Topology of efficiently controllable Banyan multistage networks

Abdou Youssef* **and Bruce Arden**† examine several control classes of Banyan multistage interconnection networks and study their structure

Due to their unique path property, Banyan multistage interconnection networks (MINs) can be self-routed using control tags. This paper introduces a number of routing control classes of MINs and studies their structure. These include the D-controllable networks where the control tags are the destination labels, the FD-controllable networks where the control tags are functions of the destination labels, and the doubly D- or FD-controllable networks which are D- or FD-controllable forward and backward. The paper shows that all D- and FD-controllable networks have a recursive structure, and that all doubly D-controllable (resp., FD-controllable) networks are strictly (resp., widely) functionally equivalent to the baseline network. The subclass of MINs where the interconnections are operations on bits or digits is also studied and shown to be doubly FD-controllable and hence equivalent to the baseline. Finally, the paper presents an efficient, parallel algorithm that relabels the terminals of the baseline to simulate any network in that subclass.

multistage interconnection networks **routing control** **topology**

Banyan multistage interconnection networks (MINs) are increasingly important in parallel computing systems. Their importance grows with the need for fast computation and with the increasing feasibility of systems of thousands of processing elements afforded by VLSI technology. Several networks of this type have been proposed and studied, such as omega and its inverse[1], the indirect binary n-cube[2], the baseline[3] and the generalized cube network[4].

*Department of Electrical Engineering and Computer Science, The George Washington University, Washington, DC 20052, USA.
†College of Engineering and Applied Science, University of Rochester, Rochester, NY 14627, USA
Paper received: 22 August 1991

The efficiency of MINs is critical to overall system performance, and depends on the speed of the routing control, among other things. As these networks have the unique path property, that is, each source has a unique path to each destination, they can be self-routed via control tags. The control efficiency depends then on the speed of control tag computation. If the control tags are stored, the resulting memory cost is prohibitive for large systems. Therefore, networks whose control tags are efficiently computable and need not be stored are of special interest. The most efficiently controllable networks are clearly those whose control tags are the destination addresses. The second most efficient are those whose control tags are simple functions of the destination tags.

Several research efforts addressed equivalence relations between MINs, using the functional or structural approach without consideration of the routing control aspect[3–7]. This paper will study the relationship between the control and the structure of MINs. The understanding of this relationship is important for the design of efficiently controllable networks. Also, new insight into functionality and network equivalence is gained from the study of the control-structure relationship. Particularly, one of the contributions is tying together and superseding the different approaches and results in References 3, 4 and 7 related to the existing multistage interconnection networks, their underlying structure, their control and their equivalence to one another.

In this paper various control schemes or classes are introduced and the structure of the networks in each class is studied. These control classes include D-controllable networks where the control tags are simply the destination tags; FD-controllable networks where the control tags are functions of the destination tags; doubly D-controllable networks which are D-controllable from input to output and from output to input; and the doubly FD-controllable networks which are FD-controllable from input to output

0141-9331/92/010003-11

and from output to input. The last two control classes include all existing MINs and are useful for two-way communication needed in shared-memory systems.

The paper focuses also on the subclass of Banyan multistage networks where the interconnections between columns are bit permutations (or digit permutations in general), which are operations that permute bits in a specified manner. These networks are called digit permutation networks. The reason for studying this subclass is twofold. First, it includes all existing Banyan multistage networks. Second, it turns out that all digit permutation networks are doubly D- or FD-controllable networks and therefore share the same underlying structure as the latter networks.

The main contributions of this paper are the following. First, the topological structure of the D-controllable and FD-controllable networks is determined and shown to be recursive. Second, it is shown that all doubly D-controllable (resp., FD-controllable) networks are strictly (resp., widely) functionally equivalent to the baseline network. This allows the baseline network to simulate any doubly FD-controllable network by relabelling the input and output terminals of the baseline. Third, the paper establishes necessary and sufficient conditions for a sequence of digit permutations to construct a digit permutation network with the unique path property. An optimal algorithm is developed to decide if a sequence of digit permutations meets the aforementioned conditions. Fourth, all digit permutation networks are shown to be doubly FD-controllable and hence widely functionally equivalent to the baseline network. In addition, the control tags are shown to be digit permutations of destination tags, thus allowing for very fast routing control. Finally, an efficient, parallel algorithm that relabels the input and output terminals of one digit permutation network in order to simulate another digit permutation network is given. Such simulation is possible when two networks are widely functionally equivalent.

The paper is organized as follows. The next section gives some preliminary definitions and fundamental concepts related to network control and equivalence. The following section explores the structure of D-controllable networks. FD-controllable networks are treated in the next section. The functional equivalence of doubly D-controllable and doubly FD-controllable networks is established in the section after and the following section studies digit permutation networks. Conclusions and future directions are given in the final section.

DEFINITIONS AND FUNDAMENTAL CONCEPTS

In this section Banyan multistage interconnection networks are specified, functional and topological relations among them are reviewed, their routing control is discussed and various control classes are defined.

Banyan multistage networks

Banyan multistage interconnection networks have N input terminals, N output terminals and k interconnected columns of (N/r) $r \times r$ crossbar switches, where $N = r^k$ and $r \geqslant 2$. Each $r \times r$ crossbar switch realizes all $r!$ permutations.

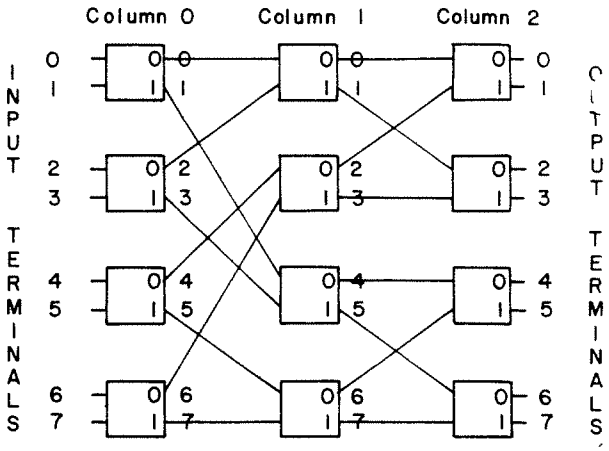

Figure 1. A Banyan multistage network in MIN(2,3)

The interconnection between every two successive columns is a permutation of $S_N = \{0,1,\ldots,N-1\}$. Similarly, the interconnection from the input terminals to the leftmost column is a permutation of S_N and is called the left-end interconnection. The interconnection from the rightmost column to the output terminals is a permutation of S_N, and is called the right-end interconnection. The connectivity of these networks is such that they have the unique path property. That is, between every input terminal i and every output terminal j there is one and only one path, which will be denoted $i \rightarrow j$. The class of these networks is denoted MIN(r,k). Omega and its inverse[1], the indirect binary n-cube[2] and the baseline network[3] are examples of such networks where $r = 2$. Figure 1 shows a network which is in MIN(2,3).

For ease of reference, the input (i.e., left) terminals of networks in MIN(r,k) are labelled $0, 1, \ldots, N-1$ from top to bottom, and so are the output (right) terminals. The input ports and output ports of each column are similarly labelled $0, 1, \ldots, N-1$ from top to bottom. The ports are also labelled locally relative to each switch: the ports (input or output) of each switch are labelled $0, 1, \ldots, r-1$, from top to bottom. The distinction between the two labels will be clear from the context. The columns are numbered $0, 1, \ldots, k-1$ from left to right, and the switches of each column are labelled $0, 1, \ldots, (N/r)-1$ from top to bottom. The labels of terminals and column ports are often represented in r-ary, each label having k r-ary digits. In this context, the local label of a switch port is represented by a single r-ary digit.

