A STUDY OF PERMUTATION NETWORKS: SOME GENERALIZATIONS AND TRADEOFFS*

A. Yavuz Oruç
Electrical Engineering Department
and
Institute for Advanced Computer Studies
University of Maryland,
College Park, MD 20740

ABSTRACT

Permutation switching is a critical element of many computer and communication systems. Within a group theoretical framework, this paper provides an indepth study of permutation networks, and examines the tradeoffs between network cost and set up or routing time. It introduces the notion of an (n, r, q)-permuter, i.e., a permuter with n inputs and r outputs that can realize all $q!$ permutations between any q of its n inputs and q of its r outputs, where $q \leq r \leq n$. This generalization accounts for a switching environment where the maximum number of simultaneous paths may be less than the actual number of inputs and outputs. It is shown that the previously known designs, such as Clos networks result in inferior realizations of (n, r, q)-permuters. Using concentrators, the paper gives new network designs that lead to (n, r, q)-permuters with asymptotically minimum cost and quadlogarithmic routing time for all $q \leq r$. More specifically, for $q = O(\lg n)$ and $q = O(n^{\epsilon})$, where $0 < \epsilon < 1$, an (n, r, q)-permuter with $O(n)$ switches is given[1]. For the same values of q, Clos designs require at least $n \lg \lg n$ and $n \lg n$ switches. Another advantage of the new designs is that they do not require complex routing schemes as Clos networks since they are inherently self-routing. It is also established that, when $q = n = r$, these same designs can be extended to permuters with $O(n \lg n)$ switches.

Index Terms: *Clos network, concentrator, coset double coset, expander, permuter, permutation network, superconcentrator, superpermuter, symmetric group.*

*This work is supported in part by the National Science Foundation under Grant No: CCR-8708864.

[1]All logarithms in this paper are in base 2, and $\lg n$ denotes $\log_2 n$.

1 Introduction

In many computer and communication systems, permutation maps are used to characterize the interconnections between various components of the system [8,14]. In this paper, we undertake a comprehensive investigation of the ways in which permutation maps can be realized in terms of a particular kind of a connector[2] that is commonly referred to as a *permutation network* in the literature. Here, we will call it a *permuter* for notational convenience, and consider three types of permuters. The first one, to be called an n-permuter, is a connector with n inputs and n outputs and can permute its inputs onto its outputs in all $n!$ ways. The second type of permuter we consider is a connector with n inputs and r outputs, where $r \leq n$, and can realize all $r!$ permutations between any r-subset of its n inputs and its r outputs. We will refer to this permuter as an (n, r)-permuter. Finally, the third type of permuter we consider is a connector with n inputs and r outputs, called an (n, r, q)-permuter, which can realize, for some positive integer $q \leq r \leq n$, all $q!$ permutations between any q-subset of its n inputs and any q-subset of its r outputs. The parameter q is called the *bandwidth* of the permuter.

All previously known permuters are n-permuters, and with the exception of cellular permutation arrays [12], they all are based on the well-known 3-stage network of Clos [2,7]. It is also possible to obtain n-permuters by using shuffle-exchange connectors [19,24,27], but these connectors can essentially be derived from Clos designs. The main motivation for providing an extension to n-permuters stems from a possibility that in certain computations, it may be sufficient to have a smaller bandwidth than n. For example, one can envision a matrix algebra computer with n^2 memory elements and n processors where the processors may be limited to access rows, columns, diagonals and $\sqrt{n} \times \sqrt{n}$ submatrices. In a more flexible setting, one may increase the number of processors to n^2, but still limit the size of access to n. Obviously, these two parallel organizations can be modeled in terms of (n^2, n)-permuters and (n^2, n^2, n)-permuters.

Accordingly, it is of both theoretical as well as practical interest to determine if and how the reduction in bandwidth impacts the cost of designing permuters. A major result of this paper is that for $q = O(n^\epsilon)$, where $0 < \epsilon < 1$, one can design (n, r, q)-

[2]The definition of a connector will be formally given in Section 2. For now, the reader may think of it as a network of crossbar switches.

permuters with $O(n)$ cost, and $O(\lg^2 n)$ set-up, or routing time. As a comparison, for the same value of q, Clos designs require $O(n \lg n)$ cost, and $O(\lg^4 n / \lg \lg n)$ routing time. A similar result is also given for (n, r)-permuters. That is, an (n, r)-permuter with $O(n + r \lg r)$ cost and $O(\lg n + \lg^2 r)$ routing time is presented. An extension of Clos network in this case requires $O((n+r) \lg r)$ cost and $O(\lg^4 n / \lg \lg n)$ routing time.

Unlike 3-stage Clos connectors, most of our permuter designs make use of concentrators and superconcentrators [1,21,10] as their building blocks. The need for such connectors in designing permuters arises rather naturally when we consider the problem in a group theoretical framework. In fact, the application of group theory to connector design is not new. Let Σ_n denote the set of all permutations over a set of n elements, i.e., the symmetric group of degree n. In [3], Benes gave a proof of rearrangeability for a 3-stage Clos connector by using the fact that any permutation group containing all transpositions, i.e., pairwise exchanges among n elements is Σ_n. In another paper [4], he showed that a 3-stage Clos connector with square switches can be algebraically represented by a product of two double cosets of certain impritimive subgroups of Σ_n. In a subsequent paper [5], Benes used this fact to obtain some new rearrangeable connectors. More recently, it was established in [16,17] that every right (or left) coset decomposition of Σ_n induces a 2-stage n-permuter. Unlike the product expression used in [3], this latter decomposition is based on the fact that any finite group can be expressed as a union of pairwise disjoint cosets of any of its subgroups.

In this paper, we further develop these ideas by considering double coset decompositions of Σ_n rather than its right and left coset decompositions. The use of double cosets was advocated earlier in a conference paper [18]. The main contrast between the connectors that were derived there, and most of those given here is the use of linear cost self-routing concentrators and superconcentrators recently constructed in [9,10]. Linear cost concentrators and superconcentrators have long been known, but the lack of fast schemes to route them prevented their use to obtain efficient permuter constructions.

Table 1 lists all nine permuter designs described in the paper. They are identified by their numbers, and types along with their costs, depths, routing costs, and routing times. Here, *cost* represents the number of constant fan-in switches and *depth* refers to the largest number of such switches on a path from an input to an output of a connector. *Routing cost* accounts for any additional circuitry, or hardware, also in

Type/Name	Design	Cost	Depth	Routing Cost	Routing Time
$n, G, 1$	3-stage	$O(n^{1.58})$	$O(n)$	$O(n^{1.58})$	$O(\lg^2 n)$
$n, GS, 2$	3-stage	$O(n \lg n)$	$O(\lg^2 n)$	$O(n \lg n)$	$O(\lg^2 n)$
$n, G, 3$	3-stage	$O(n^{1+2\lg^{-\frac{1}{2}} n})$	$O(n^{\lg^{-\frac{1}{2}} n})$	$O(n^{1+2\lg^{-\frac{1}{2}} n})$	$O(\lg^{2.5} n + n^{1/\lg^{\frac{1}{2}} n})$
$n, G, 4$	3-stage	$O(n \lg n)$	$O(\lg n)$	$O(n \lg n)$	$O(\lg^4 n / \lg\lg n)$
$n, S, 6$	2-stage	$O(n \lg n)$	$O(\lg^2 n)$	0	$O(\lg^2 n)$
$(n,r), G, 5$	3-stage	$O((n+r)\lg r)$	$O(\lg r + \lg n/r)$	$O(n \lg n)$	$O(\lg^4 n / \lg\lg n)$
$(n,r), S, 7$	2-stage	$O(n + r \lg r)$	$O(\lg n + \lg^2 r)$	0	$O(\lg n + \lg^2 r)$
$(n,r,q), S, 8$	3-stage	$O((n+r)\lg q)$	$O(\lg nr/q + \lg^2 q)$	0	$O(\lg nr/q + \lg^2 q)$
$(n,r,q), S, 9$	3-stage	$O(n + r + q \lg q)$	$O(\lg nrq^2)$	0	$O(\lg nrq^2)$

Table 1: Various permuters described in the paper.

terms of constant size switches, over and beyond network's own cost needed to compute the switch settings, and the *routing time* specifies the time it takes the routing hardware to compute the switch settings in terms of instruction steps (path delays) of constant fan-in processing elements (logic circuits). The word *design* refers to a particular arrangement of switches, and these arrangements are broadly classified as 2-stage and 3-stage designs, indicating that a connector is a cascade of either two or three stages of smaller size connectors prior to any decomposition. Another distinction we make among the nine designs is their routing schemes. Some permuter designs use global routing as indicated by the letter G, some use self-routing as indicated by S, and one uses a combination of the two schemes as indicated by GS. The first two permuters are derived from double coset decompositions of Σ_n using a pair of symmetric subgroups of Σ_n while the next three are derived from double coset decompositions of Σ_n using products of symmetric subgroups of Σ_n. The last of these is self-routing. The next two designs are (n, r)-permuters, the first one with global routing, and the second with a self-routing scheme. The last two designs are (n, r, q)-permuters and are both self-routing.

The remainder of the paper is organized as follows. The next section summarizes the basic notions that are used throughout the paper. Section 3 describes the double coset decomposition that yields the first two connectors. Section 4 computes the cost and depth of these two connectors when they are recursively decomposed. Section 5 describes the next two permuters while Section 6 describes Permuter 5. Section 7 presents the permuter designs using concentrators. Section 8 presents the (n, r, q)-

permuter designs, and the paper is concluded in Section 9.

2 Basic Concepts

This section gives definitions of various connectors that are used throughout the paper. It also briefly mentions the basic relevant algebraic facts. The reader may refer to Appendix B for a glossary of notations used in the paper.

Definition 1:

(i) An elementary (n, r)-connector, or (n, r)-switch is a complete bipartite graph with n distinguished vertices, called *inputs*, and r distinguished vertices, called *outputs*. In switching terms, an (n, r)-switch is a crossbar with n inputs and r outputs, and nr crosspoints.

