A distributed routing and wavelength assignment algorithm for real-time multicast in WDM networks

Chuanhe Huang\textsuperscript{a}, Xiaohua Jia\textsuperscript{a,\*}, Yongbing Zhang\textsuperscript{b}

\textsuperscript{a}Institute of Computer Science, Wuhan University, People's Republic of China
\textsuperscript{b}Institute of Policy and Planning Science, University of Tsukuba, Japan

Received 6 April 2001; revised 19 December 2001; accepted 28 January 2002

Abstract
Routing and wavelength assignment for online real-time multicast connection setup is a difficult task due to the dynamic change of availabilities of wavelengths and the consideration of wavelength conversion delay in wavelength division multiplexing (WDM) networks. This paper presents a distributed routing and wavelength assignment scheme for the setup of real-time multicast connections. It integrates routing and wavelength assignment as a single process, which greatly reduces the connection setup time. The proposed routing method is based on the Prim's minimum spanning tree (MST) algorithm and the \(K\)-restricted breadth-first search method, which can produce a sub-minimal cost tree under a given delay bound. The wavelength assignment uses the least-conversion and load balancing strategies.

1. Introduction
Wavelength division multiplexing (WDM) is a key technology to explore the tremendous network bandwidth. It divides the bandwidth on a fiber into many non-overlapping and parallel channels, each of them operates at a different wavelength. WDM network is a connection-oriented technology. A connection must be established between a pair of communication nodes before data transmission occurs. Connection establishment in WDM networks involves two operations: routing and wavelength assignment (RWA).

Multicast is such a communication means, by which a sender (source) sends a message to a group of receivers (destinations) simultaneously. Real-time multicast is a special kind of multicast, which requires that the communication time from source to any destination should not exceed a given delay bound. Real-time multicast has a wide range of applications in modern computer networks, such as, video conferencing, multimedia education systems, video on demand (VOD) systems, and distributed database systems.

The routing for a real-time multicast connection is to find a tree that is rooted from the source and contains all the destinations. The delay from the source to any destination along the tree should not exceed the required bound. An important requirement of such a routing tree is that the total cost of the edges in the tree is minimal. Finding such a tree is NP-hard [3,17]. A lot of research has been done on the heuristics of minimum cost and delay-bounded multicast routing trees [19].

There are several difficulties in the connection establishment for real-time multicast connections in wide area WDM networks:

1. A network node only knows the availability of wavelengths on the links incident to it. There is no such a node, which has the global knowledge about the network topology or wavelength availability of the whole network.
2. It is unknown whether there is a wavelength conversion at a node until the routing request reaches the node, and wavelength conversion delay is not negligible (sometimes significant).
3. The delay and cost factors are independent from each
other. A path with least cost may require a long delay, and vice versa.

This paper presents a distributed and sender-initiative algorithm of routing and wavelength assignment for the establishment of a real-time multicast connection in WDM networks. The algorithm first constructs a minimum cost tree to link those destinations to which the delay bound is met. Then, a K-restricted breadth-first search is used to construct a shortest delay tree for the remaining destinations. The two trees are finally merged into a single routing tree. The proposed method has the following advantages:

1. It is fully distributed. The routing and wavelength assignment is done based on local information of a node.
2. It tries to incur less wavelength conversions in a multicast connection.
3. The routing tree has sub-minimal cost under the bounded delay constraint.

2. Problem formulation

The network is modeled as an undirected graph $G(V, E)$, where $V$ is the set of nodes and $E$ is the set of fiber links. Each link $(i,j)$ is associated with three parameters:

- $\sigma_{ij} \subseteq \{1, 2, ..., W\}$, the set of available wavelengths on link $(i,j)$.
- $c_{ij}$, the cost of using link $(i,j)$.
- $d_{ij}$, the delay on link $(i,j)$.

Available wavelengths on link $(i,j)$ change dynamically. Only the end nodes of the link know the exact value of $\sigma_{ij}$. When the connection is setup, a free wavelength will be assigned to each link in the routing tree, and it will be occupied until the termination of the connection.