If W is a network of MIN(r,k) and f a permutation of S_N, f can be viewed as an interconnection and can be appended to the right end of W, forming a network denoted Wf. That is, if the permutation g is the right-end interconnection of W, then gf is the right-end interconnection of Wf. Another way of viewing Wf is as W except that the output terminals of W are relabelled by f, that is, output terminal j is relabelled by $f(j)$, for every $j = 0, 1, \ldots, N-1$. Similarly, f can be appended to th left of W, forming fW with left-end interconnection fh, where h is the left-end interconnection of W. Viewed differently, fW is the same as W except that the input terminals of W are relabelled by f^{-1}, that is, every input terminal i of W is relabelled by $f^{-1}(i)$.

It should be noted that the composition of functions is

taken from left to right, that is, $(x)fg = g(f(x))$. If $P(W)$ denotes the set of permutations realizable by W, then $P(gWf) = \{ghf | h \in P(W)\}$, which clearly follows from the definition of gWf and the left-to-right view of composition.

Finally, a network W in MIN(r,k) is said to be left bare-ended if its left-end interconnection is the identity permutation. It is said to be right bare-ended if the right-end interconnection is the identity permutation. A network is bare-ended if it is both left and right bare-ended.

Network equivalence relations

Two networks W and W' in MIN(r,k) are strictly functionally equivalent (denoted $W \equiv W'$) if they realize the same permutations. The two networks are widely functionally equivalent if they can be made to realize the same permutations by relabelling the input and/or output terminals of one of the networks, that is, if there exist two permutations g and f of S_N such that $W \equiv gW'f$.

Topological relations are defined next. To this effect, two simple operations on networks are specified. The first, called permute-links-within-switch (PL), consists of disconnecting the links connected to one side (input or output) of an $r \times r$ switch of the network and reconnecting them to different ports of the same side of the same switch. The second, called permute-switches-within-column (PS), consists of permuting the switches within a column in such a way that the links wired to a repositioned switch remain wired to it. Two networks in MIN(r,k) are strictly topologically equivalent if one network can be derived from the other by a sequence of PL and PS operations, and widely topologically equivalent if one can be derived from the other by a sequence of PL and PS operations and by relabelling the input and output terminals.

It has been shown[8, 9] that two networks in MIN(r,k) are strictly topologically equivalent if and only if they are strictly functionally equivalent. Similarly, they are widely topologically equivalent if and only if they are widely functionally equivalent. For that reason, we will often drop the terms topological and functional and speak merely of equivalence, strict or wide.

Control of Banyan multistage interconnection networks

Due to the unique path property, the networks of MIN(r,k) can be self-routed using control tags (CT). Specifically, since there is a unique path between any input terminal i and any output terminal j in a network W in MIN(r,k), there exists a unique r-ary tag $c_{k-1}c_{k-2}\ldots c_0$ which can be used to establish the path $i \rightarrow j$. To show this, let $s_0, s_1, \ldots, s_{k-1}$ be the consecutive switches through which the path $i \rightarrow j$ goes. The unique link between switch s_l and switch s_{l+1} that lies on the path $i \rightarrow j$ leaves switch s_l through some output port of local label b_l $(0 \leqslant b_l \leqslant r-1)$. Let $c_l = b_{k-1-l}$ for all $l = 0,1,\ldots,k-1$. Now that $c_{k-1}c_{k-2}\ldots c_0$ is defined, it can be used as a control tag as follows. Input terminal i sends $c_{k-1}c_{k-2}\ldots c_0$ through the control lines of W, and column l uses the r-ary digit c_{k-1-l} to link the input port of switch s_l, to which the control signal comes, to output port c_{k-1-l}. Clearly then, switch s_0 uses digit c_{k-1}, s_1 uses digit c_{k-2} and so on, establishing the path to output terminal j.

The uniqueness of the path between i and j implies the uniqueness of the tag $c_{k-1}c_{k-2}\ldots c_0$. This tag is called the control tag, denoted CT(W,i,j), or just CT(i,j) when no confusion arises. As an example, the control tag CT(2,6) is 110 (in binary) for the path $0 \rightarrow 6$ in the network of Figure 1.

It is clear that the control tag CT(W,i,j) for the path $i \rightarrow j$ depends on the structure of W. The control tags CT(W,i,j) can be computed and stored in a two-dimensional array such that the ith row is stored in the input terminal i. The amount of storage needed is $N^2k\log r$ bits, which is prohibitive for large N. Therefore, it is preferable to have networks for which the CT(i,j)s are simple to compute and need not be stored, like for instance when CT(i,j) = j, or when CT(i,j) = $f(j)$ for some easy-to-compute f.

A network W in MIN(r,k) is D-controllable if for every input terminal i and every output terminal j, CT(W,i,j) = j (in r-ary), that is, the control tags are the destination tags. The baseline and omega networks are examples of D-controllable networks. A network W in MIN(r,k) is FD-controllable if there exists a permutation f of S_N such that for every input terminal i and every output terminal j CT(W,i,j) = $f(j)$, that is, the control tags are functions of destination tags. In this case, f is called the control function of W. Omega inverse is FD-controllable with control function ρ where $\rho(x_{k-1}x_{k-2}\ldots x_0) = x_0\ldots x_{k-2}x_{k-1}$[1]. Note that in general, the control tags may be a function of both the input terminals and the output terminals.

In shared memory systems the input terminals represent processors and the output terminals memory modules. In this case, communication is from processors to memories and from memories to processors. The above two control definitions concern processor-to-memory communication only. To have efficient memory-to-processor routing, those two definitions should be extended to double controllability as follows. A network W in MIN(r,k) is doubly D-controllable if it is D-controllable from left to right (i.e., input to output) and from right to left (i.e., output to input). In particular, to go from output terminal j to input terminal i, the control tag needed is i. Similarly, a network is doubly FD-controllable if it is FD-controllable from left to right and from right to left. Double controllability can be better understood with the help of inverse networks. The inverse of network W, denoted W^{-1}, is the mirror image of W, where the input terminals of W are the output terminals of W^{-1} and *vice versa*. A network W is doubly D-controllable (resp., doubly FD-controllable) if both W and W^{-1} are D-controllable (resp., FD-controllable).

STRUCTURE OF D-CONTROLLABLE NETWORKS

In this section a subclass of MINs, called generalized recursive networks (GRN), will be defined recursively. Then, it will be shown that every network in GRN is D-controllable and conversely. That is, the structure of D-controllable networks is the same as the GRN structure.

Definition

The class of generalized recursive networks GRN(r,k) is a subclass of MIN(r,k), defined recursively as follows.

