

# Wavelength Scheduling in Time-domain Wavelength Interleaved Networks

Yan Li, Sanjay Ranka and Sartaj Sahni

Department of Computer and Information Science and Engineering

University of Florida, Gainesville, Florida 32611

Email: {yanli, ranka, sahani}@cise.ufl.edu

**Abstract**—Time-domain Wavelength Interleaved Networking (TWIN) is a new optical network architecture which achieves a good balance between scheduling flexibility and deployment cost. In this paper, we solve the wavelength assignment problem for TWIN networks using topology sharing approach. We show that determining the wavelength assignment that use the minimum number of wavelengths is a NP-Complete problem. Four greedy heuristics are presented to compute the approximated solution within reasonable time. The evaluation results show that performing sorting on destination trees and wavelengths improves the assignment results, especially under low traffic loads. However, performing sorting brings some extra overheads to the sort heuristics' running time, but overall computation costs are still acceptable.

## I. INTRODUCTION

Time-domain Wavelength Interleaved Networking (TWIN) is an optics-based transport network architecture that aims to provide cost effective optical grooming [1]–[3]. Traditional optical networks work in one of the following two modes: optical circuit switching (OCS) or optical packet switching (OPS). In OCS networks, the finest bandwidth granularity offered by an optical switch is at a wavelength level, i.e. one single wavelength on a fiber can be used by only one end-to-end traffic and cannot be shared with other traffic. This is not effective when the traffic demand is much lower than the wavelength capacity. At the other extreme, OPS networks permit sharing of optical links by traffic with different sources and destinations. These networks, which are enabled by optical-electronic-optical (OEO) conversion at each node in the network, tend to incur a relatively high system cost and transmission delay, as OEO converters are generally expensive and the conversion process is time-consuming comparing to direct circuit switching. Some techniques have been introduced to improve the utilization of optical links by simulating OPS over OCS, such as optical burst switching (OBS) [4]. However, OBS still needs high-speed optical switches and a contention algorithm at each switch.

Widjja et al. etc. proposed TWIN to overcome link utilization problems in OCS but avoid the high cost and delays resulting from OEO converters being deployed at all the optical switches [1]. TWIN performs optical grooming only at its edge switches and the network core is purely based on passive wavelength-selective switches (WSS) that route the wavelengths from their ingress ports to the appropriate egress ports [5]. In a TWIN network, the edge nodes can be either sources or destinations. A transmitter with a multi-frequency laser is located at each source node. With this transmitter, source nodes can change the wavelength of their optical signal in sub-nanoseconds

[6]. Source nodes collect data units from various clients and assemble data units for the same destination into one burst. When sending the burst, the source changes its fast-tunable laser to the wavelength uniquely assigned to that destination. The intermediate nodes route optical bursts based purely on the wavelength of the burst. When the burst is received at its destination, it is disassembled and forwarded to the corresponding clients. As current optical switches cannot separate the bursts that share the same wavelength, only traffic with the same destination may share a wavelength in the time-domain. This constraint leads to tree-like routes in the network for every destination, where the destination is the root and the sources are the leaves.

In this paper, we discuss the wavelength assignment problem for TWIN networks (TWIN-WA). We solve this problem using a two-phase process: *Tree Construction* and *Tree-Wavelength Assignment*. *Tree-Construction* groups the traffic demands with the same destination together and constructs the corresponding destination trees. *Tree-Wavelength Assignment* process assigns wavelengths to the destination trees constructed in the previous step. The goal of the *Tree-Wavelength assignment* phase is to minimize the total number of wavelengths needed to accommodate the traffic demands.

