Network Optimization Problems, pp. 387-401
Eds. D.-Z. Du and P.M. Pardalos
©1993 World Scientific Publishing Co.

Steiner Problem in Multistage Computer Networks

Sourav Bhattacharya
Bhaskar Dasgupta[1]
Computer Science Department, University of Minnesota, Minneapolis, MN 55455

Abstract

Multistage computer networks are popular in parallel architectures and communication applications. We consider the message communication problem for the two types of multistage networks: one popular for parallel architectures and the other popular for communication networks. A subset of the problem can be equated to the *Steiner tree* problem for multistage graphs. Inherent complexities of the problem is shown and polynomial-time heuristics are developed. Performance of these heuristics is evaluated using analytical as well as simulation results.

1 Introduction

Multistage interconnection networks (MINs) are popular among parallel architecture and/or communication network topologies. An $N \times log_2 N$ element MIN consists of $log_2 N$ stages of N elements each. A common pictorial view of an $N \times log_2 N$ MIN is to collect N elements in a stage (vertically) and arrange $log_2 N + 1$ such stages horizontally one after the other. MINs offer a good balance between network cost and performance. They are often characterized as intermediate $\{O(N \times log_2 N)\}$ cost networks falling within the two extreme cases: fully connected $\{O(N^2)$ cost$\}$ and bus connected $\{O(N)$ cost$\}$. Architectural and other topological properties of MIN may be found in [8].

[1]Supported in part by NSF grant CCR-9208913

1.1 Two Versions of MINs

Let $S_{i,j}$ denote the i-th stage j-th row element in an $N \times log_2 N$ MIN, $0 \leq i \leq log_2 N, 0 \leq j \leq N-1$. We consider source-to-source wrap-around MINs only, i.e., when $\forall j : S_{0,j} = S_{log_2 N,j}$. These networks can allow multiple passes using the wrap-around connections. Depending on the role of intermediate stage elements, two types of MINs are possible as outlined below:

- *Intermediate stages as switches only:* This type is popular in parallel architecture applications. Here the source end (leftmost) and the destination end (rightmost stage) constitute of processors, while the intermediate elements are bare switches which interconnect various sources and destinations. Such MINs are of commercial usage in parallel processors, e.g., the BBN Butterfly machine. We refer to MINs of this type as *type-1 MIN*.

- *Intermediate stages as processors:* This type is common in communication network applications. Here the intermediate stage elements are identical to the source or destination stage processors, i.e., they can have their own message traffic. Example of such MINs can be found in [10]. We refer to MINs of this type as *type-2 MIN*.

1.2 Communication in MINs

Depending on the number of destinations involved in a communication in MIN, three types can be classified: *one-to-one*, *one-to-many* and *one-to-all*. These are commonly known as *routing*, *multicast* and *broadcast*. In this article we focus ourselves to the multicast problem for MINs. Note that routing & broadcast are two special instances of multicast and do not offer any opportunity for traffic reduction.

The multicast problem specifies a source node and a set of k destination nodes. Without loss of generality we assume the source node to be $S_{0,0}$. Destination nodes are spread over the MIN, $1 \leq k \leq N$ ($k = 1 \equiv$ routing, $k = N \equiv$ broadcast). Objective of the multicast problem is to transmit the message from the source node to the destination nodes.

Flow-control Mechanism

For multihop networks, various form of switching and flow-control mechanisms have evolved. *Store and forward* is a traditional approach to message communication. *Virtual cut-through, wormhole, deflection routing* etc. have been subsequently proposed. A survey can be found in [11, 4]. We assume packetized message communication, where packets are independently flown through the network. Our focus is to estimate (and possibly reduce) the overall traffic overhead in message communications.

1.3 Optimality Criteria in MIN Multicast

Two possible criteria to measure the optimality of MIN multicast communication are to minimize one of the following two objective functions:

- the total traffic generated in the network (each occupied link of the network counts as one unit of traffic.
- the hops-distance between the source node and any destination node.

The *traffic* metric makes the problem equivalent to the *Steiner problem for MIN*, while the *time* metric is a different dimension altogether. These two metrics work in the dual sense. Reducing one increases the other and vice versa. Thus, we focus on the *traffic* metric only. Considerations along the *time* metric is an open problem.

2 Multistage Interconnection Networks

We consider *type 1* MINs with the cube network topology. These class of networks (e.g., baseline, delta, generalized cube, indirect binary-cube, omega, banyan [8]) have been proposed as fixed-degree alternative to hypercube architecture. They are popular in switching and communication applications. They can also emulate the performance of hypercube in most applications (e.g., the CCC architecture [12]). Let MIN_d denote a d dimensional generalized MIN.

