



ELSEVIER

Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

Computer Communications 26 (2003) 975–985

computer
communications

www.elsevier.com/locate/comcom

Optimal wavelength converter placement in arbitrary topology wavelength-routed networks[☆]

Sashisekaran Thiagarajan*, Arun K. Somani

Dependable Computing and Networking Laboratory, Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, 50010 USA

Received 4 March 2002; revised 15 July 2002; accepted 12 September 2002

Abstract

This paper describes an algorithm for optimally placing a given number of wavelength converters in all-optical networks with arbitrary topologies. We first introduce the simple network model upon which the algorithm is based. We provide a formulation of the overall network blocking probability when a given number of nodes in the network, is provided with full wavelength conversion capability. We then present our optimal converter placement algorithm and illustrate its working using a simple example. The savings offered by our algorithm in the calculation of blocking performance are analyzed. The benefits of our optimal converter placement algorithm are studied through network examples such as the path, NSFnet and the mesh-torus. Finally, some heuristics for converter placement are presented.

© 2003 Elsevier Science B.V. All rights reserved.

Keywords: Wavelength-routing; Wavelength converters; Blocking probability; Converter placement

1. Introduction

Wavelength division multiplexing (WDM) networks using wavelength-routing have emerged as a popular architectural solution for wide area networks (WANs) [1]. Unlike broadcast-and-select networks, wavelength-routing networks offer advantages such as wavelength reuse and scalability. In addition, such networks do not suffer from splitting losses and offer higher reliability as they can be reconfigured to provide alternate paths in the event of failures. The wavelength-routing network consists of *wavelength crossconnect* nodes interconnected by optic fiber links. The network provides all-optical transport of data between pairs of nodes by establishing circuit-switched connections called *lightpaths*. A lightpath usually consists of a high-bandwidth pipe, capable of gigabit per second transmission speeds. Each fiber optic link can support many lightpaths by allocating each lightpath to a wavelength.

However, no two lightpaths can be assigned the same wavelength on any given link.

Some of the key elements required to implement wavelength crossconnect nodes are passive wavelength multiplexers, demultiplexers, and switches. These crossconnect nodes can also be equipped with wavelength converters (WCs) to reduce the blocking in the network. WCs can be either full-range or limited-range with respect to their ability to converting the incoming wavelengths to outgoing wavelengths. Full-range WCs (FWCs) can convert an incoming wavelength to any outgoing wavelength. Limited-range WCs (LWCs) can convert an incoming wavelength to only a subset of outgoing wavelengths. When a crossconnect node is equipped with as many FWCs as the number of outgoing wavelengths, it is said to possess *full wavelength conversion* capability. A node equipped with LWCs is said to have *limited wavelength conversion* capability [10]. Since WCs are likely to remain costly devices, the cost of equipping all the nodes in the network with *full conversion* capability is high. Hence, a network in which some nodes are equipped with full conversion capability is more practical. Such a network is referred to as a *sparse wavelength conversion* network. Other wavelength conversion scenarios such as having a fewer number of FWCs per node [3] or equipping some (or all) of the networks nodes with LWCs are also viable alternatives

[☆] An early version of this paper appeared in Proceedings IEEE INFOCOM'99 [12].

* Corresponding author. Address: CIENA Corporation, 920 Elkridge Landing Road, Linthicum 21090, USA. Tel.: +1-410-865-8590; fax: +1-410-865-8905.

E-mail addresses: tsashi@ieee.org (S. Thiagarajan), arun@iastate.edu (A.K. Somani).

[14–16]. However, in this paper, we are concerned about deciding which of the nodes in a sparse conversion network should be equipped with full wavelength conversion capabilities so as to optimize the blocking performance. Specifically, given the number of nodes with full wavelength conversion capability, the traffic demand, the network topology and the number of available wavelengths, the goal is to find the node locations which can be equipped with full wavelength conversion so as to obtain optimum blocking performance. Henceforth, the phrase ‘placing a wavelength converter at a node’ is defined as providing the node with full wavelength conversion capability. Such a node is referred to as a *wavelength converter node*.

1.1. Background

In a network without wavelength converter nodes, it is required that data for a call arriving at the input port of a node on one wavelength has to be switched to a output port of the node at the same wavelength. To satisfy such a connection request for a call, it is necessary that the connection path, which is set up from the source to the destination has a free wavelength on all its links. This requirement is called the *wavelength continuity constraint* and results in increasing the probability of a call being blocked. On the other hand, it has been shown in Ref. [3] that WCs improve the performance of wavelength-routed networks in terms of blocking probability. The benefits of using WCs in wavelength-routed networks has been studied in Refs. [4–9].

Ramaswami and Sivarajan [4] showed that use of WCs resulted in a 10–40% increase in the amount of wavelength reuse. In addition, they also established lower bounds on the blocking probability for a network using any routing and wavelength assignment algorithm. In Ref. [5], it was shown that wavelength conversion can improve performance in large mesh networks where a path consists of many hops. Barry and Humblet [7] also studied the effects of path length, switch size and the hop number on the blocking probability in networks with and without WCs. Subramanian et al. [8] studied sparse wavelength conversion and its effects on the blocking performance. They showed that the performance of the network improved as the conversion density (probability that a node in the network is capable of full wavelength conversion) was increased from 0 to 1. But, they also found that the rate of improvement decreased with increasing conversion density. This means that it may not be necessary for all the nodes to have WCs for obtaining sufficiently high performance. On the other hand, they pointed out that it was more important to perform a detailed analysis of a given network topology in determining the number and node placement of the WCs.

1.2. Converter placement

The problem of WC placement at the network nodes in sparse wavelength conversion networks was first considered

in Ref. [11]. Optimal solutions (based on dynamic programming) for the path, bus and ring topologies were proposed for both uniform and non-uniform traffics. It was shown that considerable gains in blocking performance could be obtained when optimal placement was used compared to a random placement. In Ref. [2], a WC placement heuristic was presented which places WCs at nodes with highest average output link congestion. While the heuristic gives almost-optimal results for the NSFNET example, due to the sequential placement of WCs, it will not give near-optimal results for the simple case of a path topology with uniform traffic. In Ref. [13], three algorithms of linear complexity were presented to obtain near-optimal solutions for a path topology. They proved that optimal placement for end-to-end blocking probability is obtained when the segments on the path, have equal blocking probabilities. However, optimal placement was not always obtained since it was not always possible to divide the path into segments with equal blocking probability. Venugopal et al. [19] proposed a heuristic to place limited-range WCs based on node congestion, length of the lightpaths and nodes where conversion is high. Ref. [17] proposed a solution for obtaining the best placement of converters on a path. Ref. [18] also dealt with heuristics for full-range WC placement in networks of arbitrary topology and with arbitrary traffic patterns. In this paper, we develop an optimal solution to the converter placement problem for arbitrary network topologies by using the traffic model of Ref. [11]. The optimal solution provides the best performance so that the overall network blocking probability is minimized. Though computationally intensive, we show that the algorithm is more efficient than exhaustively searching all combinations of WC placements. We use the wavelength independence model given in Ref. [7] to evaluate the end-to-end blocking probability of a call.

