Node Placement Optimization in ShuffleNets

Kwan Lawrence Yeung, *Member, IEEE*, and Tak-Shing Peter Yum, *Senior Member, IEEE*

***Abstract*— Node placement problem in ShuffleNets is a combinatorial optimization problem. In this paper an efficient node placement algorithm, called the gradient algorithm, is proposed. A communication cost function between a node pair is defined and the gradient algorithm places the node pairs one by one, based on the gradient of the cost function. Then two lower bounds on the traffic weighted mean internodal distance $\overline{h}$ are proposed. The performance of the gradient algorithm is compared to the lower bounds as well as to some algorithms in the literature. Significant reduction of $\overline{h}$ is obtained with the use of the gradient algorithm, especially for highly skewed traffic distributions. For a ShuffleNet with $N = 64$ nodes, the $\overline{h}$ found is only 22% above the lower bound for the uniform random traffic distribution, and 14.7% for a highly skewed traffic distribution with skew factor $\gamma = 100$.**

***Index Terms*— Gradient algorithm, node placement problem, ShuffleNet, topological design.**

I. Introduction

NETWORK designs using wavelength-division multiplexing (WDM) technologies can be grouped into two classes: single-hop networks and multihop networks [1]. In a single-hop WDM network, information is transmitted directly from the source to the destination without going through intermediate nodes. The main drawback of this design is the need to use fast tunable optical transceivers, which are not mature for mass production at the current stage. Besides, the performance of a single-hop WDM network depends on the efficiency of the transmission protocol in performing pretransmission coordination. On the other hand, in a multihop WDM network only a small number of fixed optical transceivers are needed, but the information from a source node may need to go through a number of intermediate nodes before reaching its destination node.

The logical topology of a multihop WDM network is relatively independent of its physical topology. Depending on the wavelength assignment, a logical topology can be designed to optimize the network performance independent of its underlying physical networks. Literature on designing logical topology based on different physical networks can be found, for example, in [2] and [3]. The logical topology of a multihop WDM network can be either regular (symmetrical), such as ShuffleNet, Toroid, DeBruijn graph, and Hypercube, or irregular (asymmetrical). A regular topology can use simple routing rules, which is crucial for high-speed networks. Two major design issues for regular topologies are: 1) modular increase and decrease of network size and 2) adaptation to nonuniform traffic. Research on the first issue can be found, for example, in [4] and [3]. Research on the second issue can be grouped into two classes. The first class focuses on the optimal initial placement of nodes into an empty regular topology given the (average) traffic requirements between all pairs of nodes. This is known as the *node placement problem*. The commonly used objective function is to minimize the traffic weighted mean internodal distance of a network. If the network consists of equal length links, this is equivalent to minimize the mean network packet delay. It has been shown that the node placement problem for even the simplest linear bus regular topology is NP-hard [5]. Heuristic algorithms for suboptimal solutions are reported for linear bus topology [5], ring topology [6], and ShuffleNet [7]. Assume that all nodes have already been placed in the network; the second class of research for adapting to nonuniform traffic focuses on the routing/switching problem on a packet-by-packet or call-by-call basis [8], [9].

In this paper we focus on the design of an efficient node placement algorithm for ShuffleNets. A multistage shuffle-exchange network architecture was originally proposed for processor–memory and processor–processor interconnection in modular multiprocessor systems [10], [11]. Many broad-band packet switch designs are also based on this architecture [12], [13]. ShuffleNet [14] is another proposed design of shuffle-exchange networks for telecommunications applications. It is one of the most prominent examples of WDM multihop lightwave networks and has attracted a lot of research interests. Among the proposed regular multihop network topologies, ShuffleNet has the advantage of having a large connected graph with a small degree and diameter. It has been shown that ShuffleNet compares favorably with Torous network [15], and the maximum throughput supported by a ShuffleNet is higher than that supportable by an equivalent network based on de Bruijn graph [16].

In this paper an efficient node placement algorithm, called the gradient algorithm, is proposed. It has a time complexity of $O(N^3)$, where N is the network size. A communication cost function between a node pair is defined and the gradient algorithm places the node pairs one by one, based on the gradient of the cost function. Then two lower bounds on the traffic weighted mean internodal distance $\overline{h}$ are proposed. The performance of the gradient algorithm is compared to the lower bounds as well as to some algorithms in the

Manuscript received January 16, 1997; revised January 26, 1997; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor H. S. Kim.