(ii) An (n, r)-connector where $r \leq n$, is a directed acyclic multigraph with n distinguished source vertices, called *inputs*, r distinguished sink vertices, called *outputs*, and each of whose remaining vertices is either a switch, or connector itself. ||

Definition 2: Let q be a positive integer $\leq r$. An (n, r)-connector is called an (n, r, q)-permuter if it can realize all $q!$ permutations between any q of its inputs any q of its outputs. When $q = r$, an (n, r, q)-permuter is called an (n, r)-permuter, and when $q = n = r$, it is called an n-permuter. ||

Definition 3: An (n, r)-connector is called an (n, r)-concentrator if, for $i = 1, 2, \ldots, r$, it can connect any i of its inputs to some i of its outputs. It is called an (n, r)-superconcentrator, if it can connect any i of its inputs to any i of its outputs. ||

Let G be a permutation group acting on a set $N = \{1, 2, \ldots, n\}$, i.e., each element of G defines a permutation from N onto itself. For any two permutations $g_1, g_2 \in G$, define g_2g_1 as the permutation obtained by composing g_1 with g_2, i.e., for each $i \in N$, let $g_2g_1(i) = g_2(g_1(i))$. Let H_1 and H_2 be any two subgroups of G. A *double coset* of the ordered pair of groups (H_1, H_2) in G is defined as the set of elements $H_2gH_1 = \{h_2gh_1 : h_1 \in H_1, h_2 \in H_2\}$ where g is an arbitrary but fixed element in G. An important property that double cosets share with ordinary cosets is that any two double cosets are either identical or disjoint. That is, we can write

$$G = H_2g_1H_1 \cup H_2g_2H_1 \cup \cdots \cup H_2g_rH_1 \tag{1}$$

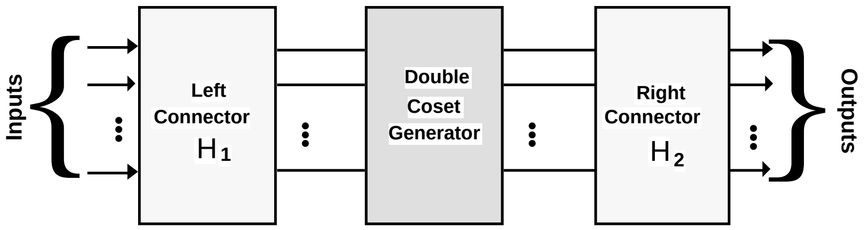

Figure 1: The double coset representation of a connector.

where any two double cosets in the expression are disjoint. Thus, the decomposition is unique for any ordered pair of subgroups (H_1, H_2) except for the choice of permutations $g_i, 1 \leq i \leq r$, that are the *coset leaders* of the decomposition. In fact, since $H_2 g_i H_1 = H_2 g_i' H_1$ for any $g_i' \in H_2 g_i H_1$, any permutation in $H_2 g_i H_1$ can serve as a coset leader for $H_2 g_i H_1$.

As Figure 1 illustrates, the double coset decompositions of permutation groups are algebraic analogs of connectors consisting of three stages. The first and third stages realize the subgroups H_1 and H_2 respectively, and the center stage, that will be called a *double coset generator*, realizes all the double coset leaders in the above decomposition thereby making the entire connector realize G. In the remainder of the paper, we will be concerned with connectors that realize the symmetric group of permutations so that G is replaced by Σ_n in (1). However, nothing special is assumed about the subgroups H_1 and H_2, and to each choice of H_1 and H_2, there corresponds a 3-stage connector that realizes Σ_n.

3 Permuters Based on a Single Symmetric Subgroup

In this section, we present two n-permuters both of which result from a double coset decomposition of Σ_n over a single symmetric subgroup. Accordingly, for a given m-subset M of N, we let $H_1 = H_2 = \Sigma_M$, where Σ_M is the symmetric subgroup of Σ_n consisting of all permutations which fix the elements in $N - M$. In connector terms, we are to use two copies of an m-permuter to obtain an n-permuter as shown in Figure 2. Since the design of an m-permuter is identical to the original problem, except for its size, it will suffice to consider the design of the double coset generator only. This will be accomplished by characterizing the double cosets of Σ_M in Σ_n.

We first note that any permutation $\pi \in \Sigma_n$ can expressed as

$$\pi = (\pi_{M,N}, \pi_{N-M,N-M}, \pi_{N-M,M}) \tag{2}$$

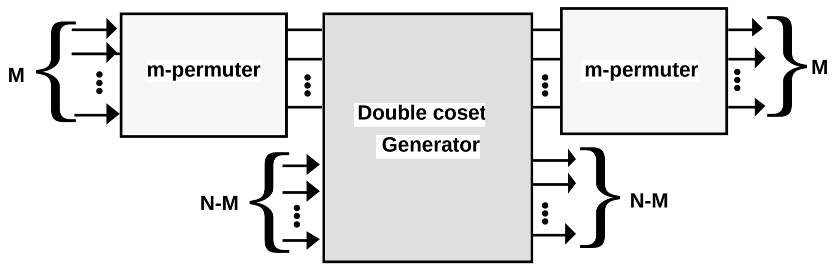

Figure 2: An n-permuter formed from two m-permuters.

where

(i) $\pi_{M,N}$ is a mapping from M into N that coincides with π when the domain of π is restricted to M, i.e., $\pi_{M,N}(i) = \pi(i)$, for all $i \in M$,

(ii) $\pi_{N-M,N-M}$ is a mapping from $N - M$ into $N - M$ that coincides with π when the domain of π is restricted to $N - M$ and its range is restricted to $N - M$, i.e., $\pi_{N-M,N-M}(i) = \pi(i)$, for all $i \in N - M$ for which $\pi(i) \in N - M$,

(iii) $\pi_{N-M,M}$ is a mapping from $N-M$ into M that coincides with π when the domain of π is restricted to $N-M$ and its range is restricted to M, i.e., $\pi_{N-M,M}(i) = \pi(i)$, for all $i \in N - M$ for which $\pi(i) \in M$.

To illustrate these maps, let $N = \{1, 2, 3, 4, 5, 6\}$, $M = \{1, 2, 3\}$, and

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 2 & 1 & 6 & 3 & 5 \end{pmatrix}$$

Then

$$\pi_{M,N} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 2 & 1 \end{pmatrix}, \ \pi_{N-M,N-M} = \begin{pmatrix} 4 & 6 \\ 6 & 5 \end{pmatrix}, \ \pi_{N-M,M} = \begin{pmatrix} 5 \\ 3 \end{pmatrix}.$$

It should be noted that these maps are not necessarily permutations. However, we can define composition maps between $\pi_{M,N}$, $\pi_{N-M,M}$ and each permutation $\pi_M \in \Sigma_M$ as $\pi_{M,N}\pi_M(i) = \pi_{M,N}(\pi_M(i))$ for all $i \in M$ and $\pi_M\pi_{N-M,M}(i) = \pi_M(\pi_{N-M,M}(i))$ for all $i \in N - M$. With these, we can then define, for any $\pi_M, \pi'_M \in \Sigma_M$, the product $\pi_M\pi\pi'_M$ as $(\pi_{M,N}\pi_M, \pi_{N-M,N-M}, \pi'_M\pi_{N-M,M})$. Accordingly, any permutation in the double coset $\Sigma_M\pi\Sigma_M$ can be represented as in (2), and conversely any permutation that can be represented as in (2) belongs to the double coset of π. Furthermore, the map $\pi_{N-M,N-M}$ remains invariant (i.e., the same) over all permutations in the double coset of π, and we have

Theorem 1: $\Sigma_M \pi \Sigma_M$ and $\Sigma_M \pi' \Sigma_M$ coincide in Σ_n if and only if $\pi_{N-M,N-M} = \pi'_{N-M,N-M}$.

Proof: The condition is obviously necessary by the preceding discussion. To prove that it is also sufficient, suppose $\pi_{N-M,N-M} = \pi'_{N-M,N-M}$. Then it must be that π and π' map the same elements from $N-M$ into $N-M$, and hence, they also map the same elements from $N-M$ to M. Thus, there exists a permutation $\pi_M \in \Sigma_M$ such that $\pi'_{N-M,M} = \pi_M \pi_{N-M,M}$. Furthermore, it is easy to see that, for any $\pi'_{M,N}$, there exists a $\pi_M \in \Sigma_M$ such that $\pi'_{M,N} = \pi_{M,N} \pi_M$. Thus, $\pi' = (\pi_{M,N}\pi_M, \pi_{N-M,N-M}, \pi_M \pi_{N-M,M})$, or $\pi' \in \Sigma_M \pi \Sigma_M$. Since any two double cosets are identical or disjoint, $\Sigma_M \pi \Sigma_M = \Sigma_M \pi' \Sigma_M$ and the statement follows. ||

The point of Theorem 1 is that the double cosets $\Sigma_M \pi \Sigma_M$ in Σ_n are in one-to-one correspondence with the bijections between the u-subsets of $N-M$ for all u such that $\max(0, n-2m) \leq u \leq n-m$. The lower bound on u follows from the fact that when $n-m \geq m$, each $\pi \in \Sigma_n$ must map at least $n-m-m = n-2m$ of the elements in $N-M$ into $N-M$; since π is one-to-one, it cannot map more than m elements from $N-M$ into M. The union of all such double cosets then gives Σ_n, that is,

Theorem 2: Let α be the number of distinct double cosets of Σ_M in Σ_n. The union of any α double cosets $\Sigma_M \pi_i \Sigma_M, i = 1, 2, \ldots, \alpha$ is Σ_n if and only if, to each bijection $\pi_{N-M,N-M}$ between any two u-subsets of $N-M$ where $\max(0, n-2m) \leq u \leq n-m$, there corresponds a $\pi_i \in \Sigma_n$ that coincides with $\pi_{N-M,N-M}$ when its domain and range are both restricted to $N-M$. ||

The number of distinct double cosets of Σ_M in Σ_n is, therefore, given by

$$\sum_{u=\max(0,n-2m)}^{n-m} \binom{n-m}{u} \binom{n-m}{u} u! \tag{3}$$

where $\binom{n-m}{u}$ is the number of u-subsets of $N-M$, and $u!$ is the number of bijections between any two such subsets[3].

Now, consider the design of a connector that realizes this decomposition. We are given two copies of an m-permuter, and we need to specify the structure of the coset generator that is placed between these two permuters as shown in Figure 2. Theorem 2

[3]For all $x \geq 0$, $\binom{x}{0}$ is taken to be 1. The $u = 0$ case in Equation (3) amounts to double cosets whose permutations map no element from $N-M$ into $N-M$, and this can only be done in one way. Of course, this happens only when $n-m \leq m$, since otherwise, i.e., if $n-m > m$, then $u \geq n-2m > 0$.