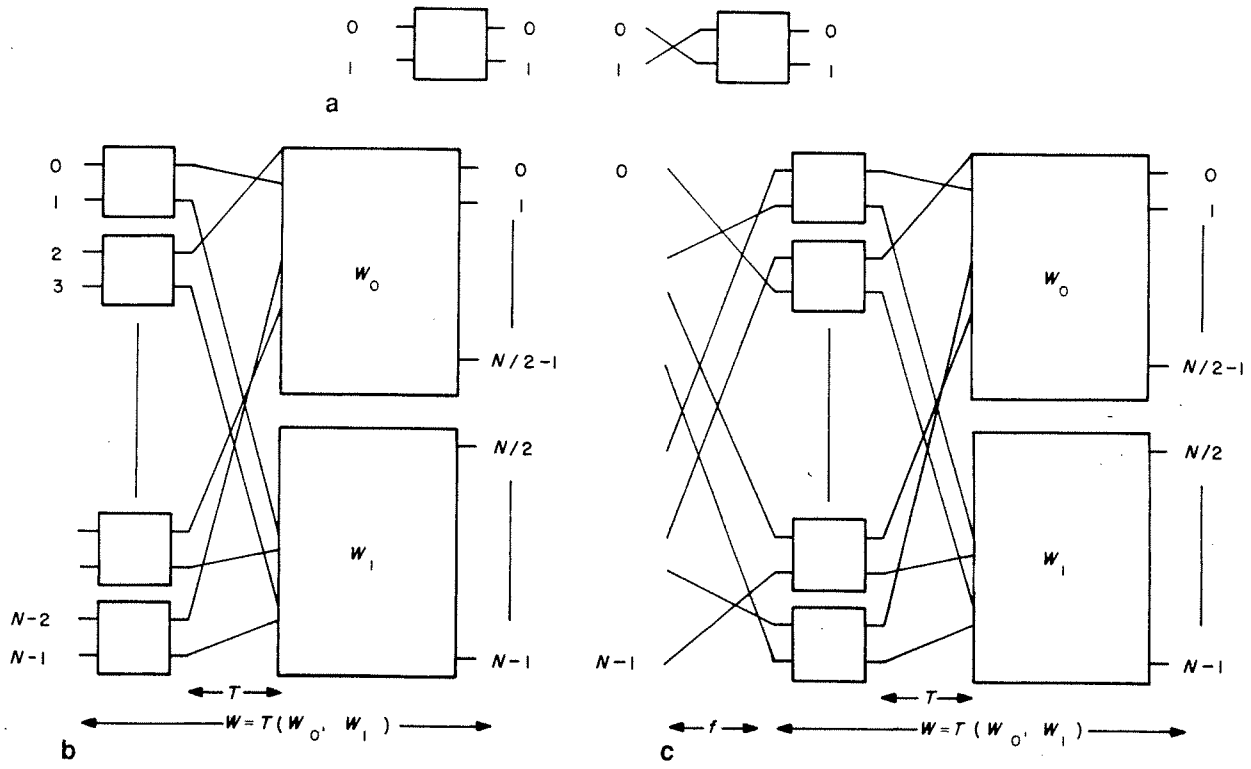

Figure 2. GRN structure

(a) $k = 1$. The networks of GRN(r,1) are mere $r \times r$ switches with left interconnections, as shown in Figure 2a, for the case $r = 2$.

(b) $k > 1$. GRN(r,k) is the class of networks of the form $W = T(W_0, W_1, \ldots, W_{r-1})$ as shown in Figure 2b, where T is the interconnection (i.e., permutation) between column 0 and column 1 of W, each of the W_is is a network in GRN($r,k-1$) such that the input terminals as well as the output terminals of W_i are labelled $ir^{k-1}, ir^{k-1}+1, \ldots, (i+1)r^{k-1}-1$, and T links the ith output port of every switch in column 0 of W to an arbitrary input terminal of W_i, for $i = 0,1,\ldots,r-1$.

(c) If W is as in (b) and f is a permutation of S_N, then fW is also in GRN(r,k) (see Figure 2c).

(d) Non-canonical form. If W is as in (c), and if the switches are permuted within columns, the resulting network is considered to be in GRN(r,k) but is said to be in a non-canonical form. The form of networks as in (c) is called canonical.

Note that the control of a GRN network is the same whether the network is in canonical form or not.

The baseline network[3], an instance of which is shown in Figure 1, is an example of a network in GRN(2,k). In particular, if we denote by $B(2,k)$ the $2^k \times 2^k$ baseline network with 2×2 switches as building blocks, then $B(2,k) = R(B(2,k-1), B(2,k-1))$, where R is the unshuffle. The baseline can be generalized to $B(r,k)$ such that $B(r,1)$ is a mere bare-ended $r \times r$ switch, and $B(r,k) = R(B(r,k-1), \ldots, B(r,k-1))$, where R is the unshuffle in the system of base r, that is, $R(x_{k-1}x_{k-2}\ldots x_0) = x_{k-2}\ldots x_1 x_0 x_{k-1}$ for every k-digit r-ary label $x_{k-1}x_{k-2}\ldots x_0$.

The following two theorems will show that GRN is the same as the class of D-controllable networks.

Theorem 1

Every network in GRN(r,k) is D-controllable.

Proof

Let W be a network in GRN(r,k). We need to show that the control tag $j_{k-1}\ldots j_0$ establishes the path $i \to j$ in W, where $j = j_{k-1}\ldots j_0$ in base r. The proof is by induction on $k \geqslant 1$.

Basis. $k = 1$. In this case W has only one $r \times r$ switch as in Figure 2a. The control tag j_0 links any input i to the output j_0, which is j.

Induction. Assume the statement is true for all networks in GRN($r,k-1$). It will be proved for the network W in GRN(r,k). Let $W = fT(W_0, W_1, \ldots, W_{r-1})$ be a network in GRN(r,k), and assume without loss of generality that it is in canonical form. The digit j_{k-1}, used to control column 0, will cause the input terminal i to link to the j_{k-1}th output port of a switch in column 0. By definition of GRN, port j_{k-1} is linked by T to some input x of $W_{j_{k-1}}$. As $j = j_{k-1}r^{k-1} + j_{k-2}j_{k-3}\ldots j_0$, it follows that j is the $j_{k-2}\ldots j_0$th output terminal of $W_{j_{k-1}}$. Since $W_{j_{k-1}}$ is in GRN($r,k-1$), it follows from the inductive hypothesis that the control tag $j_{k-2}\ldots j_0$ establishes the path $x \to j_{k-2}\ldots j_0$ in $W_{j_{k-1}}$. Consequently, the path $i \to j$, which is $i \to x \to j$, is established in W.

To establish the converse of Theorem 1, the following lemmas are needed.

Lemma 1

Let W be a D-controllable network. Then the following statements are true:

(a) For every $l \leqslant (N/r) - 1$, the output terminals $r \times l$, $r \times l + 1, \ldots, r \times l + r - 1$ are all linked to a single switch s_l in the rightmost column. Furthermore, output terminal $r \times l + t$ is linked to the output port t of s_l for $t = 0,1, \ldots, r - 1$.
(b) The switches of the rightmost column of W can be permuted (within column) so that W becomes right bare-ended.

Proof

(a) Fix $l = l_{k-1} \ldots l_1$. Then $r \times l + t = l_{k-1} \ldots l_1 t$ and hence all the paths from an arbitrary source to the destinations $r \times l + t$, $t = 0,1,2, \ldots, r - 1$, are identical up to the rightmost column because the $k - 1$ leftmost digits of the $(r \times l + t)$s are identical. Therefore, the terminals $r \times l$, $r \times l + 1, \ldots, r \times l + r - 1$ are all linked to the same switch (say switch s_l) in the rightmost column. In addition, as the rightmost digit t of $r \times l + t$ controls the rightmost column connecting the incoming input port of s_l to the output port t of s_l, it follows that output terminal $r \times l + t$ is linked to the output port t of s_l.
(b) Follows immediately from (a): move switch s_l to position l for every l.

In the next lemma, the switches of the rightmost $k - 1$ columns in a D-controllable network W in MIN(r,k) will be partitioned into r groups, such that each group forms a D-controllable network in MIN($r,k - 1$). To this effect, let $U_n = \{d_{k-1}d_{k-2} \ldots d_0 \mid d_{k-1} = n\}$. $(U_n)_{0 \leqslant n < r-1}$ is a partition of the output terminals. Let also U_n^i be the set of switches of column i ($i \geqslant 1$) that are reachable from some output terminal in U_n.

Lemma 2

Let W be a D-controllable network. Then the following statements are true:

(a) For every $n \neq m$ and for every i, $U_n^i \cap U_m^i = \emptyset$.
(b) For every i and n, the switches in U_n^i are linked forward only to switches in U_n^{i+1}.
(c) For every i and n, $|U_n^i| = r^{k-2}$.

Proof

(a) The proof is by contradiction. Let s be a switch in $U_n^i \cap U_m^i$ for some $n \neq m$. Hence, s can reach some output terminal $nd_{k-2} \ldots d_0$ in U_n and another $md'_{k-2} \ldots d'_0$ in U_m. There exists some input terminal p of W that can reach through s the output terminals $nd_{k-2} \ldots d_0$ and $md'_{k-2} \ldots d'_0$ via the control tags $nd_{k-2} \ldots d_0$ and $md'_{k-2} \ldots d'_0$, respectively, because W is D-controllable. Hence, the two tags must agree in the i leftmost digits, yielding $m = n$, and leading to a contradiction.
(b) If a switch s in U_n^i were linked to a switch in U_m^{i+1} for some $m \neq n$, then s could reach an output terminal in U_n and another in U_m. This would lead to the same contradiction as above.
(c) The proof is by backward induction on i.