In this paper, we show that the minimum number of destination trees can be constructed using a greedy approach in the *Tree Construction* phase. For the *Tree-Wavelength Assignment* problem, we prove its NP-Completeness by reducing the *Graph-Coloring* problem to it. We propose a greedy strategy that matches destination trees and wavelengths one by one. We also proposed two tree sorting methods and two wavelength sorting methods to regulate the order of tree-wavelength matching. When different tree sorting and wavelength sorting methods are applied to the tree-wavelength assignment scheme, four heuristics are presented: *MC-BF*, *MC-MF*, *MP-BF* and *MP-MF*. Extensive simulations are conducted to evaluate the performances of these heuristics. The results show that performing sorting on destination trees and wavelengths improves the assignment results, especially under low traffic loads. However, performing sorting brings some extra overhead to the sort heuristics' running time, but overall computation costs remain acceptable. In large topologies with heavy workload, the heuristic without any sorting becomes competitive as it can provide similar scheduling performance with much less computational cost.

The rest of this paper is organized as follows. In Section II, we discuss related work. In Section III, we explain the TWIN

architecture in detail and define the TWIN-WA problem formally. In Section IV, a greedy algorithm for *Tree-Construction* is presented. In Section V, we prove the NP-Completeness of *Tree-Wavelength assignment* problem and four heuristics are discussed. Section VI presents an experimental evaluation of the four heuristics for *Tree-Wavelength assignment*. Section VII gives the conclusions.

## II. RELATED WORK

Optical Circuit Switching (OCS) with wavelength-dimension multiplexing (WDM) [7] provides the most economical solution for high speed optical networks. However, the inflexible routing scheme and coarse multiplexing granularity make it only suitable for the long-lived large bulk data transfers. On the other hand, Optical Packet Switching (OPS) [8] and Optical Burst Switching (OBS) [9] have been proposed to provide sub-wavelength scheduling granularity and the capability of dynamic routing. However, the ultra-high speed optical-electronic-optical switches that are required in the OPS/OBS networks are normally expensive and difficult to maintain. The high cost in deployment and maintenance inhibit the use of OPS and OBS in modern networks.

Time-domain Wavelength Interleaved Networking (TWIN) has been proposed to fill the gaps between OCS and OPS/OBS. The architecture of TWIN is introduced in [1]. The goal of TWIN is to provide sub-wavelength granularity for traffic scheduling without using expensive high speed optical switches in the networks. TWIN achieves this by only allowing light paths with the same destination to share a wavelength. As the traffic on the same wavelength will not be split again, the economical switches used in OCS networks are able to route the bursts in TWIN networks. TWIN brings new challenges to traditional optical scheduling approaches. [10] presents some basic ideas in the routing the burst scheduling in TWIN networks and proposed a performance measurement framework. [3] investigated the optical burst scheduling problem in TWIN networks. They show that achieving the maximum throughput with zero propagation delay is equivalent to the optimal matching problem in bipartite graphs. They also demonstrate that even when propagation delay is non-negligible, a factor-2 approximate scheduling algorithm exists to maximize the throughput. Meanwhile, [11] focused on the providing better QoS in TWIN networks. They introduced an Integer Linear Programming formulation that minimize the queueing delay of the optical bursts. They also proposed the Destination Slot Set (DSS) algorithm to approximately solve the problem within reasonable time.

In this paper, we focus on the wavelength assignment problem for TWIN networks. Traditional wavelength assignment strategies take the available wavelength number as the main constraint. However, as fractional wavelength is allowed, and the general traffic flow is assumed in sub-wavelength level, TWIN wavelength assignment (TWIN-WA) is relaxed from the integer capacity constraint in traditional networks. The main concern in TWIN-WA is the conflict on topologies among multiple destination trees when they share one wavelength. Moreover, instead of assigning wavelength to each light path, TWIN assigns wavelengths to a destination tree. Traditional

wavelength assignment problem are normally equivalent to the *Bin-Packing* problem [12]. However, TWIN networks' wavelength assignment problem is a variation of the *Graph-Coloring* problem [13], as shown in Section V-A.