K. L. Yeung is with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong (e-mail: kyeung@ee.cityu.edu.ee).

T.-S. P. Yum is with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong (e-mail: tsyum@ie.cuhk.edu.hk).

Publisher Item Identifier S 1063-6692(98)04076-X.

literature. Significant reduction of $\overline{h}$ is obtained with the use of the gradient algorithm, especially for highly skewed traffic distributions. For a ShuffleNet with $N = 64$ nodes, the $\overline{h}$ found is only 22% above the lower bound for the uniform random traffic distribution, and 14.7% for a highly skewed traffic distribution with skew factor $\gamma = 100$.

In Section II the node placement problem for the ShuffleNet is formally formulated. In Section III a communication cost function for placing a node pair into a network is defined. The gradient algorithm is proposed in Section IV and an example is presented in Section V to illustrate the algorithm. In Section VI we derive two lower bounds on the weighted mean hop distance. Then the performance of the proposed algorithm is evaluated in Section VII. Some comments on the future work conclude the paper in Section VIII.

II. Node Placement Problem

ShuffleNet was first proposed in [14] and later extended in [17]. It is a communication network in which nodes in one column are connected to nodes in the next column in a perfect shuffle connection pattern by directed links. A ShuffleNet is characterized by two integer parameters p and k. In a (p, k) ShuffleNet, the total number of nodes N is equal to kp^k. They are numbered from 0 to $kp^k - 1$ and are arranged in k columns of p^k nodes each, with the kth column wrapped around to the first in a cylindrical fashion (Fig. 1). The number of transmitters and the number of receivers per node are both equal to degree p. The total number of links is kp^{k+1}. The location of a node n in a ShuffleNet can be represented by its (*row*, *column*) coordinates (a, b), where $a = n \bmod p^k$ and $b = \lfloor n/p^k \rfloor$. The p outgoing links of node n are connected to the p nodes in the next stage with respective coordinates $(c, d), (c+1, d), \cdots, (c+p-1, d)$, where $d = (b+1) \bmod k$ and $c = (a \bmod p^{k-1})p$.

Let $A = [\lambda_{ij}]$ be the traffic matrix where λ_{ij} denotes the traffic rate from node i to node j. Let $\lambda_{ii} = 0$ and $\Lambda = \Sigma_{i,j}\ \lambda_{ij}$. Let h_{ij} be the hop distance from node i to node j. The weighted mean hop distance between two nodes can be expressed as

$$\overline{h} = \frac{1}{\Lambda} \sum_i \sum_j h_{ij}, \lambda_{ij}. \tag{1}$$

The node placement problem is to assign nodes to the ShuffleNet positions such that $\overline{h}$ is minimized. For the special case where all λ_{ij}'s, except $i = j$, are equal, the minimum average hop distance is given by [7]

$$\overline{h} = \frac{kp^k(p-1)(3k-1) - 2k(p^k-1)}{2(p-1)(kp^k-1)}. \tag{2}$$

III. Communication Cost

The diameter of a ShuffleNet is the minimum hop distance between two furthest-away nodes and is equal to $2k - 1$. A ShuffleNet is cylindrically wrapped around and each node can reach back to itself after traversing through the network. We call such a traversal path a *loop*. Based on the regular topology of a ShuffleNet, each node has a *unique* shortest loop of k hops, called the k-loop. If the traversal path does not follow the k-loop, the next shortest traversal path has $2k$ hops. These $2k$-hop loops (or $2k$-loops for short) are not unique. If node j is in the k-loop of node i, node i is also in the same k-loop of node j. The same is true for the $2k$-loops. As an example, Fig. 1 shows that the loop (node 0 $\Rightarrow$ node 4 $\Rightarrow$ node 0) is the k-loop for node 0, where $k = 2$, and the loop (node 0 $\Rightarrow$ node 5 $\Rightarrow$ node 3 $\Rightarrow$ node 6 $\Rightarrow$ node 0) is a $2k$-loop of node 0. The communication cost from node i to node j is defined as the product of the traffic rates from node i to node j and the number of hops needed to travel from node i to node j. If node j is in the k-loop of node i and node i can reach node j in x hops, then node j can reach node i in $k-x$ hops, where $0 < x < k$. Under this condition, the communication cost *between* nodes i and j, denoted as $c_{ij}(x)$, is

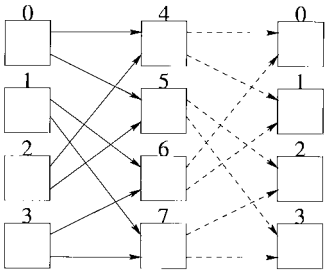

Fig. 1. A (2,2) ShuffleNet.

$$\begin{aligned} c_{ij}(x) &= x\lambda_{ij} + (k-x)\lambda_{ji} \\ &= k\lambda_{ji} + m_{ij}x, \qquad \text{for } 0 < x < k \end{aligned} \tag{3}$$

where $m_{ij} = \lambda_{ij} - \lambda_{ji}$ is the gradient of the cost function.

