$ to the following two graph coloring problems:

Weak Edge Coloring Problem (WEC problem): Given an I/O mapping graph $G(N, K, g)$ with $K_0 (< K)$ colored edges, color K edges with a set of colors such that no two edges with the same color are incident at the same vertex of $G(N, K, g)$ with the changing of the colors of the K_0 colored edges allowed. If we can find a weak edge coloring of $G(N, K, g)$ using c_1 different colors, we call this coloring *a (weak)*[2] *c_1-edge coloring* of $G(N, K, g)$.

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 c_2 different colors, we call this coloring *a strong c_2-edge coloring* of $G(N, K, g)$.

If we think the colored (resp. uncolored) edges in $G(N, K, g)$ as the existing (resp. new) connections in $B(N, x, p, \alpha)$, a solution to the WEC problem is a plane assignment for routing in an RNB network since we can reroute existing connections in such a network, and a solution to the SEC problem is a plane assignment for routing in an SNB network since rerouting existing connections is not allowed in such a network. Clearly, for the same $G(N, K, g)$, $c_1 \leq c_2$.

[2]The definition of weak edge coloring is the same as the definition of edge coloring in graph theory. Thus we omit "weak" in the following of paper.

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

4 Routing for Rearrangeable Nonblocking Networks

4.1 Rearrangeable Nonblockingness

Lemma 4 *If* $p \geq 2^{\lfloor \frac{n-x+\alpha}{2} \rfloor}$, *then* $B(N, x, p, \alpha)$ *is rearrangeable nonblocking.*

The above claim is implied by the results of [12]. It is important to note that the minimum value of p in Lemma 4 equals to the value of g in Lemma 3, where p is the number of $B(N, x, 1, \alpha)$ planes required for $B(N, x, p, \alpha)$ to be rearrangeable nonblocking.

By Lemmas 3 and 4, if we assign the connections (including existing and new connections) sharing the same modulo-g input/output group to different planes, the connections are feasible for every assigned plane. Then, the routing can be completed by setting up conflict-free connection paths within each plane.

Lemma 5 *Every bipartite graph* G *has a* Δ*-edge coloring, where* Δ *is the degree of* G.

By Lemma 5 (see a proof in [2]), if we set $g = 2^{\lfloor \frac{n-x+\alpha}{2} \rfloor}$ in $G(N, K, g)$, the plane assignments for a set of connections in RNB $B(N, x, p, \alpha)$ can be solved by finding a g-edge coloring of $G(N, K, g)$ since the degree of $G(N, K, g)$ equals to g.

4.2 Algorithm for Balanced 2-Coloring of $G(N, K, g)$

In order to solve WEC problem efficiently, we present an algorithm for a problem, named *balanced 2-coloring problem*: given an I/O mapping graph $G(N, K, g)$, color its edges with 2 colors so that every vertex is adjacent to at most $g/2$ edges with one color and $g/2$ with the other.

We choose to present our parallel algorithms for a completely connected multiprocessor system since any algorithm for this parallel computing model can be easily transformed to algorithms on more realistic multiprocessor systems. A completely connected multiprocessor system of size N consists of a set of 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.

Initially, each PE_i reads $\pi(i)$ from input i, sets value of π^{-1} in $PE_{\pi(i)}$ as i, and then performs the following two steps.

Step 1. Divide the I/O mapping graph $G(N, K, g)$ into a set of components. This step can be done by each edge finding its left edge $\bar{i}$ and right edge $\pi^{-1}(\overline{\pi(i)})$.

Step 2. Color components with two colors, red and blue, so that neighboring edges in each component have different colors.

