Photonic Network Communications, 3:3, 197–211, 2001

Minimizing Wavelength Conversions in WDM Path Establishment*

Rami Melhem, Shu Li, Taieb Znati
Computer Science Department, University of Pittsburgh, Pittsburgh, PA 15260
E-mail: {melhem,sli,znati}@cs.pitt.edu

Received August 3, 2000; Revised December 26, 2000

Abstract. This paper studies algorithms for connection establishment in wavelength division multiplexing networks. Although wavelength conversion capabilities are assumed at each node in the network, the goal of this research is to minimize or impose an upper bound on the number of wavelength conversions on the established path. Two types of schemes are investigated and compared. In the first scheme, the wavelengths are selected adaptively during path establishment on a given route, and in the second scheme, both the route and the wavelength assignment are selected optimally based on global information about path costs and wavelength availability in the network. We present two efficient algorithms for globally selecting routes and wavelengths, one finds the least cost route for a maximum number of wavelength conversions, and the other selects from among the shortest routes, the one with a minimum number of wavelength conversions. The results of comparing the two path establishment schemes show that, for dynamically changing traffic, the adaptive selection of wavelengths on a fixed route during path establishment is more beneficial than the optimal selection of the route and wavelengths prior to path establishment.

Keywords: wavelength division multiplexing, path setup, connections establishment, wavelength conversion, all-optical networks, bounded delay

1 Introduction

Connection establishment in BISDN networks is accomplished by first selecting a path in the network, and then reserving the resources along this path. However, when the rate at which connections are established and released is high, the resource availability may change rapidly, and the likelihood of establishing a connection along the selected path decreases. This difficulty can be avoided by combining route selection and resource reservation [1]. In this paper, we deal with wavelength division multiplexed (WDM) networks, where wavelengths are the primary resources in the networks.

Although different types of photonic switching networks have been reported and demonstrated, wavelength division multiplexing of optical networks has emerged as one of the most attractive approaches to data transfer in interconnection networks [2–4]. Specifically, an all-optical path between two access nodes can exploit the large bandwidth of optics, without the overhead of buffering and processing at intermediate nodes. However, it has been shown that the ability of converting between wavelengths along the same path increases the probability of successfully establishing a connection [5–8].

Current technology does not allow wavelength conversions to be performed efficiently and cost effectively in the optical domain. Hence, in the near future, the switching fibers of many optical networks will not be capable of performing wavelength conversions. In such networks, semi-optical paths may still be established if intermediate nodes are used as relays to receive a message on one wavelength and retransmit it on another. In Fig. 1, we show an intermediate node in a network with two wavelengths. In this simple example, the node is connected to its neighbors by two input links and two output links. The optical input and output signals to the local node are either multiplexed (Fig. 1a) or demultiplexed (Fig. 1b). In either case, a message that is relayed at this node because of a need for wavelength conversion will suffer a relatively large delay because of buffering, processing, and optics/electronic conversion.

Thus, wavelength conversion in networks with all-

*This work was supported in part by NSF award MIP 96-33729 to the University of Pittsburgh.

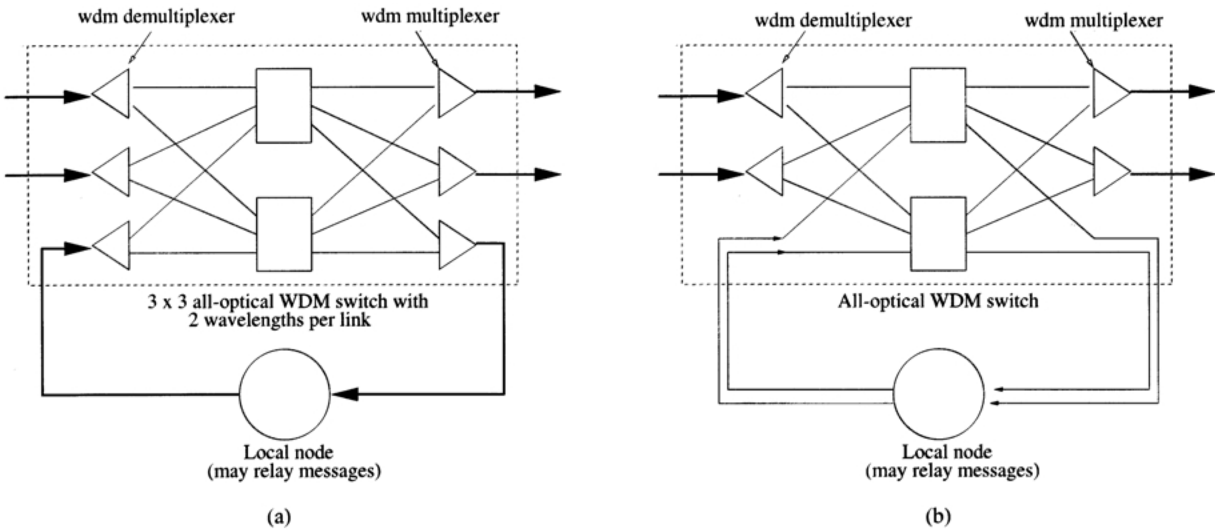


Fig. 1. A node's model in a WDM network with all-optical switches.

optical switches introduces delays at intermediate nodes which affect the quality of service on the connections. For this reason, it is desired to minimize the number of wavelength conversions along a given connection. Moreover, if certain quaility of service guarantees are required in terms of delay and jitter, then a limit on the number of wavelength conversions need to be imposed on a connection in order to meet these requirements.

The objective of this paper is the development and analysis of distributed, QoS-based routing and connection establishment protocols for WDM networks with all-optical switches. We propose a set of connection establishment algorithms which attempt either to minimize or impose an upper bound on the number of wavelength conversions along the routing path in order to meet the end-to-end delay requirements of the connection. The first class of algorithms is adaptive and uses a forward local approach to establish paths. Furthermore, the algorithms take an aggressive and greedy reservation strategy which leads to paths with a minimum number wavelength conversions.

The second class of algorithms uses a global approach and aims at optimizing the number of conversions required along the path. The optimality feature of these algorithms, combined with their polynomial time complexity, makes them very efficient to handle QoS-based routing and path establishment in high-speed WDM networks.

The rest of the paper is organized as follows: In the next section, we describe the network model. We then present a distributed algorithm for establishing a connection on a given path in the network. The algorithm dynamically selects the wavelengths on the path in a way that minimizes the number of wavelength conversions. In Section 4.1, we describe a global optimal algorithm for selecting the least cost path assuming that an upper limit is given on the number of wavelength conversions. In Section 4.2, we present a global optimal algorithm for selecting the shortest path with the minimum number of wavelength conversions. In Section 5, we discuss simulation results to compare the performance of the proposed path selection and establishment algorithms in a distributed environment where connections are established and removed dynamically. Finally, we conclude the paper in Section 6.

2 Network Model

It is assumed in this paper that W wavelengths, $\lambda_1, \ldots, \lambda_W$, are available on each link of the network and that wavelength-sensitive switches are available at each node to route signals either to the next link toward the destination, or to the local processor. The switching is performed in the optical domain but the switches are controlled by an electronic controller (see Fig. 2).

A control network, which is separate from the optical data network but shadows its topology, is used

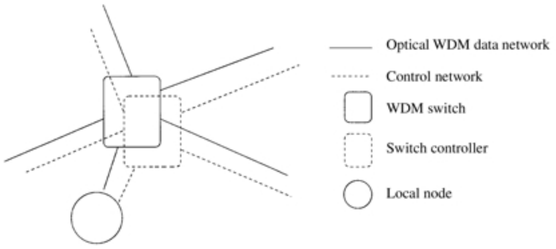


Fig. 2. The data and control networks.

for exchanging control messages. The traffic on the control network consists of small packets, and thus is lighter than the traffic on the data network. Given that switch controllers at intermediate nodes need to examine the control packets and update local bookkeeping information accordingly, the control network can be implemented as an electronic network. It is possible, however, to reserve a wavelength on the data network to exclusively exchange control messages as proposed in Ramaswami and Segall [9].

