Nonblocking Multirate Networks

Authors: Riccardo Melen and Jonathan S. Turner

An extension of classical theory of connection networks is defined and studied. This extension models systems in which multiple connections of differing data rates share the links within a network. We determine conditions under which the Clos and Cantor networks are strictly nonblocking for multirate traffic. We also determine conditions under which the Benes Network and variants of the Cantor and Clos networks are rearrangeable. We find that strictly nonblocking operation can be obtained for multirate traffic with essentially the same complexity as in the classical context.

Follow this and additional works at: http://openscholarship.wustl.edu/cse_research

Recommended Citation
http://openscholarship.wustl.edu/cse_research/759
NONBLOCKING MULTIRATE NETWORKS

Riccardo Melen and Jonathan S. Turner

WUCS-88-02

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

Abstract

An extension of the classical theory of connection networks is defined and studied. This extension models systems in which multiple connections of differing data rates share the links within a network. We determine conditions under which the Clos and Cantor networks are strictly nonblocking for multirate traffic. We also determine conditions under which the Benes Network and variants of the Cantor and Clos networks are rearrangeable. We find that strictly nonblocking operation can be obtained for multirate traffic with essentially the same complexity as in the classical context.

NONBLOCKING MULTIRATE NETWORKS

Riccardo Melen
Jonathan S. Turner

1. Introduction

In this paper we introduce a generalization of the classical theory of nonblocking switching networks to model communications systems designed to carry connections with a multiplicity of data rates. The theory of nonblocking networks was motivated by the problem of designing telephone switching systems capable of connecting any pair of idle terminals, under arbitrary traffic conditions. From the start, it was recognized that crossbar switches with \( N \) terminals and \( N^2 \) crosspoints could achieve nonblocking behavior, only at a prohibitive cost in large systems. In 1953, Charles Clos [6] published a seminal paper giving constructions for a class of nonblocking networks with far fewer crosspoints, providing much of the initial impetus for the theory that has since been developed by Benes [2,3], Pippenger [16] and many others [1,5,8,11,12,13,14].

The original theory was developed to model electro-mechanical switching systems in which both the external links connecting switches and the internal links within them, were at any one time dedicated to a single telephone conversation. During 1960’s and 1970’s technological advances led to digital switching systems in which information was carried in a multiplexed format, with many conversations time-sharing a single link. While this was a major technological change, its impact on the theory of nonblocking networks was slight, because the new systems could be readily cast in the existing model. The primary impact was that the the traditional complexity measure of crosspoint count had a less direct relation to cost than in the older technology.

During the last ten years, there has been growing interest in communication systems that are capable of serving applications with widely varying characteristics. In particular, such systems are being to designed to support connections with arbitrary data rates, over a range from a few bits per second to hundreds of megabits per second [7,10,19]. These systems also carry information in multiplexed format, but in
contrast to earlier systems, each connection can consume an arbitrary fraction of the bandwidth of the link carrying it. Typically, the information is carried in the form of independent blocks, called packets which contain control information, identifying which of many connections sharing a given link, the packet belongs to. One way to operate such systems is to select for each connection, a path through the switching system to be used by all packets belonging to that connection. When selecting a path it is important to ensure that the available bandwidth on all selected links is sufficient to carry the connection. This leads to a natural generalization of the classical theory of nonblocking networks, which we explore in this paper. Note that such networks can also be operated with packets from a given connection taking different paths; reference [20] analyzes the worst-case loading in networks operated in this fashion. The drawback of this approach is that it makes it possible for packets in a given connection to pass one another, causing them to arrive at their destination out of sequence.

In Section 2, we define our model of nonblocking multirate networks in detail. Section 3 contains results on strictly nonblocking networks, in particular showing the conditions that must be placed on the networks of Clos and Cantor in order to obtain nonblocking operation in the presence of multirate traffic. We also describe two variants on the Clos and Cantor network that are wide-sense nonblocking in the general environment. Section 4 gives results on rearrangeably nonblocking networks, in particular deriving conditions for which the networks of Benes and Cantor are rearrangeable.

2. Preliminaries

We start with some definitions. We define a network as a directed graph $G = (V, E)$ with a set of distinguished input nodes $I$ and output nodes $O$, where each input node has one outgoing edge and no incoming edge and each output node has one incoming edge and no outgoing edge. We consider only networks that can be divided into a sequence of stages. We say that the input ports are in stage 0 and for $i > 0$, a node $v$ is in stage $i$ if for all links $(u, v)$, $u$ is in stage $i - 1$. A link $(u, v)$ is said to be in stage $i$ if $u$ is in stage $i$. In the networks we consider, all output ports are in the same stage, and no other nodes are in this stage. When we refer to a $k$ stage network, we generally neglect the stages containing the input and output ports. We refer to a network with $n$ input ports and $m$ output ports as an $(m, n)$-network. We let $X_{n,m}$ denote the network consisting of $n$ input nodes, $m$ output nodes and a single internal node. In this network model, nodes correspond to the hardware devices that perform the actual switching functions and the links to the interconnecting data paths. This differs from the graph model traditionally used in the theory of switching networks, which can be viewed as a dual to our model.
When describing particular networks we will find it convenient to use a composition operation. We denote the composition of two networks $Y_1$ and $Y_2$ by $Y_1 \circ Y_2$. The composition operation yields a new network consisting of one or more copies of $Y_1$ connected to one or more copies of $Y_2$, with a link joining each pair of subnetworks. More precisely, if $Y_1$ has $n_1$ outputs and $Y_2$ has $n_2$ inputs, then $Y_1 \circ Y_2$ is formed by taking $n_2$ copies of $Y_1$ numbered from 0 to $n_2 - 1$ followed by $n_1$ copies of $Y_2$, numbered from 0 to $n_1 - 1$. Then, for $0 \leq i \leq n_1 - 1$, $0 \leq j \leq n_2 - 1$, we join $Y_1(i)$ to $Y_2(j)$ using a link connecting output port $j$ of $Y_1(i)$ to input port $i$ of $Y_2(j)$. Next, we remove the former input and output nodes that are now internal, identifying the edges incident to them and finally, we renumber the input and output nodes of the network as follows; if $u$ was input port $i$ of $Y_1(j)$, it becomes input $jn_1 + i$ in the new network; similarly if $v$ was output port $i$ of $Y_2(j)$, it becomes output $jn_2 + i$.