Each component has two specific representatives, simply referred to *Rep's*. (There is an exception: for the component with length of 1, there is only one Rep, which is itself.) For closed and open components, the Rep's are defined differently. For a closed component, we define two edges with the minimum labels as two Rep's; for an open component, if an edge e has no left edge or e's left edge has no right edge, e is defined as one Rep. Fig. 4(c) shows the Rep's of all possible types of components, where the Rep's of each component are marked as dark lines and edges are labeled by their corresponding inputs Step 2 can be done by coloring edges with the Rep's as references using the pointer jumping technique in [7]. At the beginning, each edge sets its pointer to point to the right edge of its left edge if it exists and to itself otherwise. By doing so, for a closed component or an open component with more than one edge, two disjoint directed cycles or paths are formed, each containing a Rep. For an open component, furthermore, the end pointer of every directed path is pointing to one of the Rep's. For example, Fig. 4(d) shows that the directed cycles and paths formed from the components of Fig. 4(c). Then, by performing $\lceil \lg \lceil K/2 \rceil \rceil$ times of parallel pointer jumping, each edge finds the Rep belonging to the same directed cycle or path. Finally, each edge can be colored by comparing the value of the Rep found by itself with that by its neighbor. That is, if the value of the Rep founded by an edge is no larger than its neighbor's, color the edge with red; and otherwise color it with blue. Fig. 4(b) shows a balanced 2-coloring of an I/O mapping graph of Fig. 4(c), where solid lines are colored as red and dashed lines are colored as blue.

Theorem 1 *A balanced* 2*-coloring of any* $G(N, K, g)$ *can be found in* $O(\lg K)$ *time using a completely connected multiprocessor system of* N *PEs.*

Proof. Given an I/O mapping graph $G(N, K, g)$, Step 1 can be done in $O(1)$ time using a completely connected multiprocessor system of N PEs. In Step 2, since the length of each directed cycle or path is at most $\lceil K/2 \rceil$, each edge can find a Rep by $\lceil \lg \lceil K/2 \rceil \rceil$ times of pointer jumping. Clearly, all edges in the same directed cycle or path are colored with the same color since they find the same Rep. The pointer initialization implies that each edge and its neighboring edge are in different directed cycle or path, and thus, they have different colors. By the definition of left/right edge, there are no more than $g/2$ pairs of neighboring edges incident at any vertex of $G(N, k, g)$. Thus, the coloring of all components compose a balanced 2-coloring of $G(N, k, g)$. Therefore, a balanced 2-coloring of any $G(N, K, g)$ can be found in $O(\lg K)$ time. □

4.3 Algorithm for g-Edge Coloring of $G(N, K, g)$

Based on the balanced 2-coloring algorithm, a WEC solution to any I/O mapping graph $G(N, K, g)$ with no more than g colors can be found as follows. Initially, we remove all colors for K_0 already colored edges. In initial step (i.e., step 0), we find a balanced 2-coloring of $G(N, K, g)$ using colors 0 and 1, and let G_0 and G_1 be the graphs induced by the edges with colors 0 and 1, respectively. In step 1, if the degree of G_0 and/or G_1 is no less than 2, we find a balanced 2-coloring for G_0 using colors 00 and 01, and/or