Basis. $i = k - 1$. Follows immediately from Lemma 1 (a,b).

Induction. Assume the statement is true for all values of $i = k - 1, \ldots, l + 1$. It will be proved for $i = l$. By the inductive hypothesis, we have $|U_n^{l+1}| = r^{k-2}$. Using (b), it can be concluded that all the outgoing links from the switches in U_n^l go to switches in U_n^{l+1}, and all the incoming links to the switches of U_n^{l+1} come from switches in U_n^l. Therefore, the number of the links between the switches of U_n^l and the switches of U_n^{l+1} is equal to $r|U_n^l|$ on the one hand, and to $r|U_n^{l+1}|$ on the other hand. Hence, $|U_n^l| = |U_n^{l+1}| = r^{k-2}$.

Theorem 2

Every D-controllable network in MIN(r,k) is in GRN(r,k).

Proof

Let W be a D-controllable network in MIN(r,k). It will be shown by induction on k that W is in GRN(r,k).

Basis. $k = 1$. It is obvious that W is a single switch that is right bare-ended (after Lemma 1 (b)), and hence in GRN($r,1$).

Induction. Assume the theorem is true for all values of $k < l$. It will be proved for $k = l$. By Lemma 2 (c), each U_n^i has r^{l-2} switches. Permute the switches of the rightmost column so that W becomes right bare-ended. For every $i = 1,2, \ldots, l - 2$ permute the switches of column i so that the labels of the switches of U_n^i are nr^{l-2}, $nr^{l-2} + 1, \ldots, (n + 1)r^{l-2} - 1$. Since $U_n^i \cap U_m^i = \emptyset$ for every $n \neq m$ (Lemma 2 (a)), and since the outgoing links from the switches of U_n^i go only to switches in U_n^{i+1} (Lemma 2 (b)), it follows that the subnetworks $W_n = (U_n^1, U_n^2, \ldots, U_n^{l-1})$, for $n = 0,1,2, \ldots, r - 1$, are disjoint. Hence, W can be put in the form $W = fT(W_0, W_1, \ldots, W_{r-1})$ for some f and T. It can be seen that each W_n is D-controllable. By the inductive hypothesis, each W_n is in GRN($r,l - 1$). Consequently, W is in GRN(r,l).

Therefore, the topologically defined class GRN is identical to the control-characterized class of D-controllable networks. This topological characterization will be used to show that all doubly D-controllable networks are topologically and functionally equivalent. It can also be used to develop an optimal $O(N \log_r N)$ algorithm to decide if a MIN is D-controllable[8].

STRUCTURE OF FD-CONTROLLABLE NETWORKS

The class GRN is extended in this section and shown to be identical to the class of FD-controllable networks. Recall that if W is an FD-controllable network in MIN(r,k), there is a permutation f, called the control function, such that the control tag CT(i,j) to establish the path $i \rightarrow j$ is $f(j)$.

Definition

The extended GRN, denoted EGRN(r,k) is the class of networks of the form Wf where W is in GRN(r,k) and f is a permutation of S_{rk}.

The following lemma relates the networks of the extended GRN to the FD-controllable networks and their control functions.

Lemma 3

(a) If W is in GRN(r,k) and f a permutation of $S_{r,k}$, then Wf is FD-controllable and the control tag CT(Wf,i,j) for the path $i \rightarrow j$ is $f^{-1}(j)$.
(b) If W is FD-controllable with control function g, then Wg is D-controllable.
(c) If W' is FD-controllable, then there exist a network W in GRN(r,k) and a permutation f such that $W' = Wf$.

Proof

(a) Going from i to j in Wf is the same as going from i to $f^{-1}(j)$ in W. Therefore, CT(Wf,i,j) = CT($W,i,f^{-1}(j)$) which is equal to $f^{-1}(j)$ because W is in GRN and hence D-controllable. It follows that Wf is FD-controllable and its control function is f^{-1}.
(b) CT(Wg,i,j) = CT($W,i,g^{-1}(j)$) = $g(g^{-1}(j) = j$. Therefore, W is D-controllable.
(c) Let W' be an FD-controllable network, and g its control function. Let also $W = W'g$ and $f = g^{-1}$. After (b), W is D-controllable and hence in GRN. Clearly, $Wf = (W'g)g^{-1} = W'(gg^{-1}) = W'$.

Theorem 3

A network in MIN(r,k) is FD-controllable if and only if it is in EGRN(r,k).

Proof

It follows from Lemma 3 (a,c).

DOUBLY CONTROLLABLE NETWORKS

As pointed out earlier, it is of theoretical as well as practical interest to study the networks that are efficiently controllable not just from left terminals to right terminals, but also from right terminals to left terminals (R-to-L).

Recall that by right-to-left D-controllability it is meant that if right terminal j needs to communicate to left terminal $i = i_{k-1}i_{k-2}\ldots i_0$ in a network W, then the control tag $i_{k-1}i_{k-2}\ldots i_0$ is used to establish the path as follows. The digit i_{k-1} controls the switches of column $k-1$ (i.e., the rightmost column), i_{k-2} controls the switches of column $k-2$, and so on. Right-to-left FD-controllability is defined similarly, where the control tag (denoted $CT_{R\text{-to-}L}(i,j)$) to establish the path from right terminal j to left terminal i is $f(i)$. The permutation f is then called the right-to-left control function.

So for doubly controllable networks, we make a distinction between the left-to-right control tags $CT_{L\text{-to-}R}$ (the old sense of CT) and the right-to-left control tags $CT_{R\text{-to-}L}$. We also make a distinction between left-to-right control functions and right-to-left control functions, which do not have to be identical.

Note that most of the existing networks have been shown to be doubly D-controllable or doubly FD-controllable. In particular, the baseline network, which is D-controllable, has been shown to be equal to its inverse[3], and hence it is doubly D-controllable. Omega inverse has been shown to be FD-controllable[1]. Therefore, the omega network is doubly FD-controllable. Other networks, such as the indirect binary n-cube[2] and the generalized cube network[4], have been or can be shown to be doubly FD-controllable. It has been established that omega, omega inverse, the baseline, the indirect binary n-cube and the augmented cube network are topologically and functionally equivalent[3]. In this section, this will be generalized. It will be shown that all doubly D-controllable networks are strictly functionally equivalent and all doubly FD-controllable networks are widely functionally equivalent to the baseline $B(r,k)$.

The proof of the equivalence of the doubly D-controllable networks with the baseline $B(r,k)$ involves the following steps. First, it will be established that if W is a doubly D-controllable network, then the switches of its leftmost and rightmost columns can be repositioned so that W becomes bare-ended of the form $W = T(W_0, W_1, \ldots, W_{r-1})$. Afterwards, it will be shown that the switches in column 1 can be repositioned so that T becomes identical to the corresponding interconnection R in $B(r,k)$ and all the W_is become doubly D-controllable. Finally, the proof will proceed by induction on k. These steps will be made precise next.

Lemma 4

Let W be a doubly D-controllable network in MIN(r,k). Then the following statements are true:

(a) The switches of column 0 and column $k-1$ of W can be repositioned so that W becomes bare-ended.
(b) Assume that W is bare-ended, and in canonical form $W = T(W_0, W_1, \ldots, W_{r-1})$. Then the switches of the leftmost column of each W_l can be repositioned so that T becomes identical to the corresponding interconnection R in the baseline $B(r,k)$.
(c) Assume W is as in (b) and $T = R$. Then W_i is doubly D-controllable for every $i = 0,1,\ldots,r-1$.