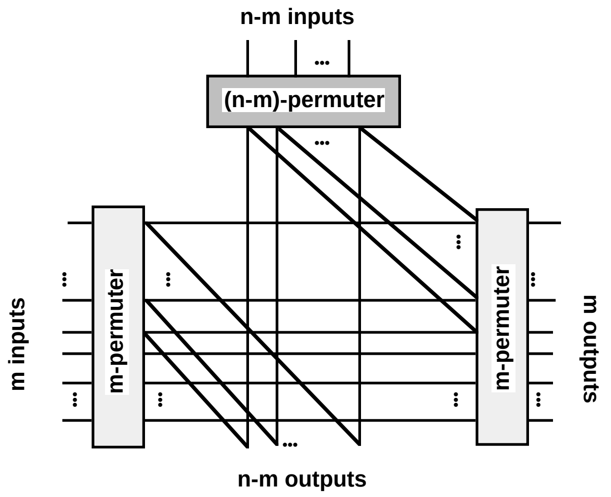

Figure 3: Permuter 1 ($m \geq n - m$).

implies that, when $u = n - m$, the coset generator must realize $(n-m)!$ permutations between its bottom $n - m$ inputs and bottom $n - m$ outputs. These represent the double cosets whose elements map all $n - m$ inputs in $N - M$ onto the $n - m$ outputs in $N - M$. An $(n - m)$-permuter suffices to realize all these $(n - m)!$ permutations as well as all bijections between any u-subset of the bottom inputs, any u-subset of the bottom outputs when $\max(0, n - 2m) \leq u \leq n - m$. Given this, the only problem that remains is to distribute the outputs of this permuter and the left m-permuter upon the inputs of the right m-permuter and the outputs in $N - M$. For this, we consider two cases.

Case 1: $m \geq n - m$. In this case, the coset generator is constructed as shown in Figure 3. Since $m \geq n - m$, no more than $n - m$ of the left inputs can be permuted onto the outputs at the bottom. Thus, connecting any $n - m$ outputs of the left m-permuter to the bottom $n - m$ outputs suffices to realize any one-to-one mapping between the left inputs and bottom outputs. These connections are arbitrarily specified between the topmost $n - m$ outputs of the left m-permuter and the bottom outputs, since the left m-permuter may be used to bring any $(n - m)$-subset of the left inputs upon any $n - m$ of its outputs. Likewise, since no more than $n - m$ inputs at the top can be permuted onto the right outputs, it suffices to connect the $n-m$ outputs of the $(n - m)$-permuter to any $n - m$ inputs of the right m-permuter. The outputs of the $(n-m)$-permuter are also connected to the bottom outputs to accomodate up to $n-m$ connections between the top inputs and bottom outputs. Finally, the m outputs of the left m-permuter are connected to the m inputs of the right permuter on a one-to-one basis. Any permutation that maps all inputs on the left onto all outputs on the right will need all the m edges provided. These m edges also suffice when fewer than m inputs are destined to outputs on the right. It, therefore, follows that

Theorem 3: The connector in Figure 3 realizes Σ_n for all $m \geq n - m$. $||$

Remark 1: This connector can be viewed as a simple extension of an n-input 3-stage Clos connector with one of the two $n/2$-input connectors deleted from its first stage as described in Waksman [26]. In fact, it becomes a Clos connector if we let $m = n - m = n/2$, and cluster the edges in the center into $(2, 2)$-switches. $||$

Case 2: $m \leq n-m$. In this case, the permuter in Figure 3 cannot be used since, if each of the m outputs of the left m-permuter is connected to a single output at the bottom, some bottom outputs will be left out. Consequently, any permutation that maps any inputs of the left m-permuter to those outputs cannot be realized by the connector. Likewise, the $(n-m)$-permuter at the top has more outputs than the right m-permuter has inputs, and so any one-to-one connection will leave some outputs of the $(n - m)$-permuter unconnected to the right m-permuter. This problem can be solved by using superconcentrators. As the following theorem establishes, using superconcentrators renders the right m-permuter vacuous and leads to a 2-stage permuter design as shown in Figure 4.

Theorem 4: The connector in Figure 4 realizes Σ_n for all $m \leq n - m$.
Proof: Consider an arbitrary, but fixed permutation $\pi \in \Sigma_n$. We must show that all three maps given in (2) can simultaneously be realized by this connector. Given the $(n-m)$-permuter at the top, it must be obvious that $\pi_{N-M,N-M}$ can be realized. Once $\pi_{N-M,N-M}$ is realized, then, using the unoccupied outputs of the $(n - m)$-permuter along with the $(n-m, m)$-superconcentrator on the right, $\pi_{N-M,M}$ can also be realized. Having realized these two maps, $\pi_{M,N}$ can be realized by considering it as comprising two submaps: (i) $\pi_{M,M}$, the map that coincides with $\pi_{M,N}$ when the range of $\pi_{M,N}$ is restricted to M, and (ii) $\pi_{M,N-M}$, the map that coincides with $\pi_{M,N}$ when the range of the $\pi_{M,N}$ is restricted to $N-M$. Given the m-permuter on the left, $\pi_{M,M}$ can be realized simply by connecting the inputs through that m-permuter to their respective outputs on the right. Finally, $\pi_{M,N-M}$ can be realized by (i) identifying its range of outputs at the bottom, (ii) concentrating these outputs via the $(n - m, m)$-superconcentrator to the unoccupied outputs of the m-permuter on the left, and then (iii) using that permuter to complete the map. $||$

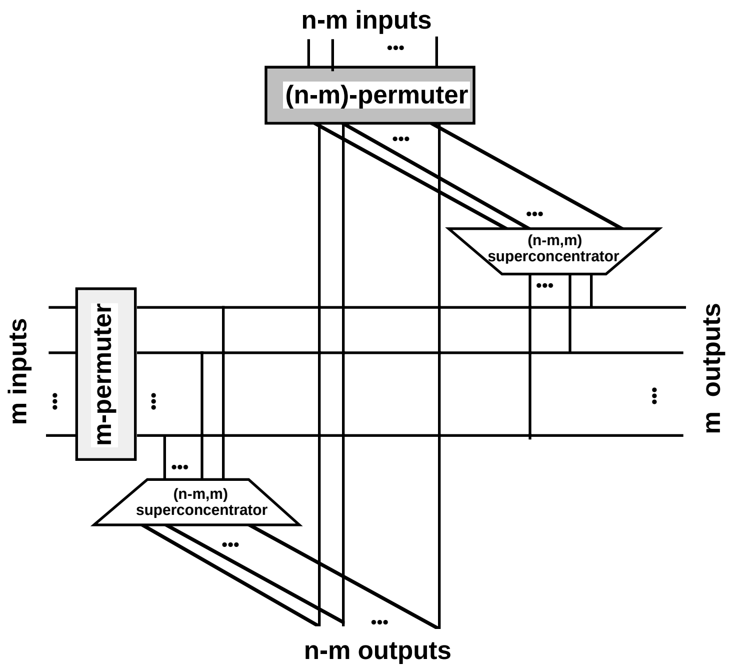

Figure 4: Permuter 2 ($m \leq n - m$).

4 Recursive Realizations

In this section, we consider the complexities of recursive decompositions of the two permuters described in the earlier section. We use four measures of complexity, one for *cost*, one for *depth*, one for *routing cost* and one for *routing time*. Cost will be tallied in terms of constant size switches, and depth will be taken as the largest number of such switches from an input to an output. Routing cost will account for any additional circuitry, or hardware, also in terms of constant size switches, over and beyond network's own cost needed to compute the switch settings, and the routing time will give the time it takes the routing hardware to compute the switch settings in terms of instruction steps (path delays) of constant fan-in processing elements (logic circuits).

It will be assumed that elementary switches are implemented in terms of binary multiplexers and demultiplexers, as shown in Figure 5. For n inputs and r outputs, it is easy to see that an (n, r)-elementary switch can be realized by attaching a binary tree of $r - 1$ nodes and $\lg r$ depth to each input, and a binary tree of $n - 1$ nodes of $\lg n$ depth to each output, and tieing the leaf nodes of the two sets of trees together. Accordingly, an elementary (n, r)-switch will have $\alpha_c(n(r - 1) + r(n - 1))$ cost and $\alpha_d(\lg n + \lg r) = \alpha_d \lg nr$ depth where α_c and α_d are some positive numbers indepen-

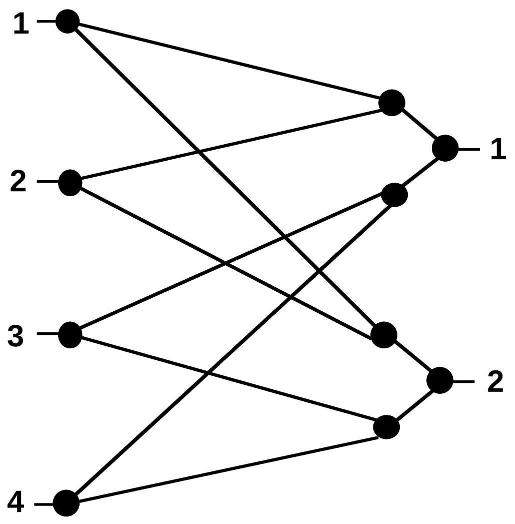

Figure 5: A realization of a $(4, 2)$-switch.

dent of n, and r.

With these assumptions, let $C_1(n)$ and $C_2(n)$ denote the costs of Permuter 1 and Permuter 2 and let $D_1(n)$, and $D_2(n)$ denote their depths, where n is the number of inputs. Let $C_{super}(n-m, m)$ and $D_{super}(n-m, m)$ denote the cost and depth of an $(n-m, m)$-superconcentrator respectively. From Figures 3, and 4, the cost and depth of the two connectors can then be expressed recursively as

$$\begin{aligned} C_1(n) &= 4\alpha_c(n-m) + C_1(n-m) + 2C_1(m);\ n/2 \le m < n & (4) \\ C_2(n) &= 2C_{super}(n-m, m) + C_2(n-m) + C_2(m);\ 2 \le m \le n/2 & (5) \\ D_1(n) &= 2\alpha_d + 2D_1(m);\ n/2 \le m < n & (6) \\ D_2(n) &= D_{super}(n-m, m) + D_2(n-m);\ 2 \le m \le n/2 & (7) \end{aligned}$$

where $C_1(2) = C_2(2) = 4\alpha_c$ and $D_1(2) = D_2(2) = 2\alpha_d$. The $4\alpha_c(n-m)$ term in the first recurrence accounts for the cost of multiplexer and demultiplexer devices switches in the coset generator, and $2\alpha_d$ in the third recurrence represents the depth of the coset generator in the first permuter. The cost of the connections in the coset generator of Permuter 2 excluding the superconcentrators is ignored.