The organization of the paper is as follows. In Section 2, we describe the network model on which our algorithm is based. In Section 3, we present the optimal converter placement algorithm. In Section 4, we analyze the performance and explain the improvement in performance over an exhaustive search algorithm. The benefits of our WC placement algorithm are studied by analyzing some network examples such as the path, NSFnet and the mesh-torus in Section 5. We also provide some simple heuristics for converter placement in Section 6. We provide our conclusions in Section 7.

2. Network model

We model the network using a directed graph $G = (V, L)$ where V , the vertices in the graph, represents the set of network nodes and L , the directed edges in the graph, represent the set of unidirectional optic fiber links in the network. Let the nodes be numbered $1, 2, \dots, N$ and let l_{ij} represents the directed link from node i to node j . There are

F wavelengths on each link and each call requires a full wavelength on each link it traverses. Each call uses a prespecified path—in this case, the shortest path—based upon hop count. All the shortest paths in the network are assumed to satisfy the *optimality principle*. This principle states that if a node x is on the optimal shortest path from node y to node z then the optimal shortest path from node x to node z follows the same route after node x as that followed by the path from y to z . If the prespecified path is available then a lightpath is established between the source and the destination nodes. If the prespecified path is not available then the call may not be routed through an alternate path and is assumed to be blocked.

Let $A = [\lambda_{ij}]$ be the traffic matrix where λ_{ij} , $i \neq j$ denotes the node-pair load from node i to node j , and $\lambda_{ii} = 0$. Let the link loads per wavelength be ρ_{ij} for link l_{ij} . Specifically, ρ_{ij} is the probability that a given wavelength is occupied by a lightpath on link l_{ij} . The link loads per wavelength per link can be obtained from the node-pair loads by

$$\rho_{ij} = \frac{\sum \lambda_{sd}}{F} \quad \forall \text{ nodes } s, d \text{ whose paths include link } l_{ij} \quad (1)$$

such that the path from s to d includes link l_{ij} provided λ_{sd} is small such that $\rho_{ij} < 1$. We assume that the load on a given wavelength on a link is statistically independent of the link loads of other wavelengths on that link and the loads on other links.

2.1. Blocking probability analysis of a single path

Let the number of converter nodes to be placed be K . Let $C = \{c(1), c(2), \dots, c(K)\}$ be the *converter placement vector* such that $1 \leq c(i) < c(i+1) \leq N, 1 \leq i < K$. The entries of C denote the placement of converters among the nodes $1, 2, \dots, N$. Now consider the path p of an end-to-end call from a source node s to a destination node d in the network. We define a *segment* to be the set of links on the path between two consecutive converter nodes or between the source (or destination) and a converter node. If the path contains no converter nodes, then it consists of a single segment between the source and destination. Then

$$f(i, j) = 1 - (1 - \{\bar{\rho}_{i_1}, \bar{\rho}_{i_1 i_2} \dots \bar{\rho}_{i_{k-1} i_k}\})^F \quad (2)$$

is the success probability in the segment from node i to node j on the path, where $\bar{\rho}_{xy} = 1 - \rho_{xy}$ for link l_{xy} and i_1, i_2, \dots, i_n are the nodes in the segment between nodes i and j . Let the number of converters placed on the nodes (not including the source and destination nodes) of the path p be k , where $0 \leq k \leq K$. Given the converter placement vector C for the network, we have $d_{sd} = (d(1), d(2), \dots, d(k))$ and $d_{sd} \subset C$, the converter placement vector for the *path* such that the entries of $d(i)$, $1 \leq i \leq k$ correspond only to the node numbers of the path p which have converters on them. This divides the path p into $k+1$ segments. Hence the success probability of the end-to-end call with the converter

placement vector being C is given by

$$S_{sd}(C) = \prod_{i=0}^k f(s_i) \quad (3)$$

where $f(s_i) = f(d(i), d(i+1))$, $1 \leq i \leq k-1$ corresponds to the success probability of the segment s_i between the converter nodes $d(i)$ and $d(i+1)$. $f(s_0)$ corresponds to the success probability of the segment between the source node and converter node $d(1)$ and $f(s_k)$ corresponds to the success probability of the segment between the converter node $d(k)$ and the destination node. The blocking probability for the path is then obtained as

$$P_{sd}(C) = 1 - S_{sd}(C) \quad (4)$$

Using the above formula, we can calculate the blocking probabilities for all the paths in the network. From this, we obtain the blocking performance $\Gamma(C)$ of the network for the converter placement vector C by

$$\Gamma(C) = \frac{\sum_{\forall s,d} \lambda_{sd} P_{sd}(C)}{\sum_{\forall s,d} \lambda_{sd}} \quad (5)$$

The above formula can now be used to calculate the blocking performance and select the best converter placement that gives the minimum blocking performance Γ in the network.

3. Converter placement algorithm

One obvious way to solving the optimal converter placement problem is to use the method of exhaustive placement and blocking performance computation. In this method, given the directed graph G with N nodes and the number of converters K to be placed, we initially obtain the set of all converter placement combinations C_l^K , $1 \leq l \leq \binom{N}{K}$. Here $\binom{N}{K}$ corresponds to the number of combinations of K converters out of N nodes. C_l^K here is the K -converter placement vector and can be further expanded as $C_l^K = (c_l^K(1), c_l^K(2), \dots, c_l^K(K))$ where $c_l^K(x)$, $1 \leq x \leq K$ corresponds to the placement entry on K of the nodes of the network. Next for each converter placement combination, we calculate the corresponding blocking performance according to Eq. (5). We then select that converter placement combination which gives the minimum blocking performance value as the optimal placement. Evidently this method is not an efficient and fast way to solve the problem.

Many factors like the amount of transit traffic at a node, path lengths, distances between converters, distances from the converters to other nodes, and the amount of traffic that needs to be switched at a node affect the optimal solution which makes converter placement in an arbitrary network a hard problem in general. It is also not always possible to make a sequence of stepwise decisions by placing converters one by one on the nodes and ending up with an optimal placement. That is, the optimal placement

combination for $K - 1$ nodes in the network may not be a proper subset of the optimal placement combination for K nodes. This can be illustrated by considering the optimal placement of converters in a bidirectional path network of 10 nodes with uniform link loads. As shown in Table 4, the optimal placement for one converter is either node 5 or node 6. On the other hand, the optimal placement for two converters is at nodes 4 and 7. In this paper, we make use of the exhaustive search method. However, we develop a method to reduce the computational requirements and still be able to find the optimal solution. Our algorithm uses the procedure defined in Sections 3.1 and 3.2.