We also allow composition of more than two networks; the composition $Y_1 \circ Y_2 \circ Y_3$ is obtained by letting $Z_1 = Y_1 \circ Y_2$ and $Z_2 = Y_2 \circ Y_3$, then identifying the copies of $Y_2$ in $Z_1$ and $Z_2$. This requires of course that the number of copies of $Y_2$ generated by the two initial compositions be the same. Note this is not the same as $(Y_1 \circ Y_2) \circ Y_3$.

A connection in a network is a triple $(x, y, \omega)$ where $x \in I$, $y \in O$ and $0 \leq \omega \leq 1$. We refer to $\omega$ as the weight of the connection and it represents the bandwidth required by the connection. A route is a path joining an input node to an output node, with intermediate nodes in $V - (I \cup O)$, together with a weight. A route $r$ realizes a connection $(x, y, \omega)$, if $x$ and $y$ are the input and output nodes joined by $r$ and the weight of $r$ equals $\omega$.

A set of connections is said to be compatible if for all nodes $x \in I \cup O$, the sum of the weights of all connections involving $x$ is $\leq 1$. A configuration for a network $G$ is a set of routes. The weight on an edge in a particular configuration is just the sum of the weights of all routes including that edge. A configuration is compatible if for all edges $(u, v) \in E$, the weight on $(u, v)$ is $\leq 1$. A set of connections is said to be realizable if there is a compatible configuration that realizes that set of connections. If we are attempting to add a connection $(x, y, \omega)$ to an existing configuration, we say that a node $u$ is accessible from $x$ if there is path from $x$ to $u$, all of whose edges have a weight of no more than $1 - \omega$.

A network is said to be rearrangeably nonblocking (or simply rearrangeable) if for every set $C$ of compatible connections, there exists a compatible configuration that realizes $C$. A network is strictly nonblocking if for compatible configuration $R$, realizing a set of connections $C$, and every connection $c$ compatible with $C$, there exists a route $r$ that realizes $c$ and is compatible with $R$. For strictly nonblocking networks, one can choose routes arbitrarily and always be guaranteed that any new connections can be satisfied without rearrangements. We say that a network is wide-sense nonblocking if there exists a routing algorithm, for which the network never blocks; that is, if we use the routing algorithm to select routes for each new connection request, it is always possible to realize a new connection by adding a route to the
current configuration.

Sometimes, improved performance can be obtained by placing constraints on the traffic imposed on a network. We will consider two such constraints. First, we restrict the weights of connections to be in the interval $[b, B]$. We also limit the sum of the weights of connections involving a node $x$ in $\mathcal{I} \cup \mathcal{O}$ to $\beta$. Note that $0 \leq b \leq R \leq \beta \leq 1$. We say a network is strictly nonblocking for particular values of $b$, $B$ and $\beta$ if for all sets of connections for which the connection weights are in $[b, B]$ and the total port weight is $\beta$, the network cannot block. The definitions of rearrangeably nonblocking and wide-sense nonblocking networks are extended similarly. The practical effect of a restriction on $\beta$ is to require that a network’s internal data paths operate at a higher speed that the external transmission facilities connecting switching systems, a common technique in the design of high speed systems. The reciprocal of $\beta$ is commonly referred to as the speed advantage for a system.

Two particular choices of parameters are of special interest. We refer to the traffic condition characterized by $B = \beta$, $b = 0$ as unrestricted packet switching (UPS), and the condition $B = b = \beta = 1$ as pure circuit switching (CS). Since the CS case is a special case of the multirate case, we can expect solutions to the general problem to be at least as costly as the CS case and that theorems for the general case should include known results for the CS case.

3. Strictly Nonblocking Networks

A three stage Clos [6] network with $N$ input and output ports is denoted by $C_{N,k,m}$, where $k$ and $m$ are parameters, and is defined as: $C_{N,k,m} = X_{k,m} \circ X_{N/k,N/k} \circ X_{m,k}$. A Clos network is depicted in Figure 1. The standard reasoning to determine the
nonblocking condition (see [6]) can be extended in a straightforward manner, yielding the following theorem.

**Theorem 3.1.** The Clos network $C_{N,k,m}$ is strictly nonblocking if

$$m > 2 \max_{b \leq \omega \leq B} \left\lfloor \frac{\beta k - \omega}{s(\omega)} \right\rfloor$$

where $s(\omega) = \max\{1 - \omega, b\}$.