The communication delay on a path consists of two components: link delay and wavelength conversion delay. The link delay $d_{ij}$ represents the delay that signal transmits from node $i$ to $j$ on link $(i,j)$. The wavelength conversion delay, denoted by $d_{c}(\lambda_x, \lambda_y)$, is the time to convert wavelength $\lambda_x$ to wavelength $\lambda_y$ at node $i$. This delay is assumed to be the same at any node for any wavelengths. If there is no wavelength conversion performed, i.e. $\lambda_x = \lambda_y$, then $d_{c}(\lambda_x, \lambda_y) = 0$. We assume that every node has a full-range wavelength converter, which is able to convert a wavelength to any other wavelengths.

We consider an online request for multicast connection setup, $R = (s, D, \Delta)$, where $s$ is the source node, $D$ the set of destinations, and $\Delta$ the delay bound. The route of a multicast connection is a tree $T$. The total cost of $T$ is defined as

$$\text{COST}(T) = \sum_{(i,j) \in T} c_{ij} \quad (1)$$

Let $P(u,v)$ denote the path from node $u$ to $v$ in $T$. The delay between $u$ and $v$ along $T$, denoted by $\text{DELAY}(u,v)$, can be represented as

$$\text{DELAY}(u,v) = \sum_{(i,j) \in P(u,v)} d_{ij} + \sum_{i \in P(u,v), j \neq u,v} d_{c}(\lambda_i, \lambda_j) \quad (2)$$

The delay from $s$ to a node $v$ along the tree is $\text{DELAY}(s,v)$. The delay of $T$ is defined as

$$\text{DELAY}(T) = \text{MAX}\{\text{DELAY}(s,d), \forall d \in D\} \quad (3)$$

The delay constraint can thus be represented as

$$\text{DELAY}(T) \leq \Delta \quad (4)$$

The problem of concern is to design a distributed routing and wavelength assignment algorithm to construct a wavelength-routed tree, whose cost is minimal under the delay constraint defined in Eq. (4).

3. Related work

Minimum cost multicast tree is a Steiner tree. Adding delay bound constraint, the problem of finding a Steiner tree with delay bound is NP-hard [17]. Only heuristics to find sub-optimal solutions are sought.

Many research results have been presented about constructing optimal multicast trees [19], most of which are based on minimum spanning tree (MST) heuristics. For example, Prim MST algorithm, Kruskal MST algorithm, Sollin MST algorithm [18]. These algorithms are suitable to constructing full spanning tree. Zhu presented an optimization multicast algorithm with delay bound constraint in Ref. [20]. The algorithm defines the link utilization as cost. It first computes shortest path tree, and then replaces the highest cost edge with a lower-cost but satisfying-delay-constraint edge until the total cost cannot be decreased. Kompella [21], Guo [22] presented QoS multicast algorithms, which first transform the delay into weighted cost, and then use Prim algorithm to construct the trees with minimal weights. Jang-Jin Wu presented a genetic multicast algorithm McMGA in Ref. [23]. All these algorithms can only be used to solve the problem for static multicast connections in conventional networks. Jia in Ref. [24] presented a distributed multicast algorithm to find delay bounded multicast trees, but the link delay and cost must be in proportion. Kompella [25], Low [26] presented distributed multicast algorithms using static metric, the latter improved the former. But message complexity is high ($O(n^3)$) and total cost of the tree is not always sub-optimal. In Ref. [27], Asaka presented a receiver-initiative distributed algorithm by searching predefined point-to-point routes, and evaluated its performance. These results cannot be applied to WDM networks directly.

Current research about routing and wavelength assignment in WDM networks mainly focuses on point-to-point communication (unicast). It is usually assumed that the
route is fixed or the route is computed firstly and then wavelengths are assigned. For example, Banerjee [1], Dutta [2] presented a virtual topology (also named logical topology) design approach, i.e. linear programming approach. Ramaswami [3], Subramaniam [4], Mokhtar [5] gave several wavelength assignment strategies and discussed their impact on the performance on predefined routes. Spath gave several routing strategies in Ref. [6]. Chlamtac [7] and Liang [8] presented wavelength graph methods of unifying the processes of routing and wavelength assignment. Jia presented a k-cut method to minimize wavelength conversion for static requests in Ref. [9]. Sinclair presented a genetic algorithm to assign wavelength in Ref. [10]. Zhu compared several wavelength assignment strategies in Ref. [11].