A network is represented by a graph, $G = (V, E)$, where V is a set of nodes (corresponding to network nodes) and E is a set of edges (corresponding to network links). An edge from a node $u \in V$ to a node $v \in V$ is denoted by $\langle u, v \rangle$. A path, $P(s, d)$, from a source node, $s \in V$, to a destination node, $d \in V$ is an ordered set of nodes $\{v_0, v_1, \ldots, v_k\}$, such that $v_0 = s, v_k = d$ and for $i = 0, \ldots, k-1$, $\langle v_i, v_{i+1} \rangle \in E$. The function $next_{P(v_0,v_k)}(v_i) = v_{i+1}$, $0 \le i \le k-1$, is used to indicate the node following v_i on $P(v_0, v_k)$.

3 Dynamic Wavelength Selection and Connection Establishment

A distributed reservation protocol is needed in networks with rapidly changing resource availability [10,11]. In WDM networks, wavelengths are the main resources to be reserved along a path. With wavelength conversion capabilities at individual nodes, a path consists of one or more segments, where the same wavelength is used on each segment. Two consecutive segments on a path use different wavelengths, and thus wavelength conversion is needed at the node that joins two segments. The first node in a segment is called the initial node for that segment.

In Qiao and Melhem [12] and Yuan, Melhem and Gupta [13], a number of dynamic path establishment protocols were studied for time-division multiplexed networks. In Qiao and Mei [14] and Yuan et al. [15], the protocols were applied to WDM networks assuming no wavelength conversion capabilities. Two classes of protocols are presented in Yuan, Melhem and Gupta [13]: a forward reservation protocol and a backward reservation protocol. In this section, we assume that the nodes have wavelength conversion capabilities, and we propose a forward reservation protocol that adaptively selects wavelengths while dynamically establishing a connection along a given path. We show that aggressive and greedy reservations lead to paths with a minimum number of segments, and thus a minimum number of wavelength conversions.

Given a selected path, the process of distributively establishing a connection along this path requires the exchange of control messages. The switch controller at each node in the network maintains a state for each wavelength on each link emerging from that node. For a wavelength λ on link $\mathscr{L}$ the state can be one of the following:

- *AVAIL*: indicates that λ is available and can be used to establish a new connection.
- *LOCK*: indicates that λ is locked by some request in the process of establishing a connection.
- *BUSY*: indicates that λ is being used in some connection to transmit data.

For a link, $\mathscr{L}$, the set of wavelengths that are in the *AVAIL* state is denoted by $Avail(\mathscr{L})$. When a wavelength, λ, is not in $Avail(\mathscr{L})$, an additional field, *id*, identifies the connection request locking λ, if λ is in the *LOCK* state, or the connection using λ, if λ is in the *BUSY* state. The identifier, *id*, consists of the source node address and a local sequence number issued by that node.

Each control message related to the establishment of a connection carries its *id*, which becomes the identifier of a successfully established connection. Four types of packets are used to establish a connection:

- *Reservation packets (RES)*, each carrying a set, *cset*, of wavelengths, an ordered list, *w_segments*, of wavelengths and a corresponding ordered list, *n_segments*, of nodes.

The set *cset* keeps track of the wavelengths that are being considered for the establishment of a connection on the current segment. The list *w_segments* contains the wavelengths used on previous segments and the list *n_segments* keeps track of the nodes at which each of the segments was initiated. Wavelengths are locked at intermediate nodes while the *RES* control packet progresses toward the destination node.

- *Acknowledgment packets (ACK)*, used to inform source nodes of the success of connection requests. An *ACK* packet contains a *w_segments* list and a *n_segments* list to record the wavelengths used on each segment of the connection, and the initial nodes of the segments, respectively. As an *ACK* control packet travels from the destination to the source, the WDM switches are configured to accommodate the connection and release the wavelengths that may have been unnecessarily locked by the corresponding *RES* control packet.
- *Negative acknowledgment packets (NACK)*, used to inform source nodes of the failure of their connection requests. While traveling to the source node, a *NACK* control packet unlocks all wavelengths that were locked by the corresponding *RES* control packet.
- *Release packets (REL)*, used to release a connection. A *REL* control packet traveling from a source to a destination changes the state of the wavelength reserved for that connection from *BUSY* to *AVAIL*.

The reservation proceeds as follows:[1] when the source node, v_0, wishes to establish a connection to node v_k on a given path $P(v_0, v_k) = \{v_0, v_1, \ldots, v_k\}$, it composes a *RES* control packet with *RES.cset* set to $Avail(\langle v_0, v_1\rangle)$, *RES.w_segments* set to empty, and $RES.n\_segments = \{v_0\}$. This control packet is then routed to v_1.

When an intermediate node, v_i, receives a *RES* packet, it performs the following operations:

1. Determine the next outgoing link, $\mathscr{L} = \langle v_i, v_{i+1}\rangle$, for *RES*,
2. If $Avail(\mathscr{L})$ is empty, then the connection cannot be established along the given path. Send a *NACK* control packet back to the source node.
3. If $RES.cset \cap Avail(\mathscr{L})$ is not empty, then no need for wavelength conversion. Extend the segment to include link $\mathscr{L}$ and set *RES.cset* to $RES.cset \cap Avail(\mathscr{L})$,
4. If $RES.cset \cap Avail(\mathscr{L})$ is empty, then wavelength conversion is required. Start a new segment:

   (a) randomly select a wavelength, $\lambda \in RES.cset$ and append λ to *RES.w_segments*,
   (b) append v_i, to *RES.n_segments*,
   (c) set *RES.cset* to $Avail(\mathscr{L})$,

5. Reserve the wavelengths in *RES.cset* on $\mathscr{L}$ by changing their state to *LOCK* and send *RES* forward on $\mathscr{L}$.

A *NACK* packet traveling to the source node unlocks all the wavelengths that were reserved by the corresponding *RES* packet. Specifically, when a node v_i receives a *NACK* packet, it performs the following operations:

1. Unlock all the wavelengths that are locked on link $\langle v_i, v_{i+1}\rangle$ by connection *NACK.id*,
2. Send the *NACK* packet to node v_{i-1}.

As the *RES* packet travels towards the destination, v_k, the path is reserved incrementally. When the *RES* packet reaches v_k, one wavelength is selected from *RES.cset* and appended to *w_segments*. This is the wavelength that is to be used on the last segment of the connection. Note that, at the destination, v_k, the number of entries in the ordered lists *w_segments* and *n_segments* equals the number of segments that will be used to establish the connection. Each entry of *n_segments* contains the first node in a segment and the corresponding entry of *w_segments* contains the wavelength that is to be used for that segment.

In order to actually establish the connection, release the unused wavelengths and inform the source node of the success of the connection establishment, the destination, v_k, composes an *ACK* control packet with *ACK.w_segments* set to *RES.w_segments* and *ACK.n_segments* set to *RES.n_segments*. It then sends this *ACK* control packet toward the source, on the path, $\vec{P}(v_k, v_0)$, which is the reverse of the path $P(v_0, v_k)$ that the *RES* packet used to travel from v_0 to v_k.

When an intermediate node v_i receives an *ACK* packet on link $\mathscr{L}$, it performs the following operations:

1. Find the last wavelength, λ in *ACK. w_segments* and the last node, v, in *ACK.n_segments*,
2. Change the state of λ on $\mathscr{L}$ to *BUSY*, and change the state of all other wavelengths that were locked on $\mathscr{L}$ for the connection *ACK.id* from *LOCK* to *AVAIL*,
3. If $v = v_i$, then,
   (a) remove λ from *ACK.w_segments*,
   (b) remove v_i from *ACK.n_segments*,
   (c) setup a wavelength conversion to λ at v_i
4. Send *ACK* to v_{i-1}.

As soon as the source receives the *ACK* packet, it can start sending data on the established connection. After all data is sent, the source node sends a *REL* packet to tear down the connection. When an intermediate node v_i receives a *REL* packet it performs the following operations:

1. Unlock the wavelength that is locked on link $\langle v_i, v_{i+1}\rangle$ by connection *REL.id*.
2. Forward the *REL* packet to node v_{i+1}.