2.1 Formulation of the Traffic Reduction Problem.

We consider multicasting on MIN_d which are *unique path networks*. Given a set of k multicast destinations (D_i, $1 \leq i \leq k$) and a source node S in MIN_d, the path from S to any particular D_i is fixed. However, it is clear that for a given set of multicast destinations, the total traffic generated in MIN_d depends on the relative order in which d different dimensions are arranged. This leads to our problem formulation as (see Section 2.1.1 for practical applicability):

> *Given a set of destination nodes, traffic optimum multicasting in MIN_d is to find a permutation of the d dimensions (each stage of MIN_d is allocated to one particular dimension value) so that the total traffic is minimized.*

Unfortunately, this problem is NP-complete as shown by the next theorem. Hence, we need to investigate the possibility of designing efficient heuristics for this problem.

Theorem 2.1 *The traffic optimum multicasting problem is NP-complete.*

Proof sketch: The problem is obviously in NP. To show NP-hardness one can reduce the *space minimized full trie* problem, which is shown to be NP-complete in [3, 6], to this problem. Details are available in [1]. □

2.1.1 Design Issues

Any hardware implementation of a MIN_d would assume an ordering among the d dimensions. In such cases, online dimension ordering (as required by the traffic reduction criterion in this paper) in a MIN_d may be argued from the practical viewpoint. We identify the following situations as practical applications.

(1) *Communication networks* often use MINs. Traditional hardware implementation of switches at every intermediate stages have been replaced using Wave-Time Division Multiplexors (WTDM) over passive stars [5]. The actual interconnection is formed by wavelength (frequency) or time-slot assignment of different nodes, i.e., by *firmware* control. A *firmware* controlled design can be changed without changing the underlying hardware. Thus, it is possible to re-order the dimensions in a MIN_d dynamically. Every stage may have to configure to at most d possible dimensions, for which the wave/time assignments can be pre-computed and stored.

(2) If the traffic pattern is known and repetitive (as may happen in periodically occurring similar message communications) then from the above optimum dimensional ordering for each multicasting instance one can derive the most common pattern and design the MIN_d using the corresponding optimum dimensional ordering. The idea here is to achieve traffic optimality for *most* multicasting instances which leads to an overall traffic reduction.

(3) *Hierarchical hypercubes* are designed for several practical reasons [9]. Such hierarchical designs limit the availability of different dimensions at any node. Only a certain set of dimensions can be availed at each node. This imposes a hierarchy among dimensions in a routing/ multicasting operation. In some other cases, even with complete hypercubes routing/ multicasting is done in hierarchical fashion, imposing a (arbitrary) desired ordering among dimensions [4]. With these applications our results and optimality ordering among dimensions can be used as a measure whether or not a particular multicast operation is generating optimal traffic. Note that a hypercube with hierarchically ordered dimensions can be treated as a MIN_d for analysis purpose and results from the latter can be used for the former.

2.2 Greedy Heuristic

Let $Reach_p$ denote the number of nodes which received a copy of the message at stage p. Let k_d be the dimension between stage p and stage $p+1$. We define an *expansion ratio* $F_{k_d} = \frac{Reach_{p+1}}{Reach_p}$. Intuitively, this fraction F_{k_d} indicate how much the size of the multicast destination is increasing at every stage. This expansion ratio depends of the dimension, stage position and on the set of all prior dimensions served already. For the sake of brevity we treat this *all previous* information as part of the stage information and denote compactly using the stage number position. Now, the total traffic equals

$$\sum_{p=1}^{p=d} Reach_p = F_{k_1} \times [1 + F_{k_2} \times [1 + \ldots + F_{k_{d-1}} \times [1 + F_{k_d}] \ldots]]$$

Our objective is to arrive at values of k_1, k_2, ..., k_d, such that the above expression is minimized. Note that F_{k_i} $(1 \leq i \leq d)$ varies between 1 and 2 and have real values.

We propose the following greedy algorithm which works stage by stage and selects one dimension in each stage. After d iterations of the algorithm the complete permutation of d dimensions in MIN_d are generated.

Greedy heuristic:

> *At each stage select dimension k_i, where $\forall j$, $F_{k_j} \geq F_{k_i}$. In case of a tie, anyone of the smallest F_{k_i} dimension may be chosen. A particular dimension k_i used in a preceding stage may not be repeated in a subsequent stage.*

Regarding time and space complexities of the above heuristic, it is easy to prove the following theorem.

Theorem 2.2 *The greedy heuristic runs in $O(k.d^3)$ time, performs $O(k.d^4)$ bitwise arithmetic operations and uses $O(k.d)$ space.*

The following theorem, whose proof can be found in [1], shows local optimality of dimension ordering of the greedy heuristic.

Theorem 2.3 *The greedy heuristic leads to a locally optimal ordering of the dimensions, i.e., orders the dimensions so that no* **adjacent pairwise interchange** *of the dimensions can minimize the total traffic.*

Table 1 compares the time and space complexities of the greedy heuristic with two other known exponential time optimum strategies[1].