In WDM networks, the routing should base on the availability of wavelengths on each link. Sahasrabuddhe presented the concept of light tree in Ref. [12], whose basic idea is to predefine some trees and assign wavelengths. When a request for multicast connection setup arrives, it is adapted into one predefined tree. But the light trees cannot always accommodate the request because of dynamic characteristics of multicast connection requests and uncertainty of multicast groups. Li [13], Malli [14], Pankaj [15] presented, respectively, methods to compute routing trees with minimum number of required wavelengths. Jia presented a routing approach for a group of static multicast requests in Ref. [16]. The approach includes four algorithms, which constructs static trees (Algorithm A), assigns wavelengths by using graph-coloring method (Algorithm B), optimizes trees and assigned wavelengths (Algorithm C and D), respectively.

Most of the existing work about multicast in WDM networks considers routing (constructing tree) and wavelength assignment independently, or assumes the metrics and network state are static and remain unchanged or the network has a special topology. The existing methods are not suitable for on-line setup of multicast connections. New methods are needed to cope with the difficulties.

4. The distributed algorithm

4.1. Overview of the proposed algorithm

The proposed distributed algorithm works in the sender-initiative way. It consists of two procedures: GenCtree and GenDtree. The procedure GenCtree constructs a minimum cost tree, Ctree, based on the Prim’s MST algorithm. Each time, the destination nearest to the tree (in terms of cost) is selected and linked to the tree if the delay bound constraint is satisfied. The construction of the routing tree terminates when all destinations are linked to the Ctree. Otherwise, the connection setup fails due to the failure of meeting the delay constraint for some of the destinations. In this case, procedure GenDtree is called to construct a shortest-delay tree, Dtree, to link all the remaining destinations by a K-restricted breadth-first search method.

After Ctree and Dtree are constructed, they are merged into one tree. During the merging operation, cycles may emerge. The algorithm eliminates cycles by removing some links in Ctree to ensure the final routing tree meeting the delay constraint.

4.2. Discussion of the algorithm

4.2.1. Data structures and notations

The following are the data structures and notations used by the algorithm.

Routing Table. Two routing tables, CRoutab and DRoutab, are kept at each node for cost and delay routing, respectively. An entry of the tables, CRoutab[d][i] / DRoutab[d], contains a set of possible outgoing links and the corresponding least cost/delay to node d from this node. The first outgoing link in a table entry is the primary path, and the remaining outgoing links are secondary paths. The tables can be constructed and maintained by using the distance vector routing scheme.

Distance from a Destination to a Tree. During the tree construction, a list of 3-tuples (treenode, dest, dist), one tuple for each destination, is used to keep track of the shortest distance from each destination to the tree constructed so far. A tuple (treenode, dest, dist) represents that the closest tree node to destination dest is treenode with distance dist.

Number of Links of a Free Wavelength. Each node records the available wavelengths on the links incident to it. For wavelength λ, NL(λ) denotes the number of links on which λ is available at the node.

4.2.2. Construction of Ctree

The routine GenCtree starts at the source node. The source node first selects the nearest destination (in terms of cost), and to link it into the Ctree. To find the right path from s to the selected destination, s selects one outgoing link leading to the destination and having a free wavelength. When the primary outgoing link cannot be used, it selects a secondary outgoing link. After the selection, it sends a CFIND message to the next-hop node via the selected link. Then, this next-hop node selects an outgoing link leading to the designated destination using the same method until the message reaches the destination. The CFIND message collects information about shortest distance from the tree to each remaining destination. The destination then selects the next destination that is closest to the Ctree and finds the corresponding treenode in the tuple of (treenode, dest, dist). It sends a message to the fork node treenode to start a new path setup. Each time when the CFIND message reaches a destination, this destination is responsible for selecting the next destination to be linked to Ctree. When a destination is failed to be linked to the Ctree due to the failure of meeting the delay bound, the node that finds failure will try another
destination. This operation is repeated until no more destinations can be linked to the Ctree, where procedure GenDtreen will be called (see Section 4.2.3).