The protocol described above is aggressive in the sense that, at the beginning of each segment, the initial set of wavelengths *RES.cset* is set to include all wavelengths that are available on the next link along the path. This increases the chances of successfully establishing a connection at the price of locking wavelengths that may end up not being used for the connection, thereby potentially reducing the overall rate of connection acceptance.

A conservative approach to reservation would set *RES.cset* at the beginning of each segment to include only one wavelength. This approach, which produces shorter segments and thus results in more wavelength conversions, does not over-reserve the network resources (wavelengths). A compromise between the aggressive and conservative approach is to use a probabilistic approach which only considers those wavelengths, that are most likely to be available along the path, for inclusion in the set *RES.cset*. In the following section, we discuss some properties of the aggressiveness of the path establishment and reservation protocols.

3.1 The Effect of Aggressiveness on Segment Length

It is possible to analyze the effect of the size of *RES.cset* on the segment length in the simple case where the probabilities of wavelength availability are independent and uniform. Specifically, assuming that ρ is the probability that a particular wavelength is available on any given link $\mathscr{L}$, then we can model the reservation process by a Markov Chain. For an initial *cset* size of m the chain has $m+1$ states, $S_1, \ldots, S_m$ and F. The discrete Markov process starts from state S_m and each transition in the process models the progress of the *RES* packet on one link. If the process is at state S_i after K transitions, then this means that the size of *RES.cset* is i after traversing K links. The state F indicates that the size of *RES.cset* is zero, and thus a failure to extend the current segment. Hence, the average number of states visited in the Markov process before being absorbed in F represents the average number of links traversed by *RES* before *RES.cset* becomes empty, and thus represents the average segment size.

From any given state S_q, the next state in the process is F with probability $(1-\rho)^q$ since failure occurs if none of the wavelengths in *RES.cset* are available on the next link. The transition from S_q to $S_i, i \leq q$ occurs with probability $C_i^q \rho^i (1-\rho)^{q-i}$, where C_i^q is the number of ways to choose the i wavelengths that are available on the next link from among the q wavelengths in *RES.cset*. In Fig. 3, we show an example of the Markov Chain for $m = 4$.

Starting at any state S_q, a standard first step analysis can be used to find K_q, the average number of states visited before absorption in state F. Specifically, denoting state F by S_0, we get:

$$K_q = \sum_{i=0}^{q} C_i^q \rho^i (1-\rho)^{q-i}(K_i + 1),$$

where $K_0 = 0$. It is easy to solve the above equation and obtain the following recursive formula for K_q.

$$K_q = \frac{\rho^n + \sum_{i=0}^{q-1} C_i^q \rho^i (1-\rho)^{q-i}(K_i+1)}{1-\rho^n}.$$

Note that K_q is the average number of links that the *RES* packet traverses before ending the current segment, assuming that the current size of *RES.cset* is q. In Fig. 4, we plot the average segment size obtained using the above analysis as a function of the initial *cset* size for different values of ρ. Clearly, the

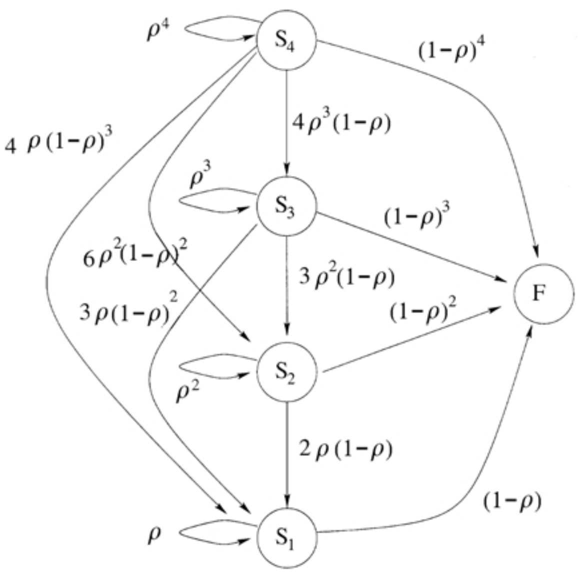


Fig. 3. A Markov Chain for *cset* size = 4.

analysis shows that the advantage of increasing the size of *cset* diminishes when the size of *cset* increases. The optimal size of *cset* should be obtained by taking into consideration the negative effect of having a large *cset* size in terms of over-reserving resources. Such a trade-off will be considered in the simulation experiments presented in Section 5.

3.2 Optimality of the Aggressive Reservation

Irrespective of the degree of aggressiveness (size of *cset*), the protocol described in this section is greedy in the sense that it does not terminate a segment prematurely. A segment is terminated only when none

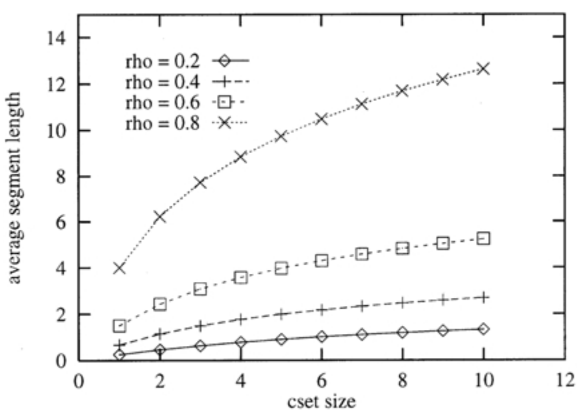


Fig. 4. The effect of aggressiveness on average segment length.

of the wavelengths in *RES.cset* are available on the next link. In the following lemma, we prove that the greedy policy, combined with aggressive reservation, is optimal.

Lemma 1: *The greedy aggressive reservation scheme results in the minimum number of segments, and thus the minimum number of wavelength conversions for establishing a connections along a given path.*

Proof: *Consider a path* $v_0, \ldots, v_k$ *and denote by* $\mathscr{L}_i$ *the link* $\langle v_{i-1}, v_i \rangle$. *Also denote by* $RES.cset(\mathscr{L}_i)$ *the value of RES.cset when RES is sent over link* $\mathscr{L}_i$. *Finally, let RES be the request packet for the aggressive greedy algorithm, and* $\bar{RES}$ *the request packet for any other algorithm.*

The aggressive greedy algorithm starts with $RES.cset(\mathscr{L}_1) = Avail(\mathscr{L}_1)$, *and determines that a wavelength conversion should take place at a node* v_c when $RES.cset(\mathscr{L}_c) \cap Avail(\mathscr{L}_{c+1}) = \emptyset$. *That is when* $\bigcap_{i=1}^{c} Avail(\mathfrak{L}_i) = \emptyset$. *At this point, it sets* $RES.cset(\mathscr{L}_{c+1}) = Avail(\mathscr{L}_{c+1})$.

If some other algorithm will determine that a conversion should take place at a node v_q, *where* $q < c$, *then it will set* $\bar{RES}.cset(\mathscr{L}_{q+1})$ to $Avail(\mathscr{L}_{q+1})$, *and* $\bar{RES}$ will reach $\mathscr{L}_{c+1}$ *with* $\bar{RES}.cset(\mathscr{L}_{c+1}) = \bigcap_{i=q}^{c+1} Avail(\mathscr{L}_i)$, *which is clearly a subset of* $Avail(\mathscr{L}_{c+1})$. *That is,* $\bar{RES}.cset(\mathscr{L}_{c+1}) \subseteq .cset(\mathscr{L}_{c+1})$, *which means that, with one wavelength conversion for each algorithm,* $\bar{RES}$ *cannot go further than RES on the path without requiring another conversion. By repeating the above argument, we reach the conclusion that the aggressive greedy algorithm will reach* v_k *with the minimum number of conversions.* □

It should be noted that it is possible to modify the aggressive greedy algorithm to still achieve the minimum number of conversions even when only some nodes in the network have wavelength conversion capabilities, while others do not [16]. In this case, if *RES.cset* becomes empty at a node without conversion capabilities, then *RES* should backtrack to the last node that has wavelength conversion capabilities and start a new segment at that node. The aggressive greedy approach, however, may not be optimal.