[14]–[16] together provide a summary on the existing wavelength assignment strategies. The most popular wavelength assignment strategies are *First-Fit* and *Best-Fit*, where the wavelengths are matched with the request according to a random order or to their remaining capacities. [14] proposed a *deferred wavelength assignment* strategy for optical networks with wavelength converter provides. This strategy improves the request accepting rate by deferring the wavelength assignment from scheduling time to actual job start time. [17] proposed the *least-conversion assignment* scheme that attempts to reduce the wavelength conversion overhead in sparse wavelength converter networks.

In TWIN networks, the simplest wavelength assignment strategy is to assign each destination tree an individual wavelength. However, this strategy requires that the number of available wavelengths be equal to the number of destination trees, the internal switches are unable to separate the traffics to different destination trees on the same wavelength. When the number of destination trees are more than the available wavelengths, we need to establish some mechanism so that multiple destination trees can share one single wavelength without collision. This process is denoted as *wavelength reuse*. Most existing wavelength reuse algorithms are Time Dimension Multiplexing based [2], [18]. However, the traffic flows in their scenarios cannot be transmitted at the max rate as the link capacity is shared by different destination tree in different time slices. This constraint may not be necessary in many scientific and commercial applications, where the job finish time is critical for the workflow. Moreover, TDM based wavelength reuse requires comprehensive optical burst scheduling algorithms, especially when the network delay is considered. Our algorithm considers the wavelength reuse problem in a different direction: multiple destination trees share one wavelength if their topologies are compatible. In our approach, once the wavelength is assigned for the destination trees, the source nodes of all assigned traffic flows can work in a full load to transmit their bursts without worrying about the flow collisions on the optical links that shared by different destination trees, which greatly simplifies the burst scheduling.

## III. NETWORK MODEL AND PROBLEM DEFINITION

On the spectrum of optical networks, TWIN networks reside between the OCS and OPS networks. Compared to these traditional optical networks, the TWIN poses following new features:

- 1) Similar to traditional OCS networks, TWIN's data bursts travel along the light path using a pre-assigned wavelength. However, a wavelength can be shared by multiple traffic flows from different sources, only if their total flow size does not exceed the wavelength capacity.
- 2) As current optical switches cannot separate the bursts that share the same wavelength, traffic on one optical link has to be routed to the same destination if they are using the same wavelength. So TWIN light paths with

same destination are grouped together as a tree structure where the destination is the root and the sources are the leaves. Wavelengths are assigned to each of these trees, rather than to a single light path.

Figure 1 shows a simple TWIN network. Two source nodes,  $S_1$  and  $S_2$  are sending traffic to two destination nodes,  $D_1$  and  $D_2$ . A 5-node communication network connects the sources and the destinations. In the network, there are 4 different light paths:  $(S_1, D_1)$ ,  $(S_2, D_1)$ ,  $(S_1, D_2)$  and  $(S_2, D_2)$ . Before the wavelengths are assigned, these light paths are grouped into 2 tree structures according to their destination, denoted as  $T_1 = \langle (S_1, S_2), D_1 \rangle$  and  $T_2 = \langle (S_1, S_2), D_2 \rangle$ , respectively.  $D_1$  is the root of  $T_1$  while  $D_2$  is the root of  $T_2$ . The network contains two wavelengths:  $W_1$  and  $W_2$ . Each destination tree has to be assigned to a wavelength before the transmission can start. In the simplest case,  $T_1$  is assigned wavelength  $W_1$  and  $T_2$  is assigned wavelength  $W_2$ . During the transmission,  $S_1$  and  $S_2$  interleave their traffic to  $D_1$  and  $D_2$  by tuning the color of their laser to the corresponding wavelength. For each node in the communication network, a routing table is maintained to indicate the outgoing port for different wavelengths. When the traffic arrive at the internal switches, routing is performed using only the rules in the routing table and the color of the incoming bursts. This guarantees that optical bursts of a given wavelength will be routed to the intended destination. For example, in Figure 1, node  $a$  must combine the traffic from node  $S_1$  and  $d$  on wavelength  $W_1$  and forward is to the link that connects to node  $b$ , according to the routing table. Node  $b$ , after receiving the bursts on wavelength  $W_1$ , will forward them to node  $D_1$ , which is their destination.