Proof

(a) Applying Lemma 1 (b) to W, we conclude that column $k-1$ can be permuted so that W becomes right bare-ended. Since column 0 of W is column $k-1$ of W^{-1} and W^{-1} is D-controllable, we can apply the same lemma on W^{-1} to make W^{-1} right bare-ended, which amounts to making W left bare-ended.
(b) Consider W_l for some arbitrary l, and let $i_{k-1}\ldots i_2$ be a $(k-2)$-digit r-ary label. Using the double D-controllability of W, it can be shown that the right ports $(i_{k-1}\ldots i_2 tl)_{0<t<r-1}$ of column 0 of W are linked to a single switch (say $s_{i_{k-1}\ldots i_2}$) in column 1 of W, such that the right port $i_{k-1}\ldots i_2 tl$ is linked to the tth left port of switch $s_{i_{k-1}\ldots i_2}$. Now if switch $s_{i_{k-1}\ldots i_2}$ is moved to position $li_{k-1}\ldots i_2$ of column 1 of W for every $i_{k-1}\ldots i_2$, its left ports get the global labels $(li_{k-1}\ldots i_2 t)_{0<t<r-1}$ which are respectively equal to the unshuffle of the labels $(i_{k-1}\ldots i_2 tl)_{0<t<r-1}$. In all, the interconnection from column 0 to column 1 of W becomes identical with the unshuffle R.
(c) As each W_l is D-controllable, it suffices to show that each W_l is D-controllable from right to left. Let $i = i_{k-1}i_{k-2}\ldots i_1$ and j be an input terminal and an output terminal of W_l, respectively, for some arbitrary l. The equivalent label of i in W is $i' = li_{k-1}i_{k-1}\ldots i_1$. As $T = R$, i' is linked to the right port $R^{-1}(i')$ of column 0 of W. The path $R^{-1}(i') \leftarrow j$ from output

terminal j to input terminal $R^{-1}(i')$ in W has to go through i'. As W is D-controllable from right to left, the control tag for this path is $R^{-1}(i') = i_{k-1}i_{k-1}\ldots i_1 l$. Hence, the subtag $i_{k-1}\ldots i_1$ establishes the subpath from j to i'. Since i' is the same port as i and $i = i_{k-1}\ldots i_1$, it follows that the control tag for the path from j to i in W_l is i. Consequently, W_l is D-controllable from right to left.

Theorem 4

All doubly D-controllable networks in MIN(r,k) are strictly equivalent to the baseline network $B(r,k)$.

Proof

Due to part (a) of the previous lemma, we can limit the proof to bare-ended doubly D-controllable networks. The proof is by induction on k.

Basis. $k = 1$. Clearly, each bare-ended doubly D-controllable network is a bare-ended single $r \times r$ crossbar switch, which is $B(r,1)$.

Induction. Assume the theorem holds for all doubly D-controllable networks in MIN($r,k-1$). Let W be a doubly D-controllable network in MIN(r,k). By the previous lemma (a,b), W can be assumed to be in a canonical form $W = T(W_0, W_1, \ldots, W_{r-1})$, where $T = R$, the corresponding interconnection in the baseline $B(r,k)$. Using part (c) of the previous lemma, each W_i is doubly D-controllable. By the inductive hypothesis, each W_i is strictly topologically equivalent to $B(r,k-1)$. It follows that W is strictly topologically equivalent to $B(r,k)$.

The following lemma will relate doubly FD-controllable networks, the baseline network and the control functions.

Lemma 5

(a) If $W = gB(r,k)f$, then $CT_{L\text{-to-}R}(W,i,j) = f^{-1}(j)$ and $CT_{R\text{-to-}L}(W,i,j) = g(i)$.
(b) If W is doubly FD-controllable, and if $CT_{L\text{-to-}R}(W,i,j) = f(j)$ and $CT_{R\text{-to-}L}(W,i,j) = g(i)$, then W is strictly topologically equivalent to $gB(r,k)f^{-1}$.

Proof

(a) Since $B(r,k)$ is in GRN(r,k), it follows that $gB(r,k)$ is also in GRN(r,k). Consequently, $CT_{L\text{-to-}R}(W,i,j) = f^{-1}(j)$, after Lemma 3. As $W^{-1} = f^{-1}B^{-1}(r,k)g^{-1} = f^{-1}B(r,k)g^{-1}$, we conclude in the same way that $CT_{R\text{-to-}L}(W,i,j) = CT_{L\text{-to-}R}(W^{-1},j,i) = g(i)$.
(b) Since W is FD-controllable and $CT_{L\text{-to-}R}(W,i,j) = f(j)$, it follows that Wf is D-controllable (after Lemma 3 (b)), and consequently $g^{-1}Wf$ is D-controllable. As W^{-1} is also FD-controllable, it can be similarly shown that $f^{-1}W^{-1}g$ is D-controllable. Hence, $g^{-1}Wf$ and $(g^{-1}Wf)^{-1}$ are both D-controllable. Using Theorem 4, we have $g^{-1}Wf \equiv B(r,k)$, and consequently, $W \equiv gB(r,k)f^{-1}$.

Theorem 5

All doubly FD-controllable networks in MIN(r,k) are widely equivalent to the baseline network $B(r,k)$.

Proof

Follows from the previous lemma.

Lemma 5 has other implications, some of which have been proved elsewhere through other methods[3] about the relations among some of the existing networks. The following theorem rediscovers these relations and generalizes them for arbitrary switch sizes $r \geq 2$.

Theorem 6

$B^{-1}(r,k) \equiv B(r,k)$, $\Omega(r,k) \equiv \rho B(r,k)$ and $\Omega^{-1}(r,k) \equiv B(r,k)\rho$ where ρ is the digit reversal ($\rho(x_{k-1}\ldots x_1x_0) = x_0x_1\ldots x_{k-1}$).

Proof

Since $B(r,k)$ is doubly D-controllable, it follows that its inverse is doubly D-controllable. Consequently, $B^{-1}(r,k) \equiv B(r,k)$, after Theorem 4. $\Omega(r,k)$ is doubly FD-controllable, its left-to-right control function is the identity permutation and its right-to-left control function is ρ^1. After Lemma 5(b), we have $\Omega(r,k) \equiv \rho B(r,k)$. Furthermore, $\Omega^{-1}(r,k) \equiv (\rho B(r,k))^{-1} \equiv B^{-1}(r,k)\rho^{-1} \equiv B(r,k)\rho$ because $\rho^{-1} = \rho$.

The equivalence among existing MINs is therefore no coincidence since they are doubly D- or FD-controllable. In the remaining part of the paper, the double controllability of these networks will be shown to be a result of their inter-column interconnections, namely bit manipulation permutations, which are operations that permute the bits of binary labels in a specified manner. The class of digit permutation networks, where the inter-column interconnections are digit permutations (or bit permutations when $r = 2$), will be defined and shown to be doubly FD-controllable and hence equivalent to the baseline network. In addition, an algorithm will be given to determine the relabelling of the terminals of one digit permutation network when it is to simulate another digit permutation network.

DIGIT PERMUTATION NETWORKS

As was just pointed out, one common feature in the definitions of the existing multistage interconnection networks is that the interconnections between columns are bit permutations. The well-known shuffle interconnection is an example. Some of the reasons for using these permutations as interconnections are their regularity, rich structure and ease of analysis.

These same reasons may tempt one to propose and study new MINs that have as intercolumn interconnections bit permutations or, in the general case where the switches are $r \times r$, digit permutations that permute digits of r-ary labels. In this section, the whole class of digit permutation networks will be studied using the concepts of D-control, FD-control and double FD-control, and taking advantage of the equivalence among all doubly FD-controllable networks.

The approach is algebraic. A relation will be derived relating the input terminal, the output terminal and the control tag that establishes the path in between. This relation will be used to find necessary and sufficient conditions for $k + 1$ digit permutations to construct a MIN that has the unique path property. Later the control

tags in digit permutation networks are shown to be functions of the destination tags only. This makes them FD-controllable. Making use of the fact that the inverse of a digit permutation network is a digit permutation network, it will be concluded that digit permutation networks are doubly FD-controllable and hence widely equivalent to the baseline network.