It is shown in Appendix A that the minimum solution of (4) is obtained when $m = n/2$, in which case

$$C_1(n) = 4\alpha_c(n^{1.58} - n), \qquad (8)$$

and for the same value of m, the solution of (6) gives

$$D_1(n) = 2\alpha_d(n-1). \tag{9}$$

These values of cost and depth can be reduced to $O(n \lg n)$ and $O(\lg^2 n)$ by using the second permuter design. We first note that a self-routing $(n-m, m)$-superconcentrator can be constructed with $\beta_c(n-m)$ cost, and $\beta_d \lg(n-m)$ depth [9,10], where β_c and β_d are small positive contants[4]. Substituting these into (5) and (7) we obtain

$$\begin{aligned} C_2(n) &= 2\beta_c(n-m) + C_2(n-m) + C_2(m);\ 0 < m < n/2 & (10) \\ D_2(n) &= \beta_d \lg(n-m) + D_2(n-m);\ 2 \le m \le n/2. & (11) \end{aligned}$$

Letting $m = n/2$ in the above equations, and solving the resulting recurrences with $C_2(2) = 4\alpha_c$, and $D_2(2) = 2\alpha_d$ yields

$$\begin{aligned} C_2(n) &= \beta_c n \lg n - (\beta_c - 2\alpha_c)n & (12) \\ D_2(n) &= \frac{\beta_d}{2}(\lg^2 n - \lg n) + 2\alpha_d. & (13) \end{aligned}$$

Now we consider the routing cost and routing time of the two permuters when $m = n/2$. We adopt the routing algorithm given in [16]. First consider Permuter 1. To keep its overall cost to $O(n^{1.58})$, we shall fix the cost of the routing algorithm to $O(n^{1.58})$. This algorithm first determines, from any given permutation and its inverse, the settings for the multiplexer and demultiplexer devices in the coset generator, next it computes the submaps for the three $n/2$-permuters, and then recursively repeats the same steps over each permuter. The multiplexer and demultiplexer devices can all be set in $O(1)$ steps by checking the most significant bits of the preimages of the bottom half of outputs. The permutations of the $n/2$-permuters can be determined by this algorithm in $\alpha_t \lg n/2$ steps with $n/2$ simple processing elements, each of cost α_r, where α_r and α_t are positive numbers independent of n. Once these maps are determined all three $n/2$-permuters can then be routed in parallel. So, if $R_1(n)$, and $T_1(n)$ denote, respectively, the routing cost and routing time of Permuter 1 for n inputs, we have

$$\begin{aligned} R_1(n) &= \alpha_r n/2 + 3R_1(n/2) & (14) \\ T_1(n) &= \alpha_t \lg n/2 + T_1(n/2). & (15) \end{aligned}$$

[4]When the cost and depth of this superconcentrator are computed in terms of binary fan-in devices, β_c is about 10, and β_d is about 4.

The solutions of these recurrences with $R_1(2) = \alpha_r, T_1(2) = \alpha_t$ give

$$\begin{aligned} R_1(n) &= \alpha_r(n^{1.58} - n) & (16) \\ T_1(n) &= \frac{\alpha_t}{2}(\lg^2 n - \lg n) + \alpha_t. & (17) \end{aligned}$$

As for the second permuter, the routing scheme described for Permuter 1 can be used to determine the permutations of its two $n/2$ permuters in $\alpha_t \lg n/2$ time with $n/2$ processors, each of cost α_r, in much the same way. Furthermore, the superconcentrators in the coset generator can be self-routed as described in [9,10] in $\beta_t \lg n/2$ time, where β_t is a positive number independent of n. Hence, if $R_2(n)$ and $T_2(n)$ denote the routing cost and routing time of Permuter 2 for n inputs, we have

$$\begin{aligned} R_2(n) &= \alpha_r n/2 + 2R_2(n/2) & (18) \\ T_2(n) &= (\alpha_t + \beta_t)\lg n/2 + T_2(n/2). & (19) \end{aligned}$$

The solutions of these recurrences with $R_1(2) = \alpha_r, T_1(2) = \alpha_t$ give

$$\begin{aligned} R_2(n) &= \frac{\alpha_r}{2} n \lg n & (20) \\ T_2(n) &= \frac{\alpha_t + \beta_t}{2}(\lg^2 n - \lg n) + \alpha_t & (21) \end{aligned}$$

5 Clos Permuters

In this section we utilize 3-stage Clos permuter designs to obtain permuters with $O(n \lg n)$ cost, $O(\lg n)$ depth and $O(\lg^4 n/ \lg\lg n)$ routing time. We also derive from this same design a permuter with $O(n^{1+2\lg^{-\frac{1}{2}} n})$ cost, $O(n^{\lg^{-\frac{1}{2}} n \lg^{1/2} n})$ depth, and $O(\lg^{2.5} n + n^{\lg^{-\frac{1}{2}} n})$ routing time. Throughout the section, we will assume $n = mk$, where m and k are positive integers.

Let $M_i; 1 \le i \le k$, be any k m-sets such that $M_i \cap M_j = \Phi$ whenever $i \ne j$, and $M_1 \cup M_2 \cup \cdots \cup M_k = N$. Let Σ_{M_i} be the symmetric group over M_i. In this more general setting, we are given $2k$ copies of an m-permuter, and we are to combine them as shown in Figure 6 to realize Σ_n. Accordingly, the double cosets of interest are of the form

$$\Sigma_{M_1}\Sigma_{M_2}\ldots\Sigma_{M_k}\,\pi\,\Sigma_{M_1}\Sigma_{M_2}\ldots\Sigma_{M_k} \qquad (22)$$

where $\pi \in \Sigma_n$. To characterize these double cosets, let us refine the representation of π given in Section 3 as $(\pi_{M_1,M_1}, \pi_{M_1,M_2}, \ldots, \pi_{M_k,M_{k-1}}, \pi_{M_k,M_k})$, where, for each pair

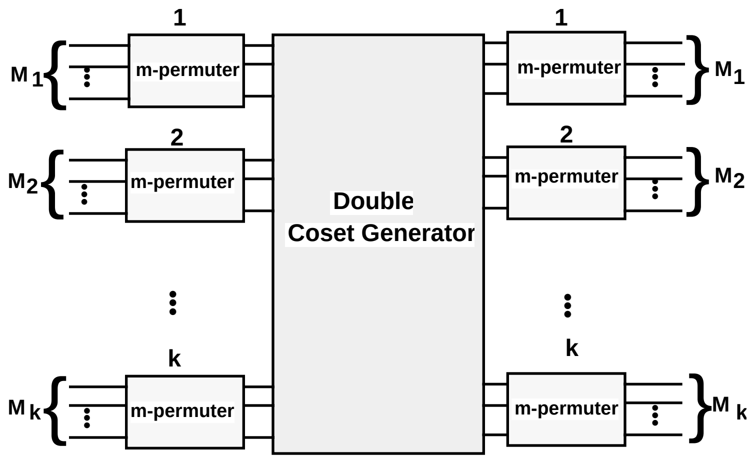

Figure 6: An n-permuter formed from $2k$ m-permuters, $k = n/m$.

$x, y, 1 \leq x, y \leq k$, π_{M_x,M_y} is a mapping from M_x into M_y that coincides with π when the domain and range of π are restricted to M_x and M_y respectively. It can be verified that a permutation in Σ_n belongs to the double coset of π if and only if it can be expressed this way. The following then becomes an analog of Theorem 1 for this more generalized double coset decomposition.

Theorem 5: $\Sigma_{M_1}\Sigma_{M_2}\ldots\Sigma_{M_k}\pi\Sigma_{M_1}\Sigma_{M_2}\ldots\Sigma_{M_k} = \Sigma_{M_1}\Sigma_{M_2}\ldots\Sigma_{M_k}\pi'\Sigma_{M_1}\Sigma_{M_2}\ldots\Sigma_{M_k}$ if and only if π_{M_x,M_y} and π'_{M_x,M_y} map an equal number of elements between M_x and M_y, for all $x, y; 1 \leq x, y \leq k$.

Proof: The proof is a straightforward extension of the proof of Theorem 1, and is omitted. $||$

A direct corollary of this fact is that the outputs of each permuter in the first stage in Figure 6 must be connected to all the permuters in the third stage. If these connections are to be of depth 1, then the design in Figure 7 results. Furthermore, these connections are also sufficient as the following statements show.

Let $R^{\pi}_{M_i,M_j}$ be the range of π_{M_i,M_j}, i.e., the subset of outputs in M_j to which π_{M_i,M_j} maps inputs from M_i. Let $K = \{1, 2, \ldots, k\}$, and $R^{\pi}_{i,K} = \{j \in K : |R^{\pi}_{M_i,M_j}| \geq 1\}$. In words, $R_{i,K}$ is the set of all m-permuters in the third stage to an output of which π maps an input from the ith m-permuter in the first stage.

Lemma 1: For any $\pi \in \Sigma_n$, there exist k inputs $x_i \in M_i, 1 \leq i \leq k$ and outputs $\pi_{M_i,M_{j_i}}(x_i), 1 \leq i \leq k$ such that $j_i, 1 \leq i \leq k$ are all distinct.