Algorithm	Time (in bit operations)	Space (in bits)
Direct Permutation	$O(k \cdot d^3 \cdot d!)$	$O(k \cdot d)$
Dynamic Programming	$O(k \cdot d^2 \cdot 2^d)$	$O(k \cdot d + 2^d \cdot d)$
Heuristic	$O(k \cdot d^4)$	$O(k \cdot d)$

Table 1: Multicast in MIN_d with k destinations: Summary of time and space complexity of different solutions.

2.3 Performance

This section analytically compares the greedy heuristic with "randomly ordered dimensions" approach as well as the optimal algorithm. Detailed proofs of all the results are available in [1]. For our analysis purpose, we characterize the *destination node set* into two classes: the "complete subcube multicast" and the "incomplete subcube multicast". Each one of these cases are explained below and worst (average) case performance of the greedy heuristic is compared with that of optimal algorithm as well as random dimension ordering. Let traffic "overhead" comparison between two approaches be denoted as the absolute difference in the traffic generated by those two individual approaches[2]. For example, Overhead(greedy, optimum) = greedy traffic - optimum traffic, Overhead(random, greedy) = traffic in "random ordering" - greedy traffic.

2.3.1 Complete Subcube Multicast

In this case the set of multicast destinations, D_i, $1 \leq i \leq k$, forms a complete subcube. Thus, $i = 2^r$, for some integer r and the set $[D_i]$ can form a r-dimensional subcube. Let $CSM(r)$ denote this situation.

Theorem 2.4 *The greedy heuristic produces optimum traffic for the complete subcube multicast case. Also, in the worst CSM(r) case,*

$$\textit{Overhead(random,optimum)}=\textit{Overhead(random,greedy)}=(d-r)\times(2^r-1).$$

The next theorem gives the probabilistic traffic overhead of "random dimension ordering" as opposed to the worst case performance as stated above.

Theorem 2.5 *In the average $CSM(r)$ case, the random dimensions ordering approach incurs a traffic overhead Overhead(random, optimum) = Overhead(random, greedy) = $\sum_{p=1}^{p=r} Q_r(p) \times (N_p - N_0)$, where*

$$N_p = \sum_{i=1}^{i=p} 2^i + 2^p \times (d-r) + 2^p \times \sum_{i=1}^{i=p} 2^i$$

$$Q_r(p) = \frac{\begin{pmatrix} r \\ p \end{pmatrix} \cdot (d-r) \cdot (d+p-r-1)! \cdot (r-p)!}{d!}$$

[2]A random dimension ordering occurs when d dimensions of MIN_d are randomly ordered.

2.3.2 Incomplete Subcube Multicast

In this case not all the 2^r destination nodes are present in the set of multicast destinations. Thus, the set of multicast destinations (D_i, $1 \leq i \leq k$) forms an incomplete r-dimensional subcube, where r is the minimum dimension value to include all those destination nodes. Let $ISM(r)$ denote this situation. The following theorems give worst case and average case performance ratios of various strategies. First we consider the case when k is a power of 2.

Theorem 2.6 *In the case of incomplete r-subcube multicast with $k = 2^j$ destinations,*

$$\begin{array}{lcl} \textit{Overhead(greedy, optimum)} & \leq & ((r-j) \times (k-1)) \\ \textit{Overhead(random, greedy)} & \leq & [(d-2r+log_2 k) \times (k-1)] \end{array}$$

Since $j = log_2 k$, Overhead(greedy,optimum)$\approx (r - log_2 k) \times k$. For a given r, this value is maximized when $k = 2^{r-1}$. Thus, the worst case performance degradation suffered by the greedy heuristic equals 2^{r-1}.

Next we generalize the value of k to a non-power of 2.

Lemma 2.7 *In $ISM(r)$, with $k = 2^j + l$ nodes ($0 \leq l \leq 2^j - 1$), Overhead(greedy, optimum) = A(d,k,r,j), where*

$$\begin{array}{lcl} A(d,k,r,j) & = & [(d-r) + \sum_{i=0}^{i=j+1} 2^i + (r-j-1) \times k] - \\ & & [(d-r) + (r-j) + \sum_{i=0}^{i=j} 2^i] \\ & \approx & 2^{j+1} + (r-j-1) \times (k-1) \end{array}$$

and, Overhead(random,greedy) = B(d,k,r,j), where

$$\begin{array}{lcl} B(d,k,r,j) & = & [(r-j) + \sum_{i=0}^{i=j} 2^i + (d-r) \times k] - \\ & & [(d-r) + \sum_{i=0}^{i=j+1} 2^i + (r-j-1) \times k] \\ & \approx & (d+j+1-2r) \times (k-1) - 2^{j+1} \end{array}$$

The following lemma gives estimates for the average performance. Let

$$P_k(r) = \frac{\begin{pmatrix} d \\ r \end{pmatrix} \cdot \sum_{j=0}^{r} (-1)^{r-j} \begin{pmatrix} r \\ j \end{pmatrix} 2^{d-j} \begin{pmatrix} 2^j \\ k \end{pmatrix}}{\begin{pmatrix} 2^d \\ k \end{pmatrix}}$$

We reuse notations of the previous lemma for brevity.

