On New Classes of Communication Networks

Julio Cezar. D. de Melo
Depto. de Eng. Eletrônica
Escola de Engenharia da UFMG
e-mail: demelo@uirapuru.cpdee.ufmg.br
Roy M. Jenevein
Dept. of Computer Science
University of Texas at Austin
e-mail: jenevein@cs.utexas.edu

Abstract

In this work a new generalized model for communication networks is proposed, based on banyan networks. These networks have been studied extensively in multiprocessing applications, and have recently been considered for application in ATM networks [11]. Through the introduction of labelling and numbering schemes for the nodes in the network, a construction algorithm is presented based on the notion of a connection formula, which makes it possible to describe an enourmous number of new networks. Some of the possible atractive topologies are discussed, along with the distance and traffic properties. Generic routing algorithms are also presented.

1 Introduction

Communication networks have been proposed since the early days of telephone systems, as a way of minimizing connection costs and improving the quality of service. The particular arrangement used is known as its topology, and it defines many of its features, like the delay in sending messages between nodes, the way messages are sent or communication links established, the type of control and routing algorithms that will be needed, and others. In this work, we will be dealing with the class of networks called Multistage Interconnection Networks, or MINs, of which many variations exist [3]. These networks are characterized by the fact that nodes are laid out in stages, and connections exist only between adjacent stages. A network can be represented by a graph, in which nodes correspond to processing elements or to data-routing switches, and edges corresponds to communication links connecting such components.

The emphasis in this work will be on the study of the distance properties of banyan networks, justified by the fact that the delay observed in the transmission of messages across an interconnection network is closely related to their distance properties. This is particularly useful when studying single-sided networks, because base-to-base distance becomes quite important, and improvements in the distance properties are essential for minimizing communication overhead. Also, the fault-tolerance properties of the network may show some improvements, due to the existence of alternative routes. Other properties of the network may improve as well, while keeping the cost of the original network, since the improvements are obtained by merely rearranging connections, not through the addition of nodes or links.

The SK-banyan [1] is a unified model of banyan networks [4, 6, 7, 8], in which the connections between adjacent levels have been *SK*ewed in relation to SW-banyan networks. This rearrangement of connections is illustrated in figure 1.

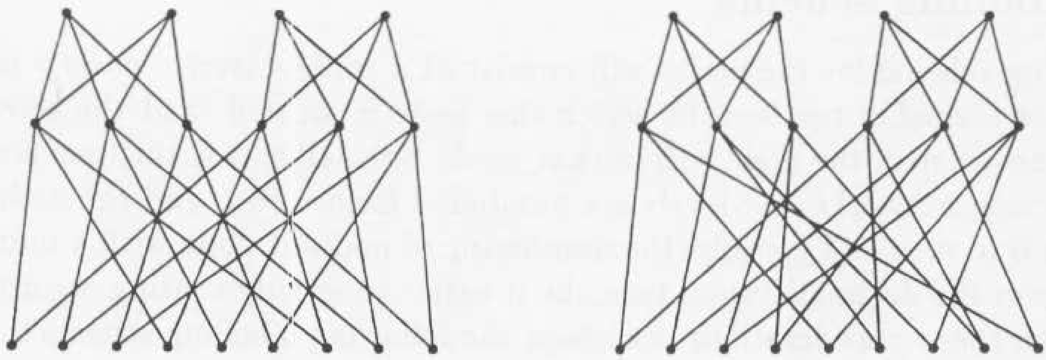


Figure 1: SW-banyan and SK-banyan.

In section 2, the definition of an SK-banyan is presented and a construction algorithm is given, allowing for the definition of a large number of topologies, of which some will be described there. In section 3, generic routing algorithms are presented, which have the advantage of being topology-independent. In section 4, distance and traffic properties are presented for some subclasses.

2 Definition of SK-banyans

In this section, the concept of SK-banyans is introduced. By allowing arbitrary, but predictable, connections between nodes at two adjacent levels in a multistage network, different topologies will emerge with distinctive properties. To formalize this procedure, a construction algorithm is initially presented in its most general form, without regard for connections between levels. Later, by using the concept of connection formulas and by restricting its number in the same network, different subclasses of networks are proposed.

We start our analysis by reviewing the basic properties of an (s, f, l) regular banyan, which were studied in detail in [10]. This graph is a Hasse diagram of a partial ordering in which the following properties hold:

- **banyan property**: there is one and only one path from any base to any apex;
- **l-level property**: all base-to-apex paths are of the same length l;

- **regularity property**: the indegree[1] of every node, except the bases, is f and the outdegree of every node, except the apexes, is s.

A *base* is defined as a node of indegree 0 and an *apex* is defined as a node of outdegree 0. Two basic results can be proved for this graph. First, the number of nodes at each level can be computed from the parameters of the graph. Second, the nodes can be distinctively numbered within each level, from 0 up to $n_i - 1$, the number of nodes at level i. Nothing can be said though, about other properties of the graph as no connection scheme between levels is defined for an (s, f, l) regular banyan. To allow for the systematic definition of connections between nodes, we present first a labelling scheme, and then a connection scheme.

2.1 Labelling scheme

The labelling scheme for the nodes will consist of a tuple <*level*, *order*> in which the first number identifies the level in which this node is located, and the second number identifies the order of the node within that level. Normally, as is the case for multistage interconnection networks, the levels are numbered from 0 to l, and the nodes are numbered from 0 to $n_i - 1$. Typically, the numbering of nodes is done with a number system different from the decimal system to make it easier to specify routing algorithms.

Based on these considerations, we adopt the following labelling scheme for an (s, f, l) regular banyan:

Level numbering:

- vertex levels are numbered using a decimal base, from 0 to l, and this number is called the (vertex) *level number*;
- the top vertex level, called the *apex* level, is numbered l;
- the bottom vertex level, called the *base* level, is numbered 0;
- edge levels are numbered like vertex levels, with level i being the edge level between vertex levels $i + 1$ and i.