Proof: The proof relies on Hall's Theorem on distinct representatives (see [11]), and is similar to that given in ([5],p. 425). Consider any $\pi \in \Sigma_n$, and any α of the sets of inputs, $M_1, M_2, \ldots, M_k$, say $M_{i_1}, M_{i_2}, \ldots, M_{i_\alpha}$, $1 \leq \alpha \leq k$. Since π is one-to-one it must map the $m\alpha$ inputs in $M_{i_1} \cup M_{i_2} \cup \ldots \cup M_{i_\alpha}$ to some αm outputs. These outputs belong to those m-permuters in the third stage that are in $R_{i_1,K} \cup R_{i_2,K} \cup \ldots \cup R_{i_\alpha,K}$. Furthermore, since each m-permuter in the third stage has m outputs, we must have $|R_{i_1,K} \cup R_{i_2,K} \cup \ldots \cup R_{i_\alpha,K}| \geq m\alpha/m = \alpha$. Since this is true for any $\alpha, 1 \leq \alpha \leq k$, Hall's theorem implies that there exist k distinct sets of outputs $M_{j_i}, j_i \in R_{i,K}, 1 \leq i \leq k$, such that π maps an input from M_i to an output in M_{j_i}. That is, there exist k pairs of inputs and outputs $(x_i, \pi_{M_i,M_{j_i}}(x_i))$, where $j_i, 1 \leq i \leq k$ are all distinct. $||$

Theorem 6: For all n, m, k subject to $n = mk$, the connector in Figure 7 realizes Σ_n.
Proof: By the above lemma, for any $\pi \in \Sigma_n$ there exist k pairs of inputs and outputs $(x_1, \pi_{M_1,M_{j_1}}(x_1)), (x_2, \pi_{M_2,M_{j_2}}(x_2)), \ldots, (x_k, \pi_{M_k,M_{j_k}}(x_k))$, where $j_1, j_2, \ldots, j_k$ are all distinct. Furthermore, when $x_i, 1 \leq i \leq k$, are removed from M_i, and $\pi_{M_i,M_{j_i}}(x_i)$ are removed from $M_{j_i}, 1 \leq i \leq k$, the lemma still holds, except for the fact that each m-permuter in the first stage now contains $m-1$ inputs, and each m-permuter in the third stage contains $m-1$ outputs. Thus, Lemma 1 can be applied m times resulting in m copies of $(x_1, \pi_{M_1,M_{j_1}}(x_1)), (x_2, \pi_{M_2,M_{j_2}}(x_2)), \ldots, (x_k, \pi_{M_k,M_{j_k}}(x_k))$. Each of these represents a submap that assigns exactly one input from each m-permuter in the first stage to an output of a distinct m-permuter in the third stage. Each submap can, therefore, be realized through a complete bipartite graph in the center stage. Since the center stage contains m nonoverlapping such graphs, all m submaps can then be realized by the connector and hence the statement. $||$

Let $C_3(n)$ and $D_3(n)$ denote the cost and depth of this permuter, respectively. A close examination of the center stage reveals that it encompasses n/k (k,k)-elementary switches. So, assuming that each elementary (k,k)-switch is realized in terms of constant fan-in devices, the center stage has $(n/k)\alpha_c(2k(k-1)) = 2\alpha_c n(k-1)$ cost and $\alpha_d \lg k^2 = 2\alpha_d \lg k$ depth. Combining these, with the recursive construction of Permuter 3, we have

$$C_3(n) = 2\alpha_c n(k-1) + 2kC_3(n/k), 2 \leq k \leq n \tag{23}$$
$$D_3(n) = 2\alpha_d \lg k + 2D_3(n/k), 2 \leq k \leq n. \tag{24}$$

The solutions of these recurrences with $C_3(2) = 4\alpha_c$, $D_3(2) = 2\alpha_d$, yield

$$C_3(n) = \alpha_c(2(k-1)+4)(n/2)^{1+\frac{1}{\lg k}} - 2\alpha_c n(k-1), \tag{25}$$
$$D_3(n) = 2\alpha_d(1+\lg k)(n/2)^{\frac{1}{\lg k}} - 2\alpha_d \lg k. \tag{26}$$

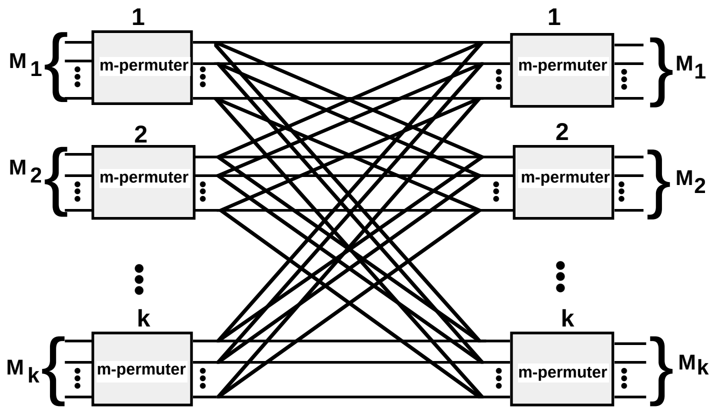

Figure 7: Permuter 3, $k = n/m$.

It can be shown that the minimum asymptotic value of $C_3(n)$ occurs at $k - 1 \approx (n/2)^{\lg^{-\frac{1}{2}}(n/2)}$ and is given by

$$C_3(n) \;=\; 2\alpha_c (n/2)^{1+2\lg^{-\frac{1}{2}}(n/2)} \tag{27}$$

and for the same value of k,

$$D_3(n) \;=\; 2\alpha_d \lg^{\frac{1}{2}}(n/2)\left((n/2)^{\lg^{-\frac{1}{2}}(n/2)} - 1\right) + 2\alpha_d (n/2)^{\lg^{-\frac{1}{2}}(n/2)}. \tag{28}$$

A variant of this permuter is the well-known Clos connector [2] that is obtained simply by replacing each of the m complete bipartite graphs in the center stage of the connector in Figure 7 by an n/m-permuter as shown in Figure 8. Since an n/m-permuter can realize each of the submaps described in the proof above, as a direct corollary of Lemma 1, we have

Theorem 7: (Slepian-Duguid, Benes [2]) For all n, m where m evenly divides n, the connector in Figure 8 realizes Σ_n. $||$

Let $C_4(n)$ and $D_4(n)$ denote the cost and depth of this permuter, respectively, where $m = n/k$ as before. Then an inspection of Figure 8 reveals that

$$C_4(n) \;=\; \frac{2n}{m} C_4(m) + mC_4(n/m), 1 \le m \le n/2 \tag{29}$$
$$D_4(n) \;=\; 2D_4(m) + D_4(n/m), 1 \le m \le n/2 \tag{30}$$

In solving these recurrences, it will be assumed that each m-permuter in the outer stages is realized as an elementary (m, m)-switch so that $C_4(m) = 2\alpha_c m(m - 1)$ and $D_4(m) = \alpha_d \lg m^2 = 2\alpha_d \lg m$. We note that other cost and depth assumptions can also be used, for example, each (m, m)-switch can be modeled as an m-input Beneš network

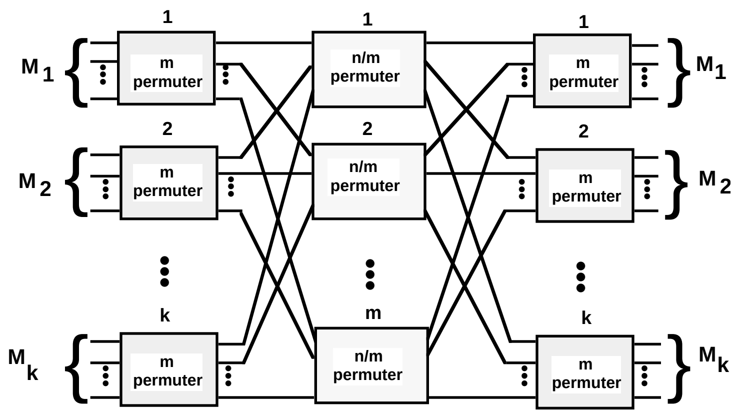

Figure 8: Permuter 4 (Clos), $k = n/m$.

with $(2, 2)$ switches. Nonetheless, it will be seen momentarily that these assumptions are sufficient to obtain an asymptotically minimum cost permuter.

With these assumptions, the above recurrences become

$$C_4(n) = 4\alpha_c n(m-1) + mC_4(n/m), 1 \leq m \leq n/2 \tag{31}$$

$$D_4(n) = 4\alpha_d \lg m + D_4(n/m), 1 \leq m \leq n/2 \tag{32}$$

whose solutions with $C_4(2) = 4\alpha_c$, and $D_4(2) = 2\alpha_d$ yield

$$C_4(n) = 4\alpha_c \frac{n(m-1)(-1+\lg n)}{\lg m} + 2\alpha_c n \tag{33}$$

$$D_4(n) = 4\alpha_d \lg n - 2\alpha_d \tag{34}$$

In particular, when $m = 2$, $C_4(n)$ becomes

$$C_4(n) = 4\alpha_c n \lg n - 2\alpha_c n \tag{35}$$

As for determining the routing cost and routing time of these permuters, we adopt the routing algorithms in [15,22]. For Permuter 3, Algorithm 5 in [22] can be used to determine the settings for the coset generator section on a completely connected n-processor computer, with each processor charged a cost of α_r, in $\alpha_t(\lg^2 n)\lg n/k$ time, where α_r, α_t are positive constants independent of n and k. Thus, if $R_3(n)$ and $T_3(n)$ denote the routing cost and routing time of Permuter 3 under this routing algorithm, we have from the construction of Permuter 3,

$$R_3(n) = \alpha_r n + 2kR_2(n/k), 2 \leq k \leq n/2 \tag{36}$$

$$T_3(n) = \alpha_t(\lg^2 n)\lg n/k + 2T_2(n/k), 2 \leq k \leq n/2. \tag{37}$$

It can be shown that the iteration of the first formula over n until it becomes 2, and with $R_3(2) = \alpha_r$ yield

$$R_3(n) \;=\; 2\alpha_r(n/2)^{1+\frac{1}{\lg k}} - \alpha_r n \tag{38}$$

As for $T_3(n)$, we shall first simplify the recurrence by noting that the first term in (37) is $\leq \alpha_t \lg^3 n$, for any $k \geq 1$. Thus,

$$T_3(n) \;\leq\; \alpha_t \lg^3 n + 2T_2(n/k), 2 \leq k \leq n/2 \tag{39}$$

Iterating this r times over n, we obtain

$$\begin{aligned}
T_3(n) &\leq \alpha_t \Sigma_{i=0}^{r-1} 2^i \lg^3 n/k^i + 2^r T_3(n/k^r) &(40)\\
&\leq \alpha_t \Sigma_{i=0}^{r-1} \lg^3 (n/k^i)^{2^i/3} + 2^r T_3(n/k^r) &(41)\\
&\leq \alpha_t \lg^3 \Pi_{i=1}^{r-1} (n/k^i)^{2^i/3} + 2^r T_3(n/k^r) &(42)\\
&\leq \alpha_t \lg^3 \left[\frac{n^{(2^r-1)/3}}{k^{(r-2)2^r/3}}\right] + 2^r T_3(n/k^r) &(43)\\
&\leq \alpha_t (\frac{2^r-1}{(r-2)2^r}) \lg^3 \frac{n}{k} + 2^r T_3(n/k^r) &(44)\\
&\leq (\frac{2\alpha_t}{r}) \lg^3 \frac{n}{k} + 2^r T_3(n/k^r),\;\; r \geq 4. &(45)
\end{aligned}$$

Letting $n/k^r = 2$, $T_3(2) = \alpha_t$, gives $r = \lg(n/2)/\lg k$, and

$$T_3(n) \;\leq\; 2\alpha_t \frac{\lg k \lg^3(n/k)}{\lg n} + \alpha_t (n/2)^{\frac{1}{\lg k}} \tag{46}$$