Definition

A permutation f of $S_N = \{0,1,\ldots,N-1\}$, where $N = r^k$, is a digit permutation in the base system r if there exists a permutation π of $S_k = \{0,1,\ldots,k-1\}$ such that $f(x_{k-1}\ldots x_1x_0) = x_{\pi(k-1)}\ldots x_{\pi(1)}x_{\pi(0)}$, where $x_{k-1}\ldots x_1x_0$ is an arbitrary k-digit r-ary label. In this case, f is denoted f_π and π is called the kernel of f_π.

Definition

A digit permutation network, denoted DPN($f_0, f_1, \ldots f_k$), is a network in MIN(r,k) where the leftmost interconnection is f_0, the rightmost interconnection is f_k, the interconnection from column $i-1$ to column i is f_i, for $i = 1,\ldots,k-1$, and all the f_is are digit permutations of S_{r^k} in the base system r.

Denote by E^i_a, where a is an r-ary digit and $i = 0,1,\ldots,k-1$, the following mapping from S_N to S_N:

$$E^i_a(x_{k-1}\ldots x_0) = x_{k-1}\ldots x_{i+1}ax_{i-1}\ldots x_0$$

that is, E^i_a replaces the ith digit by a.

Next, the relation between an arbitrary input terminal s, an arbitrary output terminal d and the control tag $c = c_{k-1}c_{k-2}\ldots c_0$ for the path $s \rightarrow d$ in a digit permutation network DPN ($f_{\pi_0}, f_{\pi_1}, \ldots, f_{\pi_k}$) will be derived. Recall that the digit c_{k-1-i} controls column i for $i = 0,1,\ldots,k-1$. Note that if the path $s \rightarrow d$ enters column i through input port $x_{k-1}\ldots x_1x_0$, it exits that column through the output port $x_{k-1}\ldots x_1c_{k-1-i}$ which is equal to $E^0_{c_{k-1-i}}(x_{k-1}\ldots x_1x_0)$. Note also that if the path exits column $i-1$ through some output port y, it then enters the next column, that is, column i, through input port $f_{\pi_i}(y)$ because the interconnection between column $i-1$ and column i is f_{π_i}. We have thus proved the following lemma.

Lemma 6

In a digit permutation network DPN($f_{\pi_0}, f_{\pi_1}, \ldots, f_{\pi_k}$) an arbitrary output terminal d is related to an arbitrary input terminal s and the control tag $c = c_{k-1}c_{k-2}\ldots c_0$ for the path $s \rightarrow d$ by the following relation: $d = (s)f_{\pi_0}E^0_{c_{k-1}}f_{\pi_1}E^0_{c_{k-2}}f_{\pi_2}E^0_{c_{k-3}}\ldots E^0_{c_0}f_{\pi_k}$.

The Es will be 'filtered' out to the right of the fs in the relation above. To that effect, the following lemma is needed. The proof is straightforward and hence omitted.

Lemma 7

$f_\alpha f_\beta = f_{\beta\alpha}$ and $E^i_a f_\pi = f_\pi E^{\pi^{-1}(i)}_a$.

The following lemma will simplify the relation in Lemma 6 and enable us to derive the necessary and sufficient conditions that the f_{π_i}s have to satisfy in order for the network to have the unique path property.

Lemma 8

Under the assumptions of Lemma 6 we have $d = (s)f_{\beta_0}E^{\beta_1^{-1}(0)}_{c_{k-1}}E^{\beta_2^{-1}(0)}_{c_{k-2}}\ldots E^{\beta_k^{-1}(0)}_{c_0}$ where $\beta_i = \pi_k\pi_{k-1}\ldots\pi_i$.

Proof

Let $g = f_{\pi_0}E^0_{c_{k-1}}f_{\pi_1}E^0_{c_{k-2}}f_{\pi_2}E^0_{c_{k-3}}\ldots E^0_{c_1}f_{\pi_{k-1}}E^0_{c_0}f_{\pi_k}$

By making repeated use of Lemma 7 on the expression of g (from right to left) we have

$$g = f_{\pi_0}E^0_{c_{k-1}}f_{\pi_1}E^0_{c_{k-2}}f_{\pi_2}E^0_{c_{k-3}}\ldots f_{\pi_{k-2}}E^0_{c_1}f_{\pi_{k-1}}f_{\pi_k}E^{\pi_k^{-1}(0)}_{c_0}$$

$$g = f_{\pi_0}E^0_{c_{k-1}}f_{\pi_1}E^0_{c_{k-2}}f_{\pi_2}E^0_{c_{k-3}}\ldots f_{\pi_{k-2}}E^0_{c_1}f_{\beta_{k-1}}E^{\beta_k^{-1}(0)}_{c_0}$$

because $\beta_k = \pi_k$ and $f_{\pi_{k-1}}f_{\pi_k} = f_{\beta_{k-1}}$

$$g = f_{\pi_0}E^0_{c_{k-1}}f_{\pi_1}E^0_{c_{k-2}}f_{\pi_2}E^0_{c_{k-3}}\ldots f_{\pi_{k-2}}f_{\beta_{k-1}}E^{\beta_{k-1}^{-1}(0)}_{c_1}E^{\beta_k^{-1}(0)}_{c_0}$$

$$g = f_{\pi_0}E^0_{c_{k-1}}f_{\pi_1}E^0_{c_{k-2}}f_{\pi_2}E^0_{c_{k-3}}\ldots f_{\beta_{k-1}\pi_{k-2}}E^{\beta_{k-1}^{-1}(0)}_{c_1}E^{\beta_k^{-1}(0)}_{c_0}$$

$$g = f_{\pi_0}E^0_{c_{k-1}}f_{\pi_1}E^0_{c_{k-2}}f_{\pi_2}E^0_{c_{k-3}}\ldots f_{\beta_{k-2}}E^{\beta_{k-1}^{-1}(0)}_{c_1}E^{\beta_k^{-1}(0)}_{c_0}$$

$$\ldots$$

$$g = f_{\beta_0}E^{\beta_1^{-1}(0)}_{c_{k-1}}E^{\beta_2^{-1}(0)}_{c_{k-2}}\ldots E^{\beta_k^{-1}(0)}_{c_0}$$

As $d = g(s)$ (from Lemma 6), the lemma follows.

The necessary and sufficient conditions as well as the relation between the control tag and the output terminal can now be easily derived as follows.

Theorem 7

Let $f_{\pi_0}, f_{\pi_1}, \ldots$, and f_{π_k} be $k+1$ digit permutations, and $\beta_i = \pi_k\pi_{k-1}\ldots\pi_i$.

(a) The digit permutation network DPN($f_{\pi_0}, f_{\pi_1}, \ldots, f_{\pi_k}$) is in MIN($r,k$), that is, it has the unique path property, if and only if $\beta_1^{-1}(0), \beta_2^{-1}(0), \ldots, \beta_k^{-1}(0)$ are pairwise distinct.

(b) The control tag $c = c_{k-1}c_{k-2}\ldots c_0$ for a path $s \rightarrow d$ in a digit permutation network DPN($f_{\pi_0}, f_{\pi_1}, \ldots, f_{\pi_k}$) which has the unique path property is $c = f_\gamma(d)$, where $\gamma(i) = \beta_{k-i}^{-1}(0)$.

Proof

(a) Let s be an input terminal, d an output terminal and $c = c_0c_1\ldots c_{k-1}$ the control tag that establishes the path $s \rightarrow d$. Let $s' = f_{\beta_0}(s)$. Using the previous lemma, we have

$$d = (s')E^{\beta_1^{-1}(0)}_{c_{k-1}}E^{\beta_2^{-1}(0)}_{c_{k-2}}\ldots E^{\beta_k^{-1}(0)}_{c_0}$$

It can be easily seen that the effect of each $E^{\beta_i^{-1}(0)}_{c_{k-i}}$ is to replace the digit in position $\beta_i^{-1}(0)$ of s' by c_{k-i}.