Node numbering:

- at each level, including the apex and the base levels, nodes are numbered from 0 to $n_i - 1$, where n_i represents the number of nodes within that level; this number is called the *order number*;
- each order number is a numeric string composed of two substrings, one of them possibly empty, the rightmost substring in base f, and the leftmost substring in base s; an order number for a node at level $l - i$ is represented, according to this format, as a sequence of digits in the form:

[1]To avoid unnecessary clutter, all graphs will be shown here without the arrows that define a directed graph; it is assumed that all edges have arrows pointing upwards.

$$(\cdots d_{i+2}d_{i+1}d_i)_s(d_{i-1}d_{i-2}\cdots)_f$$

where the indexes are defined within the range $[0, l-1]$;

- each node in the graph is identified by a tuple <*level number, order number*>

This numbering scheme is shown in Figure 2.

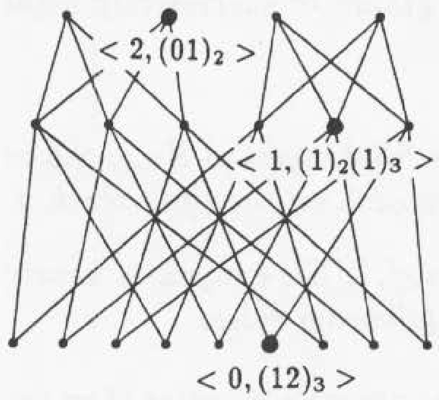


Figure 2: Level and node numbering schemes on an (s, f, l) regular banyan.

2.2 Connection scheme

The connection scheme to be used for SK-banyans will be defined in terms of a *connection formula*, which specifies which nodes at a given level are connected to a particular node at a higher level. This can also be written as a relation R between the two nodes, the relation being that if two nodes a and b are related (aRb), then there is an arc between these two nodes, the node at the upper level (a) being the final vertex, and the node at the lower level (b) being the initial vertex. We will write the relation R as "$\rightarrow$", to mean "*is connected to*". Assuming the node numbering described before, we can describe the connections as pairs of tuples in the form:

$$node < level, order\,number > \rightarrow node < level-1, order\,number > \quad (1)$$

Using the notation given previously for writing the node number as a composition of two numbers in bases s and f, we can write the connection formula above as:

$$node < l-i, (\cdots d_{i+2}d_{i+1}d_i)_s(d_{i-1}d_{i-2}\cdots)_f > \rightarrow$$
$$node < l-i-1, (\cdots d'_{i+2}d'_{i+1})_s(d'_i d'_{i-1}d'_{i-2}\cdots)_f > \quad (2)$$

where:

- $l-i$ - level number of the node at the upper level $(0 \leq i \leq l-1)$

- d_i - digits in base s or f of the order number for a node at the upper level
- d'_i - digits in base s or f of the order number for a node at the lower level
- $d'_i = F(d_i)$

2.3 Recursive definition

One way to define a graph is by giving a construction algorithm to obtain it. This method will be used here because it leads naturally to a recursive definition, which allows the construction of a graph of a large scale given subgraphs of smaller scale. We start with a basic graph, which will correspond to a one-level network, and proceed to define a recursive algorithm to obtain graphs of successively higher number of levels.

Algorithm 1

- **Basic step**: *an* $(s, f, 1)$ *SK-banyan is* $K_{n,m}$, *the complete bipartite directed graph with* n *and* m *vertices in each set, and for which* $n = s$ *and* $m = f$.
- **Recursion step**: *an* (s, f, l) *SK-banyan is constructed from an* $(s, f, l-1)$ *SK-banyan by applying the following rules:*

  1. **Multiplicity rule**: *generate* s *copies of an* $(s, f, l-1)$ *SK-banyan, numbering them from* 0 *to* $s-1$. *Name these graphs the* top graphs.
  2. **Numbering rule**: *renumber every node in copy* i *of the top graphs by attaching digit* i *(in base* s*) to its order number in the most significant position, and by increasing its level number by one.*
  3. **Null graph rule**: *generate a null graph of order* f^l, *and label every node with a number from* 0 *up to* $f^l - 1$ *(in base* f*) and assign them the level number* 0. *Name this graph the* bottom graph.
  4. **Connection rule**: *connect every base node of the top graphs to* f *nodes of the bottom graph, according to the connection formula defined for this level.*

This algorithm is illustrated graphically in Figure 3. Not all connection formulas preserve the banyan property, and it is clear from Algorithm 1 that the key to preserving this property is the definition of the connections between the base nodes of the top graphs and the nodes of the bottom graph. This is so because the only step in Algorithm 1 where a connection is specified is the connection rule, which is related only to the base edge level.

2.4 Examples of connection formulas

We now examine some examples of connection formulas, and the graphs that results from them. First, we make some observations. The set of f^{l-1} base nodes of each one of the s top graphs is called an *upper cluster*. The set of f^{l-1} nodes of the bottom graph whose node numbers have the same most significant digit is called a *lower cluster*. As was defined before, and according to Algorithm 1, there are s upper clusters and f lower

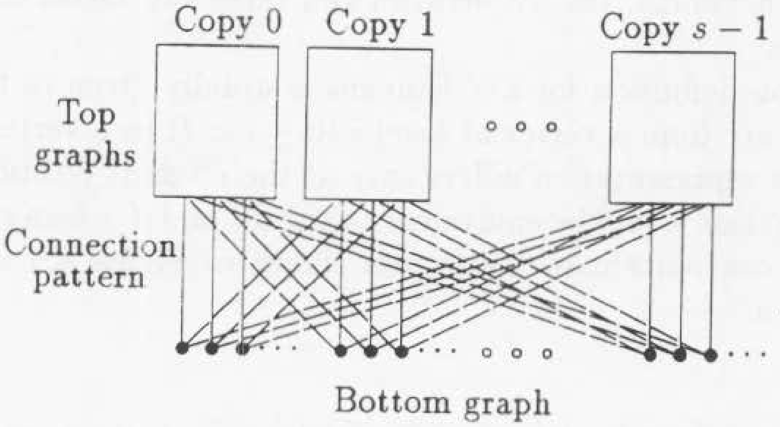


Figure 3: Illustration of the construction algorithm.

clusters at the levels that correspond to the top and bottom graphs, and as a consequence, up to $s \times f$ bijections can be defined between them. This can be conveniently represented as an $s \times f$ matrix in which entry $[i, j]$ corresponds to the bijection between an upper cluster from copy i to a lower cluster whose most significant digit is equal to j. Also, because the cardinality of lower and upper clusters is given by f^{l-1}, the bijections between them will have to be defined between sets of varying cardinality, which will increase with the number of levels. This implies that although we still are dealing with $s \times f$ matrices, the cardinality of its elements will depend on the number of levels.