4 Global Path Selection Based on Cost/Conversion Trade-Offs

In the last section, we described a protocol to dynamically select wavelengths during path establishment on a given path. Usually, such a path is determined by optimizing some cost criteria. Specifically, if a function σ assigns to each link $\mathscr{L}$ in E a cost, $\sigma(\mathscr{L})$, then a path, $P(s,d)$, from a source, s, to a destination, d, is usually chosen to minimize the cost, $cost(P(s,d))$, where $cost(P(u,v))$ is the sum of the costs of the edges traversed by $P(u,v)$.

Given a set of available wavelengths, $Avail(\mathscr{L})$, on each edge $\mathscr{L} \in E$, let us denote by $conv(P(u,v))$ the minimum number of wavelength conversions needed to establish a connection on $P(u,v)$. Clearly, selecting $P(s,d)$ first, and then assigning wavelengths on $P(s,d)$ may not minimize $conv(P(s,d))$. Alternatively, if an algorithm is used to find the path from s to d which minimizes the number of wavelength conversions, then the cost of that path may not be minimum. It is possible to combine the optimization of the cost and number of wavelength conversions by considering one of the following two criteria:

1. Minimize $cost(P(s,d)) + \gamma\, conv(P(s,d))$, where γ is a measure of the relative importance of the cost and the number of wavelength conversions,
2. Minimize $cost(P(s,d))$ subject to $conv(P(s,d)) \leq \Delta$. That is find the minimum cost path from s to d in G such that the number of wavelength conversions along this path is at most Δ.

One way for solving the first optimization problem was first described in Chlamtac, Farago and Zhang [17]. The idea is to create a new graph, $\bar{G} = (\bar{V}, \bar{E})$. The set $\bar{V}$ contains W nodes, $\bar{u}_\lambda, \lambda = 1, \ldots, W$, for each node $u \in V$. For each edge $e = \langle u, v \rangle \in E$ with cost c, there are W edges $\bar{e}_\lambda \in \bar{E}$ between nodes $\bar{u}_\lambda$ and $\bar{v}_\lambda$ for $\lambda = 1, \ldots, W$. The cost of $\bar{e}_\lambda$ is c if $\lambda \in Avail(e)$ and ∞ otherwise. In addition to the edges described above, $\bar{E}$ contains one edge between any two nodes $\bar{u}_\lambda$ and $\bar{u}_\mu$ with cost γ to reflect the relative cost of performing a conversion from wavelength λ to wavelength μ at node u. Finally, two nodes $\bar{s}$ and $\bar{d}$ are added to $\bar{G}$ and connected with zero cost edges to $\bar{s}_\lambda$ and $\bar{d}_\lambda$, respectively, for $\lambda = 1, \ldots, W$. The optimization problem is then solved by using Dijkstra's algorithm to find the minimum cost path, $\bar{P}(\bar{s}, \bar{d})$ in $\bar{G}$ from $\bar{s}$ to $\bar{d}$. Specifically, if $\langle \bar{u}_\lambda, \bar{v}_\lambda \rangle$ is on $\bar{P}(\bar{s}, \bar{d})$, then $\langle u, v \rangle$ is on $P(s,d)$ and λ is assigned to $\langle u, v \rangle$. Moreover, if $\langle \bar{v}_\lambda, \bar{v}_\mu \rangle$ is on $\bar{P}(\bar{s}, \bar{d})$, then $P(s,d)$ enters node v on wavelength λ and leaves it on wavelength μ, which implies a wavelength conversion.

The second optimization problem may be solved by applying the algorithm proposed in Kompella, Pasquale and Polyzos [18] to a graph $\hat{G}$ which is topologically identical to $\bar{G}$ but in which each edge is assigned a delay weight as well as a cost. Specifically, each edge $\bar{e}_\lambda$ in $\hat{G}$ is assigned a cost of c and a delay of zero, and each edge $\langle \bar{u}_\lambda, \bar{u}_\mu \rangle$ is assigned a cost of zero and a delay of 1. The algorithm proposed [18] finds the minimum cost paths with bounded delays between every two pairs of nodes in a given graph, assuming that delays are discretized into a finite number of possible delays. The idea of the algorithm is to apply a variation of Floyd's all-pair shortest path algorithm which keeps track of the delays on the shortest paths during the execution of the algorithm and excludes any path which exceeds the given delay bound. The complexity of Floyd's algorithm is cubic in the number of nodes in the graph, and keeping track of the delays requires Δ steps. Thus, the complexity of solving our problem using this approach is $O(\Delta \cdot W^3 \cdot |V|^3)$.

The $O(\Delta \cdot W^3 \cdot |V|^3)$ complexity is a huge price to pay if we desire to compute the minimum cost path between a specific pair of nodes, s and d, while limiting the number of wavelength conversions. In what follows we describe an algorithm to solve this problem in $O(\Delta \cdot W \cdot (|E| + |V|) \cdot \log|V|)$. The algorithm works directly on the graph G rather than on the expanded graph $\hat{G}$.

4.1 Minimum Cost Path with Bounded Number of Wavelength Conversions

Given a bound, Δ, on the number of wavelength conversions, our goal is to find from among all the paths linking s to d that have at most Δ wavelength conversions, the path which has minimum cost.

For each node $w \in V$, wavelength λ, and number of wavelength conversions, $\delta = 1, \ldots, \Delta$, define $Cost(w, \lambda, \delta)$ to be the minimum cost of any path from s to w such that the path enters node w using wavelength λ and the number of wavelength conversions along this path is at most δ. The minimum cost of the path from s to w with at most δ wavelength conversions will be denoted by $Cost_{\min}(w, \delta)$. That is

$$Cost_{\min}(w, \delta) = \min_{1 \le \lambda \le W} \{Cost(w, \lambda, \delta)\}.$$

The minimum cost of a path from s to d with at most Δ wavelength conversions is then given by $Cost_{\min}(d, \Delta)$.

The values of $Cost(w, \lambda, 0)$ for a given λ and w can be found by removing from the graph, G, any link on which wavelength λ is not available, and then finding the shortest path from s to w using Dijkstra's shortest path algorithm. Then, the values of $Cost(w, \lambda, \delta)$ can be computed recursively from $Cost(w, \lambda, \delta - 1)$ by observing that the minimum cost of entering node w on λ with at most δ wavelength conversions should satisfy the following:

- **M1:** $Cost(w, \lambda, \delta)$ is less than or equal to the minimum cost, $Cost(w, \lambda, \delta - 1)$, of entering node w on λ with at most $\delta - 1$ wavelength conversions.
- **M2:** for any neighboring node u such that $\lambda \in Avail(\langle u, w\rangle)$, the value of $Cost(w, \lambda, \delta)$ should be less than or equal to the minimum cost, $Cost_{\min}(u, \delta - 1)$, of entering u on any wavelength with at most $\delta - 1$ wavelength conversions, plus the cost of the link $\langle u, w\rangle$. In this case, a wavelength conversion to λ may be needed at node u.
- **M3:** for any neighboring node u such that $\lambda \in Avail(\langle u, w\rangle)$, the value of $Cost(w, \lambda, \delta)$ should be less than or equal to the minimum cost, $Cost(u, \lambda, \delta)$, of entering a neighboring node u on λ with at most δ wavelength conversions, plus the cost of the link $\langle u, w\rangle$.

Note that finding the values of $Cost(w, \lambda, \delta)$ that satisfy the first two of the above three conditions can be obtained directly from the values of $Cost(w, \lambda, \delta - 1)$. The resulting values of $Cost(w, \lambda, \delta)$ will be denoted by $Cost_{\text{init}}(w, \lambda, \delta)$. The values of $Cost(w, \lambda, \delta)$ for a given λ and a given δ that satisfy the third condition, M3, can be obtained from the following equations:

$$Cost(s, \lambda, \delta) = 0, \tag{1}$$

$$Cost(w, \lambda, \delta) \le \min\{Cost_{\text{init}}(w, \lambda, \delta), \min_{<u,w> \in E} \{Cost(u, \lambda, \delta) + \sigma_\lambda(u, w)\}\} \tag{2}$$

where, $\sigma_\lambda(u, w) = \sigma(u, w)$ if $\lambda \in Avail(\langle u, w\rangle)$ and $\sigma_\lambda(u, w) = \infty$, otherwise.