The criterion that the procedure GenCtree uses to select a destination is the shortest distance in terms of cost between Ctree and a destination d, defined as

$$\text{DIST}(Ctree, d) = \min\{\text{COST}(t, d)\}, \quad \forall t \in Ctree, d \in D-Ctree$$

(5)

The first link of the shortest path in terms of cost is selected as the outgoing link. If this link does not meet the wavelength or delay requirement, the next outgoing link is tried. The corresponding delay to adjacent node w from the source s through this node v, DELAY(s, w),

$$\text{DELAY}(s, w) = \text{DELAY}(s, v) + d_{vw} + d_{v}(\lambda_{v}, \lambda_{w})$$

(6)

is computed and checked at node v.

When it is found at a node that the accumulated delay exceeds the pre-specified bound or there is no free wavelength of going out, the node selects the next destination to start a path setup, and releases the reserved wavelengths along the failed path.

In order to reduce the wavelength conversion along the route, a node always tries to choose the wavelength on an outgoing link to be the same as that of the input link. If this wavelength is unavailable, the wavelength with the largest possible outgoing links in the network and reserved wavelengths. GenDtreen uses breadth-first search and Dijkstra’s shortest-path marking method to extend the tree. The DFIND message is forwarded until the given destination is included in Dtree, or a node cannot be included because of delay bound violation or wavelength unavailability.

At an intermediate node v, the criterion to select an outgoing link is the least delay to the given destination d through the possible adjacent node w. The selection function is defined as

$$f(w) = \begin{cases} 
\text{DELAY}(v - w, d), & \text{DELAY}(s, v) + \text{DELAY}(v, d) \\
+d_{v}(\lambda_{v}, \lambda_{w}) \leq \Delta \\
\infty, & \text{otherwise} 
\end{cases}$$

(8)

When a node is visited by a DFIND message the second time, GenDtreen adopts the following method to handle the duplicated message:

1. If the node has been visited by a previous DFIND message, and this DFIND message comes from a link different from the previous one, then it removes the input link with the longer delay.
2. If one or more child nodes and the corresponding links have been included in Dtree by a previous DFIND message, then GenDtreen tests the delay to those adjacent nodes and decides if they should be included in Dtree according to if the new delay is lower than the original one. When an old link of Dtree is removed, the branch is pruned.
3. Each destination that is successfully linked into Dtree sends a DFOUND message to source s to inform the advance.

The wavelength assignment strategies used by GenDtreen are:

1. If the selected link is also a link of Ctree, it uses the same wavelength assigned for that link of Ctree.
2. If the selected link is already an existing link of Dtree, it may reassign a wavelength as needed.
3. If the selected link is a new link, it uses the same strategy as GenCtree does.

4.2.4. Merging of Ctree and Dtree

After Ctree and Dtree are constructed successfully, Ctree
and Dtree are merged together as a single tree. Source node $s$ sends a MERGE message along all links in Ctree and Dtree. The message is processed node by node along the trees until a tree leaf is reached. The detailed merging algorithm is as below:

1. If Ctree and Dtree meet at node $x$, and the two input links belonging to Ctree and Dtree, respectively, are different, then the input link of Ctree is removed. The accumulated delay at the down-stream nodes of $x$ in the Ctree needs to be re-computed.
2. If the delay along the new tree violates the delay bound, the algorithm fails and stops.
3. If a link of Ctree is removed, all the up-stream links in Ctree should be removed along the reverse direction toward the source until a destination or a fork node is met.

From above algorithm, we can see that the merging operation may lead the final routing tree an invalid tree, in which some nodes violate delay bound due to wavelength conversion delay. The increased wavelength conversion delay results from the different wavelengths on the input link belonging to original Dtree and the output link belonging to original Ctree.

In the algorithm, the connection information is stored at the local node, each node (apart from source and destination) only stores information about input and output links.

### 4.3. Illustration of our algorithm

In this section, an example is used to illustrate our algorithm. The example network is in Fig. 1. $R = (F, \{B,D,E,H\}, 20)$, the conversion delay at every node is $\delta_{xy}(\lambda_s, \lambda_y) = 4$. $\alpha_{AB} = [4]$ , $\alpha_{FA} = [1, 2, 3]$ , $\alpha_{AE} = [1, 2, 3]$ , $\alpha_{BG} = [4]$ , $\alpha_{BC} = [2, 4]$ , $\alpha_{CD} = [1, 2, 3]$ , $\alpha_{CG} = [3]$ , $\alpha_{DF} = [2, 3, 4]$ , $\alpha_{DE} = [3, 4]$ , $\alpha_{BH} = [4]$ , $\alpha_{BG} = [2, 3]$ , $\alpha_{BH} = [3, 4]$.