A second observation regards the relationship between connection formulas for different levels. If they are not related at all, we have a graph whose properties are not readily obtainable, as the irregular connections between levels would be hard to model analytically. On the other hand, if we impose some relationship between the connection formulas for different levels, a regularity may arise from it that makes it possible to study the graph with a tractable model. To avoid extending the subject, we restrict the analysis here to graphs whose connection formulas are the same for all levels.

Besides following a common formula we will say that, if the bijections defined between two levels are arbitrarily chosen and bear no relation to the ones defined between other two levels, we have what will be called a *non-uniform* SK-banyan. Accordingly, if the bijections at any level are chosen according to the same procedure, the graph will be called a *uniform* SK-banyan. Again, to avoid extending the subject, we will restrict the analysis here to only the uniform cases. As it will be seen, this already covers most of the interconnection networks being studied currently, as well as a large number of cases that have not been reported before.

Besides specifying the connection formula, we will also make use of a *connection formula diagram*, which will serve as a visual aid to illustrate the action performed by the connection formula on the digits of a node's order number. In this diagram, the upper part represents the digits of a node at an upper level, and the lower part represents the digits of the nodes at the lower level to which it is connected. Thus, the upper part represents just one node, whereas the lower part represents a range of f nodes. The relation between the digits of the upper and lower level nodes will be represented by a down arrow ("$\downarrow$") to mean exactly the identity bijection, or by a thicker down arrow ("$\Downarrow$") to mean any bijection, including the identity. Following, we give the connection formulas for two different networks: the SW-banyan and three subclasses of the SK-banyan.

SW-banyans

A construction definition for SW-banyans is usually given in terms of a connection rule: there is an arc from a vertex at level i $(0 \leq i < l)$ to a vertex at level $i+1$ if and only if their digit representation differs only at the i^{th} digit position. In our connection formula notation, this would be equivalent to the d's and d''s being different only at digit i. Formally, we can write it in the format shown in Figure 4. A (3,3) SW-banyan is shown in Figure 5.

$$node < l-i, (\cdots d_{i+2}d_{i+1}d_i)_s(d_{i-1}d_{i-2}\cdots)_f > \rightarrow$$
$$node < l-i-1, (\cdots d_{i+2}d_{i+1})_s(j d_{i-1}d_{i-2}\cdots)_f >$$

$$0 \leq j \leq f-1, 0 \leq i \leq l-1$$

Connection formula diagram

d_{l-1}	$\cdots$	d_{i+1}	d_i	d_{i-1}	$\cdots$	d_0
↓	↓	↓		↓	↓	↓
d_{l-1}	$\cdots$	d_{i+1}	j	d_{i-1}	$\cdots$	d_0

Figure 4: Connection formula for SW-banyans.

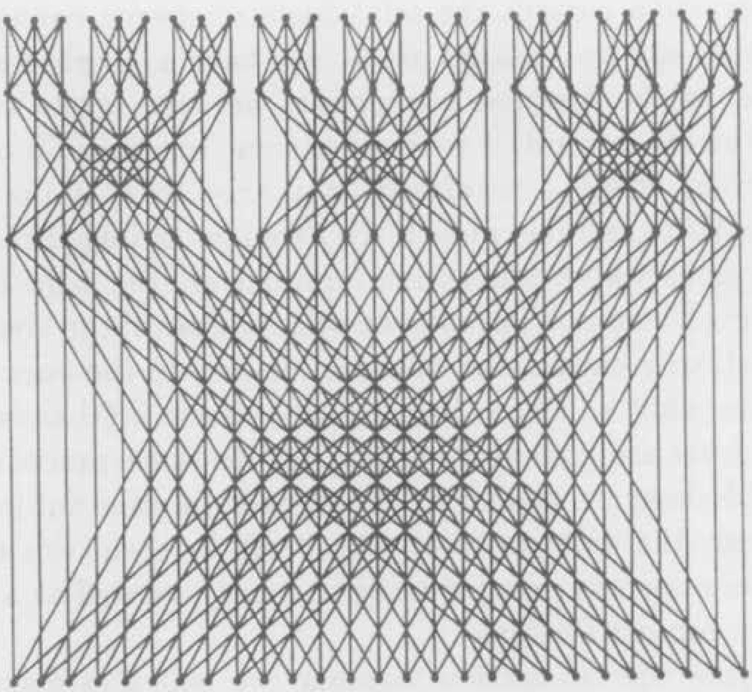


Figure 5: A (3,3) SW-banyan.

Uniform, single-digit SK-banyans

An example of a connection formula for a uniform, single-digit SK-banyan is given in Figure 6, along with its connection diagram. As can be seen, when compared to the SW-banyan's connection formula, the only difference is the bijection between digits d_{i-1} and d'_{i-1}, which is exclusively the identity for SW-banyans, and can be any bijection for SK-banyans. An example of a (3,3) uniform, single-digit SK-banyan is given in figure 7.

$$node < l - i, (\cdots d_{i+2} d_{i+1} d_i)_s (d_{i-1} d_{i-2} \cdots)_f > \rightarrow$$
$$node < l - i - 1, (\cdots d_{i+2} d_{i+1})_s (j d'_{i-1} d_{i-2} \cdots)_f >$$

$$0 \le j \le f-1, 0 \le i \le l-1, 0 \le k < i$$

Connection formula diagram

d_{l-1}	$\cdots$	d_{i+1}	d_i	d_{i-1}	$\cdots$	d_0
↓	↓	↓		⇓	↓	↓
d_{l-1}	$\cdots$	d_{i+1}	j	d'_{i-1}	$\cdots$	d_0

Figure 6: Connection formula for uniform, single-digit SK-banyans.