Similarly, if node j is in a $2k$-loop of node i, and node i can reach node j in y hops, then node j can reach node i in $2k - y$ hops, where $0 < y < 2k$. The communication cost $d_{ij}(y)$ *between* nodes i and j can be defined as follows:

$$\begin{aligned} d_{ij}(y) &= y\lambda_{ij} + (2k-y)\lambda_{ji} \\ &= 2k\lambda_{ji} + m_{ij}y \quad \text{for } 0 < y < 2k. \end{aligned} \tag{4}$$

Equations (3) and (4) show that $c_{ij}(x) = c_{ji}(k-x)$ and $d_{ij}(y) = d_{ji}(2k-y)$, or they are linear functions of x and y, respectively. If $\lambda_{ij} \geq \lambda_{ji}, c_{ij}(x)$ and $d_{ij}(y)$ are minimized when $x = 1$ and $y = 1$ respectively. Otherwise, $c_{ij}(x)$ and $d_{ij}(y)$ are minimized when $x = k-1$ and $y = 2k-1$, again respectively. If $\lambda_{ij} = \lambda_{ji}$, $c_{ij}(x)$ and $d_{ij}(y)$ are constants and are independent of x and y.

An $N = kp^k$ ShuffleNet has $N(N-1)/2$ different node pairs. They can be divided into two sets ξ_1 and ξ_2, where ξ_1 is the set of node pairs which are in the k-loop of each other, and ξ_2 is the remaining set of node pairs which are in a $2k$-hop

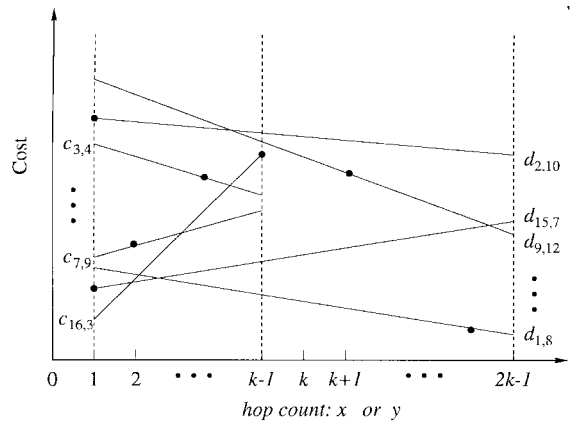

Fig. 2. $c_{ij}(x)$ and $d_{ij}(y)$ of a (p, k) ShuffleNet.

loop of each other. Substituting $c_{ij}(x)$ and $d_{ij}(y)$ into (1), we get $\overline{h} = C/\Lambda$, where

$$C = \sum_{(i,j)\in\xi_1} c_{ij}(x) + \sum_{(i,j)\in\xi_2} d_{ij}(y) \tag{5}$$

is the total communication cost. Therefore, minimizing $\overline{h}$ is the same as minimizing C.

Fig. 2 shows the cost functions of some node pairs in a ShuffleNet. The vertical axis is the cost and the horizontal axis is the hop count x or y. Each line represents the communication cost between a node pair as a function of its distance, and there are a total of $N(N-1)/2$ lines. The bullet on each line represents the relative position of the node pair in the ShuffleNet and the communication cost. The node placement problem can be restated as to find an optimal placement of the bullets on the cost function lines (one per line) such that the total cost C is minimized under the ShuffleNet topological constraints.

From Fig. 2, we can see that it is more crucial to place node pairs with cost functions having large gradients, or $|m_{ij}|$'s, than small gradients. If a node pair with large $|m_{ij}|$ can be appropriately placed such that its communication cost is minimized, a smaller overall traffic weighted mean internodal distance will be obtained. This principle leads to a new node placement algorithm called the gradient algorithm.