Source F computes the total number of free wavelengths on all incident links, yielding NL($\lambda_3$) = 3, NL($\lambda_2$) = 2, NL($\lambda_4$) = 2, NL($\lambda_1$) = 1. The static shortest distance in terms of cost to destination B, D, E, H is 11, 6, 3, and 5, respectively. F first selects E as the destination to route. The selected outgoing link is (F, A) and assigned wavelength is $\lambda_3$. After E is added, the Ctree contains path F–A–E as shown in Fig. 2. The number marked in square bracket beside a node number is the input delay of the node from source node F. Then, E selects H as the next destination to be added, with tree node E as the fork node, and path E–H is added. Wavelength conversion is performed at node E, as shown in Fig. 3. The gray node indicates that the node performs wavelength conversion. H selects F–D as next path and it is added successfully as shown by Fig. 4. The costs of paths D–C–B, H–G–B, A–B are the same (equals 10), but D–C does not perform wavelength conversion. Thus, D selects destination B and outgoing link (D, C) to be added. The node B cannot be added because of violating the delay bound. Procedure GenDtreetree is then called, tree Dtree is constructed as shown in Fig. 5, where $K = 1$ and the numbers labeled on links are the wavelengths assigned. When Ctree and Dtree are finally merged together, link (E, H) is removed. The final tree is shown in Fig. 6.

When constructing Dtree, there are two cases when processing a DFIND message. The first case is that it is the first time that a DFIND message arrives at this node. Its processing is simple and direct. The second case is that this node has received a DFIND message before. Under such condition, one scenario is that the message comes from the same link as the previous one, as node y shown in Fig. 7. When DELAY($s, x$) + $d_{xy}(\lambda_s, \lambda_y) = \Delta$, then it continues to forward DFIND message. The remaining nodes in Dtree may receive the DFIND message more than once as
this node does. Another scenario is that DFIND message comes from a different link from the previous one, as node \( x \) in Fig. 8. At node \( x \), GenDtree may select \((x,w)\) to extend Dtree. Because \((x,v)\) is linked in Dtree by a previous DFIND message, and it must be re-checked. This may add \( w \) to Dtree, which may make \( v \) and its down-stream nodes violating the delay bound due to the conversion from \((y,x)\) to \((x,v)\) at \( x \).

When merging Ctree and Dtree, cycles may emerge, as shown in Fig. 9. To eliminate the cycle, node \( y \) sends a message to \( b \) to remove tree link \((b,y)\). The path \( s\rightarrow b\rightarrow y\rightarrow w \) is changed to \( s\rightarrow a\rightarrow x\rightarrow y\rightarrow w \). This may make node \( w \) violating the delay bound due to possible conversion performed at node \( y \) from \((x,y)\) to \((y,w)\). The algorithm will return a failure in this case.

4.4. Algorithm analysis

**Theorem 1.** The tree constructed by the proposed algorithm includes source node and all destinations. It satisfies constraint (4), and has sub-minimal cost.

**Proof.** It is obvious that the final tree includes source and all destinations.

Each step one link and corresponding node is added, the path delay from source \( s \) to a newly added node will not exceed delay bound. So each (source, destination) path delay in the final tree satisfies delay bound constraint (4). Because procedure GenCtree adds a link of minimal cost path to the tree each step. Eventually, a minimal cost path from the tree to a destination is added to the tree, only GenDtree adds non-minimal cost paths. Based on the Prim’s algorithm, the total cost is sub-minimal.

What GenDtree does is to look for a shortest delay path in parallel for the remaining destinations and construct a shortest delay path tree. The merging of Ctree and Dtree usually generates a single tree and it is the solution.