3.1. Construction of auxiliary graphs

Our converter placement algorithm proceeds in phases. In the first phase, we first construct N auxiliary directed graphs, $G_j = (V, E_j)$ for each node $j \in V$ such that E_j consists of only those edges $e \in L$ which are on those paths from the remaining $N - 1$ nodes to node j . The edges are directed towards the destination node j . We consider only the shortest paths to node j from each node i , $i \in V$ and $i \neq j$. The shortest paths, in addition, satisfy the optimality property. Therefore, each graph G_j is acyclic. The first phase results in a family of auxiliary directed acyclic graphs $G_1, G_2, G_3, \dots, G_N$ for each node $1, 2, \dots, N$ acting as the destination. In each auxiliary graph G_j , there exists at least one or more nodes which have indegree of zero. We refer to these nodes as *primary source* nodes. Only the destination node has outdegree of zero. We collectively refer to the destination node and the primary source nodes as *outer* nodes of the auxiliary graph. All other nodes are referred to as *inner* nodes of the auxiliary graph. Let the number of inner nodes be $|I|$ and the number of outer nodes be $|O|$ so that $|O| + |I| = N$.

3.2. Blocking performance for an auxiliary graph

The second phase of the algorithm uses the simple principle that the blocking probabilities of the paths are not affected if converters are placed on the outer nodes in the auxiliary graph. In this phase, we first obtain all placement combinations C_l^K , $1 \leq l \leq \binom{N}{K}$, on the whole network of N nodes. Each placement combination has an associated overall blocking performance variable $\Gamma(C_l^K)$ which is initialized to zero. Next, for each graph G_j , we obtain the blocking probabilities of the paths to the destination node by placing a subset of converters on the inner nodes of the graph. The idea is that if m converters are placed on the inner nodes and $K - m$ converters are placed on the outer nodes in any placement combination then the blocking probability on paths to node j in the auxiliary graph G_j depends only on the m converters placed on the inner nodes. We exhaustively place all combinations of m converters, where m is varied from a *lower limit* to a *higher limit* (as given in Eq. (7)) and calculate the blocking performances,

$\Gamma_j(E_i^m)$ for G_j , where E_i^m is the i th combination of placing m converters on the inner nodes of the graph G_j . The value of i varies from 1 to $\binom{|I|}{m}$, the number of combinations of m out of $|I|$ inner nodes. E_i^m , also known as the m -converter placement vector, can be further expanded as $E_i^m = (e_i^m(1), e_i^m(2), \dots, e_i^m(m))$, where $e_i^m(x)$, $1 \leq x \leq m$ corresponds to the placement entry of an inner node of the graph. The blocking performance $\Gamma_j(E_i^m)$ for the auxiliary graphs is defined as

$$\Gamma_j(E_i^m) = \frac{\sum_{\forall s, s \neq j} \lambda_{sj} P_{sj}(E_i^m)}{\sum_{\forall s, d} \lambda_{sd}} \tag{6}$$

The division by the sum of *all* node-pair loads is done so that the blocking performances obtained from all the auxiliary graphs can be directly added together to obtain the overall blocking performance of the entire network. Note that blocking performance of each auxiliary graph is calculated by using the blocking probability of only $N - 1$ paths all of which end at a single destination node. The *lower limit* and *higher limit* values for m are obtained as follows:

$$\text{lower limit} = \max\{0, K - |O|\}$$

$$\text{higher limit} = \min\{|I|, K\} \tag{7}$$

Next we obtain all combinations D_n^{K-m} of $K - m$ (m varied from the *higher limit* to the *lower limit*) converters placed only on the outer nodes. Here n varies from 1 to $\binom{|O|}{K-m}$. Now to each combination E_i^m , we append each of the combinations D_n^{K-m} to get K -converter placement combinations C_l^K on the whole network. All the K -converter placement combinations which are obtained from appending each E_i^m to any D_n^{K-m} will have the same blocking performance of $\Gamma_j(E_i^m)$. Hence for each of the C_l^K obtained, the blocking performance $\Gamma_j(E_i^m)$ is simply added to the corresponding overall blocking performance variable of $\Gamma(C_l^K)$. Recall that this can be done because the only converter positions that vary in all these combinations are in the outer nodes which do not affect the blocking probability of these paths. The above procedure provides the blocking performances corresponding to the auxiliary graph G_j for all the $\binom{N}{K}$ converter placement combinations. However, we calculate the blocking probability for only

$$\text{Path}_{\text{calc}}^{G_j} = \sum_{m=\text{lower limit}}^{\text{higher limit}} \binom{|I|}{m} \tag{8}$$

combinations. It is interesting to note that the above procedure partitions the $\binom{N}{K}$ converter combinations in to $\text{Path}_{\text{calc}}^{G_j}$ *equivalence classes* where all of the elements in each of the equivalence classes have the same blocking probability and each equivalence class represents a single inner node combination of converter placement. This procedure is carried out on all auxiliary graphs G_j , $1 \leq j \leq N$. The blocking performance values in each case are calculated and accumulated to the appropriate overall blocking performance variables. We have now obtained

the overall blocking performances for all $\binom{N}{K}$ placement combinations.

The overall blocking performance variable corresponding to a single K -converter placement combination gives us a measure of the call-blocking for all $N(N - 1)$ paths from all sources to all destinations for that particular placement combination. We now select that converter placement combination which results in the minimal blocking performance value as the optimal solution. The pseudocode for the algorithm is given in Table 1. The function