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

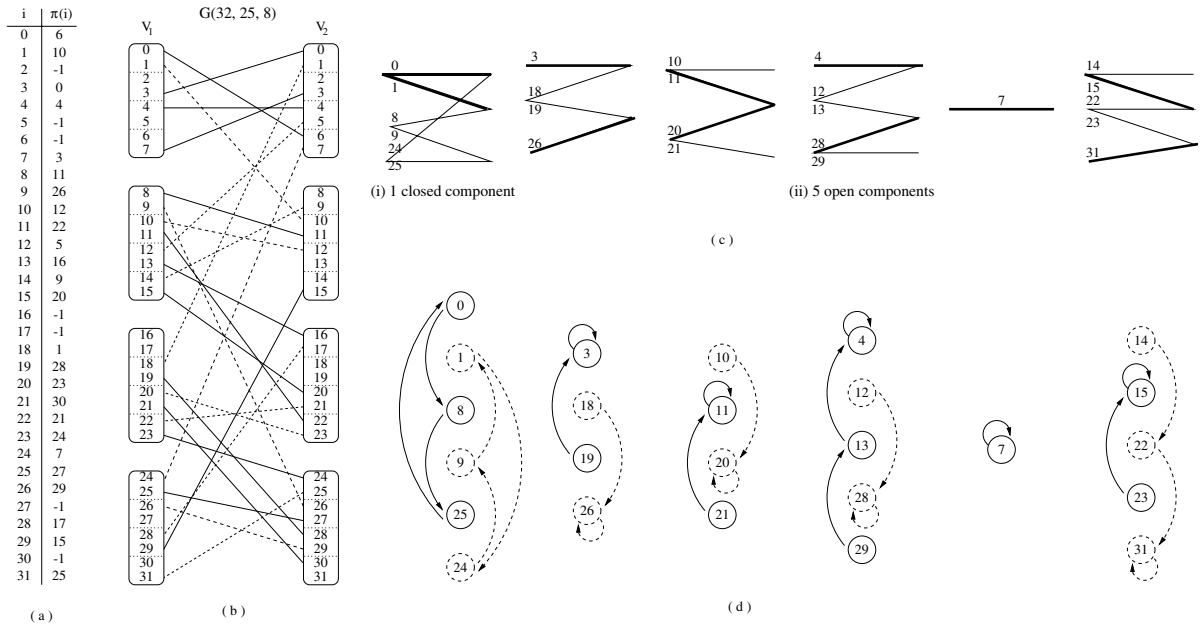

Figure 4. Finding a balanced 2-coloring (a) An I/O mapping (b) A balanced 2-coloring of an I/O mapping graph $G(32, 25, 8)$ (c) A set of components (d) Pointer initialization for pointer jumping.

find a balanced 2-coloring for G_1 using colors 10 and 11. This process is recursively continued in a binary tree fashion until a solution to WEC is reached. More formally, in each recursive step i, $1 \leq i \leq \lg g$, we find a balanced 2-coloring for each graph G_z using colors $z0$ and $z1$ (i.e., concatenate 0 or 1 with z) if the degree of G_z is no less than 2, where z is a binary representation of an integer in $\{0, 1, \cdots, 2^i - 1\}$ and the color of $E(G_z)$ in step $i - 1$.

Theorem 2 *For any I/O mapping graph $G(N, K, g)$, a g-edge coloring can be found in $O(\lg g \cdot \lg K)$ time using a completely connected multiprocessor system of N PEs.*

Proof. There are $K(\leq N)$ edges in $G(N, K, g)$. Since $g = 2^k$, we can prove the theorem by an induction on k. If $k = 1$, it is true since a balanced 2-coloring is a 2-edge coloring by Theorem 1. Assume that for any $k < m \leq n$, the theorem holds. Now, we prove that the theorem holds for $k = m$. First, we find a balanced 2-coloring of $G(N, K, g)$, which can be done in $O(\lg K)$ time by Theorem 1. Let G_0 and G_1 be the graphs induced by the edges of two different colors from this balanced 2-coloring. By the definition of balanced 2-coloring, we know that $\Delta(G_0) \leq g/2$ and $\Delta(G_1) \leq g/2$. By the hypothesis, we can find a $(g/2)$-edge coloring for each of G_0 and G_1 in $O((k-1) \cdot \lg K)$ time on a completely connected multiprocessor system of $|E(G_0)|$ and $|E(G_1)|$ PEs, respectively, which can be carried out simultaneously since $E(G_0) \cap E(G_1) = \emptyset$. The $(g/2)$-edge colorings of G_0 and G_1 compose a g-edge coloring of $G(N, K, g)$, which takes total $O(k \cdot \lg K)$ time for a completely connected multiprocessor system of N PEs. □

4.4 Parallel Routing in a Plane

We have shown how to assign a plane to each connection in an RNB $B(N, x, p, \alpha)$. In this section, we show how connections are routed within each plane.