Now, recalling that $k \approx n^{\lg^{-\frac{1}{2}} n}$ minimizes the cost of Permuter 3, if we let $k = n^{\lg^{-\frac{1}{2}} n}$ in (38) and (46), then

$$\begin{aligned}
R_3(n) &= 2(n/2)^{1+\lg^{-\frac{1}{2}}(n/2)} - \alpha_r n &(47)\\
T_3(n) &\leq 2\alpha_t(\lg^{2.5} n + (n/2)^{\lg^{-\frac{1}{2}}(n/2)}). &(48)
\end{aligned}$$

The routing cost and routing time of Permuter 4 can be determined by a similar analysis. Here, we shall just consider these complexities for $k = n/2$, in which case, the parallel routing algorithm given in [15] can be directly applied. The routing cost and routing time of this algorithm are $\alpha_r n^{1+1/s}$ and $\alpha_t s \lg^3 n$, respectively, for any $s; 1 \leq s \leq \lg n$. To match the routing cost to the cost of the permuter, we let $s = \lg n/\lg\lg n$, which then gives

$$\begin{aligned}
R_4(n) &= \alpha_r n \lg n &(49)\\
T_4(n) &= \frac{\alpha_t \lg^4 n}{\lg\lg n}. &(50)
\end{aligned}$$

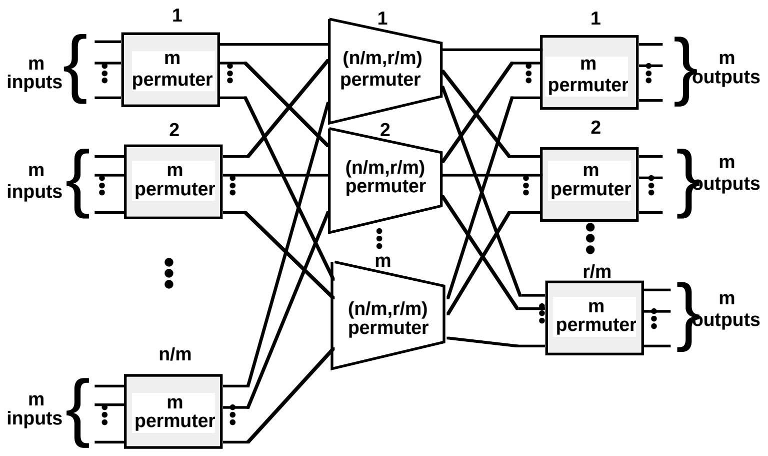

Figure 9: Permuter 5.

6 (n, r)-Permuters

Any of the n-permuters that have been described so far can be generalized to an (n, r)-permuter, where $r \leq n$. To illustrate this, consider the generalization of Permuter 4. As before, suppose that $n = mk$, and also suppose that $r = mk'$ where k, k' are some positive integers, and m is the size of the switches in the outer stages. An (n, r)-permuter can be obtained from an n-input Clos connector simply by deleting $k - k'$ of the k permuters in the third stage, and replacing the n/m-permuters in the center stage by $(n/m, r/m)$-permuters as shown in Figure 9. The proof that the resulting connector can permute any r-subset of its n inputs onto its r outputs is a direct corollary of the proof of Theorem 6 and is omitted. We just note that, in the proof, Lemma 1 must be applied to the output side of the network as $r \leq n$.

Now, if $C_5(n, r)$ and $D_5(n, r)$ denote the cost and depth of this (n, r)-permuter, respectively, we can generalize Equations (31) and (32) as

$$C_5(n, r) = 2\alpha_c(n + r)(m - 1) + mC_5(n/m, r/m) \tag{51}$$

$$D_5(n, r) = 4\alpha_d \lg m + D_5(n/m, r/m), \tag{52}$$

where the first terms in the recurrences give the cost and depth of the two outer stages combined together under assumption that the m-permuters in these stages are realized in terms of (m, m) elementary switches. Iterating these recurrences i times, we get

$$C_5(n, r) = 2\alpha_c i(n + r)(m - 1) + m^i C_5(n/m^i, r/m^i) \tag{53}$$

$$D_5(n, r) = 4\alpha_d i \lg m + D_5(n/m^i, r/m^i). \tag{54}$$

We note that $r \leq n$ so that if, after i iterations, $r/m^i = 2$, then $i = \frac{-1+\lg r}{\lg m}$, and

$n/m^i = 2n/r$. Hence

$$C_5(n,r) = \frac{2\alpha_c(n+r)(m-1)}{\lg m}(-1+\lg r) + r/2C_5(2n/r,2) \tag{55}$$
$$D_5(n,r) = 4\alpha_d(-1+\lg r) + D_5(2n/r,2), \tag{56}$$

where $C_5(2n/r,2)$ and $D_5(2n/r,2)$ denote the cost and depth of a $(2n/r,2)$-permuter. If we realize this latter permuter by an elementary $(2n/r,2)$-switch then $C_5(2n/r,2) = \alpha_c((2n/r)+2(2n/r-1)) = \alpha_c(6n/r-2)$ and $D_5(2n/r,2) = \alpha_d \lg 4n/r$. Upon substituting these in the above equations, we obtain

$$C_5(n,r) = \frac{2\alpha_c(n+r)(m-1)}{\lg m}(-1+\lg r) + \frac{r\alpha_c}{2}(6n/r-2) \tag{57}$$
$$D_5(n,r) = 4\alpha_d(-1+\lg r) + \alpha_d \lg 4n/r. \tag{58}$$

In particular, when $m = 2$, the cost becomes

$$C_5(n,r) = 2\alpha_c(n+r)(-1+\lg r) + \frac{r\alpha_c}{2}(6n/r-2). \tag{59}$$

It is noted that, when $r = n$, Equations (57) and (58) reduce to Equations (33) and (34). It is also noted that this permuter can be routed in much the same way as Permuter 4 when $m = 2$, and hence, assuming $r = O(n)$, its routing cost and routing time are of the same order as the routing cost and routing time of for Permuter 3.

7 Permuters Using Concentrators

Permuter designs described so far have all been based on dividing both a permuter's inputs and outputs into disjoint sets, and tieing these sets of inputs (outputs) to distinct switches. One drawback of this scheme is that it makes it difficult to map inputs to outputs in a distributed, or self-routing fashion, leading to complex routing schemes. In this section, we consider an alternative network design wherein only the outputs are divided. We show that this design can be optimized to obtain a permutation network with $O(n \lg n)$ cost and $O(\lg^2 n)$ depth that can self-route any permutation in $O(\lg^2 n)$ time[5]. An (n,r)-permuter construction using a concentrator is also given. For certain values of r, this new (n,r)-permuter construction outperforms Permuter 5 described in the previous section.

[5]We must note that, even though Permuter 2 match all these three complexities, routing it requires a combination of both global and self-routing schemes.

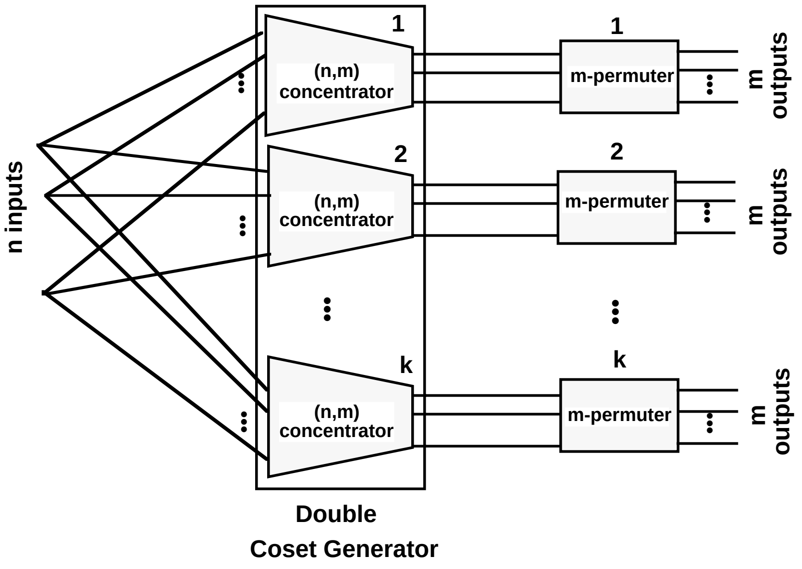

Figure 10: Permuter 6, $k = n/m$.

In group theoretic terms, avoiding the division of the inputs into disjoint sets amounts to replacing the product of subgroups on the left handside side of the decomposition given in Equation (22) by the trivial subgroup, i.e., the set $\{e\}$ where e is the identity permutation. Thus, the permuter corresponding to this decomposition is obtained by removing all k m-permuters in the first stage of Figure 8. As for the double coset generator in the center, it is easy to see that all one needs to do is to concentrate every m-subset of the inputs on the left upon the inputs of each m-permuter in the third stage. This results in the permuter design shown in Figure 10. Each (n, m)-concentrator allows any m of the n inputs on the left to reach the m inputs of the m-permuter to which it is cascaded together. Thereafter the concentrated m inputs can be permuted to their ultimate destinations by the m-permuter.

The (n, m)-concentrators in Figure 10 can be realized in terms of the self-routing concentrator described in [9]. For n inputs, and m outputs, $m \leq n$, this concentrator has $\gamma_c n$ cost, $\gamma_d \lg n$ depth, and $\gamma_t \lg n$ routing time, where $\gamma_c, \gamma_d, \gamma_t$ are small positive constants (≤ 5) independent of n (see [9]).

Now, let $C_6(n)$ and $D_6(n)$ denote the cost and depth of this permuter, and $C_{concen}(n, n/k)$ and $D_{concen}(n, n/k)$ denote the cost and depth of an $(n, n/k)$-concentrator, where $k = n/m$ as before. Then, from Figure 10,

$$C_6(n) = kC_{concen}(n, n/k) + kC_6(n/k) \quad (60)$$

$$D_6(n) = D_{concen}(n, n/k) + D_6(n/k), \quad (61)$$

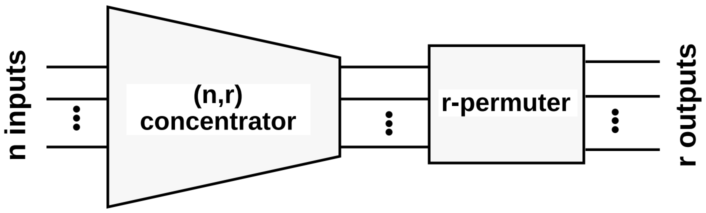

Figure 11: Permuter 7.

or substituting $\gamma_c n$ for $C_{concen}(n, n/k)$ and $\gamma_d \lg n$ for $D_{concen}(n, n/k)$,

$$C_6(n) = k\gamma_c n + kC_6(n/k) \tag{62}$$

$$D_6(n) = \gamma_d \lg n + D_6(n/k), \tag{63}$$

Solving these recurrences with $k = 2$, $C_6(2) = 4\alpha_c$, and $D_6(2) = 2\alpha_d$ yields

$$C_6(n) = 2\gamma_c n \lg n - 2(\gamma_c - \alpha_c)n \tag{64}$$

$$D_6(n) = \frac{\gamma_d}{2} \lg n(\lg n + 1) - (\gamma_d - 2\alpha_d). \tag{65}$$

We note that the routing cost of this permuter is zero since it is self-routing. Its routing time is the same as its depth, except for a constant factor.