Before describing the algorithm which enforces conditions M1, M2 and M3, we first present an algorithm which uses a dynamic programming approach to compute Equations (1) and (2) given the initial values of $Cost_{\text{init}}(w, \lambda, \delta)$.

Algorithm NO_CONV (s, λ, δ)
  $Cost(w, \lambda, \delta) = Cost_{\text{init}}(w, \lambda, \delta)$ for every node $w \in V$;
  $Cost(s, \lambda, \delta) = 0$;
  Mark all nodes in V;
  Repeat until all nodes are un-marked;
    Select a marked node, u with minimum $Cost(u, \lambda, \delta)$;
    Un-mark u;
    For each w such that $\langle u, w\rangle \in E$ Do
      $Cost(w, \lambda, \delta) = \min\{Cost(w, \lambda, \delta), Cost(u, \lambda, \delta) + \sigma_\lambda(u, w)\}$;

For a given λ and a given δ, the above algorithm uses the same approach as Dijkstra's shortest path algorithm to recursively update the values of $Cost(w, \lambda, \delta)$ for every node w in the graph G. In fact, it is straight forward to see that for $\delta = 0$ and $Cost_{init}(w, \lambda, 0) = \infty$, the algorithm reduces to Dijkstra's shortest path algorithm for computing the least cost path from s to any other node w using wavelength λ. Specifically, using $\sigma_\lambda(\mathscr{L})$ on the links of G rather than $\sigma(\mathscr{L})$ amounts to removing from G any link on which λ is not available.

Next, we present the algorithm which computes, $Cost_{\min}(w, \delta)$, the least cost path from s to any node w assuming a maximum of δ wavelength conversions. The algorithm finds the minimum cost of the path rather than the path itself. It is straight forward to add backward pointers in the algorithm to find the paths with minimum costs.

Algorithm BOUNDED_CONV
1. $Cost(w, \lambda, 0) = \infty$ for all $w \in V$ and $\lambda = 1, \ldots, W$;
2. $Cost(s, \lambda, 0) = 0$ for $\lambda = 1, \ldots, W$;
3. For $\lambda = 1, \ldots, W$ Do
   Apply NO_CONV $(s, \lambda, 0)$;

/* starting with the initial values $Cost(w, \lambda, 0) = \infty$*/;

4. For every $w \in V$

$Cost_{\min}(w, 0) = \min_{1 \le \lambda \le W}\{Cost(w, \lambda, 0)\}$;

5. For $\delta = 1, \ldots, \Delta$ Do

5.1. For every $w \in V$ and $\lambda = 1, \ldots, W$ Do

$Cost(w, \lambda, \delta) = Cost(w, \lambda, \delta - 1)$;

5.2. For all $\langle u, w \rangle \in E$ Do

$Cost(w, \lambda, \delta) = \min\{Cost(w, \lambda, \delta),$
$Cost_{\min}(u, \delta - 1) + \sigma_\lambda(\langle u, w \rangle)\}$

5.3. For $\lambda = 1, \ldots, W$ Do

Apply NO_CONV (s, λ, δ);

/* starting with the current values of $Cost(w, \lambda, \delta)$*/;

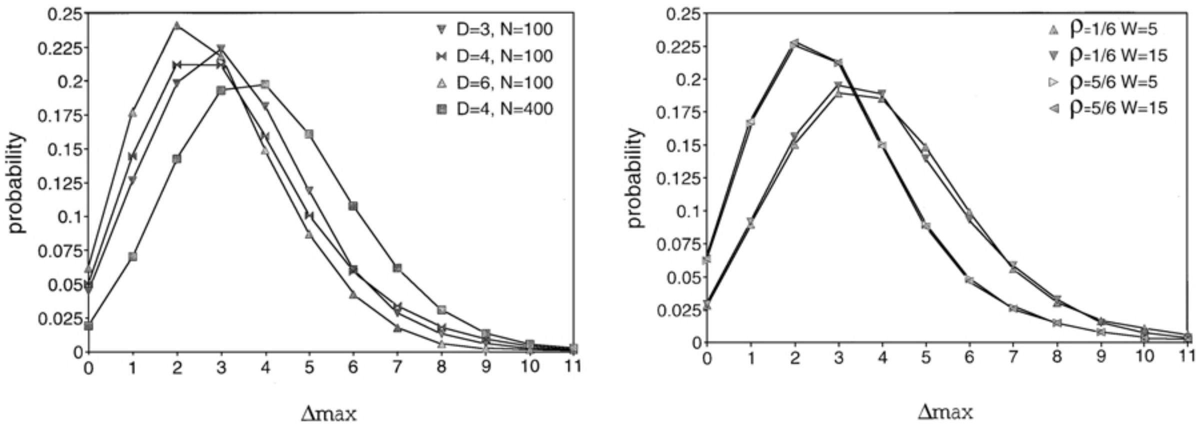
5.4. For every $w \in V$

$Cost_{\min}(w, \delta) = \min_{1 \le \lambda \le w}\{Cost(w, \lambda, \delta)\}$;

Steps 5.1, 5.2 and 5.3 in the above algorithm respectively implement the three conditions M1, M2 and M3 itemized earlier in this section.

We have implemented the above algorithm to measure the effect of the number of allowable wavelength conversions on the minimum cost of the path. We applied the algorithm to randomly generated graphs with arbitrary sources and destinations, and arbitrary wavelength availability on the links. In the experiments reported in this section, we considered randomly generated graphs with an average of N nodes each and each node having an average degree of D. The value of N was varied between 100 and 400 and the value of D was varied between 3 and 6. We assumed W wavelengths and we randomly specified the cost on each link, and assumed that each wavelength is available on any given link with a probability ρ. Finally, we selected sources and destinations randomly and we measured the percentage increase in the cost of the cheapest path from source to destination as a function of the decrease of the number of allowable wavelength conversions. More specifically, let $P(s, d)$ be the least cost path for given s and d and let $\Delta_{\max}(s, d)$ be the minimum number of wavelength conversions on this path. Let also $P_r(s, d)$ be the least cost path from s to d with at most $\Delta_{\max}(s, d) - r$ wavelength conversions. Hence, the cost penalty of reducing the number of allowable conversions by r is given by $\pi(r, s, d) = \frac{cost(P_r(s,d))}{cost(P(s,d))}$.

Two issues are faced when trying to study the average effect of reducing the number of allowable wavelength conversions. The first issue relates to the fact that different source/destination pairs have different values of $\Delta_{\max}(s, d)$, and thus the value of r cannot exceed $\Delta_{\max}(s, d)$ for a given source and destination. Hence, the average effect of tightening the allowable number of wavelength conversions should be taken over source/destination pairs with a given value of $\Delta_{\max}(s, d)$. In Fig. 5, we show the probability distribution obtained in our experiments for the value of $\Delta_{\max}(s, d)$ over randomly generated s



(a) $\rho = 0.5$ and $W = 10$ (b) $N = 100$ and $D = 4$

Fig. 5. The probability density distribution of $\Delta_{\max}(P(s, d))$.

and d. The experiments show that the probability distribution of $\Delta_{\max}(s,d)$ does not heavily depend on the graph size, N, the node degree, D, or the probability of wavelength availability, ρ.

The second issue faced when studying the average effect of reducing the number of allowable wavelength conversions relates to the fact that there may not be a path $P_r(s,d)$ if r is close to $\Delta_{\max}(s,d)$, that is if we tighten too much the number of allowable conversions. We solve this issue by assuming that $cost(P_r(s,d)) = \infty$ if $P_r(s,d)$ cannot be found, and then computing, $\bar{\pi}_{\Delta_m}(r)$, the harmonic mean of the cost penalty, $\pi(r,s,d)$, over source/destination pairs with $\Delta_{\max}(s,d) = \Delta_m$. In Fig. 6, we plot $\bar{\pi}_{\Delta_m}(r)$ for $N = 100$, $D = 4$, $W = 10$ and $\rho = 0.5$. This and similar results for different values of N, D, W and ρ indicate that the cost penalty increases exponentially when r approaches Δ_m. That is, when the number of allowable wavelength conversions approaches zero.