IV. The Gradient Algorithm

The gradient algorithm is a greedy algorithm that places node pairs one by one, starting from the one with the largest gradient onto the first available position that minimizes the communication cost between the node pair. Suppose a node pair with $\lambda_{ij} > \lambda_{ji}$ is to be placed. To decide whether node j is to be placed in a k-loop position or a $2k$-loop position, we simply check if $c_{ij}(x) < d_{ij}(y)$. Using (3) and (4), this reduces to checking

$$x < y + \frac{k\lambda_{ji}}{(\lambda_{ij} - \lambda_{ji})}. \tag{6}$$

In the following, the node placement algorithm is described:

Input: $\boldsymbol{A} = [\lambda_{ij}]$ and an $N = kp^k$, empty ShuffleNet
Output: A node placement pattern

---

1. For $i = 1$ to N do
   for $j = 1$ to N do
   $m_{ij} = \lambda_{ij} - \lambda_{ji}$.
   if $m_{ij} < 0$, $m_{ij} = -1$.
   if $i = j$, $m_{ij} = -1$.
2. Find the sorted list S of $\{m_{ij}\}$ such that the first element is the maximum.
3. While not all $m_{ij} = -1$, choose the first nonnegative entry and denote it by m_{ij},
3.1 If both nodes i and j are placed,
   set $m_{ij} = -1$; Goto Step 3.
3.2 Else if node j can be placed at 1-hop from node i and they are in the k-loop of each other,
   place node i (if it is not placed) and node j (if it is not placed); $m_{ij} = -1$.
3.3 Else if both nodes have not been placed, or only node i is placed,
   if both nodes have not been placed,
   place node i at the first available position.
   find an empty position in the k-loop of node i s.t. it has the shortest distance $x\ from$ node i.
   find an empty position in a $2k$-loop of node i s.t. it has the shortest distance $y\ from$ node i.
   if $x < y + k\lambda_{ji}/(\lambda_{ij} - \lambda_{ji})$,
   place node j at the found position in the k-loop; $m_{ij} = -1$.
   else
   place node j at the found position in the $2k$-loop; $m_{ij} = -1$.
3.4 Else if only node j is placed,
   find an empty position in the k-loop of node j s.t.
   it has the shortest distance $x\ to$ node j.
   find an empty position in a $2k$-loop of node j s.t.
   it has the shortest distance $y\ to$ node j.
   if $x < y + k\lambda_{ji}/(\lambda_{ij} - \lambda_{ji})$,
   place node i at the found position in the k-loop; $m_{ij} = -1$.
   else
   place node i at the found position in the $2k$-loop; $m_{ij} = -1$.

Step 1 calculates the gradients of the cost functions. Step 2 sorts the gradients into an ordered list S with $O(N^2 \log N)$ operations using, for example, QuickSort [18]. Step 3 is a while-loop which excecutes at most N times for placing the N nodes. Consider a worst-case execution of Step 3. Suppose both nodes are not placed. First we need to check at most N positions for the two nodes to be placed in a one-hop separation in their k-loop. Assume that this fails, then node i is