Assume first that the network has the unique path property. If $\beta_1^{-1}(0), \beta_2^{-1}(0), \ldots, \beta_k^{-1}(0)$ are not pairwise distinct, then $\{\beta_1^{-1}(0), \beta_2^{-1}(0), \ldots, \beta_k^{-1}(0)\}$ is a proper subset of $\{0,1,\ldots,k-1\}$, and therefore, there exists some j in $\{0,1,\ldots,k-1\}$ such that $j \neq \beta_i^{-1}(0)$ for all i.

Consequently, the digits in the jth digit position of s' and d must always agree. It follows that for a fixed s, and thus fixed s', no matter what control tag we use, we can never reach any output terminal d whose jth digit differs from that of s'. This contradicts the unique path property.

Conversely, if $\beta_1^{-1}(0), \beta_2^{-1}(0), \ldots, \beta_k^{-1}(0)$ are pairwise distinct, then

$$\{\beta_1^{-1}(0), \beta_2^{-1}(0), \ldots, \beta_k^{-1}(0)\} = \{0,1,\ldots,k-1\},$$

and therefore the mapping γ where $\gamma(i) = \beta_{k-i}^{-1}(0)$ is a permutation of $\{0,1,\ldots,k-1\}$. Furthermore,

$$d = (s')E_{c_{k-1}}^{\gamma(k-1)} E_{c_{k-2}}^{\gamma(k-2)} \ldots E_{c_0}^{\gamma(0)}$$

implying that the digit in position $\gamma(i)$ of d is c_i, that is, $d_{\gamma(k-1)}d_{\gamma(k-2)}\ldots d_{\gamma(0)} = c_{k-1}c_{k-2}\ldots c_0$. Therefore, $c = f_\gamma(d)$ and $d = f_{\gamma^{-1}}(c)$. As $f_{\gamma^{-1}}$ is a permutation (a digit permutation) of S_N, it follows that for a fixed input terminal s there corresponds to every control tag c one and only one output terminal. Therefore, the network has the unique path property.

(b) The relation $c = f_\gamma(d)$ has just been proved in (b).

Part (a) of the previous theorem leads to a simple algorithm to determine if a sequence of $k+1$ digit permutations construct a digit permutation network that has the unique path property. The following procedure, called Compute-Control, takes as input a sequence of $k+1$ kernels $\pi_0, \pi_1, \ldots, \pi_k$ (simply represented by the array $\pi_{0..k}$) and computes the mapping γ such that $\gamma(i) = \beta_{k-i}^{-1}(0)$ as defined in the previous theorem. Another function, called Is-MIN, checks to see if γ is a permutation of $\{0,1,\ldots,k-1\}$ by doing a bucket sort on $\gamma(0), \gamma(1), \ldots, \gamma(k-1)$ and checking if any of the 'buckets' $0,1,\ldots,k-1$ is empty. If γ is a permutation, Is-MIN returns true, indicating that network DPN$(f_{\pi_0}, f_{\pi_1}, \ldots, f_{\pi_k})$ has the unique path property; otherwise, it returns false. Note that Compute-Control was separated from Is-MIN because it will be used later in the terminal relabelling algorithm for network simulation.

Procedure

```
Compute-Control (π0..k: input; γ: output)
begin
(1) β_k^-1 := π_k^-1;
(2) γ(0) := β_k^-1(0);
(3) for i = k − 1 to 1 step −1 do
(4)     β_i^-1 := π_i^-1 β_{i+1}^-1;
    (Comment: β_i = π_k π_{k-1} ... π_i = β_{i+1} π_i and hence
    β_i^-1 := π_i^-1 β_{i+1}^-1)
(5)     γ(k − i) := β_i^-1(0);
end
```

Function

```
Is-MIN (π0..k: input)
begin
(1) Compute-Control (π0..k, γ);
(2) for i = 0 to k − 1 do
(3)     b(i) := −1; (Comment: Initialize the buckets to
        empty)
(4) for i = 0 to k − 1 do
(5)     b(γ(i)) := γ(i);
(6) for i = 0 to k − 1 do
(7)     if b(i) = −1 then
(8)         return(false);
(9) return(true);
end
```

Listing 1. An algorithm to decide if DPN$(f_{\pi_0}, f_{\pi_1}, \ldots, f_{\pi_k})$ has the unique path property

Time complexity

As the inversion of a permutation and the composition of two permutations can be done in linear time, Compute-Control clearly takes $O(k^2)$ time, which is $O(\log_r^2 N)$. Similarly, Is-MIN takes $O(\log_r^2 N)$ time. Consequently, the time to check if $k+1$ digit permutations form a DPN that has the unique path property is $O(\log_r^2 N)$, which is optimal because the size of the input $\pi_{0..k}$ to be examined is $O(k^2)$.

Control of digit permutation networks

Part (b) of the last theorem shows that digit permutation networks are FD-controllable and that their control functions are digit permutations (f_γ) which can be easily derived from the constituent digit permutations.

One consequence of the fact that the control function is a digit permutation f_γ is the increased control efficiency. One way of controlling a DPN with control function f_γ is to design the switch so that the destination labels can be used to set the switches as follows: the switches in column i use digit $\gamma(k-i+1)$ of the destination label as control digit, for $i = 0, 1, \ldots, k-1$. Another way is to provide some additional hardware in the input terminals to permute the digits of the output terminals according to γ and produce the control tag. As k is relatively small in practice, the additional amount of hardware is small. A third way is to compute $f_\gamma(d)$ in software every time a path $s \rightarrow d$ is to be estasblished. This requires a small amount of memory to store γ (not f_γ) at every input terminal. This software control is cheaper than the previous two ways but is a little slower

As will be shown later, all DPNs are widely equivalent. Therefore, any DPN can simulate any other DPN. In the simulation, the terminals of the simulating network are relabelled and the control function has to change to that of the simulated network. This is not feasible if the control is implemented in hardware as explained above. However, if the control is implemented in software, the simulation can be carried out easily by importing the procedure of the control function from the network to be simulated. This makes software control preferable.

Equivalence of digit permutation networks

Since $(f_\pi)^{-1}$ is $f_{\pi^{-1}}$, it follows that the inverse of DPN$(f_{\pi_0}, f_{\pi_1}, \ldots, f_{\pi_k})$ is the digit permutation network DPN$(f_{\pi_k^{-1}}, f_{\pi_{k-1}^{-1}}, \ldots, f_{\pi_1^{-1}}, f_{\pi_0^{-1}})$. Since also every digit permutation network is FD-controllable, we conclude that every digit permutation network is doubly FD-controllable. Consequently, every digit permutation network in MIN(r,k) is widely equivalent to the baseline network $B(r,k)$, after Theorem 4. This result is a generalization of the functional equivalence of the existing networks[3].

This equivalence reveals the structure of digit permutation networks and shows that the baseline network can simulate (i.e., become strictly functionally equivalent to) any digit permutation network by relabelling the terminals of the baseline network.

Simulation among digit permutation networks

If two networks in MIN(r,k) are widely equivalent, then the terminals of one network can be relabelled so that it can realize the permutations realizable by the other network. The problem is to find the new labels of terminals.

We have shown in Lemma 5 that if a network W in MIN(r,k) is doubly FD-controllable with left-to-right control function f and right-to-left control function g, then W is strictly equivalent to $gB(r,k)f^{-1}$. Consequently, the baseline network can simulate W by relabelling the input terminals of $B(r,k)$ by g^{-1} (i.e., each input terminal i is relabelled $g^{-1}(i)$) and relabelling the output terminals by f^{-1}. Generally, if W' is another doubly FD-controllable network in MIN(r,k) with left-to-right control function f' and right-to-left-control function g', then W' is strictly equivalent to $g'B(r,k)f'^{-1}$, and consequently, W' is strictly equivalent to $g'g^{-1}Wff'^{-1}$. Therefore, W can simulate W' by relabelling the input terminals of W by gg'^{-1} and the output terminals of W by ff'^{-1}. Therefore, the problem of finding the new labels of terminals of one network to simulate another network reduces to finding the left-to-right and right-to-left control functions of both networks. As the control functions of digit permutation networks are digit permutations f_γs, it is enough to find the γs of the control functions.