Improper choice of the initial wavelength or the wavelength assignment strategy may lead to a lot of wavelength conversions. Thus, the path delay may exceed the delay bound. Though the algorithm tries to substitute some links or paths when possible, because it does not try to reassign wavelengths for the formed tree, it is still possible that the algorithm cannot find a tree. When merging Ctree and Dtree, some links of Ctree is possibly removed, and this may lead the merged tree invalid, and the algorithm does not retry to search alternate path for removed destinations. So, it is possible that the algorithm cannot find a solution, though there exists one. Because the algorithm uses a heuristics, i.e. least-conversion and least-load, to assign wavelength for each link, this scenario can be avoided usually.

5. Simulations

The purpose of the simulations is to demonstrate the quality of the routing trees generated by our algorithm.
The network graphs used in the simulations are constructed by using the approach given in Ref. [28]. The nodes are distributed randomly over a rectangular coordinate grid. Each node is placed at a location with integer coordinates. A link between two nodes $u$ and $v$ is added by using the probability function $P((u,v)) = \lambda \exp(-d(u,v)/\rho L)$, where $d(u,v)$ is the distance between $u$ and $v$, $L$ is the maximum distance between any two nodes, and $0 < \lambda \leq 1$, $0 < \rho \leq 1$. Larger values of $\lambda$ result in graphs with higher link densities, while small values of $\rho$ increase the density of short links relative to longer ones. In our simulations, $\rho$ is set to 0.6, $\lambda$ is set to 0.5, the degree of any node is limited to not exceeding 5. The network size is fixed at 200 nodes. Graphs are generated and tested until a connected graph is found. Link cost is randomly generated and ranges between 1 and 15, link delay is in between 1 and 10. Number of wavelengths on every link is 8. In our simulations, except special circumstances, $\Delta = 0.2$ of the network size, the wavelength availability on each link is set to 50%, $K = 1$. The topology does not change throughout the simulations.

Three algorithms are simulated and compared: SPT, MST, and our algorithm dRWA. MST is computed by using our algorithm with no real-time constraints. The network cost is simulated against four parameters: $\Delta$, $|D|$, wavelength availability and relative conversion delay (defined later). Establishment time is simulated against $|D|$. Relative number of conversions is simulated against the wavelength availability.

Fig. 10 shows the network cost versus real-time bound $\Delta$. Conversion delay is set to 0.5 times of the average link delay. During the simulation, traffic load of the network is not considered. The minimum meaningful delay bound is defined as $\Delta_{\text{min}} = \max\{(\text{DELAY}(s,d), \forall d \in D)\}$ + (average number of conversions) × (conversion delay). DELAY$(s,d)$ is the shortest path delay from $s$ to $d$. The average number of conversions is set to half of the network diameter. The $\Delta$ value starts from $\Delta_{\text{min}}$, incremented by $\Delta_{\text{min}}/4$ each time. In Fig. 10, the curves of SPT and MST remain constant, because both of them are not restricted by $\Delta$. The curve of our method is in between the curves of the SPT and of the MST. Its high end is close to the SPT’s curve and the low end close to the MST’s curve. With a smaller $\Delta$, more MST paths fail and are replaced by the SPT paths. This makes the routing tree wider (bush-like), thus a higher network cost. As $\Delta$ increases, more destinations are linked into the tree via MST paths, which results in the decrease of the network cost. When $\Delta$ is large enough that it does not restrict routing any more, the final routing tree becomes an MST.

Fig. 11 shows the network cost versus group size. The value of $\Delta$ is set to $\Delta_{\text{min}} + \Delta_{\text{min}}/4$. Group size is always made not greater than 30% of the total nodes, because multicast applications running in a wide area network usually involve only a small number of nodes in the network. As group size increases, a routing tree contains more destinations, resulting in an increase of the network cost. The SPT’s curve is above the other two and rises much faster than the others. This is because SPT does not consider any path sharing. The performance of our method is close to that of the MST’s. The curves of our method and the MST’s rise slower as the increase of group size (the rise is not linear). This is because the destinations in a bigger group have a higher probability of path sharing.