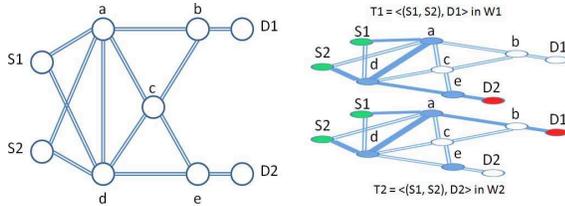


Fig. 1. An example of TWIN network.

For traffic whose required bandwidth is fraction to the wavelength capacity, TWIN networks will greatly facilitate their scheduling by providing more flexible and finer-grained routing and wavelength assignment scheme. Most optical networks still use static routing in the high speed mode as changing the routes on-the-fly incurs very high overheads. Therefore, in this paper, we also assume that the path for each source/destination pair as pre-computed, and focus our research on the wavelength assignment problem for TWIN networks (*TWIN-WA*). Given a TWIN network  $G = \langle V, E \rangle$ , a traffic demand is defined as  $r = (s, d, bw)$ , where  $s, d \in V$  is the source and destination node of the traffic flow, and  $bw \in (0, 1]$  is the fraction of the wavelength capacity required. *TWIN-WA* takes a set of traffic demands  $R$  as input. The goal is to accommodate the all demands  $r \in R$  using a minimum number of wavelengths.

As described in Section I, *TWIN-WA* is solved using a 2-step process. In the *Tree-Construction* phase, we construct the destination trees and in the *Tree-Wavelength assignment*

phase, we perform the wavelength assignment. During the *Tree-Construction* phase, the traffic demands in  $R$  are grouped together by their destination. In each group, the corresponding light paths are merged together to form a destination tree. As fractional job assignment is allowed, a simple greedy algorithm will generate the destination tree set with minimum size. *Tree-Wavelength assignment* algorithms assign each destination tree a wavelength. We show that finding an optimal assignment that uses the fewest number of wavelength is NP-Hard. Several heuristics are then proposed for wavelength assignment.

#### IV. TREE CONSTRUCTION

In a TWIN network  $G = \langle V, E \rangle$ , a destination tree for node  $D_i$  is denoted as  $T(D_i) = \langle S, D_i \rangle$ , where  $\langle S \rangle$  is the set of all source nodes in  $T(D_i)$ . Given the set of traffic demands  $R$ , we need to first construct destination trees from the light paths before we can actually assign the wavelength. This process is called *Tree-Construction*. The goal of this process is to minimize the total number of the destination trees in the result set  $T(D)$ .

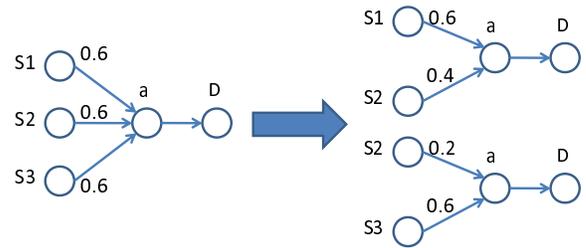


Fig. 2. An example of TWIN Tree Construction.

In this paper, we allow the traffic request  $(s, d, bw)$  to be partitioned into multiple sub-requests that can be assigned to different destination trees. This is reasonable as most modern optical switches are capable of transmitting/receiving data bursts on different wavelengths simultaneously. As long as the destination nodes are capable of package ordering and re-assembly, fulfilling one request with multiple data flows is totally feasible. On the other hand, if we simply merging all the light paths with the same destination, the resulting destination tree may not be admissible to the network, as the total flow size for one destination may exceed the wavelength capacity. Figure 2 shows an example of tree construction. Three source nodes,  $S_1$ ,  $S_2$  and  $S_3$ , are to send data to node  $D$  simultaneously. The data rate at each source node is 0.6. If we merge all 3 light paths into one destination tree, the total traffic on link  $(a, D)$  would exceed the wavelength capacity. So the demands have to be split into two separate destination trees, i.e.  $T_0(D) = \langle S_1, S_2 \rangle, D$  and  $T_1(D) = \langle S_2, S_3 \rangle, D$ . Moreover, when composing the destination trees, we should try to use up all the wavelength capacities, as the unutilized capacity cannot be shared by other destination trees. Based on the above observations, we proposed the following greedy algorithm to compute the minimum destination tree set, as shown in Figure 3