4.2 Shortest Path with the Minimum Number of Wavelength Conversions

In this section, we consider the cost of a path to be its length expressed as the number of links on the path. Specifically, if $\sigma(\mathscr{L}) = 1$ for any link $\mathscr{L} \in E$, then the cost of a path becomes its length. Given a source, s and a destination, d, there may be more than one minimum length (shortest) path from s to d. In this section, we present an algorithm that will select from among the shortest paths from s to d, the one that minimizes the number of wavelength conversions.

In order to confine the selection of routes to shortest paths, we define the sub-graph, $G_{s,d} = \{V_{s,d}, E_{s,d}\}$, of G whose nodes and edges lie on a shortest path from s to d. This sub-graph can be built in $O((|V| + |E|)\log|V|)$ and is independent of the wavelength availability.

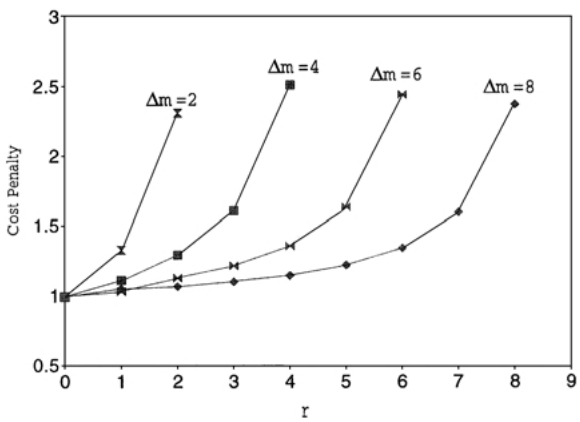


Fig. 6. The effect of tightening the number of conversions on the cost penalty.

If the shortest distance between s and d is k, then we divide the nodes in sub-graph $G_{s,d}$ into $k+1$ disjoint layers as follows:

- $V_0 = \{s\}$,
- $V_{i+1} = \{v;\ <u,v> \in E_{s,d}, u \in V_i, v \notin \bigcup_{j=0}^{j=i} V_j\}$, $i = 0, 1, \ldots k-1$.

That is, V_i is the set of nodes that are at distance i from s. Note that there cannot be any edge in $G_{s,d}$ between two nodes u and v that are in the same set V_i, since by definition, an edge between u and v cannot be on the shortest path to either u or v. Fig. 7 shows an example graph, and Fig. 8 shows the sub-graph $G_{s,d}$ for specific nodes s and d. Fig. 8 also shows the layers, $V_{0,} \ldots, V_5$, of $G_{s,d}$.

An efficient algorithm is proposed to find a shortest path from s to d on which a connection can be established with a minimal number of wavelength conversions. The algorithm uses $mwc(w,\lambda)$ to denote the minimum number of wavelength conversions needed to enter node w using λ on some path in $G_{s,d}$. The value of $mwc(w,\lambda)$ is ∞ if it is not possible to enter node w using wavelength λ.

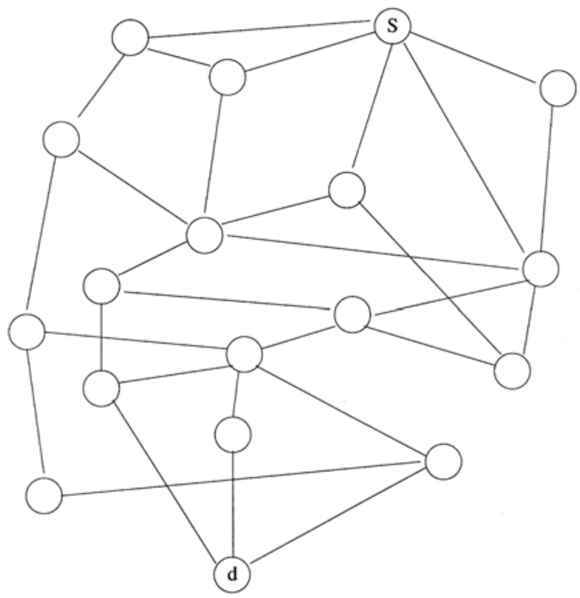


Fig. 7. A graph G.

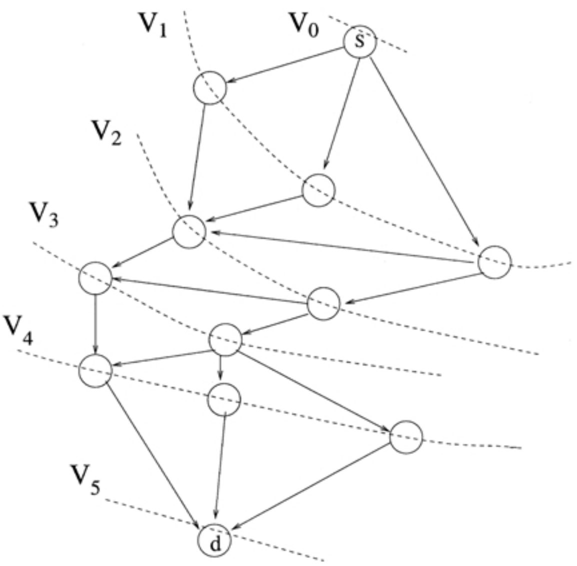


Fig. 8. The sub-graph $G_{s,d}$ of the graph given in Fig. 7 and its layers.

Given the sets *Avail*() for the links of $G_{s,d}$, a dynamic programming approach can be used to compute $mwc(w, \lambda)$ for each node, w, in the graph and each wavelength, λ. The wavelength with the minimum value of $mwc(d, \lambda)$ will then determine the path from s to d. Initially, $mwc(w, \lambda) = \infty$ for every node in $G_{s,d}$, except for the source node, s, for which $mwc(s, \lambda) = 0$ for $\lambda = 1, \ldots, W$.

In order to be more specific, let w be a node in V_{i+1} which can be reached only from node u in V_t. In this case, the value of $mwc(w, \lambda)$ is:

$$mwc(w,\lambda) = \begin{cases} \infty & \text{if } \lambda \notin Avail(u,w) \\ \min\{mwc(u,\lambda), \min_\mu\{mwc(u,\mu)+1\}\} & \text{if } \lambda \in Avail(u,w). \end{cases}$$

In words, we cannot use λ to enter node w if $\lambda \notin Avail(u, w)$. However, if $\lambda \in Avail(u, w)$ and node u can be entered on λ after k wavelength conversions, then node w can also be entered on λ after k wavelength conversions. Finally, if $\lambda \in Avail(u, w)$ and node u cannot be entered on λ, that is if $mwc(u, \lambda) = \infty$, then we can enter u on any other wavelength μ, convert to λ at u and enter w on λ after $mwc(u, \mu) + 1$ wavelength conversions. Of course, the value of μ that leads to the minimum number of wavelength conversions should be used.

So far, we have considered only the case in which w can be reached from only one node in V_i. If w can be reached from q nodes $u_j, j = 1, \ldots, q$, in V_i, then we compute for each j the value of $mwc_j(w, \lambda)$ to be:

$$mwc_j(w,\lambda) = \begin{cases} \infty & \text{if } \lambda \notin Avail(u_j,w) \\ \min\{mwc(u_j,\lambda), \min_\mu\{mwc(u_j,\mu)+1\}\} & \text{if } \lambda \in Avail(u_j,w). \end{cases}$$

Then, we compute $mwc(w, \lambda)$ for each λ by considering the path to w that leads to the minimum number of wavelength conversions. Namely,

$$mwc(w,\lambda) = \min_{1 \le j \le q} \{mwc_j(w,\lambda)\}.$$

After finding the wavelength that minimizes $mwc(d, \lambda)$, the path from s to d that uses the minimum number of conversions can be determined by back-tracking through $G_{s,d}$. The following algorithm finds such a path, P.