Table 1
Algorithm

```

begin
  for each node  $j \in V$ ,
    Construct_Auxiliary_Graph( $G_j$ ),
    //  $G_j = (V, E_j)$  is constructed such that  $E_j$  consists
    // of only those edges that are on those shortest paths
    // from all nodes  $i, i \in V$  and  $i \neq j$ .
  end for
   $\mathcal{C}^K = \text{Get\_All\_Combinations}(N, K)$ 
  //  $\mathcal{C}^K = \{C_1^K, C_2^K, \dots\}$  is the set of all combinations
  // of  $K$  converters on the set of  $N$  nodes.
  for each  $C_p^K$  in  $\mathcal{C}^K$ 
     $\Gamma(C_p^K) = 0$ 
  end for
  for each graph  $G_j$ ,
     $I \leftarrow \phi, O \leftarrow \phi$ 
    //  $I$  and  $O$  denote the set of inner and outer nodes
    // respectively (initially null) for each node  $x$  in  $G_j$ ,
    if ( $\text{indegree}(x) = 0$ ) OR ( $\text{outdegree}(x) = 0$ ) then
       $O \leftarrow O \cup \{x\}$ 
    else
       $I \leftarrow I \cup \{x\}$ 
    end if
  end for
  //  $|I|$  &  $|O|$  denote the no. of inner & outer nodes
  // respectively
   $\text{lower\_limit} = \max\{0, K - |O|\}$ 
   $\text{higher\_limit} = \min\{|I|, K\}$ 
  for  $m = \text{lower\_limit}$  to  $\text{higher\_limit}$ 
     $\mathcal{E}^m = \text{Get\_All\_Combinations}(I, m)$ 
    //  $\mathcal{E}^m = \{E_1^m, E_2^m, \dots\}$  is the set of all combinations
    // of  $m$  converters on the set of  $I$  nodes.
     $\mathcal{D}^{K-m} = \text{Get\_All\_Combinations}(O, K - m)$ 
    //  $\mathcal{D}^{K-m} = \{D_1^{K-m}, D_2^{K-m}, \dots\}$  is the set
    // of all combinations of  $K - m$  converters
    // on the set of  $O$  nodes.
    for each  $E_i^m \in \mathcal{E}^m$ 
       $\Gamma_j(E_i^m) = \text{Get\_Blocking\_Performance}(E_i^m)$ 
      for each  $D_n^{K-m} \in \mathcal{D}^{K-m}$ 
         $C_{temp}^K = \text{Combination}(E_i^m, D_n^{K-m})$ 
        // The above function denotes the union of the
        // combination of nodes in  $E_i^m$  and  $D_n^{K-m}$ 
         $\Gamma(C_{temp}^K) = \Gamma(C_{temp}^K) + \Gamma_j(E_i^m)$ 
      end for
    end for
  end for
  end for
   $C_{opt} = C_n^K$  such that  $\Gamma(C_n^K)$  is  $\text{Min}_{\forall p} \{\Gamma(C_p^K)\}$ 
  //  $C_{opt}$  denotes the optimal placement combination.
end.

```

Get_All_Combinations(I, m) obtains the set of all combinations of m converters on the set of I nodes. The function **Get_Blocking_Performance**(E_i^m) gives the blocking performance of the auxiliary graph G_j for the $N - 1$ paths to the destination node j .

3.3. Example

We will illustrate the working of the algorithm using the following example. We consider five WDM crossconnect nodes interconnected by point-to-point bidirectional fiber links in an arbitrary mesh topology as shown in Fig. 1(a). We also assume that the node traffic is uniform and all the node-pair loads from each of the nodes to each other is equal to 0.1. The shortest paths information available is assumed to be as shown in Table 2. An auxiliary graph G_E formed with only those edges which are on the shortest paths from nodes A, B, C, and D to E, is shown in Fig. 1(f). Similar

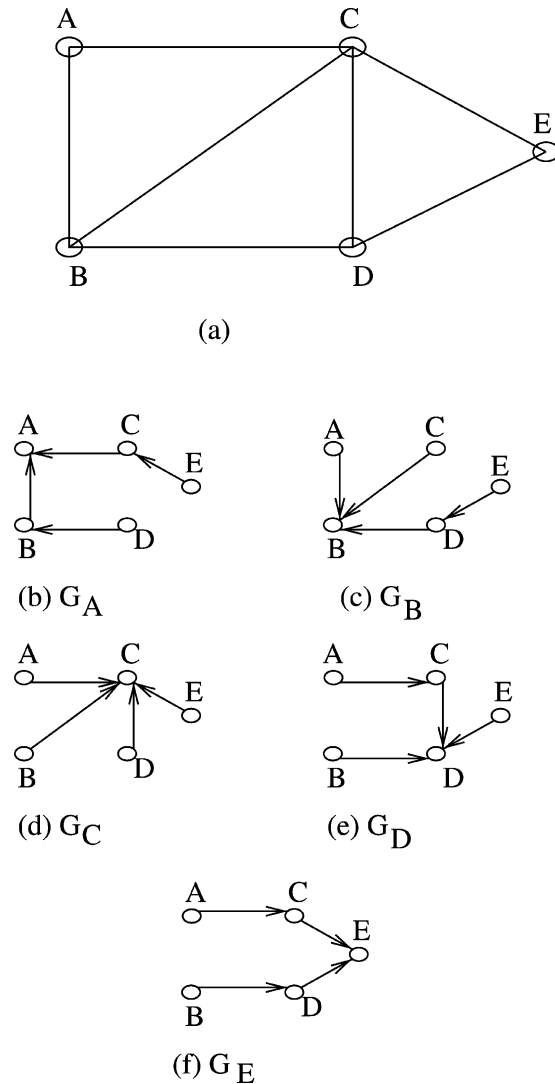


Fig. 1. WDM network with five nodes and its auxiliary graphs G_A , G_B , G_C , G_D and G_E with destination nodes A, B, C, D and E, respectively.

Table 2
Shortest path information for network shown in Fig. 1

| | |
|--|--|
| From B to A: $B \rightarrow A$ | From D to C: $D \rightarrow C$ |
| From C to A: $C \rightarrow A$ | From E to C: $E \rightarrow C$ |
| From D to A: $D \rightarrow B \rightarrow A$ | From A to D: $A \rightarrow C \rightarrow D$ |
| From E to A: $E \rightarrow C \rightarrow A$ | From B to D: $B \rightarrow D$ |
| From A to B: $A \rightarrow B$ | From C to D: $C \rightarrow D$ |
| From C to B: $C \rightarrow B$ | From E to D: $E \rightarrow D$ |
| From D to B: $D \rightarrow B$ | From A to E: $A \rightarrow C \rightarrow E$ |
| From E to B: $E \rightarrow D \rightarrow B$ | From B to E: $B \rightarrow D \rightarrow E$ |
| From A to C: $A \rightarrow C$ | From C to E: $C \rightarrow E$ |
| From B to C: $B \rightarrow C$ | From D to E: $D \rightarrow E$ |

auxiliary graphs obtained for nodes A, B, C and D acting as the destination are also shown in Fig. 1.

In auxiliary graph G_E , nodes C and D are the inner nodes and nodes A, B, and E are the outer nodes. Let us assume that the number of converters to be placed is two. Hence the lower and upper limits for m are 0 and 2, respectively. The blocking performance values have to be calculated four times, once for $m = 0$, twice for $m = 1$ (one converter placed on either node C or D), and once for $m = 2$. Thus $E^0 = \{(\phi)\}$, $E^1 = \{(C), (D)\}$, and $E^2 = \{(C, D)\}$ and $D^{K-2} = \{(\phi)\}$, $D^{K-1} = \{(A), (B), (E)\}$ and $D^{K-0} = \{(A, B), (A, E), (B, E)\}$. The blocking performance obtained when $m = 0$ is assigned to converter placement combinations (A,B), (A,E), and (B,E). The blocking performance obtained when $m = 1$ and node C has a converter is assigned to converter placement combinations (A,C), (B,C), and (E,C). The blocking performance value when $m = 1$ and node D has a converter is assigned to converter placement combinations (A,D), (B,D), and (E,D). When $m = 2$, there is only one placement combination of (C,D). The G_E column in Table 3 gives the blocking performance values for the auxiliary graph in Fig. 1(f). This procedure is also carried out for the remaining four auxiliary graphs corresponding to the nodes A, B, C, and D taken as the destinations, respectively. The total blocking performance for the whole network is calculated by adding up the values corresponding to a placement combination for all five graphs. Table 3

gives all the blocking performance values for each placement combination for all auxiliary graphs (G_A , G_B , G_C , G_D and G_E). The overall blocking performance for the network is also shown in Table 3. The placement combination, which gives the minimum overall blocking performance value is taken as the optimal placement combination. In Table 3, we find the optimal placement combination of (C,D) gives the minimum overall blocking performance of 6.84e-04.