Concentrators can also be used to construct more effficient (n, r)-permuters. As an example, consider the construction given in Figure 11. The concentrator on the left can map any subset of r inputs to the r inputs of the permuter on the right which can then permute them in any one of $r!$ ways to the r outputs, making the overall construction an (n, r)-permuter.

Let $C_7(n, r)$ and $D_7(n, r)$ denote the cost and depth of this (n, r)-permuter in that order. Assuming that the concentrator is implemented in terms of the self-routing concentrator construction described in [9] we have,

$$C_7(n, r) = \gamma_c n + C_7(r, r) \tag{66}$$

$$D_7(n, r) = \gamma_d \lg n + D_7(r, r). \tag{67}$$

Now, assuming that the r-permuter is implemented using the previous permuter design these equations become

$$C_7(n, r) = \gamma_c n + 2\gamma_c r \lg r - 2(\gamma_c - \alpha_c)r \tag{68}$$

$$D_7(n, r) = \gamma_d \lg n + \frac{\gamma_d}{2} \lg r(\lg r + 1) - (\gamma_d - 2\alpha_d). \tag{69}$$

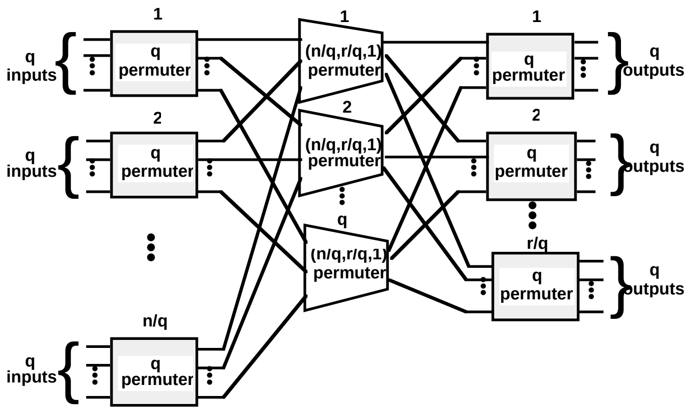

Figure 12: Permuter 8.

As in the previous permuter, this permuter is also self-routing, and hence its routing cost is zero, and its routing time is the same as its depth except for a constant factor. Furthermore, contrasting this (n,r) permuter with Permuter 5, we see that, when $r = O(n)$, both permuters have $O(n \lg n)$ cost, and $O(\lg^2 n)$ routing time, while Permuter 5 has a smaller depth than this permuter. For other values of r, this permuter outperforms Permuter 5. For example, when $r = O(\lg n)$, this permuter has $O(n)$ overall cost, and $O(\lg n)$ depth and routing time. For the same value of r, Permuter 5 has $O(n \lg\lg n)$ cost, $O(\lg n)$ depth, and $O(\lg^2 n)$ routing time. More strikingly, the cost of this permuter remains $O(n)$, even when $r = O(n^\epsilon)$ for any $0 < \epsilon < 1$. This is because, for any $\epsilon, 0 < \epsilon < 1$, $\lg n^\epsilon \leq n^{1-\epsilon}$, for sufficiently large n. Thus, for $r = O(n^\epsilon)$, and sufficiently large n, the second term in (68) is $\leq 2\gamma_c n^\epsilon n^{1-\epsilon} = O(n)$. Since the other two terms are also $O(n)$, the cost remains a linear function of n. In contrast, the cost of Permuter 5 becomes $O(n \lg n)$ for the same value of r.

8 (n,r,q)-Permuters

In this section, we give two (n,r,q)-permuter designs We note that any (n,r)-permuter design can be used as an (n,r,q)-permuter. In fact, when $q = O(r)$, one can justify realizing an (n,r,q)-permuter in terms of an (n,r)-permuter. Nonetheless, when q is less than r (in order of magnitude terms), we establish below that using an (n,r)-permuter as an (n,r,q)-permuter is not the best one can do.

As a first alternative, we mention that the (n,r)-permuter described in Section 6 can be modified to obtain an (n,r,q)-permuter at a lower cost. This can be achieved by replacing the m-permuters in the first and third stages with q-permuters, where $q \leq m$, and reducing the number of permuters in the center to q as shown in Figure 12. Since each map to be realized involves no more than q inputs and q outputs, it is easy to see that, if each of the connectors in the center stage is an $(n/m, r/m, 1)$-permuter, then any one-to-one assignment of any q inputs onto any q outputs can be realized by realizing each (input,output) pair in the assignment on a distinct $(n/q, r/q, 1)$-permuter in the center stage. Hence, the overall connector is an (n, r, q)-permuter.

Let $C_8(n, r, q)$ and $D_8(n, r, q)$ denote the cost and depth of this permuter, respectively, and $C_{permut}(q)$ and $D_{permut}(q)$ denote the cost and depth of each of the permuters in the first and third stages. Then from Figure 12,

$$\begin{aligned} C_8(n, r, q) &= (n + r)/qC_{permut}(q) + qC_8(n/q, r/q, 1) && (70) \\ D_8(n, r, q) &= 2D_{permut}(q) + D_8(n/q, r/q, 1). && (71) \end{aligned}$$

An $(n/q, r/q, 1)$-permuter with about $\alpha_c(n/q+r/q)$ cost and $\alpha_d(\lg n/q+\lg r/q)$ depth can be obtained by tieing the root nodes of a binary tree with n/q leaf nodes and another binary tree with r/q leaf nodes together, and designating the leaf nodes of the first one as inputs and those of the second as outputs. As for the q-permuters in the outer stages, they can be realized by using Permuter 6 design given in Section 7. The minimum cost of this design is given in Equation (64). Substituting q for n in this equation yields $C_{permut}(q) \approx 2\gamma_c q \lg q$, and for this cost, Equation (65) yields $D_{permut}(q) \approx \frac{\gamma_d}{2} \lg q(\lg q + 1)$.

Using these facts with Equations (70) and (71) gives

$$\begin{aligned} C_8(n, r, q) &\approx (n + r)\{2\gamma_c \lg q + \alpha_c\} && (72) \\ D_8(n, r, q) &\approx \gamma_d \lg q(\lg q + 1) + \alpha_d(\lg n/q + \lg r/q). && (73) \end{aligned}$$

As for routing, the q-permuters in the outer stages can be self-routed in roughly $\frac{\gamma_t}{2} \lg q(\lg q + 1)$ time, where γ_t is a constant independent of q, and the $(n/q, r/q, 1)$-permuters in the center stage can be self-routed in $\alpha_t(\lg n/q+\lg r/q)$ time without any additional cost. Summing these together, the routing time $T_8(n, r, q)$ of this permuter is

$$T_8(n, r, q) \approx \gamma_t \lg q(\lg q + 1) + \alpha_t(\lg n/q + \lg r/q). \qquad (74)$$

Assuming that $r = O(n)$, these expressions suggest the following facts. When $q = O(1)$, this design yields an (n, r, q)-permuter with $O(n)$ cost, $O(\lg n)$ depth, and $O(\lg n)$ routing time. When $q = O(\lg n)$, it yields an (n, r, q)-permuter with $O(n \lg \lg n)$ cost, $O(\lg n)$ depth, and $O(\lg n)$ routing time. Finally, when $q = O(n^\epsilon)$, where $0 < \epsilon < 1$

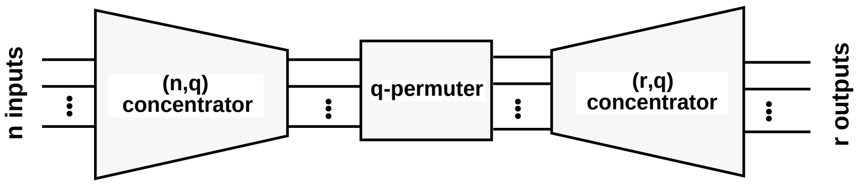

Figure 13: Permuter 9.

it yields an (n, r, q)-permuter with $O(n \lg n)$ cost $O(\lg^2 n)$ depth, and $O(\lg^2 n)$ routing time.

The first two cases show that this design is superior to realizing an (n, r, q)-permuter on an (n, r)-permuter. However, when $q = O(n^\epsilon)$, the same design leads to an (n, r, q)-permuter whose cost has the same order of complexity as that of an (n, r)-permuter. An (n, r, q)-permuter with fewer switches. can be obtained by using the connector design shown in Figure 13. Given that the (n, q)-concentrator on the left can connect any q of the n inputs to its q outputs, and the (r, q)-concentrator on the right can connect any q of its r outputs to its q inputs, it is easy to see that this connector is an (n, r, q)-permuter. Now, if $C_9(n, r, q)$ and $D_9(n, r, q)$ denote the cost and depth of this connector respectively, then directly from the figure,

$$C_9(n,r,q) \quad = \quad C_{concen}(n,q) + C_{permut}(q) + C_{concen}(r,q) \tag{75}$$

$$D_9(n,r,q) \quad = \quad D_{concen}(n,q) + D_{permut}(q) + D_{concen}(r,q) \tag{76}$$

where $C_{concen}(n, q)$, and $D_{concen}(n, q)$ are the cost and depth of the concentrator on the left, $C_{concen}(r, q)$, and $D_{concen}(r, q)$ are the cost and depth of the concentrator on the right, and $C_{permut}(q)$, and $D_{permut}(q)$ are the cost and depth of the permuter in the center. Using the concentrator design given in [9], we can bound the cost and depth of this (n, r, q)-permuter. Noting that a self-routing q-permuter can be realized via Permuter 6 design with no more than $2\gamma_c q \lg q - 2(\gamma_c - \alpha_c)q$ cost and $\frac{\gamma_d}{2} \lg q(\lg q + 1)$ depth, we have

$$C_9(n,r,q) \quad \leq \quad \gamma_c(n+r) + 2\gamma_c q \lg q \tag{77}$$

$$D_9(n,r,q) \quad \leq \quad \gamma_d(\lg n + \lg r) + \frac{\gamma_d}{2} \lg q(\lg q + 1). \tag{78}$$

It is obvious from these equations that, unlike the previous design, for $q = O(\lg n)$, as well as $q = O(n^\epsilon), 0 < \epsilon < 1$ this design yields an (n, r, q)-permuter with $O(n)$ cost. In fact, for $q = O(n^{0.5})$, even if the q-permuter in the center is realized as an elementary (q, q)-switch with q^2 crosspoints, the cost of the overall permuter remains

$O(n)$. Furthermore, given that all three components in this permuter design are self-routing, its routing cost is zero, and its routing time is the same as its depth, except for a constant factor.