Lemma 6 *Let C be a set of feasible connections for $B(N, x, 1, \alpha)$. If each connection in C is set up in the first and last x stages such that the output link in stage i and the input link in stage $\lg N - i$ on each connection are connected with the same subnetwork $B(N, x, 1/2^{i+1}, \alpha)$, $0 \leq i \leq x - 1$, then C can be routed by self-routing in the middle $\lg N - x$ stages.*

Proof. By the topology of $B(N, x, 1, \alpha)$, we know that each connection must pass through the same subnetwork $B(N, x, 1/2^i, \alpha)$, $0 \leq i \leq \lg N - 1$. Since the middle $\lg N - x$ stages of $B(N, x, 1, \alpha)$ consists of 2^x Baseline network $BL(\frac{N}{2^x})$, this lemma is true. □

Theorem 3 *Let C be a set of K feasible connections of $B(N, x, 1, \alpha)$. Then C can be correctly set up in $O(x \lg K + \lg N)$ time using a completely connected multiprocessor system of N PEs.*

Proof. By Lemma 6, what we only need to do is to set up C correctly in the first and last x stages for $x \geq 1$. By the topology of $B(N, x, 1, \alpha)$, we know that the output link in stage i and the input link in stage $\lg N - i$ on each connection are connected with the same subnetwork $B(N, x, 1/2^{i+1}, \alpha)$, $0 \leq i \leq x - 1$. Thus, we need to decide which subnetwork to be used for each connection

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

since there are 2^i $B(N, x, 1/2^i, \alpha)$'s. This can be reduced to a 2-edge coloring of a bipartite graph with degree of 2. For each subnetwork $B(N, x, 1/2^i, \alpha)$, $0 \leq i \leq x-1$, we construct an I/O mapping graph $G(N/2^i, K_i, 2)$, where K_i is the number of connections passing through it. We color the edges of $G(N/2^i, K_i, 2)$ with two different colors and assign the connections (edges) with the same color to pass through the same subnetwork $B(N, x, 1/2^{i+1}, \alpha)$. Specifically, in each step i, $0 \leq i \leq x-1$, we run g-edge coloring algorithm for 2^i $G(N/2^i, K_i, 2)$'s with $g = 2$. By Theorem 2, each step can be done in $O(\lg K)$ time. Thus, the time to set up K feasible connections in the first and last x stages is $O(x \lg K)$. By Lemma 6, we can set up the connections in the middle $\lg N - x$ stages by self-routing, which takes $\lg N - x$ time. Therefore, the total time to route K feasible connections of $B(N, x, 1, \alpha)$ is $O(x \lg K + \lg N)$ using a completely connected multiprocessor system of N PEs. □

4.5 Overall Routing Performance

Theorem 4 *For any RNB $B(N, x, p, \alpha)$, we can correctly route K connections (including existing and new connections) in $O((x + \lg p) \lg K + \lg N)$ time using a completely connected multiprocessor system of N PEs.*

Proof. By Theorem 2, we can find a p-edge coloring of the I/O mapping graph $G(N, K, p)$ in $O(\lg p \lg K)$ time. By Lemma 3, we assign the connections with the same color to the same plane. In each plane $B(N, x, 1, \alpha)$, by Theorem 3, we can set up the connections in $O(x \lg K + \lg N)$ time. Thus, the total time complexity is $O((x+\lg p) \lg K+\lg N)$. □

By Theorem 4, the routing time for setting up $O(K)$ connections in an RNB $B(N, x, p, \alpha)$ is improved to $O((x + \lg p) \lg K + \lg N)$ from $\Omega(K \log N)$. By Lemma 4, for an RNB $B(N, 0, p, \alpha)$ and an RNB $B(N, n-1, p, \alpha)$, the minimum number of planes of Baseline network and Benes network, equals to $2^{\lfloor \frac{n+\alpha}{2} \rfloor}$ and $2^{\lfloor \frac{1+\alpha}{2} \rfloor}$, respectively. Consequently, we can route connections in $O(\lg^2 N)$ time for both $B(N, 0, p, \alpha)$ and $B(N, n-1, p, \alpha)$. For the RNB $B(N, n-1, p, 0)$, which is the electronic Benes network, this performance is the same as the best known results reported in [11, 13].