Lemma 2.8 *In $ISM(r)$, with $k = 2^j + l$ nodes ($0 \leq l \leq 2^j$ -1), on an average case:*

$$\begin{array}{lcl} \textit{Overhead(greedy, optimum)} & = & \sum_{p=0}^{p=r-1} P_{k=2^p}(r) \times A(d,k,r,p) \\ \textit{Overhead(random, greedy)} & = & \sum_{p=0}^{p=r-1} P_{k=2^p}(r) \times B(d,k,r,p) \end{array}$$

The above result for the average case is based on equal distribution of destination nodes among the given nodes.

2.4 Simulation Performance

Often a MIN_d design implicitly assumes ascending or descending order among its d dimensions. For example, in a MIN_3 an increasing order would be [0,1,2], while a decreasing order would be [2,1,0]. Such is also the usual practice in hierarchical routing in hypercube, where dimensions are treated one after another alike distinct stages of MIN_d. Clearly, these *linearly ordered dimensions* approaches cannot generate traffic optimal multicasting in MIN.

We compare performance of the greedy heuristic with the *linearly ordered dimensions* heuristic approach and demonstrate the advantages. We show for randomly generated M multicast destinations how much traffic can be reduced (on an average) if the locally optimal greedy dimension ordering approach proposed in this paper is followed. We present simulation results towards this.

Our simulation implemented four situations: the exhaustive optimum traffic generation approach, greedy approach, linearly increasing and linearly decreasing (hereafter referred as 'increasing' and 'decreasing' respectively). Performance of the "increasing" and "decreasing" cases are found similar, and hence we report only the "increasing" case. We consider three different dimension values 4, 5 and 6 (i.e., MIN_4, MIN_5 and MIN_6). Number of multicast destinations are varied as 1%, 2%, 5%, 10%, 20%, 50%, 80%, 90%, 95% and 99% of the total number of nodes in the cube. For each multicast set size 30 random distributions were generated and averages taken to introduce the effect of *large numbers*. Thus, each algorithm is run 300 times with every dimension - leading to a total of 3600 experiments.

The proposed greedy algorithm produces optimal multicast traffic in most ($\approx$ 90%) of the cases. Thus the greedy heuristic is "almost always" optimum. We present simulation results showing the *miss-rate*, i.e., the percentage of test runs in which the proposed greedy algorithm does not coincide with the optimum multicast algorithm. Even in cases where the greedy algorithm deviates from optimum solution, the deviation is found to be small. This section also shows the relationship of *optimal* traffic cost (T) to the number of multicast destinations (M). We present simulation results to show variation of T for different values of M.

Let T_O, T_G and T_I denote the traffic generated using the *optimum* (exhaustively generated), *greedy* and *linearly ordered increasing* approaches respectively. Thus, the traffic overhead using *greedy* and *increasing* approaches equal $(T_G - T_O)$ and $(T_I - T_O)$ respectively. Fig.1a shows the average value of these parameters for MIN_d with d=4, 5, 6.

Let "miss-rate" be defined as the percentage of simulation runs in which a particular heuristic approach (e.g., the greedy approach or the increasing approach) differs from the optimum approach. This percentage shows the frequency by which the heuristic deviates from the optimum solutions. A low value of miss-rate indicates that the corresponding heuristic is 'almost always' optimum. Fig.1b shows the miss-rate *greedy*, *increasing* heuristics for different dimensions.