*Proof.* Suppose we wish to add a connection $(x, y, \gamma)$ to an arbitrary configuration $C$. Let $u$ be the stage 1 node adjacent to $x$ and note that the sum of the weights on all edges out of $u$ is at most $\beta(k - 1) + (\beta - \gamma) = \beta k - \gamma$. Consequently, the number of edges out of $u$ that carry a weight of more than $(1 - \gamma)$ is $\leq \lfloor(\beta k - \gamma)/s(\gamma)\rfloor$, and hence the number of inaccessible middle stage nodes is

$$\leq \left\lfloor \frac{\beta k - \gamma}{s(\gamma)} \right\rfloor \leq \max_{b \leq \omega \leq B} \left\lfloor \frac{\beta k - \omega}{s(\omega)} \right\rfloor < \frac{m}{2}$$

That is, less than half the middle stage nodes are inaccessible from $x$. By a similar argument, less than half the middle stage nodes are inaccessible from $y$, implying that there is at least one middle stage node accessible to both. $\square$

Let us examine some special cases of interest. If we let $b = B = \beta = 1$, the effect is to operate the network in CS mode and the theorem states that we get nonblocking operation when $m \geq 2k - 1$ as is well-known. In the UPS case, the condition on $m$ becomes $m > 2(\beta/(1 - \beta))(k - 1)$. So $m = 2k - 1$ is sufficient here also if $\beta = 1/2$.

Using Theorem 3.1, we can construct a wide-sense nonblocking network for unrestricted traffic by placing two Clos networks in parallel and segregating connections in the two networks based on weight. In particular if we let $m = 4k - 1$, the network $X_{1,2} \circ C_{N,k,m} \circ X_{2,1}$ is wide-sense nonblocking if all connections with weight $\leq 1/2$ are routed through one of the Clos subnetworks and all the connections with weight $> 1/2$ are routed through the other.

A $k$-ary Benes network [2], built from $k \times k$ switching elements (where $\log_k N$ is an integer) can be defined recursively as follows: $B_{k,k} = X_{k,k}$ and $B_{N,k} = X_{k,k} \circ B_{N/k,k} \circ X_{k,k}$ (see Figure 2). A $k$-ary Cantor network of multiplicity $m$ is defined as $K_{N,k,m} = X_{1,m} \circ B_{N,k} \circ X_{m,1}$. Note that this definition is expressed differently from those given in [5,13], but we find it preferable as it shows clearly the close relationship between these two structures. Figure 3 depicts a binary Cantor network of multiplicity three with one of its Benes subnetworks highlighted. The next theorem captures the condition on $m$ required to make the Cantor network strictly nonblocking.
THEOREM 3.2. The Cantor network $K_{N,k,m}$ is strictly nonblocking if
\[ m \geq 2 \frac{\beta}{s(B)} \frac{k - 1}{k} \log_k N \]

Proof. Suppose we wish to add a connection $(x, y, \omega)$ to an arbitrary configuration. Note that there are $mN/k$ nodes in the middle stage of the network. We will show that more than half of these nodes are accessible from $x$ if $m$ satisfies the inequality in the statement of the theorem.

Define $W_i$ to be the set of all edges $(u, v)$ in stage $i$, for which $u$ is accessible from $x$, but $v$ is not. Define $\lambda_i$ to be the sum of the weights on all edges in $W_i$ and note...
that \( \lambda_i \geq |W_i|s(\omega) \). If we let \( h = \log_k N \), then the number of middle stage nodes (stage \( h + 1 \)) that are not accessible from \( x \) is given by

\[
\sum_{i=2}^{h} k^{h-i} |W_i| \leq \frac{1}{s(\omega)} \sum_{i=2}^{h} k^{h-i} \lambda_i
\]

It is easily verified that

\[
\sum_{i=2}^{h} k^{h-i} \lambda_i \leq (\beta - \omega) k^{h-2} + \sum_{i=2}^{h} k^{h-i} \left( k^{i-1} - k^{i-2} \right) \beta
\]

To see this, note that each term in the summation on the left gives the weight used for blocking at stage \( i \) weighted by \( k^{h-i} \). The terms in the summation on the right, give an upper bound on the total weight of the traffic that could possibly block connections from \( x \) at stage \( i \), similarly weighted by \( k^{h-i} \). The initial term on the right corresponds to the weight from input port \( x \) that is available for blocking. The right side of the above inequality

\[
= \left( \frac{\beta - \omega}{k} \right) \left( \frac{N}{k} \right) + \beta \left( \frac{k-1}{k} \right) \left( \frac{N}{k} \right) \log_k (N/k) < \beta \left( \frac{k-1}{k} \right) \left( \frac{N}{k} \right) \log_k N
\]

Combining this with the first inequality above, we have that the number of inaccessible middle stage nodes is

\[
< \left( \frac{\beta}{s(\omega)} \right) \left( \frac{k-1}{k} \right) \left( \frac{N}{k} \right) \log_k N \leq \left( \frac{\beta}{s(B)} \right) \left( \frac{k-1}{k} \right) \left( \frac{N}{k} \right) \log_k N \leq (mN/2k)
\]

That is, fewer than half the middle stage nodes are inaccessible from \( x \). By a similar argument, fewer than half the middle stage nodes are inaccessible from \( y \), meaning that there exists an available route from \( x \) to \( y \).