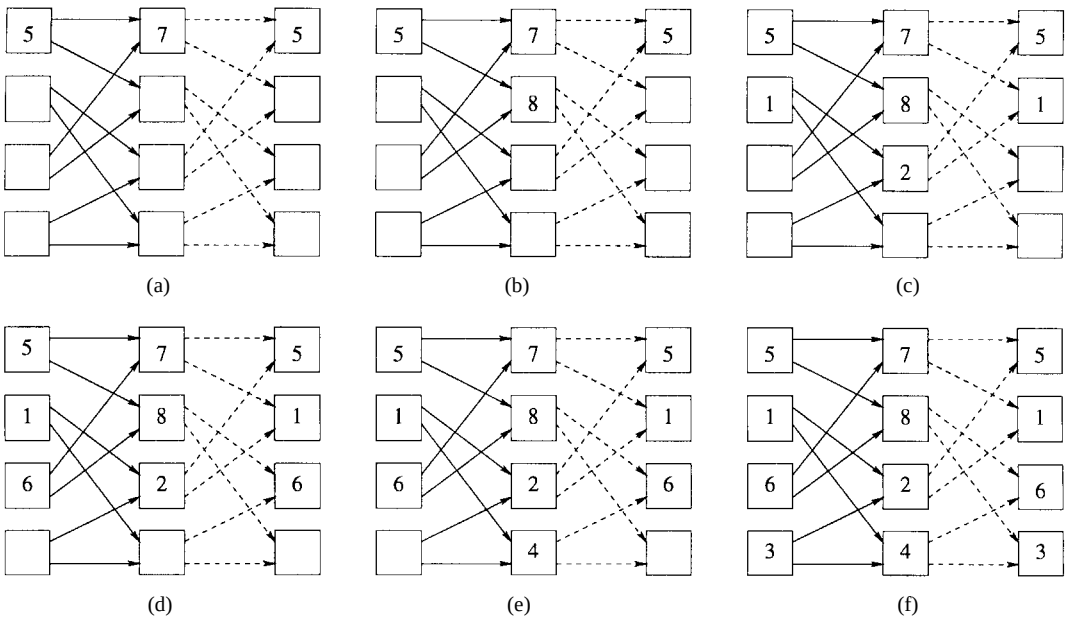

Fig. 3. An example of placing nodes into a (2,2) ShuffleNet using the gradient algorithm. (a) Node 5 to node 7. (b) Node 5 to node 8. (c) Node 1 to node 2.(d) Node 5 to node 6. (e) Node 1 to node 4. (f) Node 1 to node 3.

placed at the first available position. Next we need to check at most $N-1$ positions for node j. A simple calculation is then done for choosing the k-loop or a $2k$-loop position. The worst-case complexity for a single execution of Step 3 is therefore $O(N^2)$. The time complexity of the gradient algorithm is therefore $O(N^3)$.

V. An Example

Consider an example of placing eight nodes, numbered from 1 to 8, into a (2,2) empty ShuffleNet. Let the traffic matrix A be given by

$$A = \begin{bmatrix} 0 & 51.4 & 17.6 & 30.9 & 0 & 0 & 0 & 0 \\ 0.5 & 0 & 0.9 & 0.2 & 0.7 & 0.2 & 0.5 & 0.1 \\ 0.1 & 0.4 & 0 & 0.3 & 0.4 & 1.0 & 0.5 & 0.8 \\ 0.6 & 0.8 & 0.8 & 0 & 0.8 & 0.2 & 0.6 & 0.3 \\ 0 & 0 & 0 & 0 & 0 & 34.7 & 91.7 & 52.0 \\ 0.4 & 0.6 & 0.8 & 0.9 & 0.9 & 0 & 0.9 & 0.7 \\ 0.8 & 0.6 & 0.4 & 0.4 & 0.2 & 0.8 & 0 & 0.5 \\ 0.5 & 1.0 & 0.1 & 0.2 & 1.0 & 0.7 & 0.4 & 0 \end{bmatrix}.$$

Following Step 1 of the gradient algorithm, the gradient matrix M can be found

$$M = \begin{bmatrix} -1 & 50.9 & 17.5 & 30.3 & 0 & -1 & -1 & -1 \\ -1 & -1 & 0.5 & -1 & 0.7 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 0.4 & 0.2 & 0.1 & 0.7 \\ -1 & 0.6 & 0.5 & -1 & 0.8 & -1 & 0.2 & 0.1 \\ 0 & -1 & -1 & -1 & -1 & 33.8 & 91.5 & 51.0 \\ 0.4 & 0.4 & -1 & 0.7 & -1 & -1 & 0 & 0 \\ 0.8 & 0.1 & -1 & -1 & -1 & -1 & -1 & 0.1 \\ 0.5 & 0.9 & -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}.$$

Starting from the node pair whose cost function has the largest gradient $m_{57} = 91.5$, i.e. from node 5 to node 7, node 7 is placed at one hop downstream of node 5 and both nodes are in the k-loop of each other, as shown in Fig. 3(a). Next we consider the node pair with the second-largest gradient $m_{58} = 51$; the best position for placing node 8 has been occupied by node 7 so we have to consider if it is better to place node 8 in the k-loop with hop count $x = 2$, or to place node 8 in the $2k$-loop with hop count $y = 1$. From the inequality (6), the $2k$-loop position with $y = 1$ is preferred and thus node 8 is placed as shown in Fig. 3(b). Continue this procedure until all nodes have been placed. Each node placement has been shown in Fig. 3.

The weighted mean hop distance $\overline{h}$ for this example is found to be 1.26 by (1). The lower bound is 1.23 using the method to be described in the next section. We can see that the weighted mean hop distance is only 2.4% above the lower bound. If the greedy algorithm [7] is used, $\overline{h} = 1.80$, which is 46.3% above the lower bound.