Observation 1: The greedy approach misses the optimum solution rarely (e.g.,

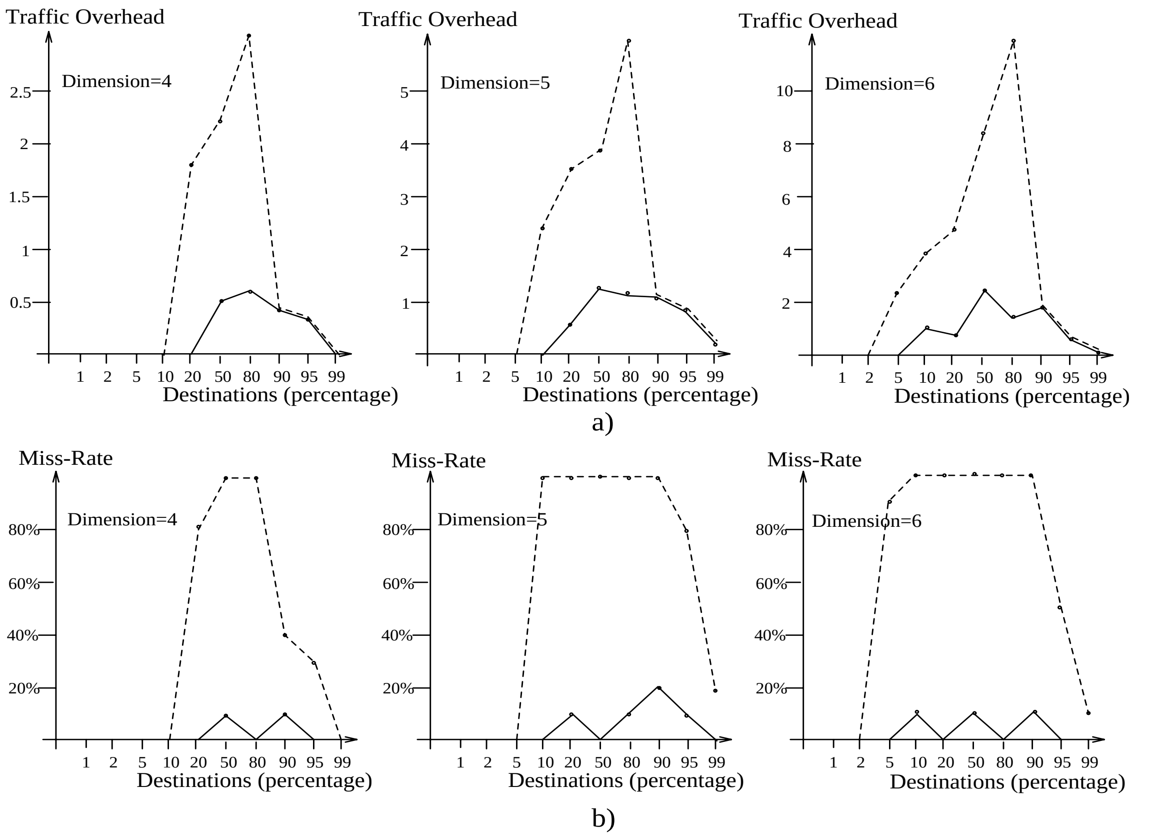

Figure 1: Greedy (solid line) and Increasing (dashed line) heuristic performance: a) Traffic overhead (≡traffic in heuristic - optimal traffic), b) Miss-rate.

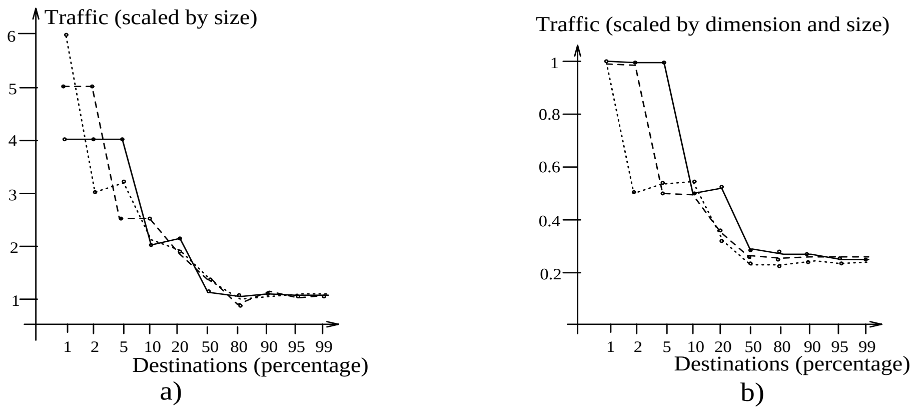

Figure 2: Optimum traffic (per destination node) variation with different multicast sizes (solid line for dimension=4, dashed line for dimension=5, dotted line for dimension=6).

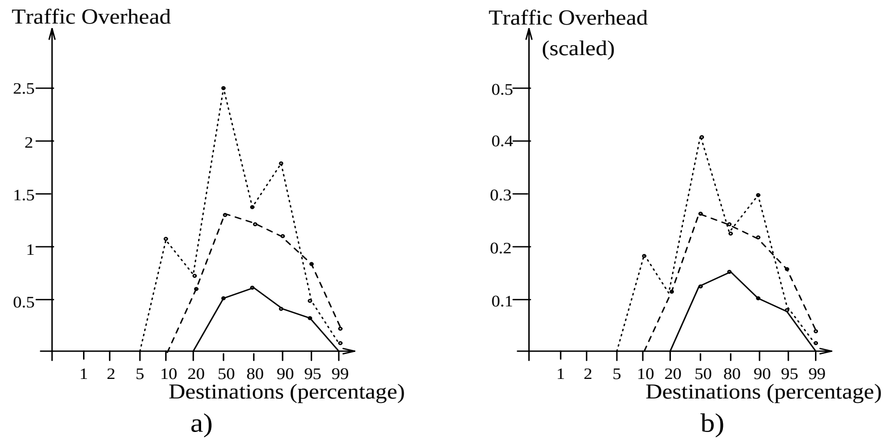

Figure 3: a) Traffic overhead of Greedy heuristic, b) Scaled traffic overhead (soild line for dimension=4, dashed for 5, dotted for 6).