4. Algorithm analysis

To show the correctness of our algorithm, note that to calculate the blocking performance for each auxiliary graph G_j , $1 \leq j \leq N$, we calculate the path blocking probability for just $(N - 1)$ paths. We can also easily see that for each auxiliary graph, we obtain $\binom{N}{K}$ placement combinations from the union of each and every one of the placement combinations of m converters on the inner nodes with each and every one of the placement combinations of $K - m$ converters on the outer nodes where the value of m is as given in Eq. (7). Hence our algorithm obtains the path blocking probability of $\binom{N}{K} N(N - 1)$ paths. In the case of finding out the optimal placement and best blocking performance for all possible combinations using the method of exhaustive placement and blocking performance computation, the path blocking probability is also calculated for a total number of $\binom{N}{K} N(N - 1)$ paths, where $\binom{N}{K}$ is the total number of placement combinations and $N(N - 1)$ is the total number of paths in the graph. On the other hand, we save on the calculation of the blocking performance for many of the converter placements, since all converter placement combinations that differ only in the converter positions of outer nodes are assigned the same blocking performance value.

To quantify these savings, we first define the average number of inner nodes, N_I , for the network as $\sum_{j=1}^N N_1^j / N$, where N_1^j is the number of inner nodes of auxiliary graph G_j . Then the average number of outer nodes is N_O and

Table 3
Blocking performance for different auxiliary graphs and total blocking performance for the network

| Combination | Blocking performance | | | | | Overall network |
|-------------|----------------------|----------|----------|----------|----------|-----------------|
| | G_A | G_B | G_C | G_D | G_E | |
| (A,B) | 0.000201 | 0.000259 | 0.000069 | 0.000249 | 0.000341 | 0.001119 |
| (A,C) | 0.000264 | 0.000259 | 0.000069 | 0.000109 | 0.000201 | 0.000902 |
| (A,D) | 0.000341 | 0.000119 | 0.000069 | 0.000249 | 0.000264 | 0.001042 |
| (A,E) | 0.000341 | 0.000259 | 0.000069 | 0.000249 | 0.000341 | 0.001259 |
| (B,C) | 0.000124 | 0.000259 | 0.000069 | 0.000109 | 0.000201 | 0.000762 |
| (B,D) | 0.000201 | 0.000119 | 0.000069 | 0.000249 | 0.000264 | 0.000902 |
| (B,E) | 0.000201 | 0.000259 | 0.000069 | 0.000249 | 0.000341 | 0.001119 |
| (C,D) | 0.000264 | 0.000119 | 0.000069 | 0.000109 | 0.000124 | 0.000684 |
| (C,E) | 0.000264 | 0.000259 | 0.000069 | 0.000109 | 0.000201 | 0.000902 |
| (D,E) | 0.000341 | 0.000119 | 0.000069 | 0.000249 | 0.000264 | 0.001042 |

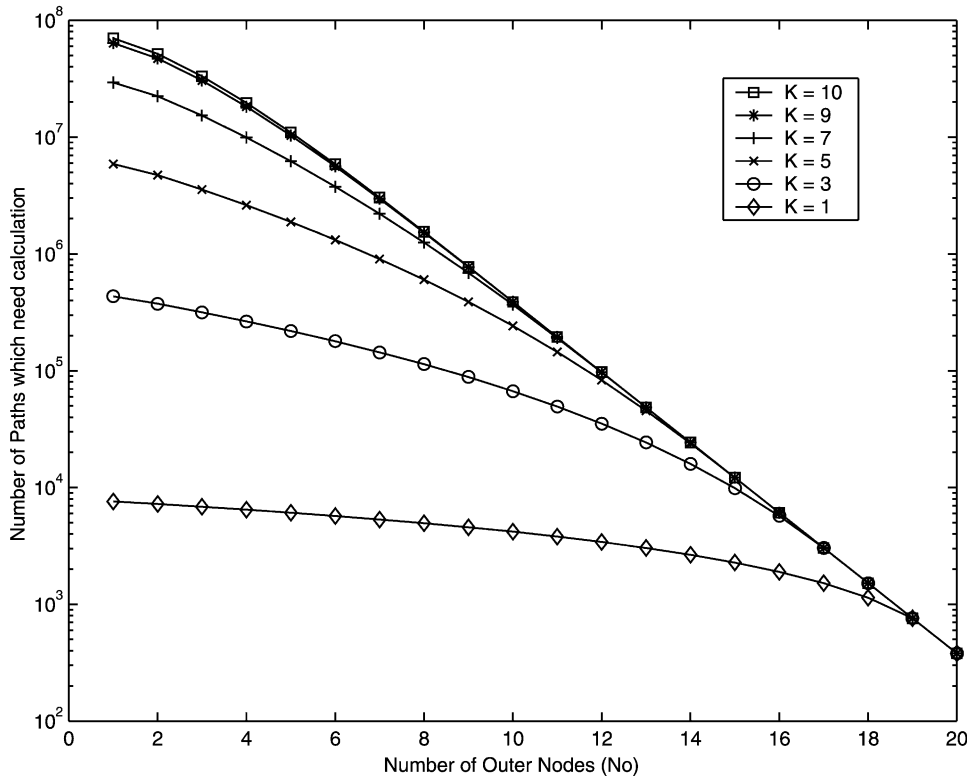


Fig. 2. P_{calc} vs. N_O for $N = 20$ and $1 < K < 20$. Note: P_{calc} value for $N_O = 1$ corresponds to the case when the exhaustive algorithm is used. $N_O \geq 2$ corresponds to P_{calc} values when our algorithm is used.

$N_O = N - N_I$ where N is the total number of nodes in the graph. It must be mentioned that there are at least two outer nodes in every auxiliary graph. This is because each auxiliary graph is a directed acyclic graph and there is one node with outdegree zero, i.e. the destination node and at least one node with indegree zero, a primary source node. Also as mentioned before, the blocking performance for each of the auxiliary graphs is calculated by computing the blocking probability of just $(N - 1)$ paths. Hence the number of paths, P_{calc} for which the blocking probabilities need to be calculated are

$$P_{\text{calc}} = (N - 1) \sum_{j=1}^N \sum_{m=\text{lower limit}}^{\text{higher limit}} \binom{N_I^j}{m} \quad (9)$$

where the *lower limit* and *higher limit* variables for varying m are obtained according to the conditions specified in Section 3.1. The first term, $(N - 1)$ denotes the number of paths present in each auxiliary graph G_j for which the blocking probability has to be calculated, and the last summation term denotes the total number of placement combinations for which the blocking performance of the auxiliary graph needs to be calculated.