VI. Two Lower Bounds

A tight lower bound on $\overline{h}$ allows us to check and to assess the quality of the solutions provided by the node placement algorithms. The weighted mean internodal distance $\overline{h}$ from (1) consists of the summation of $N(N-1)$ terms of $h_{ij}\lambda_{ij}$. These $N(N-1)$ terms can be partitioned into N sets, where set i contains the $N-1$ communication costs from node i to all other nodes. It corresponds to the traffic on row i of the traffic matrix A.

Next we show how to construct for set i an optimal node placement pattern—optimal in the sense that the one-way communication cost for set i, denoted H_i, is minimum. A directed binary tree (e.g., Fig. 4) which is a subgraph of ShuffleNet, can be constructed with the root at node i. The arrow on the line indicates the traffic flow direction. The nodes at level h of the binary tree correspond to the nodes which can be accessed by node i in h hops in a corresponding ShuffleNet. Therefore, the number of nodes at level h of the binary tree is equal to p^h for $h < k$ and $(p^k - p^{i-k})$ for $h \geq k$.

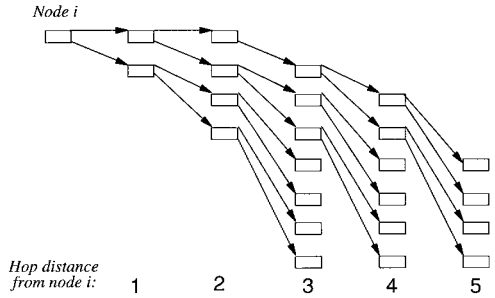

Fig. 4. A binary tree for a (2,3) ShuffleNet considering only the one-way traffic from node i to all other nodes.

Based on the one-way traffic generated from the root node i, a binary tree which minimizes H_i can be found by first ranking all other nodes in descending order of λ_{ij}. Then starting from the first node in the list, nodes are placed one by one onto the tree, starting from the first stage. H_i is minimized because the node with the heaviest traffic from node i is placed at the shortest hop distance from the root.

Theorem 1: A lower bound h_L on $\overline{h}$ is

$$h_L = \frac{1}{\Lambda}\sum_{i=1}^{N} H_i.$$

Proof: Suppose an optimal node placement pattern is found such that the weighted mean internodal distance $\overline{h}$ is minimized. Let H_i' be the total cost of the set i elements in the optimal placement pattern. Since H_i is obtained from the optimal construction of a node placement pattern based on the one-way traffic, less constraints are imposed on the construction as compared to the *overall* optimal pattern. Therefore, $H_i \le H_i'$ is always true. Because h_L is obtained by averaging over all H_i's, it is a lower bound on $\overline{h}$.

Following a similar approach, we can partition the $N(N-1)$ terms from (1) into N sets, where set i contains the $N-1$ communication costs from all other nodes to node i. Here each set corresponds to a column in the traffic matrix A. Similarly, an optimal binary tree can be constructed from set i with node i as the root. The difference is that the node with the heaviest traffic to node i is placed as close to the root as possible. From Theorem 1, a lower bound based on columnwise partitioning of the traffic matrix can be obtained.

VII. Performance Evaluation

In this section we evaluate the performance of the gradient algorithm and compare it to the lower bound and to those algorithms in [7] for both uniformly and nonuniformly distributed internodal traffic. The same numerical examples used in [7] are adopted here. For uniformly distributed random traffic, the traffic rate from any node to any other nodes is a uniformly distributed random number between 0–1. For nonuniformly distributed random traffic, k nodes are configured as database servers—each serves a disjoint set of $p^k - 1$ nodes. Traffic rate from a nonserver node to any other nodes is a uniformly distributed random number between 0–1. The traffic rate from a server node to any other nodes it serves is a uniformly distributed random number between 0 and γ, where γ is a given traffic skew factor.

TABLE I
Weighted Mean Hop Distance Under Uniformly Distributed Random Traffic

N	(p,k)	Random	BM	Greedy	Gradient	h_L	%
8	(2,2)	2.00	1.80-1.92	1.98	1.98	1.64	20.7
24	(2,3)	3.25	3.08-3.20	3.23	3.23	2.63	22.8
64	(2,4)	4.63	4.46-4.59	4.62	4.61	3.78	22.0

TABLE II
Weighted Mean Hop Distance Under Nonuniformly Distributed Random Traffic with $\gamma = 10$