9 Concluding Remarks

The paper considered permutation type connections within the context of a more general model wherein the number of inputs and outputs need not be equal and/or the number of simultaneous paths through a permuter may be less than the actual number of inputs and outputs. In all, nine different permuter designs– five n-permuters, two (n, r)-permuters, and two (n, r, q)-permuters– are presented. Two of these designs are Clos networks, one is an extension of a Clos network, and the remaining six are new designs.

A key result of the paper has been the construction of an (n, r)-permuter with $O(n + r \lg r)$ cost and $O(\lg n + \lg^2 r)$ routing time. This is the first known (n, r)-permuter construction that maintains an asymptotically minimum cost over all values of $r, 1 \leq r \leq n$. The paper also constructed an (n, r, q)-permuter with $O(n + r + q \lg q)$ cost and $O(\lg n + \lg r + \lg q^2)$ depth and routing time. This is also the first known (n, r, q)-permuter construction that maintains asymptotically minimum cost over all values of $q, 1 \leq q \leq r$.

Acknowledgments: The author thanks R. Greenberg, M. Chien, and C. Y. Lee of University of Maryland for their critical comments.

APPENDIX A

In what follows it is proved that $C_1(n)$, i.e., the cost of the connector in Figure 4 is $\Omega(n^{1.58})$ for all $m; n/2 \le m < n$.

Given that $n/2 \le m < n$, we let $m = qn$, where $1/2 \le q < 1$, and rewrite Equation (4) in Section 4 as

$$C_1(n) = 4\alpha_c(1-q)n + C_1((1-q)n) + 2C_1(qn) \tag{79}$$

Now, given $1/2 \le q < 1$ it follows that $\frac{1-q}{q} \le 1$, and hence, for $n >> 1$, and some $\alpha > 1$, $C_1((1-q)n) \ge (\frac{1-q}{q})^\alpha C_1(qn)$ where α depends on q, but not on n. Upon replacing $C_1((1-q)n)$ with $(\frac{1-q}{q})^\alpha C_1(qn)$ in Equation (79), one obtains

$$C_1(n) \ge 4\alpha_c(1-q)n + (\left(\frac{1-q}{q}\right)^\alpha + 2)C_1(qn) \tag{80}$$

The solution of this recursion with $C_1(2) = 4\alpha_c$, and the approximation $-1+\lg n \approx \lg n$ reveals that

$$C_1(n) \ge 4\alpha_c(1-q)\left(\frac{n^{1-\frac{\lg\left[(\left(\frac{1-q}{q}\right)^\alpha+2)q\right]}{\lg q}}}{\left((\frac{1-q}{q})^\alpha + 2\right)q - 1}\right) + 4\alpha_c n^{\frac{-\lg\left[\left(\frac{1-q}{q}\right)^\alpha+2\right]}{\lg q}}. \tag{81}$$

Elementary calculus shows that, for all $\alpha > 1$, the minimum value of second summand of $C_1(n)$ over $1/2 \le q < 1$ occurs at $q = 1/2$, and it is asymptotic to $n^{1.58}$. Therefore, $C_1(n) = \Omega(n^{1.58})$ for all $q, 1/2 \le q < 1$, or equivalently, for all $m, n/2 \le m < n$.

APPENDIX B

The notation and symbols used in the paper are listed below.

N	The set of integers $\{1, 2, \ldots, n\}$.
$\|M\|, m$	The number of elements in set M.
$N - M$	The set of elements in N but not in M.
$n - m$	The number of elements in set $N - M$.
$\pi_{X,Y}$	A mapping from set X into set Y.
m-set	A set of m elements.
G, H_1, H_2	Finite groups.
Σ_n	The symmetric group of permutations of degree n.
Σ_M	The symmetric group of permutations over set M.
$\Sigma_{M_1}\Sigma_{M_2}\ldots\Sigma_{M_k}$	The product of k symmetric groups defined over sets $M_1, M_2, \ldots, M_k$.
$R^{\pi}_{M_i,M_j}$	The range of M_i under π that coincides with M_j.
$R^{\pi}_{i,K}$	The set of all permuters in the third stage to which π maps an input from the ith permuter in the first stage.
$C_i(n)$	Cost of ith n-permuter.
$D_i(n)$	The depth of ith n-permuter.
$C_i(n, r)$	Cost of ith (n, r)-permuter.
$D_i(n, r)$	The depth of ith (n, r)-permuter
$C_i(n, r, q)$	The cost of ith (n, r, q)-permuter
$D_i(n, r, q)$	The depth of ith (n, r, q)-permuter
$C_{concen}(n, r)$	The cost of an (n, r)-concentrator.
$D_{concen}(n, r)$	The depth of an (n, r)-concentrator.
$C_{super}(n, r)$	The cost of an (n, r)-superconconcentrator
$D_{super}(n, r)$	The depth of an (n, r)-superconconcentrator.
$\lg n$	$\log_2 n$.
$\alpha_c, \beta_c, \gamma_c$	Cost constants.
$\alpha_d, \beta_d, \gamma_d$	Depth constants.
$\alpha_t, \beta_t, \gamma_t$	Routing time constants.
α_r	Routing cost constant.

References

[1] L. A. Bassalygo, "Asymptotically optimal switching circuits," *Problems of Information Transmission,* Vol. 17, No. 3, July-Sep. 1981, pp. 206-211.

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

[3] V. Beneš, "Toward a group theoretic proof of the rearrangeability theorem for Clos' network," *BSTJ*, Vol. 55, No. 4, Apr. 1975, pp. 797-804.

[4] V. Beneš, "Applications of group theory to connecting networks," *BSTJ*, Vol. 54, No. 2, Feb. 1975, pp. 407-420.

[5] V. Beneš, "Proving the rearrangeability of connecting networks by group calculations," *BSTJ*, Vol. 54, No. 2, Feb. 1975, pp. 421-434.

[6] F.R. K. Chung, "On concentrators, superconcentrators, generalizers, and nonblocking networks," *BSTJ*, Vol. 58, No. 8, 1979, pp. 1765-1777.

[7] C. Clos "A study of non-blocking switching networks," *BSTJ*, Vol. 32, 1953, pp. 406-425.

[8] T. Feng, "A survey of interconnection networks," *Computer*, Vol. 14, Dec. 1981, pp. 12-27.

[9] C. Y. Jan and A. Y. Oruç , " Fast self-routing permutation switching on an asymptotically minimum cost network," *Technical Report*, UMIACS T1291-127, CS-TR2753, University of Maryland, College Park. 14, Dec. 1981, pp. 12-27.

[10] C. Y. Jan and A. Y. Oruç, " Linear cost concentrators and superconcentrators with logarithmic concentration time," In preparation.

[11] P. Hall, "On representatives of subsets," J. London Math. Soc., 1935, pp. 26-30.

[12] W. H. Kautz, et. al., "Cellular interconnection arrays," *IEEE Trans. Comput.*, Vol. C-17, May 1968, pp.443-451.

[13] D. H. Lawrie, "Access and alignment of data in an array processor," *IEEE Trans. Comput.*, Vol. C-30, May 1981, pp. 324-332.

[14] G. M. Masson, G. Gingher, and S. Nakamura, “A sampler of circuit switching networks,” *Computer*, Vol. 12, June 1979, pp. 9-48.

[15] D. Nassimi and S. Sahni, ”Parallel algorithms to set-up the Benes permutation network,” *IEEE Trans. Comput.*, Vol. C-31, Feb. 1982, pp. 148-154.

[16] A. Y. Oruç, “Designing cellular permutation networks through coset decompositions of symmetric groups,” *Journal of Parallel and distributed Computing*, Academic Press Pub., Vol. 4, No. 2, Aug. 1987.

[17] A. Y. Oruç and S. Scheneider, “2-stage realizations of symmetric groups,” *In Proc. of 1987 Allerton Conference*, Urbana, IL., pp. 1037-1039.

[18] A. Y. Oruç, “Multistage permutation network design with double coset generators,” *In Proc. of International Conference on Parallel Processing*, Aug. 1987, St. Charles, IL., pp. 824-827.

[19] D. S. Parker, Jr., “Notes on shuffle/exchange-type switching networks,” *IEEE Trans. Comput.*, Vol. C-29, Mar. 1980, pp. 213-222.

[20] M. C. Pease, “The indirect binary n-cube microprocessor array,” *IEEE Trans. Comput.*, Vol. C-26, May 1976, pp. 458-473.

[21] M. S. Pinsker, “On the complexity of a concentrator,” *In Proc. 7th Int. Teletraffic Congress*, Stockholm 1973, pp. 318/1-318/4.

[22] G. F. Lev, N. Pippenger, and L. G. Valiant “A fast parallel algorithm for routing in permutation networks,” *IEEE Transactions on Computers,* Vol. C–30, No. 2, Feb. 1981, pp. 93–100.

[23] H. J. Siegel and S. D. Smith, “Study of multistage SIMD interconnection networks,” *In 5th Annual Symposium on Computer Architecture,* Apr. 1978, pp. 223-229.

[24] H. S. Stone, “Parallel processing with the perfect shuffle,” *IEEE Trans. Comput.*, Vol. C-20, Feb. 1971, pp. 153-161.

[25] C. D. Thompson, “Generalized interconnection networks for parallel processor intercommunication,” *IEEE Trans. Comput.*, Vol. C-27, Dec. 1978, pp. 1119-1125.

[26] W. A. Waksman, "A permutation network," *JACM*, May 1968, pp. 159-163.

[27] C. L. Wu, and T. Feng, "On a class of multistage interconnection networks," *IEEE Trans. Comput.*, Vol. C-29, Aug. 1980, pp. 694-702.