**Corollary 3.1.** The Benes network \( B_{N,k} \) is strictly nonblocking if

\[
\beta \leq \left[ \frac{2}{s(B)} \frac{k-1}{k} \log_k N \right]^{-1}
\]

**Proof.** Substitute 1 for \( m \) in the statement of the theorem and solve for \( \beta \).

When we apply the theorem to the CS case for \( k = 2 \), we find that the condition on \( m \) reduces to \( m \geq \log_2 N \) as is well known. For the UPS case with \( k = 2 \), we have \( m \geq 2(\beta/(1 - \beta)) \log_2 N \); that is, we again need a speed advantage of two to match the value of \( m \) needed in the CS case.

We can construct wide-sense nonblocking networks for \( \beta = 1 \) by increasing \( m \). We divide the connections into two subsets, with all connections of weight \( \leq 1/2 \) segregated from those with weight \( > 1/2 \). Applying Theorem 3.2 we find that \( m \geq 4((k-1)/k) \log_k N \) is sufficient to carry each portion of the traffic, giving a total of \( 8((k-1)/k) \log_k N \) subnetworks.
4. Rearrangeably Nonblocking Networks

As mentioned earlier, a $k$-ary Benes network [2], can be defined recursively as follows: $B_{k,k} = X_{k,k}$ and $B_{N,k} = X_{k,k} \circ B_{N/k,k} \circ X_{k,k}$. The Benes network is rearrangeable in the CS case [2] and efficient algorithms exist to reconfigure it [12,14]. In this section, we show that under certain conditions, the Benes network can be rearrangeable for multirate traffic as well. We start by reviewing a proof of rearrangeability for the CS case, as we will be extending the technique for this case to the general environment.

Consider a set of connections $C = \{c_1, \ldots, c_r\}$ for $B_{N,k}$, where $c_i = \{x_i, y_i, 1\}$ and there is at most one connection for each input and output port. The recursive structure of the network allows us to decompose the routing problem into a set of subproblems, corresponding to each of the stages in the recursion. The top level problem consists of selecting for each connection, one of the $k$ subnetworks $B_{N/k,k}$ to route through. Given a solution to the top level problem, we can solve the routing problems for the $k$ subnetworks independently. We can solve the top level problem most readily by reformulating it as a graph coloring problem. To do this, we define the connection graph $G_C = (V_C, E_C)$ for $C$ as follows.

$$V_C = \{u_j, v_j | 0 \leq j < N/k\}$$

$$E_C = \{\{u_{[x_i/k]j}, v_{[y_i/k]j}\} | 1 \leq i \leq r\}$$

To solve the top level routing problem, we color the edges of $G_C$ with colors $\{0, \ldots, k-1\}$ so that no two edges with a common endpoint share the same color. The colors assigned to the edges correspond to the subnetwork through which the connection must be routed. Because $G_C$ is a bipartite multigraph with maximum vertex degree $k$, it is always possible to find an appropriate coloring [4,9]. In brief, given a partial coloring of $G_C$, we can color an uncolored edge $\{u, v\}$ as follows. If there is a color $i \in \{0, \ldots, k-1\}$ that is not already in use at both $u$ and $v$, we use it. Otherwise, we let $i$ be any unused color at $u$ and $j$ be any unused color at $v$. We then find a maximal alternating path from $v$; that is a longest path with edges colored $i$ or $j$ and $v$ as one of its endpoints. Because the graph is bipartite, the alternating path must end at some vertex other than $u$ or $v$. Then, we interchange the colors $i$ and $j$ for all edges on the path and use $i$ to color the edge $\{u, v\}$.

To prove results for rearrangeability in the presence of multirate traffic, we must generalize the graph coloring methods used in the CS case. We define a connection graph $G_C$ for a set of connections $C$ as previously, with the addition that each edge is assigned a weight equal to that of the corresponding connection. We say that a connection graph is $(\beta, k)$-permissible if the edges incident to each vertex can be partitioned into $k$ groups whose weights sum to no more than $\beta$. A legal $(\beta, m)$-coloring of a connection graph is an assignment of colors in $\{0, \ldots, m - 1\}$ to each edge so that at each vertex $u$, the sum of the weights of the edges of any given color is no more than $\beta$. 

Now, suppose we let $Y = Y_1 \circ Y_2 \circ Y_3$, where $Y_1$ is a $(k,m)$-network, $Y_2$ is an $(N/k, N/k)$-network and $Y_3$ is an $(m,k)$-network and also let $0 \leq \beta_1 \leq \beta_2 \leq 1$. Then if $Y_1$, $Y_2$, $Y_3$ are rearrangeable for connection sets with $\beta \leq \beta_2$, $Y$ is rearrangeable for connection sets with $\beta \leq \beta_1$, if every $(\beta_1, k)$-permissible connection graph for $Y$ has a legal $(\beta_2, m)$ coloring.

Our first use of the coloring method is in the analysis of $B_{N,k}$. We apply it in a recursive fashion. At each stage of the recursion, the value of $\beta$ may be slightly larger than at the preceding stage. The key to limiting the growth of $\beta$ is the algorithm used for coloring the edges of the connection graph at each stage. We describe that algorithm next.