Our *Tree-Construction* algorithm first groups the traffic demands according to their destination. This can be done by

```

TreeConstruction(G, R)
{
  results = {};
  Group the traffic demands according to their destinations.
  for (each destination group  $DG(D_i)$ )
  {
    Initialize a new destination tree  $T_j(D_i)$ .
     $T_j(D_i).capacity = wavelength\_capacity$ .
     $curTree = T_j(D_i)$ .
    for+ (each traffic demands  $r$  in  $DG(D_i)$ )
    {
      Merge the light path from  $r.s$  to  $r.d$  into
       $T_j(D_i)$ .
      if( $r.bw < curTree.capacity$ )
         $curTree.capacity -= r.bw$ .
      else
      {
        if( $r.bw > curTree.capacity$ )
          Insert a new demand ( $s, d,$ 
           $r.bw - capacity$ ) into  $DG(D_i)$ .
        Add  $curTree$  into  $results$ .
        Initialize a new destination tree  $T_{j+1}(D_i)$ .
         $curTree = T_{j+1}(D_i)$ .
      }
    }
  }
  return results;
}

```

Fig. 3. The greedy algorithm for *Tree-Construction*

simply scanning the demand set once. For each group, the corresponding destination trees are constructed greedily. If adding the current light path to the current destination tree would exceed its wavelength capacity, we split the current request into two sub-requests. The first part joins the current tree and uses all its remaining capacity. The second part starts a new destination tree into which we attempt to merge the remaining paths in the current group. The optimality of this greedy algorithm is obvious as the number of result trees is minimized for each destination nodes. The time complexity of this tree construction algorithm is  $O(|V| * |R|)$ , where  $|R|$  is the size of the traffic demand set and  $|V|$  is the number of nodes in the network, which bounds length of all possible light paths.

## V. TREE-WAVELENGTH ASSIGNMENT

When a data burst is ready to be sent out, the source node need to know which wavelength it will use to transmit the burst for its intended destination. In TWIN network, this is decided by the *tree-wavelength assignment* process. In traditional optical networks, wavelengths are assigned to specific light paths. However, in TWIN networks each destination tree is assigned a wavelength. In this section, we discuss different strategies of assigning wavelengths to destination trees. The destination trees are constructed in the previous *tree constructing* phase. Our goal is to minimize the number of wavelengths that we use

to accommodate all the trees. In Section V-A, we introduce the generic form of the *tree-wavelength assignment* problem and prove that computing the optimal tree-wavelength assignment is NP-Hard. In Section V-B, four greedy heuristics are proposed to approximately solve the problem in reasonable time.

### A. Generic Form of the Tree-Wavelength Assignment Problem

We note that in TWIN networks, two destination trees that share some links cannot be assigned to the same wavelength, as the TWIN switches will not be able to distinguish their traffic. So, trees that have common links are considered in *conflict* for wavelength assignment. On the other hand, trees that do not share any link can be assigned the same wavelength without interference. Such trees are said to be *compatible*.

Another observation for *tree-wavelength assignment* is that a destination tree may be assigned more than one wavelength. That is, some source-destination paths may use one wavelength while the other paths use a different wavelengths. In particular, we can divide a destination tree into a compatible part and a conflict part with respect to a current wavelengths that has already been assigned to some other trees, and assign the compatible part to the current wavelength. Note that the split always starts from the source nodes (leaf nodes), and ends at the destination (root). Since the destination nodes is able to receive data flow from multiple wavelengths simultaneously, splitting destination tree as described does not affect the correctness of the data transmission. However, it provides more flexibility when we resolve the conflicts among destination trees.