N	(p,k)	Random	BM	Greedy	Gradient	h_L	%
8	(2,2)	2.03	1.59-1.78	1.90	1.52	1.37	10.9
24	(2,3)	3.26	2.78-2.97	3.20	2.79	2.31	20.8
64	(2,4)	4.63	4.38-4.61	4.59	4.31	3.56	21.1

TABLE III
Weighted Mean Hop Distance Under Nonuniformly Distributed Random Traffic with $\gamma = 50$

N	(p,k)	Random	BM	Greedy	Gradient	h_L	%
8	(2,2)	2.03	1.36-1.59	1.85	1.27	1.22	4.1
24	(2,3)	3.26	2.44-2.83	3.12	2.18	1.91	14.1
64	(2,4)	4.63	4.02-4.54	4.51	3.59	3.06	17.3

The four algorithms proposed in [7] are: greedy, local, global, and iterative. The time complexities for the first three algorithms are $\max[O(N^2p), O(N^2\log N)]$, $O(N^{2p}p^{k-1})$, and $O(N^3)$, respectively, and that for each iteration of the iterative algorithm is $O(N^4)$.

Table I summarizes the weighted mean hop distance for an N-node ShuffleNet under uniformly distributed random traffic with $N = 8$, 24, and 64. Each value in the table is obtained by averaging over 100 experiments, where each experiment is carried out with a different (randomly generated) uniformly distributed traffic matrix. The column under "Random" shows the results obtained by randomly placing nodes into the ShuffleNet. These results match the theoretical average hop distance shown in (2). Column "BM" is the range of the mean hop distance obtained from the four algorithms in [7]. Generally speaking, the greedy algorithm has the poorest performance among the four. For nonuniformly distributed traffic, the iterative algorithm has the best performance.

Note that column "Greedy" shows the results of the greedy algorithm of our implementation. There are slight discrepencies with that quoted in [7] because the algorithm stated there is not precise enough to allow exact reproduction of the results. The results for the gradient algorithm are obtained under conditions similar to the greedy algorithm, i.e., using the same random number generators and seeds. The column under "h_L" shows the larger of the column-based and the row-based lower bounds. The last column in the table is the percentage of $\overline{h}$ found by the gradient algorithm above the corresponding lower bound.

From Table I, we see that under uniformly random traffic distribution, all algorithms including the random placement algorithm give similar performance, with the global algorithm having a slight advantage. The $\overline{h}$ obtained by the gradient algorithm is at most 22.8% above the lower bound.

TABLE IV
WEIGHTED MEAN HOP DISTANCE UNDER NONUNIFORMLY DISTRIBUTED RANDOM TRAFFIC WITH $\gamma = 100$

N	(p,k)	Random	BM	Greedy	Gradient	h_L	%
8	(2,2)	2.03	1.32-1.57	1.84	1.21	1.19	1.7
24	(2,3)	3.26	2.32-2.76	3.10	1.98	1.77	11.9
64	(2,4)	4.63	3.72-4.49	4.47	3.19	2.78	14.7

Under nonuniformly random traffic distribution with skew factor $\gamma = 10$, 50, and 100, Tables II–IV show that the gradient algorithm consistently gives shorter $\overline{h}$ than the other algorithms. Moreover, the percentage improvement increases with the network size N and skew factor γ. From the last column, we can see that the percentage difference between $\overline{h}$ of the gradient algorithm and h_L is at most 21.1% for $\gamma = 10$, 17.3% for $\gamma = 50$, and 14.7% for $\gamma = 100$, showing that the gradient algorithm works better for highly skewed traffic.

VIII. CONCLUDING REMARKS

In conclusion, we would like to discuss several related issues with further investigation. In addition to the communication cost between the two nodes under consideration, other information, such as the total communication cost between all placed nodes and the node currently to be placed, can be used for further improvement of the gradient algorithm. The lower bounds proposed in this paper only consider the one-way traffic generated by or destined to a particular node. It is worthwhile to look into ways of making better use of the traffic information for obtaining a tighter lower bound. The gradient algorithm and the lower bounds can possibly be generalized to other regular topologies, e.g., Toroid.

REFERENCES

[1] A. S. Acampora and M. J. Karol, "An overview of lightwave packet networks," *IEEE Network Mag.*, vol. 3, pp. 29–41, Jan. 1989.

[2] M. Gerla, M. Kovacevic, and J. Bannister, "Multilevel optical networks," *Proc. IEEE ICC'92*, Nov. 1992, pp. 1168–1172.