Algorithm MIN_CONV

$mwc(w, \lambda) = \infty$ for all $w \in G_{s,d}$ and $\lambda = 1, \ldots, W$;
$mwc(s, \lambda) = 0$ for all $\lambda = 1, \ldots, W$
For each level $V_i, i = 1, \ldots, k$ Do
  For each node $w \in V_i$ and each λ Do
    Let H_w be the set of nodes in V_{i-1} connected to w;
    For each node u_j in H_w compute $mwc_j(w, \lambda)$;
    Compute $mwc(w, \lambda) = \min_{u_j \in H_w}\{mwc_j(w, \lambda)\}$;
select λ such that $mwc(d, \lambda) = \min_\mu\{mwc(d, \mu)\}$;
$w = d$;
Repeat
  If there is a $u \in H_w$ so that $mwc(w, \lambda) = mwc(u, \lambda)$
    Then set $next_P(u) = w$ and $w = u$
    Else find $u \in H(w)$ and μ such that $mwc(w, \lambda) = mwc(u, \mu) + 1$ and set $next_P(u) = w, w = u$ and $\lambda = \mu$;
Until $w = s$;

The complexity of the algorithm MIN_CONV is $O((|E_{s,d}| + |V_{s,d}|)W)$.

5 Comparing Dynamic Wavelength Assignment with Global Path and Wavelength Selection

After a global algorithm is used to select a path, $P(s, d)$, and assign wavelengths such as to minimize or bound the number of wavelength conversions, a simple protocol can be used to reserve the wavelengths and establish a connection on $P(s, d)$.

However, in a dynamic environment where the wavelength availability changes, the information in the sets *Avail*() used by the global algorithms, BOUNDED_CONV and MIN_CONV, may not be up-to-date. Since gathering information about the state of the network cannot be done instantaneously, the information about the availability of wavelengths that the global algorithm uses may not reflect the current state of the network.

Using a global algorithm to select a path from s to d has the drawback of selecting the route and assigning wavelengths based on out-dated information about wavelength availability. This may be rather harmful if the wavelength availability changes frequently. We have run simulation experiments to compare the dynamic selection of wavelengths at path establishment time with the global selection of route and wavelengths before path establishment. The experiments simulate mesh-like networks with 100 nodes each, where each node randomly generates requests to communicate with random destinations. Two parameters, the load that each node generates, and the average number of packets in a message, L, determine the request interarrival time at each node. Both the control traffic and the data traffic were simulated on a control and a data network, respectively, with the assumption that it takes one time unit to route one control packet between two nodes on the control network and one time unit to route a data packet on an all-optical path segment. We assumed that each link can support 10 wavelengths.

The following two schemes were simulated to establish a connection from a source to a destination:

- A fixed routing algorithm which selects wavelengths dynamically at path establishment time as described in Section 3. The initial size of *RES.cset* is varied from 1 to 10.
- A routing algorithm which selects the route and wavelengths using the global algorithm MIN_CONV of Section 4.2 with the assumption that each node in the network updates its information about the wavelength availability on every link in the network every T time units. A simple signaling protocol is used to establish a connection along the selected path using the selected wavelengths.

Both schemes described above aim at minimizing the number of wavelength conversions along a connection. In order to enforce a bound on the number of wavelength conversions, we accept only connections with a number of wavelength conversions less or equal to a given value, Δ. We will only report on the results of the simulations for $\Delta = 1$ and 3. Other values of Δ give similar results.

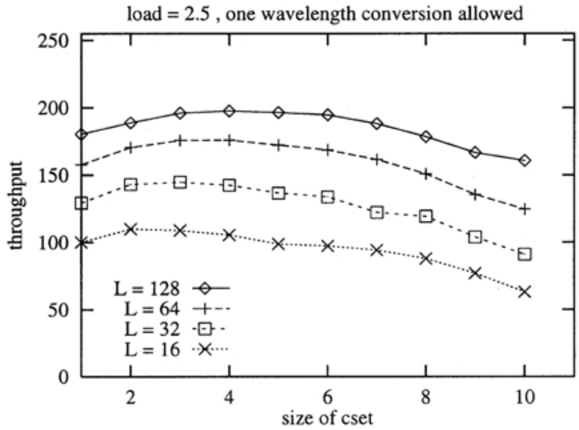


Fig. 9. Effect of aggressiveness on throughput for $W = 10$, and $\Delta = 1$ (fixed routing).

In Figs. 9 and 10, we show the effect of the initial size of *RES.cset* on the throughput of the system for different average message sizes. As indicated in Section 3, increasing the size of *cset* increases the probability of successfully establishing a connection but also locks wavelengths unnecessarily, which affects the establishment of other connections in the system. Although the optimal *cset* size depends on the message size, we observed from this and other experiments that a *cset* size of 3 or 4 is usually

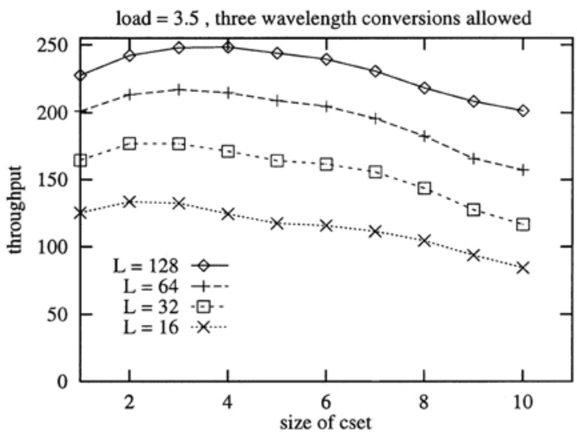


Fig. 10. Effect of aggressiveness on throughput for $W = 10$, and $\Delta = 1$ (fixed routing).

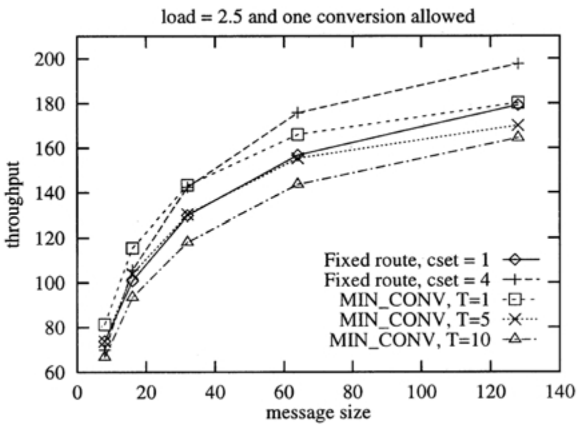


Fig. 11. Comparing the throughputs of the connection establishment algorithms for $\Delta = 1$.

close to the optimal. Clearly, the two figures indicate that increasing the number of allowable wavelength conversions increases the throughput of the system.

In Figs. 11 and 12, we compare the global selection of route/wavelength with the dynamic selection of wavelengths on a fixed route. First, observe that, as expected, the performance of the global route selection deteriorates when the global information is updated less frequently (larger T). Also, when $cset = 4$, which is close to the optimal cset size, the dynamic wavelength selection algorithm outperforms the global algorithm. The only exception is for small message sizes ($L \leq 16$) and only when it is assumed that the global algorithm has very accurate information about the network state ($T = 1$). In fact, the advantage of dynamic wavelength selection increases with the number of allowable wavelength conversions, because the flexibility of adapting the reservation process to the current network state is more valuable when wavelength conversion is needed.

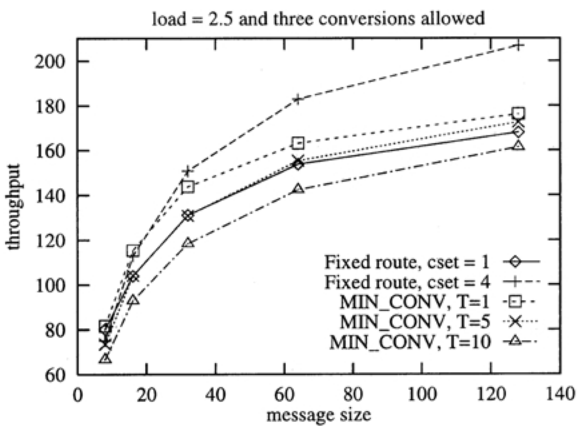


Fig. 12. Comparing the throughputs of the connection establishment algorithms for $\Delta = 3$.