Based on the above observations, the generic form of the *tree-wavelength assignment* problem is as follow: Given a set of destination trees  $DT = (t_0, \dots, t_i)$  on a TWIN network  $G < V, E >$ , minimize the total number of wavelengths that are needed to accommodate all the trees in  $DT$ , without violating the following constraints: 1). Destination trees that share a wavelength should be compatible with each other. 2). Destination tree obtained from the *tree construction* phase is either assigned a single wavelength, or split into several parts with each part being assigned to different wavelengths.

*Theorem 1:* The above *tree-wavelength assignment* problem is NP-Hard.

*Proof:* We prove this by reducing the *Graph-Coloring* problem to the tree assignment problem. *Graph-Coloring* is a well-known NP-Complete problem. Given a graph  $G < V, E >$ , we want to color all the vertices with a minimum number of colors such that no two adjacent vertices have the same color.

We first construct a corresponding TWIN-WA instance based on a *Graph-Coloring* instance  $G < V, E >$ . For each node  $v_i$  in  $G$ , we initialize a corresponding tree  $t_i$ , which only contains its root node  $r_i$ . For each link  $(v_i, v_j)$  in  $G$ , we insert a new edge  $(n_{ij\_1}, n_{ij\_2})$  to both trees  $t_i$  and  $t_j$ . We append this new edge to the last inserted node in the tree, so the tree has a chain-like structure. Figure 4 gives a simple example. Node  $v_1$  and  $v_2$  are adjacent in  $G$ . So we have edge  $(n_{12\_1}, n_{12\_2})$  appended to nodes  $r_1$  and  $r_2$  for trees  $t_1$  and  $t_2$  respectively. For the same reason, edge  $(n_{13\_1}, n_{13\_2})$  is appended to node  $n_{12\_2}$  in  $t_1$  and node  $r_3$  in  $t_3$ . After we finish the above steps for all links in  $E$ , we have a destination tree set  $DT = (t_1, t_2, \dots, t_n)$ .

We construct a TWIN network  $G_t$  from  $DT$  by merging the topology of all the trees in  $DT$ . In the example, we obtain a 7-node graph  $G_t$  by merging trees  $t_1$ ,  $t_2$  and  $t_3$  in  $DT$ .

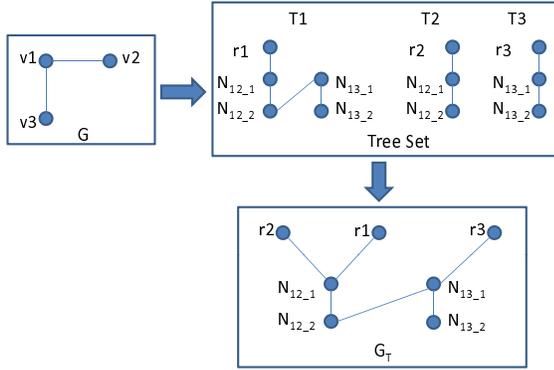


Fig. 4. Reduction from *Graph-Coloring* problem to *tree-wavelength assignment* problem.

From the construction of  $DT$  and  $G_t$ , we can see that if two vertices  $v_i$  and  $v_j$  are adjacent in  $G$ , tree  $t_i$  and  $t_j$  must have a common link  $n_{ij_1}$  and  $n_{ij_2}$ , which means  $t_i$  and  $t_j$  are in conflict in the *tree-wavelength assignment* process for network  $G_t$ . On the other hand, if two trees  $t_i$  and  $t_j$  are in conflict for wavelength assignment, they must share the edge from  $n_{ij_1}$  to  $n_{ij_2}$  and that edge is the only link that is common to both trees. From the construction, there must be a link between vertices  $v_i$  and  $v_j$  in  $G$ . Meanwhile, if the trees are all in the shape of a chain, as in our construction, splitting a tree brings no benefit to the wavelength assignment process. Therefore, in the optimal assignment for  $DT$  on  $G_t$ , every tree in  $DT$  is assigned to a single wavelength.