Let $G_C = (V_C, E_C)$ be an arbitrary connection graph. For each vertex $u$, let $C_u$ be the set of edges involving $u$. Next, number the edges in $C_u$ from zero, in non-increasing order of their weight and let $C_u^i \subseteq C_u$ comprise the edges with indices in the range $\{ik, \ldots, (i+1)k - 1\}$ for $i \geq 0$. Our coloring algorithm assigns unique colors to edges in each subset $C_u^i$. In particular, given a partial coloring of $G_C$, we color an uncolored edge $\{u, v\}$ belonging to $C_u^i$ and $C_v^j$ as follows. If there is a color $a \in \{0, \ldots, k-1\}$ that is not already in use within $C_u^i$ and $C_v^j$, we use it. Otherwise, we let $a_1$ be any unused color within $C_u^i$ and $a_2$ be any unused color within $C_v^j$. We then find a maximal constrained alternating path from $v$; that is a longest path with edges colored $a_1$ or $a_2$ with $v$ as one of its endpoints and such that for every interior vertex $w$ on the path, the path edges incident to $w$ belong to a common set $C_w^b$. Because the graph is bipartite, the last edge cannot be a member of either $C_u^i$ or $C_v^j$. Given the path, we interchange the colors $a_1$ and $a_2$ for all edges on the path and use $a_1$ to color the edge $\{u, v\}$. We refer to this as the CAP (constrained alternating path) algorithm. We can route a set of connections through $B_{N,k}$ by applying CAP recursively. Our first theorem gives conditions under which this routing is guaranteed not to exceed the capacity of any link in the network.

**Theorem 4.1.** The CAP algorithm successfully routes all sets of connections for $B_{N,k}$ for which

$$\beta \leq \left[1 + \frac{k-1}{k}(B/\beta) \log_k(N/k)\right]^{-1}.$$

**Proof.** Let $G_C$ be any $(\beta_1, k)$-permissible connection graph with maximum edge weight $B$ and $\beta_1 \leq 1 - B(k-1)/k$. We start by showing that the CAP algorithm produces a legal $(\beta_2, k)$-coloring for some $\beta_2 \leq \beta_1 + B(k-1)/k$.

Let $u$ be any vertex in $G_C$. Since each color is used at most once for each subset $C_u^i$ of the edges at $u$, the largest weight that can be associated with any one color at $u$ is bounded by the sum of the weights of the heaviest edges in $C_u^i$ for all $i$. Because the edges were assigned to the $C_u^i$ in non-increasing order of weight, the total weight of like-colored edges at $u$ is at most $B + (k\beta_1 - B)/k = \beta_1 - B(k-1)/k$. 

Given this, if we route a set of connections through $B_{N,k}$ by recursive application of the CAP algorithm, we will succeed if

$$\beta + \left(\frac{k-1}{k}\right) B \log_k(N/k) \leq 1$$

or equivalently, $\beta \leq \left[1 + ((k-1)/k)(B/\beta) \log_k(N/k)\right]^{-1}$. №

As an example, if $N = 2^{16}$, $k = 4$ and $B = \beta$, it suffices to have $\beta \leq 0.16$. We can improve on this result by modifying the CAP algorithm. Because the basic algorithm treats each stage in the recursion completely independently, it can in the worst-case concentrate traffic unnecessarily. The algorithm we consider next attempts to balance the traffic between subnetworks when constructing a coloring. We describe the algorithm only for the case of $k = 2$, although extension to higher values is possible.

Let $G_C$ be a connection graph for $B_{N,2}$. $G_C$ comprises vertices $u_0, \ldots, u_{(N/2)-1}$ corresponding to nodes in stage one of $B_{N,2}$ and vertices $v_0, \ldots, v_{(N/2)-1}$ corresponding to nodes in stage $2(\log_2 N - 1)$. We have an edge from $u_i$ to $v_j$ corresponding to each connection to be routed between the corresponding nodes of $B_{N,2}$. We note that for $0 \leq i < N/4$, the nodes corresponding to $u_{2i}$ and $u_{2i+1}$ have the same successors in stage two of $B_{N,2}$. Similarly, the nodes in $B_{N,2}$ corresponding to $v_{2i}$ and $v_{2i+1}$ have common predecessors. We say such vertex pairs are related.

Let $a$ and $b$ be any pair of related vertices in $G_C$. The idea behind the modified coloring algorithm is to balance the coloring at $a$ and $b$ so that the total weight associated with each color is more balanced, thus limiting the concentration of traffic in one subnetwork. The technique used to balance the coloring is to constrain it so that when appropriate, the edges of largest weight at $a$ and $b$ are assigned different colors, and hence the corresponding connections are routed through distinct subnetworks. For any vertex $v$ in $G_C$, let $\omega_0(v) \geq \omega_1(v) \geq \cdots$ be the weights of the edges defined at $v$, let $W_0(v) = \sum_{i \geq 0} \omega_{2i}$, $W_1(v) = \sum_{i \geq 0} \omega_{2i+1}$ and $W(v) = W_0(v) + W_1(v)$. Also, let $x(v) = W_0(v) - W_1(v)$.

The modified CAP algorithm proceeds as follows. For each pair of related vertices $a$ and $b$ in $G_C$, if $x(a) + x(b) > B$, add a dummy node $z$ to $G_C$ with edges of weight two connecting it to $a$ and $b$. We then color this modified graph as in the original CAP algorithm and on completion we simply ignore the added nodes and edges. The effect of adding the dummy node is to constrain the coloring at $a$ and $b$ so that the edges of maximum weight are assigned distinct colors. We apply this procedure recursively except that in the last step of the recursion we use the original CAP algorithm.