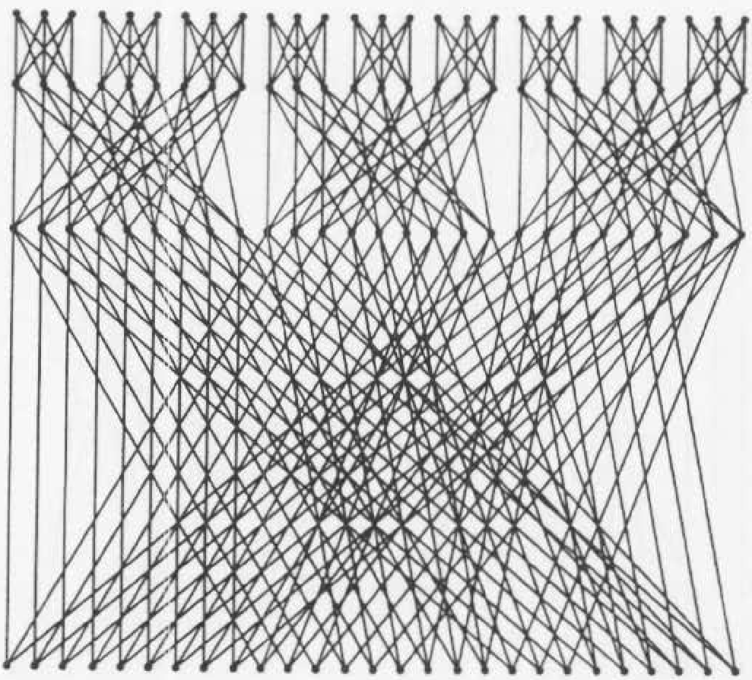


Figure 7: A (3,3) uniform, single-digit SK-banyan.

Uniform, multiple-digit SK-banyans

This subclass of SK-banyans is generated when a connection formula is used such that two or more adjacent digits are submitted to bijections. The basic principle is to submit the digits to the right of digit i to the same or different bijections. In the first case, we call the resulting graph a uniform, *multiple-digit simple* SK-banyan. In the second case, the graph is called uniform, *multiple-digit compound* SK-banyan. In the first case, the bijection that is used to compute the value of d'_{i-1} is also used to compute all the other digits from d'_{i-2} up to d'_0. In the second case, the bijections to compute each of these digits may be different, the one used for computing digit d'_k being itself dependent upon computation of digit d'_{k+1}. An example of the connection formula for the uniform, multiple-digit compound case is given in figure 8, and an example of a graph generated by it is given in figure 9.

CC-banyans

This class of banyan networks has different connection rules than those for SW-banyans. They present a richer set of shift-rotate permutations than SW-banyans, as

$$node < l-i, (\cdots d_{i+2}d_{i+1}d_i)_s(d_{i-1}d_{i-2}\cdots)_f > \rightarrow$$
$$node < l-i-1, (\cdots d_{i+2}d_{i+1})_s(j d'_{i-1} d'_{i-2}\cdots)_f >$$

$$0 \leq j \leq f.-1, 0 \leq i \leq l-1$$

Connection formula diagram

d_{l-1}	$\cdots$	d_{i+1}	d_i	d_{i-1}	$\cdots$	d_0
$\downarrow$	$\downarrow$	$\downarrow$		$\Downarrow\Rightarrow$	$\Downarrow\Rightarrow$	$\Downarrow$
d_{l-1}	$\cdots$	d_{i+1}	j	d'_{i-1}	$\cdots$	d'_0

Figure 8: Connection formula for uniform, multiple-digit compound SK-banyan.

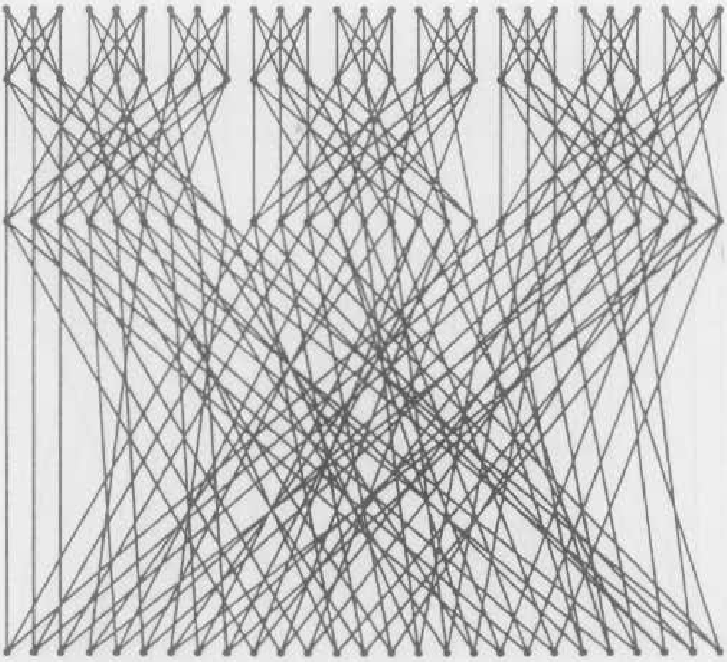


Figure 9: A (3,3) uniform, multiple-digit compound SK-banyan.

well as better distance properties for the base nodes. In [2], it was proven that CC-banyans are one special case of uniform, multiple-digit compound SK-banyans, with a given set of bijections. In this way, the study of this latter subclass can be applied to CC-banyans, including the definition of non-rectangular CC-banyans.