1 out of 30 cases or at most 2 out of 30 cases). Thus it is an 'almost always optimum' algorithm. The number of mismatches increases as the dimension increases. However, the existing dimension ordering approaches (e.g., increasing or decreasing) have high miss-rate.

Observation 2: For small (or large) number of multicast destinations all three heuristics yield optimum (or near optimum) traffic solutions. This is because, for small number of destinations *expansion ratio* (refer Section 2.2) is almost always 1 and regardless of the actual heuristic used, a near-optimum strategy is effected. Similarly, with large fraction of nodes as destinations, the *expansion ratio* is almost always 2 and regardless of the actual heuristic used, a near-optimum strategy is effected.

Observation 3: Traffic overhead of greedy approach is superior to "increasing" approach.

Observation 4: Traffic overhead of greedy approach increases with dimension (Fig. 3a). This is expected since at higher dimensions, each source-destination multicast involves larger traffic amount. At the same time it can also be attributed to the inherent characteristic of the greedy approach, i.e., the greedy approach deviates from optimality more with increasing dimension.

These two factors are distinguished by scaling the traffic overhead using dimension value. The idea is to normalize the traffic overhead using the corresponding dimension value. The scaled traffic overhead (Fig. 3b) using the greedy heuristic for different dimensions also show that traffic overhead of greedy approach increases with dimension.

Fig.2 shows the optimum traffic load variation for different multicast sizes. The idea is to explore relationship (if any) between the total amount of traffic (T) required for a M destination multicast. We show the average traffic (i.e., traffic per destination node) reported from our simulations for cube sizes 4, 5 and 6. Then, we scale the traffic requirement using the corresponding dimension value. This plot also shows similar trend as the absolute traffic plot.

Observation 5: Optimum multicast traffic decreases with increasing multicast destination size. Initially, with small M, every destination requires nearly one unit of traffic per stage. This is because the destinations nodes are sparse and on an average no common message traffic can be shared by two destinations. However, with increasing M this ratio decreases until M reaches 50% of the cube size. Beyond this range of M the value of 'optimum traffic' per multicast destination becomes almost a constant indicating high availability of destination nodes and frequent ability to share message links most efficiently.

3 Multistage Communication Networks

We consider *type 2* MINs in this section. Among several topologies we choose the shuffle connection and the multistage binary cube connection. Without loss of generality we assume that $S_{0,0}$ is the source node, while the k destinations nodes are spread over the $(log_2 N + 1)$ stages and N rows.

Type-1 MINs are unique path networks. This required us to re-order dimensions in order to have traffic reduction. The online dimension re-ordering led to practical feasibility questions (which is addressed in Section 2.1.1) and related issues. However, type-2 MINs are not unique path networks. They allow multiple paths between source and any destination. Hence, traffic reduction can be achieved even without any online topological reconfiguration.

Optimality Criterion

Given an $N \times log_2 N$ type-2 multistage communication network (connected using a particular topology T) with a source node S and k destination nodes D_i $(1 \leq i \leq k)$, the objective is to find a path from the source node to each one of the destinations such that one of the following objective functions is minimized:

- Total *traffic*.
- *Time* (in hops) between source and each destination.

The first objective equates the problem to the *Steiner tree* problem for the topology T. The second objective has not been investigated so far.

3.1 Multistage Shuffle Network

We consider a shuffle network with $N \times log_2 N$ PEs, arranged as $log_2 N$ stages of N PEs each. Let such a network be called a $log_2 N$-shuffle, Let $PE_{i,j}$ be the i-th row PE in the j-th stage, $0 \leq i \leq N-1$, $0 \leq j \leq log_2 N - 1$. $log_2 N$-Shuffle is a cyclical