5 Routing for Strictly Nonblocking Networks

5.1 Strict Nonblockingness

The following lemma can be easily derived from the results of [18].

Lemma 7 *If*

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

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

For an SNB network, we can set up 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 by no means to be simpler than that in an RNB network when we need to set up the new connections simultaneously. In this section, we present a parallel algorithm based on graph coloring to speed up routing time.

Based on the discussions in Section 3, we know that the routing problem for an SNB $B(N, x, p, \alpha)$ can be solved by finding a strong edge coloring of the I/O mapping graph $G(N, K, g)$ with $g = 2^{\lfloor \frac{n-x+\alpha}{2} \rfloor}$.

Lemma 8 *Any graph G has a strong $(2\Delta - 1)$-edge coloring, where Δ is the degree of G.*

Proof. Consider coloring edges in an arbitrary order. Since each edge in G is adjacent to at most $2\Delta - 2$ edges, any uncolored edge in G can always be assigned a color so that the total number of colors used is no more than $2\Delta - 1$. □

We consider a subclass of SNB networks, $B(N, 0, p^*, \alpha)$ with $p^* = 2^{\lfloor \frac{n+\alpha}{2} \rfloor + 1} - 1$. By Lemma 7, we know that $B(N, 0, p^*, \alpha)$ is an SNB network. Since each plane of $B(N, 0, p^*, \alpha)$ is a Baseline network, the routing of connections in any plane can be done by self-routing. Thus, the problem of setting up connections in $B(N, 0, p^*, \alpha)$ is reduced to finding a plane for each new connection so that all connections, including existing ones, are conflict-free. By Lemmas 3 and 8, this can be done by finding a strong $(2g-1)$-edge coloring for $G(N, K, g)$ of $B(N, 0, p^*, \alpha)$ with K_0 existing connections and $K - K_0$ new connections, where $g = 2^{\lfloor \frac{n+\alpha}{2} \rfloor}$. In the next subsection, we present an algorithm to find a strong $(2g-1)$-edge coloring of $G(N, K, g)$.

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

A *matching* is defined as a set of edges that does not contain any adjacent edges. Conceptually, a strong $(2g-1)$-edge coloring of $G(N, K, g)$ with $K_0 (< K)$ colored edges can be done in the following two steps.

Step 1: find a set of matchings in $G(N, K - K_0, g)$;

Step 2: color matchings one by one without changing the existing colors.

It is easy to see that the edges with the same color compose a matching for any g-edge coloring of $G(N, K - K_0, g)$. Thus, Step 1 can be done by finding a g-edge coloring of $G(N, K - K_0, g)$, which divides $K - K_0$ uncolored edges (corresponding to new connections) into at most g matchings. By Theorem 2, it takes $O(\lg g \cdot \lg(K - K_0))$ time using a completely connected multiprocessor system of N PEs. In $G(N, K, g)$, each edge is adjacent to at most $2g - 2$ edges, and hence, there are at most $2g - 2$ colored edges adjacent to each edge in a matching. Thus we can color every edge in the matching by one of the unused colors. This can be done by parallel searching for a free color among $2g - 1$ colors, which takes $O(\lg g)$ time. Since no

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

two edges are adjacent in a matching, by coloring g matchings one by one in $G(N, K, g)$, a strong $(2g-1)$-edge coloring of $G(N, K, g)$ is found. Therefore, we have the following claim.