Now, let  $k$  be the minimum number of colors we need to color  $G$  and  $m$  be the minimum number of wavelengths we need to accommodate all the trees in  $DT$ . Based on the above observation, we claim that  $k = m$ . First, we show that  $k$  wavelengths is sufficient, if we can color  $G$  using at most  $k$  colors. Our wavelength assignment scheme is to assign tree  $t_i$  the wavelength  $W_j, j \leq k$  if the corresponding vertex  $v_i$  in  $G$  is colored using color  $C_j$ . Since  $t_i$  will not be split, and all the vertices in  $G$  that are colored with  $C_j$  cannot be adjacent to each other, we can guarantee that  $t_i$  will be compatible to any other trees that are assigned the wavelength  $W_j$ . Now, we show that  $G$  also can be colored without conflict using at most  $m$  different colors, whenever  $m$  wavelengths are sufficient for the constructed tree set  $DT$ . For each node  $v_i$  in  $G$ , if its corresponding tree  $t_i$  is assigned to wavelength  $W_j$ , it will be colored with  $C_j$ . Since there is no conflict in  $W_j$ , nodes with color  $C_j$  will not be adjacent to each other in  $G$ . Therefore the color of  $v_i$  is valid.

From the above statements, *Graph-Coloring* can be reduced to the *tree-wavelength assignment* problem in polynomial steps. So *tree-wavelength assignment* is a NP-Hard problem. ■

## B. Greedy Heuristics

In this section, we propose a set of greedy heuristics to compute an approximately optimal assignment in reasonable

time. These heuristics have a similar main process when computing the wavelength assignment. However, they differ from each other in the order the input destination trees and the existing wavelengths are assigned.

The main idea of our greedy heuristics is as follows. The destination trees in  $DT$  are checked one by one according to the *tree sorting order*. A destination tree is matched against already assigned wavelengths according to the *wavelength sorting order*. For tree  $t_i$  and wavelength  $W_j$ , if part of  $t_i$  can fit into wavelength  $W_j$ ,  $t_i$  is divided and a part of it is assigned the  $W_j$ . The rest of  $t_i$  is then matched against the wavelengths  $W_{j+1}$  and so on. If all the in-use wavelengths together cannot accommodate  $t_i$ , a new wavelength is opened for the unassigned part of  $t_i$ .

We propose 2 different approaches to sort the destination trees.

- 1) **Most Conflicts Tree First (MC):** The trees are sorted in decreasing order of to the number other trees in  $DT$  with which they have a conflict, denoted as  $CN_i$ . This is sorting criterion is based on the idea that if we assign trees with more conflicts first, we may reach the minimum number of required wavelengths very quickly. Then, for those trees with less conflicts, there is a higher chance that they will fit into the existing wavelengths.
- 2) **Most Processed Tree First (MP):** Let  $P_i$  be the number of conflicted trees of  $t_i$  that have already been assigned wavelengths. Instead of choosing trees with larger  $CN_i$  values, we pick up trees that has higher  $P_i$  values. Each time after a tree is assigned, the  $P_i$  values of all the unassigned trees are updated and the one with the largest  $P_i$  value is chosen as the next tree to be assigned wavelengths. When multiple trees have the same  $P_i$ , the tie breaker will be the value of their  $CN_i$  value. The thought behind this ordering is similar to the *MC* ordering. Moreover, *MP* order is hoped to improve the *MC* order by keeping the priorities synchronized with the result of the existing assignments.

We also propose two sorting orders for wavelengths.