Assuming the following arbitrary values for a network: $N = 10$, $N_O^j = 4$, $N_I^j = 6$, for all G_j , and $K = 3$, an exhaustive search for the optimal placement needs $\binom{10}{3} \cdot 10(10 - 1) = 10,800$ path calculations of the blocking probability. On

the other hand, using the algorithm we need $10(10 - 1) \cdot (1 + 7 + 21 + 35) = 5760$ path calculations of the blocking probability to get the optimal placement. We plot P_{calc} as a function of the number of outer nodes ($N_O \geq 2$) for various K for in Fig. 2. The value for $N_O = 1$ corresponds to the case when the exhaustive algorithm is used. We find that for all K , P_{calc} is lower than the total number of paths which need to be calculated for the exhaustive algorithm. As the number of outer nodes increases, the P_{calc} decreases rapidly. Hence the saving in calculation of blocking probability offered by the optimal algorithm increases as the number of outer nodes increases. In general, an increase in network connectivity or a decrease in the network diameter usually results in more

Table 4

Optimal placement in a bidirectional path of 10 nodes with uniform link loads $\rho = 0.05$ and $F = 3$

| K | Optimal placement | Blocking probability | Algorithm efficiency (%) |
|-----|------------------------------------|----------------------|--------------------------|
| 1 | (5) or (6) | 0.002414 | 18 |
| 2 | (4,7) | 0.001490 | 32 |
| 3 | (3,5,7) or (4,6,8) | 0.001035 | 42 |
| 4 | (3,5,6,8) | 0.000789 | 48 |
| 5 | (3,5,6,7,8) or (3,4,5,6,8) | 0.000647 | 50 |
| 6 | (3,4,5,6,7,8) | 0.000505 | 48 |
| 7 | (3,4,5,6,7,8,9) or (2,3,4,5,6,7,8) | 0.000410 | 42 |
| 8 | (2,3,4,5,6,7,8,9) | 0.000315 | 32 |

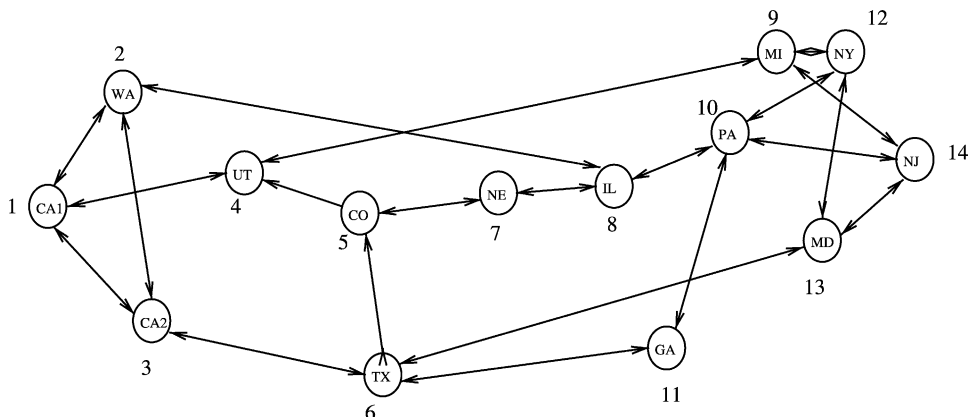


Fig. 3. The NSFnet with 14 nodes and 21 edges.

number of outer nodes in the network and results in an increase in the improvement offered by the algorithm.

5. Numerical results

We will first define the efficiency of our algorithm as the percentage of number of paths for which the blocking probability need not be calculated when our algorithm is used to the total number of paths for all possible converter combinations when the exhaustive method is used:

$$\frac{P_{total} - P_{calc}}{P_{total}} \times 100 \tag{10}$$

where P_{calc} is as given in Eq. (9) and $P_{total} = \binom{N}{K} N(N - 1)$. As illustrated in Section 4, this efficiency depends on the average number of outer nodes of the network and the number of converters to be placed on the network.

Consider the optimal placement of converters in a bidirectional path network of $N = 10$ nodes. Table 4 gives the optimal placement solutions along with their blocking probability and the efficiency of the algorithm. The optimal placement column in the table indicates the nodes where a

converter should be placed. It is obvious that the blocking probability decreases as the number of converters K increases. However, the efficiency of the algorithm increases initially until $K = \lceil N/2 \rceil$ and then decreases again. When $K = 5$, the blocking probability for 50% of the paths need not be calculated. We provide the optimal converter placement results and efficiency of the algorithm for 4×4 mesh-torus and the 14 node NSFNET shown in Fig. 3 in Tables 5 and 6, respectively. The nodes in mesh-torus are numbered from 1 to 16, from left to right and from top to bottom. The node numbers of the NSFNET are shown in Fig. 3. The NSFNET is a WAN developed under the auspices of the National Science Foundation (NSF). NSFNET replaced ARPANET and until 1995, was the main government network linking universities and research facilities. It is observed that the algorithm works more efficiently with the mesh-torus and the NSFNET. Moreover, the efficiency of the algorithm increases to a very high percentage when K increases to $\lceil N/2 \rceil$. This confirms the fact that the algorithm works better than the exhaustive case for networks with high connectivity and high average number of outer nodes.

6. Heuristics

The optimal algorithm reduced the computations to achieve optimal WC placement. However, it is still

Table 5
Optimal placement in a 4×4 mesh-torus of 16 nodes with uniform node-pair loads $\lambda = 0.1$ and $F = 5$. (For $K = 6$ and $K = 7$ there are a total of 20 and 16 optimal placement combinations. Due to space constraints, we just list one for each)

| K | Optimal placement | Blocking probability | Algorithm efficiency (%) |
|---|---|----------------------|--------------------------|
| 1 | (6,7,10,11) | 0.010481 | 50 |
| 2 | (6,11) or (7,10) | 0.008345 | 75.83 |
| 3 | (6,10,11) or (6,7,11) or (7,10,11) or (6,7,10) | 0.007012 | 88.57 |
| 4 | (6,7,10,11) | 0.005678 | 94.56 |
| 5 | (6,7,10,11) and any one of (2,3,5,8,9,12,14,15) | 0.004916 | 97.25 |
| 6 | (5,6,7,8,10,11),... | 0.004154 | 98.41 |
| 7 | (5,6,7,8,10,11,15),... | 0.003392 | 98.88 |
| 8 | (2,6,7,8,9,10,11,15) or (3,5,6,7,10,11,12,14) | 0.002630 | 99.01 |