**Theorem 4.2.** The modified CAP algorithm successfully routes all sets of connections for $B_{N,2}$ for which

$$\beta \leq \left[1 + \frac{1}{4}(B/\beta) \log_2 N\right]^{-1}$$
Proof. Let $a$ and $b$ be related vertices with $\omega_0(a) \geq \omega_0(b)$. Let $z_1 = \max\{W(a), W(b)\}$ and let $z_2$ be the total weight on edges colored 0 at $a$ and $b$. If $x(a) + x(b) \leq B$, no dummy vertex is added and we have that

$$z_2 \leq W_0(a) + W_0(b) \leq (z_1 + x(a))/2 + (z_1 + x(b))/2 \leq z_1 + B/2$$

Similarly, if $x(a) + x(b) \geq B$, a dummy vertex is added and we have that

$$z_2 \leq \omega_0(a) + W_1(a) + W_1(b) \leq \omega_0(a) + (z_1 - x(a))/2 + (z_1 - x(b))/2 \leq z_1 + B/2$$

Thus, the total weight on a node in stage $i$ is at most $2\beta + (i - 1)B/2$. In particular, this holds for $i = \log_2 N - 1$. Also note that for a link $(u, v)$ in stage $j \leq \log_2 N - 2$, the maximum weight is at most $B$ plus half the weight on $u$. For a link $(u, v)$ in stage $\log_2 N - 1$, the weight is at most $B/2$ plus the maximum weight at $u$, since in this last step the original CAP algorithm was used. Consequently, no link carries a weight greater than $\beta + (B/4)\log_2 N$. \(\square\)

Theorem 4.2 implies for example that if $\beta = B = 0.2$, a binary Benes network with $2^{16}$ ports is rearrangeable. Theorem 4.1, on the other hand gives rearrangeability in this case only if $\beta$ is limited to about 0.118. It turns out that we can obtain a still stronger result by exploiting some additional properties of the original CAP algorithm.

**Theorem 4.3.** The CAP algorithm successfully routes all sets of connections for $B_{N,k}$ for which

$$\beta \leq \left[\max\{2, \lambda - \ln|\beta/B|\}\right]^{-1}$$

where $\lambda = 2 + \ln\log_k(N/k)$.

So, for example if $k = 4$, $N = 2^{16}$ and $\beta/B = 2$, we can have $\beta = 0.3$. The proof of Theorem 4.3 requires the following lemmas.

**Lemma 4.1.** Let $r$ be any positive integer. If a set of connections for $B_{N,k}$ is routed by repeated applications of the CAP algorithm, no link will carry more than $r$ connections of weight $> \beta/(r + 1)$.

Proof. By induction; the condition is true by definition for the external links. If the assertion holds at a given level of recursion, the connection graph for the next stage will have at most $rk$ edges of weight greater than $\beta/(r + 1)$ at any given node $u$. These edges are all contained in $C_u^0 \cup \cdots \cup C_u^{r-1}$, implying that the CAP algorithm will use a single color for at most $r$ of them. \(\square\)

If $\ell$ is a link in $B_{N,k}$, we define $S^j_{\ell}$ to be the set of links $\ell'$ in stage $j$ for which there is a path from $\ell'$ to $\ell$. If a given set of connections uses a link $\ell$, we refer to one connection of maximum weight as the primary connection on $\ell$ and all others as secondary connections. We note that if the CAP algorithm is used to route a set of connections through $B_{N,k}$, then if there are $r + 1$ connections of weight $\geq \omega$ on a link $\ell = (u, v)$, there are at least $1 + kr$ connections of weight $\geq \omega$ on the links entering $u$. 
LEMMA 4.2. Let \(0 \leq i \leq \log_k(N/k)\), let \(\ell\) be a stage \(i\) link in \(B_{N,k}\) carrying connections routed by the CAP algorithm and let the connections weights be \(\omega_0 \geq \omega_1 \geq \cdots \geq \omega_h\). For \(0 \leq t \leq h\) and \(0 \leq s \leq \min\{i, t\}\), there are at least \((t-s+1)k^s + sk^{s-1}\) connections of weight \(\geq \omega_t\) on the links in \(S_t^{i-s}\).

Proof. The proof is by induction on \(s\). When \(s = 0\), the lemma asserts that there are \(t+1\) connections of weight \(\geq \omega_t\) which is trivially true. Assume then that the lemma holds for \(s-1\); that is, there exist \((t-s+2)k^{s-1} + (s-1)k^{s-2}\) connections of weight \(\geq \omega_t\) on the links in \(S_t^{i-s+1}\). Because \(|S_t^{i-s+1}| = k^{s-1}\), by the pigeon-hole principle, at least \((t-s+1)k^{s-1} + (s-1)k^{s-2}\) of these are secondary connections. This implies that there are at least

\[k^{s-1} + k \left[ (t-s+1)k^{s-1} + (s-1)k^{s-2} \right] = (t-s+1)k^s + sk^{s-1}\]

connections of weight \(\geq \omega_t\) in \(S_t^{i-s}\). □

Proof of Theorem 4.3. Consider an arbitrary set of connections for \(B_{N,k}\) satisfying the bound on \(\beta\) given in the theorem, and assume that the CAP algorithm is used to route the connections. Let \(\ell\) be any link in stage \(i\), where \(i \leq \log_k(N/k)\), and let the weights of the connections on \(\ell\) be \(\omega_0 \geq \cdots \geq \omega_h\). Let \(r\) be the positive integer defined by \(\beta/(r+1) < B \leq \beta/r\) (equivalently, \(r = \lfloor \beta/B \rfloor\)). By Lemma 4.2, \(S_0^0\) carries connections with a total weight of at least

\[\omega_0 + k\omega_1 + k^2\omega_2 + \cdots + k^{i-1}\omega_{i-1} + k^i(\omega_i + \cdots + \omega_h)\]