Optimal SK-banyans

Different bijections used in the connection formula may lead to networks with widely varying properties, namely, distance, traffic, and fault-tolerance properties. Both matrix and group-theoretic formulations were defined in [2] to represent such bijections, allowing for the use of a more synthetic expression for a network, and more importantly, allowing the definition of criteria upon which one might determine whether a given network has the "best" properties. In the case of distance and traffic properties, optimality criteria were established that considered two specific constraints: first, that the graph would be *base-symmetric*, meaning that all base nodes would have the same distance and traffic properties, providing an isotropic view of the network; and second, that the distance distribution of each base node would lead to the lowest average base-to-base distance, as specified by a lower bound defined there. Networks belonging to the subclass of uniform. single-digit SK-banyans were found that satisfy both conditions, and for this reason were called *optimal SK-banyans.* They are taken into consideration in the next section when comparisons among the subclasses are discussed.

3 Routing algorithms

In this section, a general set of routing algorithms for SK-banyans is presented for the four cases:

- apex-to-apex routing
- apex-to-base routing
- base-to-apex routing
- base-to-base routing

They are presented here for two reasons. First, because they are based on the connection formulas, they are general enough to be used for *any* SK- banyan network, regardless of its connection formula or bijection. If another routing algorithm has not yet been defined for a given subclass, they can be used as an interim algorithm until a more efficient one is found. Secondly, due to the recursive construction algorithm and the fact that these are l- level networks, for all but the base-to-base case they are the most efficient algorithms possible, and thus there is no need to specify other algorithms for these cases.

We assume for all cases that a routing tag is to be generated such that a *source node* n_s can send a message to a *destination node* n_d by using this routing tag. Because these nodes will be either at the apex or at the base, their level numbers will be dropped in the following treatment. They are assumed to have the appropriate values depending on the context.

Before the routing algorithms are defined, we must define a labelling scheme for the edges of a node, such that a correspondence can be established between them and the digits in the routing tag. It should be observed that, in this case, edges will have different labels depending on the adjacent nodes. That happens because messages are allowed to move in both directions, from base to apex and the inverse, and also because the graphs may not be rectangular.

Definition 1 *For a node at vertex level i, an edge will be labelled as edge j if it connects this node to another node at level $i+1$ $(i-1)$ such that the latter has digit j at position $l-i-1$ $(l-i)$.*

This is the same as to say that an edge adjacent to a node at level i will be labelled j if it connects this node to a node at upper (or lower) cluster j. A simple diagram, shown in Figure 10, illustrates this edge labelling scheme. As can be seen, the same edge may have different labels.

Given this labelling scheme, which is defined for all nodes in the network, a routing tag will be interpreted at each node according to the following rule:

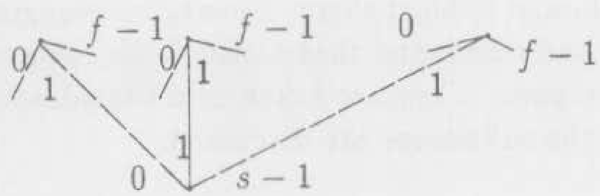


Figure 10: Edge labelling for SK-banyans.

Definition 2 *At level i, a message will be transmitted along a given edge according to the value of digit $l - i - 1$ of its routing tag.*

For nodes at level l, digit 0 of the routing tag will define the route for a message going basewards. By the same token, digit $l - 1$ will define the route for nodes at level 0 for messages going apexwards.

3.1 Apex-to-apex routing

The routing is done in two steps. First, a tag is generated so that the message is sent to any node at a level i given by:

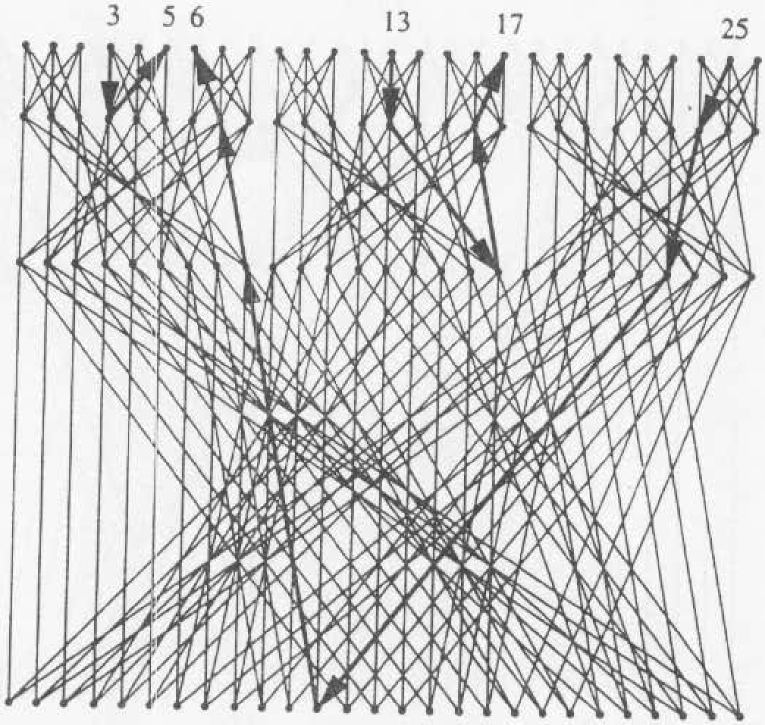
$$i = l - d_{\oplus}(a, b) - 1$$

where $d_{\oplus}(a, b)$ represents the digit distance between a e b.

Next, the routing algorithm for base-to-apex routing is applied from this intermediate node to the destination node utilizing the least significant digits of the routing tag (from digit 0 up to digit $l - i - 1$). It should be noted that in this case, because of the arbitrary definition of the first part of the routing, f^{l-i} different routing tags can be generated. To avoid preference among the reflecting nodes, a "round-robin" or a random selection with uniform distribution can be adopted to pick the reflecting node for every message. This can also be used with some load balancing algorithm to alleviate congestion in heavily loaded links. Figure 11 shows examples of the generation of tags for apex-to-apex routing. In this and other similar tables that follow, the "-" indicates digits in the routing tag that are not used. As mentioned, the baseward part is arbitrary, and in the example the values shown were chosen arbitrarily. Also, if the digit distance between the source and destination nodes is equal to l, that means that any base node can be used as the reflecting node.