[3] P. P. To, T. S. Yum, and Y. W. Leung, "Multistar implementation of expandable shufflenets," *IEEE/ACM Trans. Networking*, vol. 2, pp. 345–351, Aug. 1994.

[4] N. F. Maxemchuk, "Regular mesh topologies in local and metropolitan area networks," *AT&T Tech. J.*, vol. 64, no. 7, pp. 1659–1685, Sept. 1985.

[5] S. Banerjee, B. Mukherjee, and D. Sarkar, "Heuristic algorithms for constructiong optimized structures of linear multihop lightwave," *IEEE Trans. Commun.*, vol. 42, pp. 1811–1826, Feb./Mar./Apr. 1994.

[6] S. Banerjee and B. Mukherjee, "The photonic ring—Algorithms for near-optimal node arrangements," presented at the Fifth MAN Workshop, Taormina, Italy, May 1992.

[7] ———, "Algorithms for optimized node placement in shufflenet based multihop lightwave networks," in *Proc. IEEE INFOCOM'93*, 1993, pp. 557–564.

[8] K. C. Lee and V. O. K. Li, "Routing for all-optical networks using wavelengths outside erbium-doped fiber amplifier bandwidth," *Proc. IEEE INFOCOM'94*, vol. 2, Toronto, Ont., Canada, 1994, pp. 946–953.

[9] K. N. Sivarajan and R. Ramaswami, "Lightwave networks based on de bruijin graphs," *IEEE/ACM Trans. Networking*, vol. 2, pp. 70–79, Feb. 1994.

[10] T. Y. Feng, "Data manipulating functions in parallel processors and their implementations," *IEEE Trans. Comput.*, vol. C-23, pp. 309–318, Mar. 1974.

[11] D. H. Lawrie, "Access and alignment of data in an array processor," *IEEE Trans. Comput.*, vol. C-30, pp. 1145–1155, Dec. 1975.

[12] M. Kumar and J. R. Jump, "Performance of unbuffered shuffle-exchange networks," *IEEE Trans. Comput.*, vol. C-35, pp. 573–578, June 1986.

[13] S. C. Liew and T. T. Lee, "$n \log n$ dual shuffle-exchange network with error-correcting routing," *IEEE Trans. Commun.*, vol. 42, pp. 754–765, Feb./Mar./Apr. 1994.

[14] A. S. Acampora, "A multichannel multihop local lightwave networks," *Proc. IEEE GLOBECOM'87*, Tokyo, Japan, 1987, pp. 1459–1467.

[15] E. Ayanoglu, "Signal flow graphs for path enumeration and deflection routing analysis in multihop networks," in *Proc. IEEE GLOBECOM'89*, Dallas, TX, Nov. 1989, pp. 1022–1029.

[16] K. Sivarajan and R. Ramaswami, "Multihop lightwave networks based on de bruijn graphs," in *Proc. IEEE INFOCOM'91*, Bal Harbour, FL, Apr. 1991, pp. 1001–1011.

[17] M. G. Hluchyj and M. J. Karol, "Shufflenet: An application of generalized perfect shuffles to multihop lightwave networks," in *Proc. IEEE INFOCOM'88*, 1988, pp. 379–390.

[18] A. M. Tenenbaum, Y. Langsam, and M. J. Augenstein, *Data Structures Using C*. Englewood Cliffs, NJ: Prentice-Hall, 1990, p. 330.

Kwan Lawrence Yeung (S'93–M'95) received the B.Eng. and Ph.D degrees in information engineering from The Chinese University of Hong Kong, Shatin, Hong Kong, in 1992 and 1995, respectively.

He joined City University of Hong Kong as an Assistant Professor in May 1995. His research interests include personal and mobile communication systems, high-speed networking, broad-band packet switches, and lightwave networks.

Tak-Shing Peter Yum (S'76–A'78–SM'86) received the B.Sc., M.Sc., M.Ph., and Ph.D. degrees from Columbia University, NY, in 1974, 1975, 1977, and 1978, respectively.

He was with Bell Telephone Laboratories for two and a half years and then with National Chiao Tung University, Taiwan, for two years before joining The Chinese University of Hong Kong, Shatin, Hong Kong, in 1982. He has published original research on packet-switched networks with contributions in routing algorithms, buffer management, deadlock detection algorithms, message resequencing analysis, and multiaccess protocols. In recent years he has branched out to work on the design and analysis of cellular networks, lightwave networks, and video distribution networks. He believes that the next challenge is designing an intelligent network that can accommodate the needs of individual customers.