Theorem 5 *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 $O(\lg g \lg(K - K_0) + g \lg g)$ time using a completely connected multiprocessor system of N PEs.*

5.3 Performance Analysis

Theorem 6 *For a strictly nonblocking network $B(N, 0, p, \alpha)$ with $p \geq 2^{\lfloor \frac{n+\alpha}{2} \rfloor + 1} - 1$, we can establish K connections from K idle inputs to K idle outputs in $O(\lg p \lg K + p \lg p)$ time using a completely connected multiprocessor system of N PEs.*

Proof. In $G(N, K, p)$, we assume the edges corresponding to the existing connections in the i-th plane of $B(N, 0, p, \alpha)$ have been colored with color i and the edges corresponding to the new connections have not been colored yet. By Theorem 5, we can find a strong $(2p-1)$-edge coloring of $G(N, K, p)$ in $O(\lg p \lg K + p \lg p)$ time using a completely connected multiprocessor system of N PEs. We assign each new connection with color i to the i-th plane of $B(N, 0, p, \alpha)$. By Lemma3, these new connections can be set up by self-routing in $O(\lg N)$ time. □

By Lemma 7, we can derive the minimum number of planes, $p_{\min}$, in $B(N, 0, p, \alpha)$.

Compared with $B(N, 0, p_{\min}, \alpha)$, the hardware redundancy of $B(N, 0, p^*, \alpha)$ is shown as follows.

$$p^* - p_{\min} = \begin{cases} 0, & \text{if } \alpha = 0 \text{ and } n \text{ is odd} \\ \sqrt{N}/2, & \text{if } \alpha = 0 \text{ and } n \text{ is even} \\ 0, & \text{if } \alpha = 1 \text{ and } n \text{ is even} \\ \sqrt{2N}/2, & \text{if } \alpha = 1 \text{ and } n \text{ is odd} \end{cases}$$

The hardware cost of $B(N, 0, p^*, \alpha)$, in terms of the number of SEs, is higher than that of $B(N, 0, p_{\min}, \alpha)$ in half of the cases, but both have the same hardware complexity of $\Theta(N^{1.5} \lg N)$. The routing time for setting up $O(N)$ connections, however, is improved to sublinear $O(\sqrt{N} \lg N)$ from $\Omega(N \lg N)$.

6 Concluding Remarks

One major contribution of this paper is the design and analysis of parallel routing algorithms for a class of nonblocking switching networks, $B(N, x, p, \alpha)$'s. 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. The pointer jumping and binary searching, which dominate the complexity of the proposed algorithms, can be reduced to sorting on realistic parallel computing structures. It is interesting to note that the sorting can be implemented in Banyan-type network in $O(\lg^2 N)$ time [10]. Thus the proposed algorithms can set up connections in $B(N, x, p, \alpha)$ with a slow-down factor $O(\lg^2 N)$ on a Banyan-type network, whose complexity is no larger than one plane of $B(N, x, p, \alpha)$.

References

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

[2] J.A. Bondy and U.S.R. Murty, *Graph Theory with Applications*, Elsevier North-Holland, 1976.

[3] J. Duato, S. Yalamanchili and L. Ni, *Interconnection Networks - A Engineering Approach*, Morgan Kaufmann, 2003.

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

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

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

[7] J. Jaja, *An Introduction to Parallel Algorithms*, Addison-Wesley, 1992.

[8] C.T. Lea, "Multi-log2N Networks and Their Applications in High-Speed Electronic and Photonic Switching Systems", *IEEE Trans. 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 Trans. 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 Trans. on Computers*, vol. 30, pp. 93-100, Feb. 1981.

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

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

[14] K. Padmanabhan and A. Netravali, "Dilated Network for Photonic Switching", *IEEE Trans. on Communications*, vol. COM-35, no. 12, pp. 1357-1365, Dec. 1987.

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

[16] F.M. Suliman, A.B. Mohammad, and K. Seman, "A Space Dilated Lightwave Network-a New Approach", *Proc. of IEEE 10th International Conference on Telecommunications (ICT 2003)*, vol. 2, pp. 1675-1679, 2003.

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

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

[19] J.E. Watson et al., "A Low-Voltage 8×8 Ti:$LiNbO_3$ Switch with a Dilated Benes Architecture," *IEEE J. of Lightwave Technology*, vol. 8, pp. 794-800, May 1990.