Since the total weight on \(S^0_0\) is at most \(\beta k^i\), we have

\[\beta k^i \geq \sum_{j=0}^{i-1} k^j \omega_j + k^i \sum_{j=i}^{h} \omega_j\]

From this and Lemma 4.1, we have that

\[\sum_{j=0}^{r-1} \omega_j + \sum_{j=r}^{i-1} \omega_j + \sum_{j=i}^{h} \omega_j \leq Br + \beta \sum_{j=r+1}^{i-1} \frac{1}{j} + \beta \leq 2\beta + \beta \frac{\log_k(N/k)}{j} \frac{1}{j}\]

If \(\lfloor \beta/B \rfloor \geq \log_k(N/k)\), the summation vanishes and we have that the weight on \(\ell\) is \(\leq 2\beta\). Otherwise, the weight is bounded by

\[\leq \beta (2 + \ln \log_k(N/k)/r) = \beta (\lambda - \ln \lfloor \beta/B \rfloor)\]

So, if \(\beta\) satisfies the bound in the statement of the theorem, the weight on \(\ell\) is no more than one. By a similar argument, the weight on any link in stage \(j\) for \(j \geq \log_k N\) is at most one. □

We now turn our attention to the Cantor network and give conditions for rearrangeability in that case.
**Theorem 4.4.** Let \( \epsilon > 0 \) and \( |\beta/B| \leq \log_k(N/k) \). \( K_{N,k,m} \) is rearrangeable if
\[
m \geq [(1 + \epsilon)(\lambda - \ln|\beta/B|)] + 2(2 + \log_2 \lambda + \log_2(B/c))
\]
where \( \lambda = 2 + \ln \log_k(N/k) \) and \( c = 1 - \beta \lambda/(1 + \epsilon)(\lambda - \ln|\beta/B|) \).

The proof of Theorem 4.4 requires several lemmas.

**Lemma 4.3.** Let \( \alpha, r \geq 1 \) with \( r \) and \( \alpha r \) integers. \( B_{N,k} \) is rearrangeable for sets of connections with weights \( \omega \) that satisfy \( \beta/(\alpha r + 1) < \omega \leq \beta/r \) and \( \alpha \leq e^{(1/\beta)-1} \).

**Proof.** By Lemma 4.2, if \( B_{N,k} \) is routed using the CAP algorithm, no link contains more than \( \alpha r \) connections. The sum of the weights of the connections on any given link is
\[
\leq r \left( \frac{\beta}{r} \right) + \frac{\beta}{r+1} + \cdots + \frac{\beta}{\alpha r} = \beta + \beta \sum_{i=r+1}^{\alpha r} \leq \beta(1 - \ln \alpha) \leq 1 \] \( \square \)

**Lemma 4.4.** Let \( \alpha, r \geq 1 \) with \( r \) and \( \alpha r \) integers. \( K_{N,k,m} \) is rearrangeable for sets of connections with weights \( \omega \) that satisfy \( \beta/(\alpha r + 1) < \omega \leq \beta/r \) and \( \alpha \leq \exp \left[ \frac{\alpha r m}{\beta(r + m - 1)} - 1 \right] \).

**Proof.** The connections can be distributed among the \( m \) Benes subnetworks using the CAP algorithm; the resulting maximum port weight on the subnetworks is
\[
\gamma' \leq \frac{\beta}{m} + \frac{(m-1)\beta}{mr} = \frac{\beta(r + m - 1)}{mr}
\]
By Lemma 4.3, each subnetwork can be successfully routed if
\[
\alpha \leq \exp \left[ \frac{\alpha r m}{\beta(r + m - 1)} - 1 \right] \leq e^{(1/\gamma')-1} \] \( \square \)

**Lemma 4.5.** \( K_{N,k,m} \) is rearrangeable if \( m \geq 2(2 + \log_2(B/b)) \).

**Proof.** Define \( h, i \) by letting
\[
\frac{\beta}{2^{h+1}} < B \leq \frac{\beta}{2^h} \quad \text{and} \quad \frac{\beta}{2^{i+1}} < b \leq \frac{\beta}{2^i}
\]
By Lemma 4.4, two of the Benes subnetworks are sufficient to route connections with weights in the interval \( (2^{-(i+1)}\beta, 2^{-i}\beta) \) for any \( j \geq 0 \). For \( h \leq j \leq i \), we devote two subnetworks for the connections with weights in \( (2^{-(i+1)}\beta, 2^{-j}\beta) \). The total number of subnetworks required is at most \( 2(i - h + 1) \leq 2(2 + \log_2(B/b)) \leq m \) \( \square \)
LEMMA 4.6. $K_{N,k,m}$ is rearrangeable if

$$\frac{(\beta - B)\lambda}{1 - B\lambda} \leq m \leq \frac{\beta}{B}$$

where $\lambda = 2 + \ln \log_k(N/k)$.