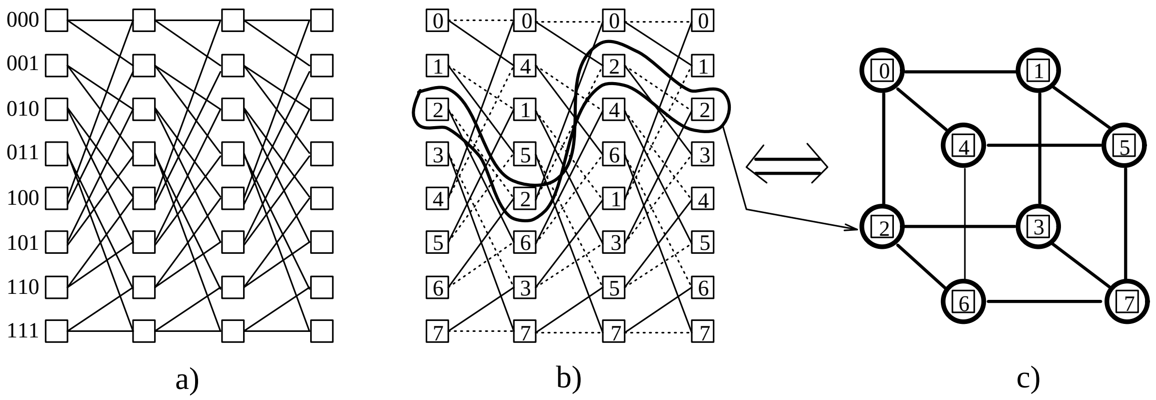

Figure 4: NP-completeness: a) Example 3-stage shuffle network, b) Restricted instance of the shuffle graph - dotted lines indicate zero-cost edges - nodes which are grouped together are tagged with identical integer, c) Equivalence to a 3-cube multicast.

wrap around network, with log_2N-th stage $\equiv$ stage 0. Formally, the binary shuffle connectivity is defined as $(N = 2^n)$ [10]

$$PE_{i,j} \text{ has } \textit{outgoing} \text{ links to } PE_{(2i+p)\ mod\ N,(j+1)\ mod\ n}, \text{ where } p \in \{0,1\}$$

Theorem 3.1 *The problem of traffic optimal shuffle multicast tree generation is NP-complete.*

Proof sketch: It can be shown that a special case of this problem is the problem of *optimal traffic multicast for hypercube* (which is known as NP-complete[7]) by setting the costs of some edges to zero and thereby identifying some nodes together. Details can be found in [2]. Fig. 4 pictorially depicts the idea. □

3.2 Multistage Cube Network

We consider a multistage cube network with $N \times log_2N$ PEs, arranged as log_2N stages of N PEs each. Let such a network be called a log_2N-stage cube. Let $PE_{i,j}$ be the i-th row PE in the j-th stage, $0 \leq i \leq N-1$, $0 \leq j \leq log_2N - 1$. log_2N-stage cube is a cyclical wrap around network, with log_2N-th stage $\equiv$ stage 0. Formally, the binary multistage connectivity is defined as $(N = 2^n)$[8]

$$\begin{array}{lll} PE_{i,j} \text{ has } \textit{outgoing} \text{ links to } & PE_{i,(j+1)\ mod\ n} & \textit{and} \\ & PE_{i+(1-2r)\times 2^j,(j+1)\ mod\ n} & \text{where } r = i\text{-th bit in } j \end{array}$$

Note that this multistage cube network is a particular instance of the generalized MIN_d considered in Section 2, where the dimensions are increasing from left to right. Also, intermediate stage nodes are active PEs here, unlike in the MIN_d of Section 2 (which are bare switches).

```
define $Dist(D,a)$ = MIN $\{ H(D_j,a) : D_j \in D \}$
$\forall n, (n \notin D) \wedge (\exists D_j \in D : H(D_j,n) = 1)$ Do
        $count(n) = |\{d_k \in DL : Dist(D \cup \{n\}, d_k) < Dist(D, d_k)\}|$
$select$ $n$ such that $count(n)$ is maximum;
```

Figure 5: Greedy algorithm for type-2 MINs: selection of the next step message recipient node.

The proof of the following theorem is essentially similar to theorem 3.1. Details can be found in [2].

Theorem 3.2 *The optimal traffic multicast problem is NP-complete.*

3.3 Greedy Heuristic

We developed a greedy heuristic for *type-2* MINs. This heuristic is applied to both the type-2 shuffle MIN as well as type-2 multistage cube MIN. We describe this greedy heuristic first and then demonstrate its performance using simulation results.

The greedy heuristic is an iterative process selecting one node in each iteration. Every time a node is selected it is included in a set D (representing the set of nodes which received a copy of the message so far). Initially D set only includes S, i.e., the source node. The algorithm stops when $\forall i\ 1 \leq i \leq k\ D_i \in D$. Let DL denote the set of k destinations, and $H(a,b)$ equal the *shortest distance* between nodes a and b. Let $Dist(D,a)$ be a function indicating the shortest distance from any node in D to a. Each step of the greedy iteration chooses a node n as in Fig. 5.

3.4 Simulation Performance

We simulated the greedy algorithm in the multistage shuffle MIN as well as in the multistage cube MIN. Its performance is compared with the exhaustively generated optimum algorithm in the respective architectures. The number of destinations is varied as 1%, 2%, 5%, 10%, 20%, 50%, 80%, 90%, 95% and 99% of the total system size. In each case 50 random set of destinations were generated; both the greedy & optimum algorithms are run and their results compared.

Two metrics are used to characterize performance of the greedy heuristic: the *number of miss* and the *average overhead*. The former denote how often (out of the 50 runs) the greedy algorithm fails to produce optimal result, while the latter indicates average deviation of the greedy result when a miss occurs. A low miss rate indicates that the greedy algorithm is *almost always optimum*, while a low *average overhead* indicates that the greedy algorithm *almost always produces a near-optimum solution.* Let T_G (T_O) denote the traffic produced by the greedy (optimum) algorithm.