6 Conclusion

The distributed algorithm presented in Section 3 minimizes the number of wavelength conversions when establishing a connection on a given path. This algorithm performs greedy wavelength reservations and may be adjusted to different levels of aggressiveness in locking wavelengths during path establishment. The best results are usually obtained when only a few wavelengths (from two to four) are locked for a particular connection during path establishment.

Two algorithms are presented to simultaneously select a route and assign wavelengths for a connection, assuming global knowledge of the wavelength availability, and the cost of using network paths. The first algorithm finds the least cost path for a given allowable number of wavelength conversions, while the second finds the shortest path with the minimum number of wavelength conversions. Both algorithms can be easily extended to the cases where only a few nodes in the network have wavelength conversion capabilities [16] or where the wavelength conversion capabilities at each node are limited [19].

Note that, by globally selecting the best path and wavelengths before path establishment, the path establishment protocol does not have the flexibility of adaptively selecting the wavelength according to actual availability. We compared the adaptive wavelength selection on a fixed path with the global selection of paths and wavelength. The results indicate that, because wavelength availability change dynamically, the capability of adaptively selecting the wavelengths is more crucial than the capability of selecting the best route before path establishment. It would be interesting to study means to adaptively determine both the route and the wavelength during path establishment.

Notes

1. We denote the fields of a packet by *packet.field*. For example, *RES.id* denotes the *id* field of the *RES* packet.

References

[1] I. Gidon, R. Rom, Multi-path routing combined with resourcee reservation. IEEE INFOCOM'97, vol. 1, Kobe, Japan, (April 1997), pp. 92–100.

[2] I. Chlamtac, A. Ganz, G. Karmi, Lightpath communications: an approach to high-bandwidth optical wans, IEEE Trans. on Communications, vol. 40, no. 7, (July 1992), pp. 1171–1182.

[3] P. Dowd, K. Sivalingam, A multi-level wdm access protocol for an optically interconnected parallel computer. IEEE INFOCOM'94, vol. 1, Toronto, Canada, (June 1994), pp. 400–408.

[4] B. Mukherjee, D. Banerjee, S. Ramamurthy, A. Mukherjee, Some principles for designing wide area optical networks. IEEE/ACM Trans. on Networking, vol. 4, no. 5, (October 1996), pp. 684–696.

[5] R. Barry, P. Humblet, Models of blocking probability in all-optical networks with and without wavelength changers. IEEE INFOCOM'95, vol. 2, Boston, MA, (April 1995), pp. 402–412.

[6] E. Karasan, E. Ayanoglu, Effect of wavelength routing and selection algorithms on wavelength conversion gain in WDM optical networks. In IEEE/ACM Trans. on Networking, vol. 6, no. 2, (April 1998), pp. 186–196.

[7] R. Pankaj, R. Gallager, Wavelength requirements of all-optical networks. IEEE/ACM Trans. on Networking, vol. 3, no. 3, (August 1995), 269–280.

[8] B. Ramamurthy, B. Mukherjee, Wavelength conversion in WDM networking. In IEEE J. on Selected Areas in Communications, vol. 16, no. 7, (July 1998), pp. 1061–1073.

[9] R. Ramaswami, A. Segall, Distributed network control for wavelength routed optical networks. In IEEE INFOCOM'96, San Francisco, CA, vol. 1, (March 1996), pp. 138–147.

[10] I. Cidon, I. S. Gopal, A. Segall, Connection establishment in high-speed networks. IEEE/ACM Trans. on Networking, vol. 1, no. 4, (August 1993), pp. 469–482.

[11] Awerbuch et al., Distributed control for Paris. In Proceedings of 9th Annual ACM Symp. on Principles of Distributed Computing, (1990), pp. 145–160.

[12] C. Qiao, R. Melhem, Reconfiguration with time division multiplexing MINs for multiprocessor communications. IEEE Transactions on Parallel and Distributed Systems, vol. 5, no. 4, (April 1994), pp. 337–352.

[13] X. Yuan, R. Melhem, R. Gupta, Distributed path reservation algorithms for multiplexed all-optical interconnection networks. In Proceedings of the IEEE Symp. High Performance Computer Architecture, San Antonio, TX, (Feb. 1997).

[14] C. Qiao, Y. Mei, Wavelength reservation under distributed control. In IEEE/LEOS summer topical meeting: Broadband Optical Networks, (August 1996).

[15] X. Yuan, R. Melhem, R. Gupta, Y. Mei, C. Qiao, Distributed control protocols for wavelength reservation and their performance evaluation. In Photonic Network Communications, vol. 1, no. 3, (Nov. 1999), pp. 207–218.

[16] S. Subramaniam, M. Azizoglu, A. K. Somani, Connectivity and sparse wavelength conversion in wavelength-routing networks. In IEEE INFOCOM'96, San Francisco, CA, vol. 1, (March 1996), pp. 148–155.

[17] Chlamtac, A. Farago, T. Zhang, Lightpath (wavelength) routing in large WDM networks. IEEE J. on Selected Areas in Communications, vol. 14, no. 5, (May 1996), pp. 909–913.

[18] V. Kompella, J. Pasquale, G. Polyzos, Multicast routing for multimedia communication, In IEEE/ACM Trans. on Networking, vol. 1, no. 3, (June 1993), pp. 286–292.

[19] V. Sharma, E. Varvarigos, Limited wavelength translation in all-optical WDM networks. INFOCOM'98, San Francisco, CA, vol. 1, (March/April 1998).

Rami Melhem received a B.E. in Electrical Engineering from Cairo University in 1976, an M.A. degree in Mathematics and an M.S. degree in Computer Science from the University of Pittsburgh in 1981, and a Ph.D. degree in Computer Science from the University of Pittsburgh in 1983. He was an Assistant Professor at Purdue University prior to joining the faculty of The University of Pittsburgh in 1986, where he is currently a Professor of Computer Science and Electrical Engineering and the Chair of the Computer Science Department. His research interests include Real-Time and Fault-Tolerant Systems, Optical Interconnection Networks, High Performance Computing and Parallel Computer Architectures. Dr. Melhem served on program committees of numerous conferences and workshops and was the general chair for the third International Conference on Massively Parallel Processing Using Optical Interconnections. He was on the editorial board of the IEEE Transactions on Computers and served on the advisory boards of the IEEE technical committees on Parallel Processing and on Computer Architecture. He is the editor for the Plenum Book Series in Computer Science and is on the editorial board of the IEEE Transactions on Parallel and Distributed systems. Dr. Melhem is a fellow of IEEE and a member of the Association for Computing Machinery.

Shu Li got his BS from Beijing University (China) in 1990, MS from Chinese Academy of Science in 1993, and Ph.D. from University of Pittsburgh (USA) in chemistry in 1999. He started doing researches in network problems in the department of Computer Science at the University of Pittsburgh in 1999. He got his M.S. from University of Pittsburgh (USA) in computer science in 2000. He is now a development engineer (ShuL@CoManage.net) with CoManage Corporation (www.comanage.net).

Taieb F. Znati obtained a Ph.D. Degree in Computer Science at Michigan State University, East Lansing, in April 1988, and a Master of Science Degree at Purdue University, West Lafayette, Indiana. In 1988, he joined the University of Pittsburgh where he currently is a professor in the Department of Computer Science with a joint appointment in Telecommunications in the Department of Information Science. His current research interests focus on the design of network protocols for wired and wireless communication networks to support multimedia applications' QoS requirements, the design and analysis of medium access control protocols to support distributed real-time systems, and the investigation of fundamental design issues related to distributed systems. He is the general chair of CNDS'00 and the 33rd Annual Simulation Symposium and as a program chair of numerous conferences and workshops in networking and distributed multimedia systems. He is frequently invited to present lectures and tutorials in networking and distributed multimedia related topics, in the United States of America and abroad. Dr. Znati is a member of the Editorial Board of the International Journal of Parallel and Distributed Systems and Networks, a member of the Editorial Board of the Journal on Wireless Systems and Mobile Computing, a member of the Technical Program Committee of Infocom 20001, and a member of the Steering Committee and Advisory Committee of the PDCS 2000. He is currently on leave from the University of Pittsburgh to serve as a senior program director for Networking Research at the National Science Foundation.