Proof. We distribute the traffic among the $m$ Benes subnetworks using the CAP algorithm. The resulting maximum port weight is $\beta'$ where

$$\beta' \leq \frac{\beta}{m} + \frac{m-1}{m}B = B + \frac{\beta - B}{m}$$

By Theorem 4.3, the maximum weight on any link is at most

$$\beta'((\lambda - \ln[\beta'/B]) \leq B\lambda + \frac{(\beta - B)\lambda}{m} \leq 1$$

Proof of Theorem 4.4. Let $B' = c/\lambda$. By Lemma 4.5, all the traffic with weight $> B'$ can be handled using

$$2(2 + \log_2 \lambda + \log_2 B/c)$$

of the Benes subnetworks. By Lemma 4.6, the remaining traffic can be carried using

$$\frac{(\beta - B')\lambda}{1 - B'\lambda} = \frac{\beta\lambda - c}{1 - c} \leq [(1 + \epsilon)(\lambda - \ln[\beta'/B])]$$

subnetworks. □

Theorem 4.4 holds when $[\beta/B] \leq \log_k(N/k)$. When this condition does not hold, $K_{N,k,m}$ is rearrangeable with $m$ between one and three, depending on the value of $\beta$. In particular, if $\beta \leq 1/2$, $m = 1$ is sufficient using Theorem 4.1. If $(1/2) < \beta \leq 1 - 1/(2 \log_k(N/k))$, $m = 2$ is sufficient since in this case the traffic can be split among the two subnetworks so that each experience a maximum port weight of at most $1/2$.

The graph coloring methods used to route connections for $B_{N,k}$ can also be applied to networks that “expand” at each level of recursion. Let $C^*_{k,h,m} = X_{k,k}$ and for $N = k^i$, $i > 1$, let $C^*_{N,k,m} = X_{k,m} \circ C^*_{N/k,N/k} \circ X_{m,k}$. The following theorem gives conditions under which $C^*_{N,k,m}$ is rearrangeable.

THEOREM 4.5. $C^*_{N,k,m}$ is rearrangeable if

$$\beta \leq \left[1/\gamma^c + \frac{m - 1}{m} B \frac{1 - 1/\gamma^c}{\beta 1 - 1/\gamma}\right]^{-1}$$

where $\gamma = m/k$ and $c = \log_k(N/k)$.

Proof. We use the CAP algorithm to route the connections. If we let $\beta_i$ be the largest resulting weight on a link in stage $i$ for $1 \leq i \leq \log_k(N/k)$, we have

$$\beta_i \leq B + \frac{k\beta_{i-1} - B}{m} = (\beta_{i-1}/\gamma) + \frac{m - 1}{m}B \leq (\beta_0/\gamma^i) + \frac{m - 1}{m}B \frac{1 - (1/\gamma)^i}{1 - 1/\gamma} \leq 1$$

So, for example, $C^*_{N,k,2k-1}$ is rearrangeable if $B \leq 1/2$. 

\newpage
5. Closing Remarks

In recent years, there has been a growing interest in switching systems capable of carrying general multirate traffic, in order to be able to support a wide range of applications including voice, data and video. A variety of research teams have constructed high speed switching systems of moderate size [7,10,19,21], but little consideration has yet been given to the problem of constructing very large switching systems using such modules as building blocks. The theory we have developed here is a first step to understanding the blocking behavior of such systems.

In this paper, we have introduced what we feel is an important research topic and have given some fundamental results. There are several directions in which our work may be extended. While we have good constructions for strictly nonblocking networks, we expect that our results for rearrangeably nonblocking networks can be improved. In particular, we suspect that the Benes network can be operated in a rearrangeable fashion with just a constant speed advantage. Another interesting topic is nonblocking networks for multipoint connections. While this has been considered for space-division networks [1,8,11,17], it has not been studied for networks supporting multirate traffic. Another area to consider is determination of blocking probability for multirate networks. We expect this to be highly dependent on the particular choice of routing algorithm.

References


Approximation Algorithms for the Shortest Common Superstring Problem

Jonathan S. Turner *

jst@wucs.UUCP

July 18, 1986
WUCS-86-16

Abstract

The object of the shortest common superstring problem (SCS) is to find the shortest possible string that contains every string in a given set as substrings. As the problem is NP-complete, approximation algorithms are of interest. The value of an approximate solution to SCS is normally taken to be its length, and we seek algorithms that make the length as small as possible. A different measure is given by the sum of the overlaps between consecutive strings in a candidate solution. When considering this measure, the object is to find solutions that make it as large as possible. These two measures offer different ways of viewing the problem. While the two viewpoints are equivalent with respect to optimal solutions, they differ with respect to approximate solutions. We describe several approximation algorithms that produce solutions that are always within a factor of two of optimum with respect to the overlap measure. We also describe an efficient implementation of one of these, using McCreight's compact suffix tree construction algorithm. The worst-case running time is $O(m \log n)$ for small alphabets, where $m$ is the sum of the lengths of all the strings in the set and $n$ is the number of strings. For large alphabets, the algorithm can be implemented in $O(m \log m)$ time by using Sleator and Tarjan's lexicographic splay tree data structure.

*The research described here was supported by the National Science Foundation (grant #DCR-8409435) and the National Institutes of Health (grant #RR-01380)