Table 6
Optimal converter placement in the NSFNET (14 nodes) with uniform node-pair loads $\lambda = 0.1$ and $F = 5$

| K | Optimal placement | Blocking probability | Algorithm efficiency (%) |
|---|-------------------|----------------------|--------------------------|
| 1 | (4) | 0.014554 | 52.04 |
| 2 | (4,6) | 0.010329 | 77.70 |
| 3 | (4,6,8) | 0.007436 | 89.72 |
| 4 | (4,6,8,10) | 0.005742 | 95.05 |
| 5 | (4,5,6,8,10) | 0.004698 | 97.30 |
| 6 | (4,5,6,8,9,10) | 0.003722 | 98.17 |
| 7 | (2,4,5,6,8,9,10) | 0.002777 | 98.40 |

Path-based Index heuristic

```
//P = {p1, p2, ..., pj} be the set of paths, one for each s-d pair.
//wi is the weight of node vi where vi ∈ V, the set of nodes in the graph.
//K is the number of converters.
//lj is the length of path pj.
1. Initialize for all vi ∈ V, wi = 0
2. For each pj ∈ P
3.     For each node vi that lies on path pj, except the source and destination.
4.         wi = wi + 1
5.     EndFor
6. EndFor
7. Sort the nodes according to their weight and place converters on K nodes with the highest weights.
```

Fig. 4. Path-based index heuristic.

expensive. Although converter placement is not optimal when attempted in a sequential manner, sometimes the solutions obtained can be close to the optimal placement. In this section, we provide four heuristics for placement of converters on the nodes of the network. All these heuristics are based on associating a weight to the nodes of the network. Converters are placed on the nodes in the order of increasing weight. The four heuristics vary in the manner, the weight on the nodes are assigned.

1. The path-based index heuristic given in Fig. 4 calculates the weight of the nodes by the number of the prespecified paths passing through it.
2. The second heuristic, called the *path-length-based index* (PLI) heuristic calculates the weight of the nodes as the sum of path lengths of all the source–destination prespecified paths passing through it. The path length is an important parameter in determining the performance of the network. The effects of path length on the blocking probability of WDM networks has been studied in Ref. [7]. The blocking probability of a network with and without WCs tends to increase as path length increases or as the number of hops from source to destination increases. WCs decrease the blocking probability by reducing the average hop lengths and minimizing the congestion on the links. Hence the PLI heuristic uses the path length as the selection parameter. This heuristic can be obtained by replacing line 4 in Fig. 4 with $w_i = w_i + l_j$ where l_j is the length of the path p_j .

Inner Node Index heuristic

```
//wi is the weight of node vi where vi ∈ V, the set of nodes in the graph.
//K is the number of converters.
1. For each node vj ∈ V,
2.     Construct_Auxiliary_Graph(Gj)
3.     For each node ui ∈ Gj,
4.         If ui is an inner node then wi = wi + 1
5. EndFor
6. Sort the nodes according to their weight and place converters on K nodes with highest weights.
```

Fig. 5. Inner node index heuristic.

3. The third heuristic, called the *traffic-path-length product* (TPLP) heuristic calculates the index of a node by using the sum of the products of traffic rate and the path length of each source–destination path that passes through the node. This heuristic can again be obtained by replacing line 4 in Fig. 4 with $w_i = w_i + l_j \times \lambda_{sd}$, where λ_{sd} is the traffic flowing on the path p_j corresponding to the source s and destination d .
4. The final heuristic, called the *inner node index* (IN) is based on our optimal algorithm. After the first phase of creating the auxiliary graphs and separating the nodes of each auxiliary graph into inner and outer nodes, instead of calculating the blocking probabilities, we count the number of times each node is an inner node and assign that as the weight of the node. If the node has a count of zero, this means the node is an outer node in all the auxiliary graphs and can be completely excluded for converter placement. The steps of the heuristic are given in Fig. 5.

The results of the four heuristics on the bidirectional path network, mesh-torus network and the NSFnet are tabulated in Tables 7–9, respectively.

If we need to select K nodes for converter placement, a simple task would be to select the best K nodes according to their weight. Assume $K = 4$, using the results in Table 7 for the bidirectional path, the best four node choices for converter placement in a 10 node path are (4,5,6,7) except in the case of IN heuristic, where all nodes with the same

Table 7
Node weights obtained using heuristics in a bidirectional path of 10 nodes with uniform link loads $\rho = 0.05$

| Node | PI | PLI | TPLP | IN |
|------|----|-----|----------|----|
| 1 | 0 | 0 | 0.000000 | 0 |
| 2 | 16 | 88 | 0.655000 | 9 |
| 3 | 28 | 154 | 1.010000 | 9 |
| 4 | 36 | 198 | 1.215000 | 9 |
| 5 | 40 | 220 | 1.320000 | 9 |
| 6 | 40 | 220 | 1.320000 | 9 |
| 7 | 36 | 198 | 1.215000 | 9 |
| 8 | 28 | 154 | 1.010000 | 9 |
| 9 | 16 | 88 | 0.655000 | 9 |
| 10 | 0 | 0 | 0.000000 | 0 |

weight. For the mesh-torus in Table 8, the best four choices for converter placement are (6,7,10,11) in the first three heuristics and any four of (2,3,6,7,10,11,14,15) in the case of IN heuristic. For the NSFnet the best four choices are (4,6,8,10) for all four heuristics. Another solution would be to first select a set of nodes, say N_H , where $K < N_H$, as the best possible positions for converter placement. Now we can just calculate $\binom{N_H}{K}$ combinations and obtain the optimal placement positions. It should be noticed that the results obtained using the heuristics are consistent with what we obtained using the optimal algorithm. We also note that heuristic IN does not work well for regular topologies. We can also improve the performance of the heuristics for mesh networks by excluding nodes of degree two, which source or sink relatively low traffic compared to other nodes. The reason is that these nodes act more as passthrough nodes for wavelengths and are less likely to be a good choice for converter placement.