3.2 Apex-to-base routing

In this case, the routing depends on the connection formula defined for a given network and on the bijections used. The basic principle is to build the routing tag backwards, starting at digit $l - 1$ and proceeding to digit 0. Digit $l - 1$ is always given by the digit $l - 1$ of n_d, regardless from where the message is coming from, because this is the edge between levels 1 and 0 along which the message must traverse to reach it. The next digit of the routing tag $(l - 2)$ can be computed applying the same principle *if the node at level*



n_s	n_d	Routing tags	
		baseward	apexward
012_3 (5)	010_3 (3)	$--0_3$	$--0_3$
111_3 (13)	122_3 (17)	-12_3	-22_3
221_3 (25)	020_3 (6)	011_3	020_3

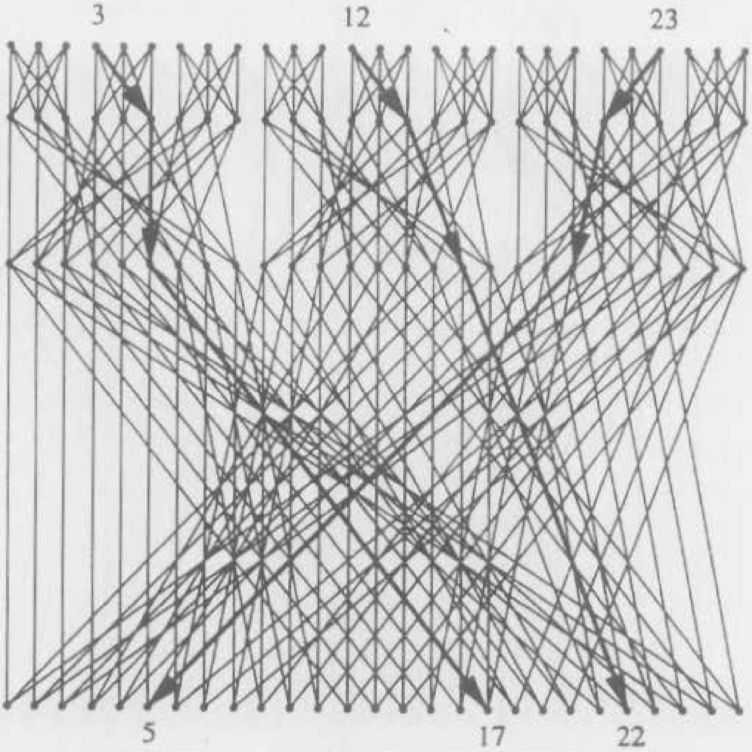
Figure 11: Examples of apex-to-apex routing in SK-banyans.

1 *to which n_d is connected can be identified.* Given the fact that the connection formulas are bijections, then for every image its *inverse* can be computed by applying the inverse bijection. Then, the image of n_d under the inverse bijection can be computed, and its $l-2$ digit will be the corresponding digit in the routing tag. This process can be applied successively to determine the final value for the routing tag. The following algorithm illustrates these operations for a uniform, single-digit SK-banyan.

begin
$t_{l-1} := n_{d_{l-1}}$
for $k := l-2$ **downto** 0 **do**
$t_k := n_{d_k}\ (bijection[n_{s_{k+1}}, t_{k+1}])^{-1}$
end;

In the computation of t_k, the last term represents the inverse of the bijection defined between the lower and upper cluster whose indexes are as given. Examples of apex-to-base routing are shown in Figure 12. The evaluation of the routing tag for one of the examples is given as follows:
source node: $n_s = 3(010)_3$
destination node: $n_d = 17(122)_3$

$$t_2 = n_{d_2} = 1$$
$$t_1 = n_{d_1}\ (bijection[n_{s_2}, t_2])^{-1} =$$



n_s	n_d	Routing tag
010_3 (3)	122_3 (17)	112_3
212_3 (23)	012_3 (5)	000_3
110_3 (12)	211_3 (22)	222_3

Figure 12: Examples of apex-to-base routing in uniform, single-digit SK- banyans.

$$
\begin{aligned}
&= n_{d_1}\ (bijection[0,1])^{-1} = 2\ (\sigma_1)^{-1} = 1 \\
t_0 &= n_{d_0}\ (bijection[n_{s_1}, t_1])^{-1} = \\
&= n_{d_0}\ (bijection[1,1])^{-1} = 2\ (\iota)^{-1} = 2
\end{aligned}
$$

$$
t = (t_2\, t_1\, t_0)_3 = (112)_3
$$

3.3 Base-to-apex routing

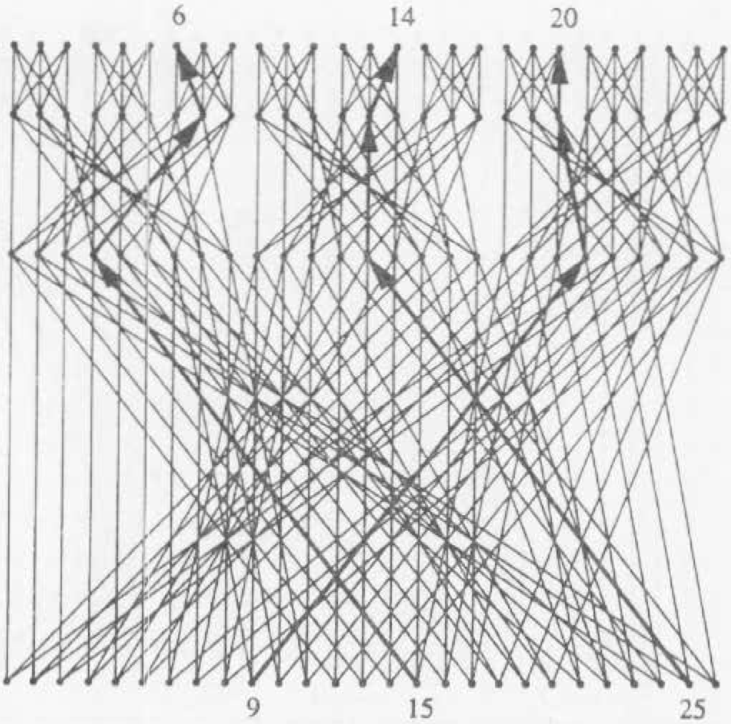
This routing is the same regardless of the bijections or the connection formula. This is so because, as seen by a node at a lower level, the construction algorithm divides the upper clusters into s disjoint subgraphs, each of which follows the conventional s-ary routing scheme. The routing tag is then given by the destination node's order number itself:

$$
t := n_d
$$

As mentioned before, in the case of apex-to-apex routing, the apexward part of the routing tag will be equal to the $l-i-1$ less significant digits of t as given above. Figure 13 shows some examples of base-to-apex routing.

3.4 Base-to-base routing

This case is the case for which the algorithm as given here may prove to be inefficient in comparison with other algorithms. This is because, contrary to the relation between



n_s	n_d	Routing tag
100_3 (9)	202_3 (20)	202_3
120_3 (15)	020_3 (6)	020_3
221_3 (25)	112_3 (14)	112_3

Figure 13: Examples of base-to-apex routing in SK-banyans.

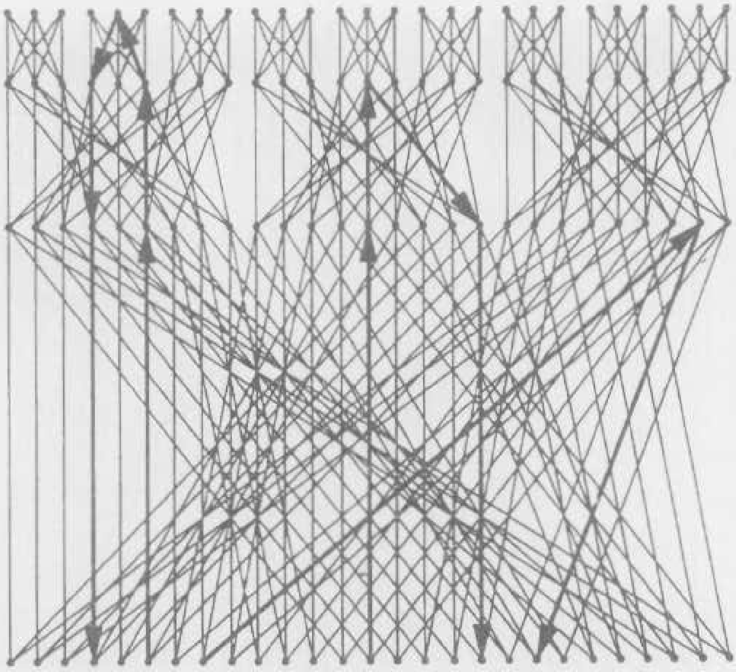
apex nodes, the relation between base nodes may not be recursively defined. Then, to compute the routing tag in this case, there is no recourse other than compute the intersection between the sets of both destination and source nodes at each level, starting at level 1, until a non-empty intersection between them is found. A node from this intersecting set can then be selected as the reflecting node for the message, with a base-to-apex routing followed by an apex-to-base routing.

For the apexward movement, the routing tag can be computed easily, as in the case of base-to-apex routing. The only difference is that only i digits will be used, which determines how far up the message will go. For the baseward movement, the routing tag can be computed using the apex-to-base routing algorithm, except that only i digits need be computed and that the source node number should be the intermediate node's number, instead of an apex node's number. As an example of this routing scheme, Figure 14 shows some cases of routing tags for a uniform, single-digit SK-banyan. As in the case of apex-to-apex routing, if there is more than a reflecting node, one of them is selected. Also, if the intersecting set is empty up to level $l-1$, that means that any apex node can be used as the reflecting node.

4 Average distance and traffic properties

In this section we compare the distance and traffic properties of some classes of SK-Banyans.

Analytical expressions for the base-to-base average distance for SW-, CC-, and optimal SK-banyans were derived in [2]. Table 1 shows these expressions. For $f = 2$, CC-



n_s	n_d	Intersecting set (node)	Routing tags apexward	Routing tags baseward
021_3 (7)	201_3 (19)	$< 1, \{7, 10, 25\} >$ (25)	$2{-}{-}_3$	$2{-}{-}_3$
111_3 (13)	122_3 (17)	$< 2, \{1, 6, 9, 13\} >$ (13)	$11{-}_3$	. $21{-}_3$
012_3 (5)	010_3 (3)	{apex nodes} (4)	011_3	010_3

Figure 14: Examples of base-to-base routing in uniform, single-digit SK- banyans.

and optimal SK-banyans have the same distance properties, for any number of levels, and the average distance for SW-banyans is higher than the average distance of both. For higher values of fanout and number of levels, the average distance for CC-banyans approaches that of SW-banyans. This shows that optimal SK-banyans are, with regard to average distance, a more efficient topology for single-sided networks than the other two topologies.

Regarding traffic properties, expressions for the link traffic density for base-to-base communication for the same three classes (SW, CC and optimal SK-banyans) were also derived in [2]. These expressions are tabulated in Table 2. They assume a uniform distribution in which a base node communicates with all the other base nodes. Taking

SW-banyan	$\bar{d} = \frac{2}{(f-1)f^l}[l\,f^{l+1} - (l+1)\,f^l + 1]$
CC-banyan	$\bar{d} = \frac{2}{(f-1)f^l}[lf^{l+1} - (l+2)f^l + l(f-1) + 2]$
Optimal SK-banyan	$\bar{d} = \frac{2}{(f-1)f^l}[(l-1)\,f^{l+1} - l\,f^l + f + l(f-1)^2]$

Table 1: Expressions for the base-to-base average distance.