- *Number of miss* $= |T_G \neq T_O|$.
- *Average overhead* $= \sum \frac{T_G - T_O}{T_O}$, where $T_G \neq T_O$.

Table 2 shows the performance of the greedy algorithm in multistage shuffle MIN, while Table 3 shows the same for multistage cube MIN. As can be observed from these two tables, the greedy algorithm has low miss rate (particularly for the multistage cube MIN) and low traffic overhead.

Systems	Metrics	Destinations									
		0.01	0.02	0.05	0.1	0.2	0.5	0.8	0.9	0.95	0.99
3-stage Shuffle	Number of miss	1	3	6	8	11	24	16	14	12	9
	Average overhead	1.01	1.03	1.23	1.4	1.52	1.6	1.56	1.31	1.19	1.06
4-stage Shuffle	Number of miss	1	4	9	11	16	32	27	19	12	10
	Average overhead	1.12	1.15	1.33	1.47	1.63	1.82	1.58	1.39	1.28	1.17
5-stage Shuffle	Number of miss	2	5	10	14	21	37	29	23	17	11
	Average overhead	1.14	1.23	1.41	1.53	1.71	2.09	1.69	1.45	1.32	1.21

Table 2: Multistage Shuffle Multicast: Performance of the Greedy heuristic over the optimal solution.

Systems	Metrics	Destinations									
		0.01	0.02	0.05	0.1	0.2	0.5	0.8	0.9	0.95	0.99
3-stage Cube	Number of miss	1	2	3	3	4	7	5	3	1	1
	Average overhead	1.01	1.01	1.01	1.01	1.04	1.1	1.04	1.01	1.01	1.01
4-stage Cube	Number of miss	2	2	3	4	6	10	5	4	3	1
	Average overhead	1.01	1.01	1.01	1.03	1.11	1.18	1.09	1.03	1.01	1.01
5-stage Cube	Number of miss	2	3	6	7	9	12	8	6	3	2
	Average overhead	1.01	1.01	1.02	1.04	1.14	1.23	1.12	1.03	1.01	1.01

Table 3: Multistage Cube Multicast: Performance of the Greedy heuristic over the optimal solution.

4 Conclusion

Multistage networks are popular for parallel architecture and/or communication network applications. We formulate the traffic optimum multicasting problem for multistage networks. Optimum traffic multicasting problem is NP-complete. Several greedy heuristics are proposed and their performances are shown using analytical as well as simulation methods. This work considered just *traffic* optimality in MIN multicasting. Optimality issues of *time* metric in MIN multicasting is left as an interesting open problem.

Acknowledgments: We thank Gary Elsesser, Lionel M. Ni and Wei-Tek Tsai for helpful discussions.

References

[1] S. Bhattacharya, G. Elsesser, W.T. Tsai and D.Z. Du, Multicasting in Generalized Multistage Interconnection Networks, to appear in *Journal of Parallel and Distributed Computing*, 1993.

[2] S. Bhattacharya and L.M. Ni, Multicasting in Multistage Communication Networks, *preprint*, 1992.

[3] D. Comer and R. Sethi, Complexity of TRIE Index Construction, *17th Annual Symposium on FOCS*, 1976, pp 197-207.

[4] W. J. Dally and C. L. Seitz, Deadlock-Free Message Routing in Multiprocessor Interconnection Networks, *IEEE Transactions on Computers*, C-36 (1987), pp. 547-553.

[5] P. Dowd, Random Access Protocols for High-Speed Interprocessor Communication Based on an Optical Passive Star Topology, *Journal of Lightwave Technology*, 9 (1991), pp. 799-808.

[6] M.R. Garey M.R. and D.S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completeness*, (Freeman, San Fransisco, CA, 1979).

[7] L.R. Foulds and R.L. Graham, The Steiner Problem in Phylogeny in NP-Complete, *Advances in Applied Mathematics*, 3 (1982), pp. 43-49.

[8] K. Hwang and F.A.Y. Briggs, *Computer architecture and parallel processing*, (New York : McGraw-Hill, c1984).

[9] K. Hwang and J. Ghosh, Hypernets: A Communication-Efficient Architecture for Constructing Massively Parallel Computers, *IEEE Transactions on Computers*, C-36 (1987), pp. 1450-1466.

[10] M.G. Hluchyj and M.J. Karol, ShuffleNet: An Application of Generalized Perfect Shuffles to Multihop Lightwave Networks, *Journal of Lightwave Technology*, 9 (1991).

[11] P. Kermani and L. Kleinrock, Virtual cut-through: A new computer communication switching technique, *Computer Networks*, 3 (1979), pp. 267-286.

[12] F.P. Preparata and J. Vuillemin, The Cube-Connected Cycles: A Versatile Network for Parallel Computation, CACM, 24 (1981), pp. 300-309.