Table 8
Node weights obtained using heuristics in a 4×4 mesh-torus of 16 nodes with uniform node-pair loads $\lambda = 0.1$

| Node | PI | PLI | TPLP | IN |
|------|----|-----|----------|----|
| 1 | 9 | 24 | 2.400000 | 3 |
| 2 | 17 | 48 | 4.800000 | 11 |
| 3 | 17 | 48 | 4.800000 | 11 |
| 4 | 9 | 24 | 2.400000 | 3 |
| 5 | 17 | 48 | 4.800000 | 3 |
| 6 | 25 | 72 | 7.200000 | 11 |
| 7 | 25 | 72 | 7.200000 | 11 |
| 8 | 17 | 48 | 4.800000 | 3 |
| 9 | 17 | 48 | 4.800000 | 3 |
| 10 | 25 | 72 | 7.200000 | 11 |
| 11 | 25 | 72 | 7.200000 | 11 |
| 12 | 17 | 48 | 4.800000 | 3 |
| 13 | 9 | 24 | 2.400000 | 3 |
| 14 | 17 | 48 | 4.800000 | 11 |
| 15 | 17 | 48 | 4.800000 | 11 |
| 16 | 9 | 24 | 2.400000 | 3 |

Table 9
Node weights obtained in the NSFNET (14 nodes) with uniform node-pair loads $\lambda = 0.1$

| Node | PI | PLI | TPLP | IN |
|------|----|-----|----------|----|
| 1 | 12 | 32 | 3.200000 | 6 |
| 2 | 14 | 38 | 3.800000 | 6 |
| 3 | 14 | 38 | 3.800000 | 6 |
| 4 | 25 | 69 | 6.900000 | 9 |
| 5 | 20 | 54 | 5.400000 | 6 |
| 6 | 34 | 90 | 9.000000 | 10 |
| 7 | 3 | 7 | 0.700000 | 3 |
| 8 | 21 | 57 | 5.700000 | 7 |
| 9 | 16 | 42 | 4.200000 | 6 |
| 10 | 24 | 62 | 6.200000 | 8 |
| 11 | 3 | 7 | 0.700000 | 3 |
| 12 | 14 | 36 | 3.600000 | 6 |
| 13 | 8 | 20 | 2.000000 | 4 |
| 14 | 0 | 0 | 0.000000 | 0 |

7. Conclusion

In this paper, we presented an algorithm for optimally placing a given number of converters in all-optical networks (AONs) with arbitrary topologies. We also developed a network model to evaluate the blocking performance of such networks. The algorithm was divided into two phases. The first phase was the creation of auxiliary graphs using each node in the network as the destination node. In the second phase, we detailed the procedure for obtaining the blocking performance for all the converter placement combinations by calculating the blocking performance for only some combinations. We used the principle that all converter placement combinations that differ only in the outer nodes can be assigned the same blocking performance value. The algorithm was further explained using a simple network example. The savings in calculation was analyzed and compared with the exhaustive case. It was found that the savings in calculation introduced by our algorithm increases rapidly as the number of outer nodes increases. Since the number of outer nodes is directly dependent on the network connectivity, it was inferred that an increase in network connectivity results in an increase in the improvement offered by the algorithm. The benefits of our algorithm was further confirmed by studying the optimal converter placement on examples such as the path, mesh-torus and the NSFNET. The efficiency of the algorithm was defined and tabulated for each of the networks. It was found that the algorithm works very well (offering efficiency of more than 95%) than the exhaustive method on networks with high connectivity and when the number of converters to be placed is about half that of the number of nodes in the network. Finally we presented four heuristics which could be used in lieu of the optimal algorithm to obtain solutions close to the optimal solution.

Acknowledgements

This work was supported by the NSF under grant ANI-9973102 and by famous project funded by Defense Advanced Research Agency under Prime Award No. N66001-00-1-8949 through George Washington University.

References

- [1] R. Ramaswami, K. Sivarajan, *Optical Networks: A Practical Perspective*, Morgan Kaufmann, Los Altos, CA, 1998.
- [2] B. Mukherjee, *Optical communication networks*, McGraw-Hill Series on Computer (1997).
- [3] K.C. Lee, O.K. Li, A wavelength-convertible optical network, *IEEE Journal on Lightwave Technology* 11 (1993) 962–970.
- [4] R. Ramaswami, K.N. Sivarajan, Routing and wavelength assignment in all-optical networks, *IEEE/ACM Transactions on Networking* 3 (5) (1995) 489–500.
- [5] M. Kovačević, A.S. Acampora, Benefits of wavelength translation in all-optical clear-channel networks, *IEEE Journal on Selected Areas in Communications* 14 (5) (1996) 868–880.
- [6] A. Birman, Computing approximate blocking probabilities for a class of all-optical networks, *IEEE Journal on Selected Areas in Communications* 13 (1996) 852–857.
- [7] R.A. Barry, P.A. Humblet, Models of blocking probability in all-optical networks with and without wavelength changers, *IEEE Journal on Selected Areas in Communications* 14 (5) (1996) 858–867.
- [8] S. Subramaniam, M. Azizoglu, A.K. Somani, All-optical networks with sparse wavelength conversion, *IEEE/ACM Transactions on Networking* 4 (4) (1996) 544–557.
- [9] S. Subramaniam, A. Somani, M. Azizoglu, R.A. Barry, A Performance Model for Wavelength Conversion with Non-Poisson Traffic, *Proceedings of IEEE INFOCOM'97*, Kobe, Japan, April, 1997.
- [10] R. Ramaswami, G.H. Sasaki, Multiwavelength Optical Networks with Limited Wavelength Conversion, *Proceedings of IEEE INFOCOM'97*, April, 1997.
- [11] S. Subramaniam, M. Azizoglu, A.K. Somani, On the Optimal Placement of Wavelength Converters in WAVELENGTH-routed Networks, *Proceedings of IEEE INFOCOM'98*, April, 98, 1998.
- [12] S. Thiagarajan, A.K. Somani, An Efficient Algorithm for Optimal Wavelength Converter Placement on Wavelength-Routed Networks with Arbitrary Topologies, *Proceedings of IEEE INFOCOM'99*, March, 1999.
- [13] L. Li, A.K. Somani, Efficient Algorithms for the Wavelength Converter Placement on All-Optical Networks, *Proceedings of Conference on Information Sciences and Systems*, Maryland, March, 1999.
- [14] J. Yates, J. Lacey, D. Everitt, M. Summerfield, Limited-Range Wavelength Translation in All-Optical Networks, *Proceedings of IEEE INFOCOM'96*, March, vol. 3, 1996, pp. 954–961.
- [15] V. Sharma, E.A. Varvarigos, Limited Wavelength Translation in All-Optical WDM Mesh Networks, *Proceedings of IEEE INFOCOM'98*, March, vol. 2, 1998, pp. 893–901.
- [16] T. Tripathi, K. Sivarajan, Computing Approximate Blocking Probabilities in Wavelength-Routed All-Optical Networks with Limited Range Wavelength Conversion, *Proceedings of IEEE INFOCOM'99*, March, vol. 1, 1999, pp. 321–328.
- [17] Y. Zhou, G. Rouskas, H. Perros, Blocking in Wavelength-Routing Networks; Part I: The Single Path Case, *Proceedings of IEEE INFOCOM'99*, March, vol. 1, 1999, pp. 321–328.
- [18] G. Xiao, Y.W. Leung, Algorithms for allocating wavelength converters in all-optical networks, *IEEE/ACM Transactions on Networking*, August 7 (4) (1999) 545–557.
- [19] K.R. Venugopal, M. Shivakumar, P. Sreenivasa, A Heuristic for Placement of Limited Range Wavelength Converters in All-Optical Networks, *Proceedings of IEEE INFOCOM'99*, March, 1999.