Fig. 12 shows the network cost versus the average wavelength availability. The value of $\Delta$ is fixed to $\Delta_{\text{min}} + \Delta_{\text{min}}/4$. Group size is always made not greater than 30% of the total nodes, because multicast applications running in a wide area network usually involve only a small number of nodes in the network. As group size increases, a routing tree contains more destinations, resulting in an increase of the network cost. The SPT’s curve is above the other two and rises much faster than the others. This is because SPT does not consider any path sharing. The performance of our method is close to that of the MST’s. The curves of our method and the MST’s rise slower as the increase of group size (the rise is not linear). This is because the destinations in a bigger group have a higher probability of path sharing.
As the wavelength availability increases, more wavelengths on one link are available, and more links in the network are usable for the routing tree, less wavelength conversion is performed, and this results in a decrease of the network cost. The simulation result shows that when the wavelength availability is greater than about 30% on each link (average about 2.4 free wavelengths), the network cost is not influenced significantly by the available wavelengths. In other words, when the load is not greater than 70%, the routing is not influenced significantly by the load.

Fig. 13 shows the network cost versus the relative conversion delay. The relative conversion delay is defined as the rate of conversion delay to the average link delay of the whole network. The value of \( \Delta \) is set to \( \Delta_{\text{min}} + \Delta_{\text{min}}/4 \). The result shows that the network cost of the SPT’s and our algorithm increases very much slowly with the increase of relative conversion delay, the network cost of the MST’s is constant. The curve of our algorithm is almost merged with the curve of the MST’s. This reflects the fact that the conversion delay influences mainly the success rate of the routing but not network cost when the delay bound is fixed. The effect of increasing relative conversion delay is somewhat similar to that of decreasing \( \Delta \).

Fig. 14 shows the establishment time against \( |D| \). The value of \( \Delta \) is set to \( \Delta_{\text{min}} + \Delta_{\text{min}}/4 \). The establishment time consists of the delays of the three major steps of routing: the traversing time of the CFIND message (i.e. the time for the construction of the Ctree), the traversing time of the DFIN message (i.e. the time for the construction of the Dtree) and the traversing time of the MERGE message (i.e. the time for merging the Ctree and the Dtree). As \( |D| \) increases, the time needed to establish the connection increases. This is because for a larger multicast group, the routing tree that contains all nodes in \( D \) becomes wider and deeper. It takes more time to construct Ctrees and Dtrees in the distributed fashion. However, from the curve in Fig. 14, we can see the increase of establishment time is much slower than that of \( |D| \).

Fig. 15 shows the relationship between the relative number of conversions and the wavelength availability. The relative number of conversions is defined as the rate of the total number of conversions to total links of the routing tree (accordingly, the total number of wavelengths used). The value of \( \Delta \) is set to \( \Delta_{\text{min}} + \Delta_{\text{min}}/4 \). As the wavelength availability increases, more wavelengths on links are available, and less wavelength conversions are performed. The simulation result shows that our algorithm performs less conversion than MST does, but more than SPT does. When the wavelength availability is low, the probability of performing conversion is very high, and this leads to a high probability of routing failure due to conversion delay.

The simulation results show that the routing trees generated by our algorithm have much less network cost than SPT. They also suggest that SPT is not appropriate to be used as routing trees in WDM networks, especially in the case where the available wavelengths are scarce and the group size is large.

6. Conclusions

This paper presents a scheme for the establishment of real-time multicast connections in WDM networks. It can construct a sub-minimal cost multicast tree under the pre-specified delay bound, with the consideration of wavelength conversion delay along the route. The proposed scheme is fully distributed.

As the future work, two topics require to be further investigated. The first topic is to reduce the connection establishment time. In the proposed method, it takes three steps sequentially to setup a connection, namely construction of Ctree, construction of Dtree and the merge of the two trees. The establishment time is relatively long. A method that either reduces the steps or conducts the three steps in parallel will be more desirable. The second topic is to consider the networks with limited wavelength conversion...
capability. In the proposed method, we assume every node has full-range of wavelength conversion capability. Since wavelength converters still remain to be expensive devices, it is too costly to equip every node with a full-range wavelength converter in wide area networks (WAN). Even in the case when a node is equipped with a converter, the wavelength conversion capability is usually limited (not full-ranged). Delay bounded multicast routing in such networks with limited wavelength conversions becomes much more complicated and needs to be further investigated.

Acknowledgments

The authors wish to thank the anonymous referees for their very valuable comments and constructive suggestions that helped improve the quality of the paper greatly. This work is supported in partial by CityU SRG grant 7001121, JSPS, and Cheung Kong Scholars scheme at WuHan University, China.

References