SW-banyan	$T_k = 2(f^{l-1} - f^{k-2})$
CC-banyan	$T_k = 2(f^{l-1} - 2f^{k-2} + f^{-1})$
Optimal SK-banyan	$T_k = 2(f^{l-1} - f^{k-1} - f^{-1} + 1)$

Table 2: Expressions for the link traffic density.

the expressions for the link traffic density, one immediate result is that, for the same value of f and l, T_k is a monotonically decreasing function of k. This implies that the busiest links are the ones at the lowest levels, and as such, none of these networks will suffer from the traditional bottleneck that affects single- or multiple-tree networks. Another interesting property, to be attributed to the existence of multiple- trees at the top of every level and through which traffic is distributed evenly, as opposed to a single tree network, is that the growth of traffic at the busiest link is a linear function of the number of base nodes, *i.e.*, $T_1 = O(N)$. The best asymptotic behavior achieved by tree networks is $N^{1.5}$, for both Hypertree-I [5] and KYKLOS-II with H-II routing strategy [9].

It should be expected that optimal SK-banyans would fare better than both SW- and CC-banyans, and that is indeed the case. Consider for instance the link traffic density at the highest level in the network, T_l, and assume $f > 2$. For SW- and CC-banyans, the expressions for T_k, $k = l$, yield values which are still dependent on l, whereas for optimal SK-banyans, T_k reduces to $T_l = 2(1 - f^{-1})$, which is a constant independent of the number of levels in the network.

Table 3 shows some values of the link traffic density computed for the three banyans, for $f = 2, 4, 8$ and $l = 6$, along with the utilization factor of links at each level in relation to the busiest link in the network. Optimal SK-banyans shows the lowest values for all levels, and they decrease faster than the other banyans. Also, the utilization factor of links at the apex level, as just explained, is just a fraction of the factors at the lower levels, as well as negligible in comparison to the other banyans. This is specially useful in the case of fault-tolerant designs, where the low utilization of links at high levels may provide extra paths in case of faults at lower levels, allowing the traffic to be distributed through these links with little degradation in the performance of the system.

4.1 Conclusions

From the results presented in the previous section, one can appreciate the wide variety of topologies provided by SK-banyans. Besides SW- and CC-banyans, which have been studied extensively, and have a very large range of applications, optimal SK-banyans have superior properties that may be useful in many of these applications.

The formulation of multistage interconnection networks in terms of labelling and connection schemes and a recursive construction algorithm allowed the definition of a common model for both existing and new subclasses of banyan networks. The use of a

	$s=2,\ f=2$					
	SW		CC		SK	
k	T_k	u (%)	T_k	u (%)	T_k	u (%)
6	32	51	1	2	1	2
5	48	76	33	52	33	52
4	56	89	49	78	49	78
3	60	95	57	90	57	90
2	62	98	61	97	61	97
1	63	100	63	100	63	100

	$s=4,\ f=4$					
	SW		CC		SK	
k	T_k	u (%)	T_k	u (%)	T_k	u (%)
6	1536	75	1025	50	2	< 1
5	1920	94	1793	88	1538	75
4	2016	98	1985	97	1922	94
3	2040	100	2033	99	2018	99
2	2046	100	2045	100	2042	100
1	2048	100	2048	100	2048	100

	$s=8,\ f=8$					
	SW		CC		SK	
k	T_k	u (%)	T_k	u (%)	T_k	u (%)
6	57344	88	49152	75	2	< 1
5	64512	98	63488	97	57346	88
4	65498	100	65280	100	64514	98
3	65520	100	65504	100	65410	100
2	65534	100	65532	100	65522	100
1	65536	100	65536	100	65536	100

Table 3: Link traffic density for some classes of SK-banyans (fanout = 2, 4 e 8).

connection formula to define the connections between adjacent levels introduced a large number of topologies, besides the existing ones.

The expressions for distance and traffic properties of some of the subclasses have been determined, and can be used as a basis for evaluation of potential applications in communication networks. Further research is underway to characterize traffic properties under dynamic conditions, with both packet and circuit switching.

References

[1] J. C. D. de Melo and R.M. Jenevein, "SK-Banyans: A Unified Class of Banyan Networks," *Proceedings of the 1986 International Conference on Parallel Processing*, pp. 100-107.

[2] J. C. D. de Melo, *A Unified Theory of Banyan Networks*, PhD Dissertation, University of Texas at Austin, 1989.

[3] T. Feng, "A Survey of Interconnection Networks," *Computer*, December 1981, pp. 12-27.

[4] L. R. Goke and G. J. Lipovski, "Banyan Networks for Partitioning Multiprocessor Systems," *Proceedings of the First Annual Symposium on Computer Architecture*, 1973, pp. 21-28.

[5] J. R. Goodman and C. H. Sequin, "Hypertree : A Multiprocessor Interconnection Topology," *IEEE Transactions on Computers*, Vol C-30, No. 12, December 1981, pp. 923-933.

[6] R. M. Jenevein and J. C. Browne, "A Control Processor for a Reconfigurable Array Computer," *Proceedings of the 9th. Annual Symposium on Computer Architecture*, April 1982, pp. 81-89.

[7] G. J. Lipovski and M. Malek, *Parallel Computing, Theory and Comparisons*, John Wiley & Sons, 1987.

[8] M. Malek and E. Opper, "Multiple Fault Diagnosis of SW-Banyan Networks," *13th. Annual International Symposium on Fault-Tolerant Computing*, June 1983, pp. 446-449.

[9] B. L. Menezes and R.M. Jenevein, "KYKLOS : A Linear Growth Fault-tolerant Interconnection Network," *Proceedings of the 14th. International Conference on Parallel Processing*, 1985, pp. 498-502.

[10] U. V. Premkumar, *A Theoretical Basis for the Analysis and Partitioning of Regular SW-Banyans*, Ph.D. Dissertation, University of Texas at Austin, August 1981.

[11] R. Rooholamini, V. Cherkassky and M. Garven, "Finding the Right ATM Switch for the Market", *Computer*, April 1994, pp. 16-28.