The following algorithm takes as input two digit permutation networks W and W' represented by the sequences of their defining kernals $\pi_{0..k}$ and $\pi'_{0..k}$, respectively, and relabels the terminals of W to simulate W'. It calls the Compute-Control procedure to compute γ and τ such that f_γ and f_τ are the left-to-right and right-to-left control function of W, respectively. It also computes the corresponding γ' and τ' of W'. As the input terminals of W have to be relabelled with $f_\tau f_{\tau'^{-1}}$, which is equal to $f_{\tau'^{-1}\tau}$, the algorithm computes $\tau'^{-1}\tau$. Similarly, as the output terminals of W have to be relabelled with $f_\gamma f_{\gamma'^{-1}}$, which is equal to $f_{\gamma'^{-1}\gamma}$, the algorithm computes $\gamma'^{-1}\gamma$. Finally, the algorithm does the relabelling. Note that f_τ is the left-to-right control function of $W^{-1} = \mathrm{DPN}(f_{\pi_k^{-1}}, f_{\pi_{k-1}^{-1}}, \ldots, f_{\pi_1^{-1}})$. Therefore, τ is computed by the procedure Compute-Control with input $\pi^{-1}_{k..0}$. The same applies to τ'.

Procedure

Simulate (W,W')
begin
(1) Compute-Control ($\pi_{0..k},\gamma$); (Comment: In case $W = B(r,k)$, γ and τ = are the identity permutation and need not be computed)
(2) Compute-Control ($\pi^{-1}_{k..0},\tau$);
(3) Compute-Control ($\pi'_{0..k},\gamma'$);
(4) Compute-Control ($\pi'^{-1}_{k..0},\tau'$);
(5) Compute $\gamma'^{-1}\gamma$ and $\tau'^{-1}\tau$;
(6) for $i = 0$ to $N-1$ do
(7) relabel input terminal i of W by $f_{\tau'^{-1}\tau}(i)$;
(8) relabel input terminal i of W by $f_{\gamma'^{-1}\gamma}(i)$;
end

Listing 2. A terminal relabelling algorithm for network simulation

Time complexity

Steps 1–4 take $O(k^2)$ each, as we saw before. Step 5 takes $O(k)$ as permutation inversion and composition take linear time. Steps 7 and 8 take $O(k)$ each as the relabelling of a node consists of permutating its k digits. Therefore, step 6 takes $O(Nk)$, which is the dominant term. It follows that the time the algorithm takes is $O(Nk)$, which is $O(N \log_r N)$.

Parallel relabelling

Step 6 can be done in parallel by making each terminal i compute its own label. This can be done by broadcasting $\gamma'^{-1}\gamma$ and $\tau'^{-1}\tau$ to all the terminals after step 5, and then requiring each terminal i to permute the digits of its label according to $\tau'^{-1}\tau$ if the terminal is an input, and according to $\gamma'^{-1}\gamma$ if the terminal is an output. This makes step 6 take the same time $O(k^2)$ as steps 7 and 8 which are computed at the individual terminals. Thus, the overall parallel time for the algorithm is $O(k^2)$, that is, $O(\log_r^2 N)$. It should be noted that in a shared memory system the input terminals are processors and the output terminals are memory modules with no processing power. In this case, the label of the output terminal i is computed by the input terminal i and then sent to the output terminal i. The parallel complexity remains $O(\log_r^2 N)$.

CONCLUSIONS

This paper has introduced and examined several control classes of the Banyan multistage networks, namely, D-control, FD-control, double D-control and double FD-control. The first two classes are efficiently controllable since the control tags needed to establish source-destination paths are destination addresses or a function thereof. The last two classes are useful for two-way communication in processor–memory networks. The structure of D- and FD-controllable networks was shown to be recursive, and that of doubly D- or FD-controllable networks was shown to be strictly or widely equivalent to the baseline network. Also the class of MINs whose interconnections are digit permutations, which includes all existing MINs, was found to be doubly FD-controllable and hence widely equivalent to the baseline. An immediate consequence of this equivalence is that the doubly controllable networks, including the digit permutation networks, have fundamentally the same functionality as the baseline, for even in the case of wide equivalence, the baseline can simulate any of these networks by appropriately relabelling the terminals of the baseline. An efficient algorithm for such relabelling was given, and the control of the relabelled baseline was shown to remain efficient, especially when the simulated network is a digit permutation network.

Future work includes the examination of the structure and functionality of MINs whose control tags are easy-to-compute functions of both source and destination tags. The purpose is to find efficiently controllable MINs that may be fundamentally different from the existing ones

while capable of realizing a larger set of *useful* permutations.

REFERENCES

1 **Lawrie, D K** 'Access and alignment of data in an arrary processor' *IEEE Trans. Comput.* Vol C-24 (December 1975) pp 1145–1155

2 **Pease, M C** 'The indirect binary n-cube multiprocessor array' *IEEE Trans. Comput.* Vol C-26 (May 1976) pp 458–473

3 **Wu, C L and Feng, T Y** 'On a class of multistage interconnection networks' *IEEE Trans. Comput.* Vol C-29 No 8 (August 1980) pp 694–702

4 **Siegel, H J** 'A model of SIMD machines and a comparison of various interconnection networks' *IEEE Trans. Comput.* Vol C-28 No 12 (December 1979) pp 907–917

5 **Agrawal, D P, Kim, S-C and Swain, N K** 'Analysis and design of nonequivalent networks' *IEEE Trans. Comput.* Vol 37 (February 1988) pp 233–237

6 **Oruc, A Y and Oruc, M Y** 'On testing isomorphism of permutation networks' *IEEE Trans. Comput.* Vol C-34 (October 1985) pp 958–962

7 **Siegel, H J and Smith, S** 'Study of multistage interconnection networks' *Proc. Fifth Annual Symp. Comp. Arch.* (April 1978) pp 223–229

8 **Youssef, A** *Properties of Multistage Interconnection Networks* PhD dissertation, Princeton University (February 1988)

9 **Youssef, A and Arden, B** 'Equivalence between functionality and topology of log N-stage Banyan networks' *IEEE Trans. Comput.* Vol 39 No 6 (June 1990) pp 829–832

10 **Feng, T** 'A survey of interconnection networks' *Computer* Vol 14 (December 1981) pp 12–27

Abdou Youssef was born on January 12, 1960. He received a BS degree in mathematics from The Lebanese University, Lebanon, in 1981, and MA and PhD degrees from Princeton University, Princeton, NJ, in 1985 and 1988, respectively. He taught for a year in 1982 at the Institute of Applied Sciences, The Lebanese University. He has been an Assistant Professor in the Department of Electrical Engineering and Computer Science at The George Washington University, Washington DC, since 1987. His research interests include interconnection networks and computer architecture, parallel processing, algorithms, and fault tolerance.

Bruce Arden received a BS (EE) degree with highest distinction from Purdue University in 1949. After service in the Navy, he was employed by the Allison Division of General Motors to numerically solve jet engine design problems on the early digital computers. Subsequently, he returned to academia at the University of Michigan and received a PhD (EE) in 1965. He joined the faculty at Michigan and rose to Professor (and Chairman) in the Computer and Communication Sciences Department in 1970. In 1973 he accepted the position of Professor (and Chairman) of Electrical Engineering and Computer Science at Princeton University. He became Princeton's Arthur LeGrand Doty Professor of Engineering in 1984. In 1986 he moved to the University of Rochester as Professor of Electrical Engineering and Computer Science and Dean of the College of Engineering and Applied Science. At these institutions, Professor Arden continued to publish, individually and with students, in the areas of compiling algorithms, operating systems, computer architecture, and interconnection strategies.