- 1) **Best-Fit Wavelength First (BF):** The in-use wavelengths are sorted in the decreasing order of the number of links in the network that do not use this wavelength. This order is updated every time a tree-wavelength assignment is completed.
- 2) **Most-Fit Wavelength First (MF):** Every time before a destination tree is being assigned, the existing wavelengths are sorted by the size of the subtree they can accommodate for the current tree. We measure the subtree size by counting the number of source nodes that can be contained in the current wavelength. If one wavelength can hold a larger number of the source nodes and their corresponding light paths, it will have higher priority during the matching. The wavelengths re-ordering is triggered at runtime whenever the current tree is changed. Either a split on the current tree, or a new tree is taken out from  $DT$  for assignment. We also note that there is no need to completely sort all the wavelengths during the updates. The only wavelength we are interested in is the one that can accommodate the largest subtree. Therefore

we only need to find the Most-Fit wavelengths, rather than sort all wavelengths.

Combining the different tree sorting and wavelength sorting methods together, we obtain 4 different heuristics for *tree-wavelength assignment*: *MC-BF*, *MC-MF*, *MP-BF* and *MP-MF*. The complexity of each of our heuristics is as follows:

- 1) **MC-BF**: Let  $|V|$  be the number of vertices in the TWIN network and  $|T|$  be the number of destination trees in *DT*. When we determine the conflicts between each pair of trees, it takes  $O(|V|)$  time as each tree may contain at most  $|V| - 1$  edges. Since every pair of trees in *DT* is checked, counting the conflicts for the whole *DT* set takes  $O(|V| \times |T|^2)$  time. The sorting takes another  $O(|T| \log(|T|))$  time. Therefore computing the *MC* order takes  $O(|V| \times |T|^2)$  time. During the assignment process, the maximum number of wavelength needed is  $|T|$ . So the number of matches for each destination tree is  $O(|T|)$ . For each tree wavelength pair, it takes  $O(|V|)$  time to match them. So the processing time for one single destination tree is bounded by  $O(|V| \times |T|)$ . To maintain the *BF* order, we need to update the wavelength capacities and sort them. It takes another  $O(|T| \log(|T|))$  time. So the overall processing time for one destination tree is  $O(|V| \times |T| + |T| \log(|T|))$ . The total complexity for *MC-BF* algorithm is  $O(|V| \times |T|^2 + (|V| \times |T| + |T| \log(|T|)) \times |T|) = O(|V| \times |T|^2)$ .
- 2) **MC-MF**: To find the Most-Fit wavelengths to the current tree, we need to match the tree against all the wavelength. This takes  $O(|V| \times |T|)$  time. A destination tree will split at most  $|V| - 1$  times during the assignment, so  $O(|V|^2 \times |T|)$  time is taken to process one destination tree. The overall complexity for *MC-MF* is  $O(|V| \times |T|^2 + |V|^2 \times |T|) = O(|V|^2 \times |T|^2)$ , where  $O(|V| \times |T|^2)$  is the *MC* sorting time and  $O(|V|^2 \times |T|^2)$  is the tree-wavelength matching time.
- 3) **MP-BF**: If the tree order is updated dynamically, extra  $O(T)$  operations are added to the processing of each destination tree. However, these extra operations do not change the asymptotic complexity for the tree-wavelength matching process. The overall *MP-BF* complexity is the same as for *MC-MF*:  $O(|V| \times |T|^2)$ .
- 4) **MP-MF**: Similar to *MP-BF*, the extra operations required to maintain the *MP* order is dominated by the other tree-wavelength assignment operations. Thus, these extra operations do not affect the asymptotic complexity of *MP-MF*, which is still  $O(|V|^2 \times |T|^2)$ .

## VI. EVALUATION

### A. Experimental Framework

In this section, we measure the performance of the wavelength assignment heuristics described in Section V and evaluate how different sorting schemes affect the performances in various scenarios. Besides comparison on the optimality of their assignments, we also measure the execution time of each heuristic and study how execution time varies with network size and workloads. We implemented a *no-sort* version of the greedy heuristics that does not do the sorting steps for either the destination trees or the wavelengths. By comparing the *no-sort*

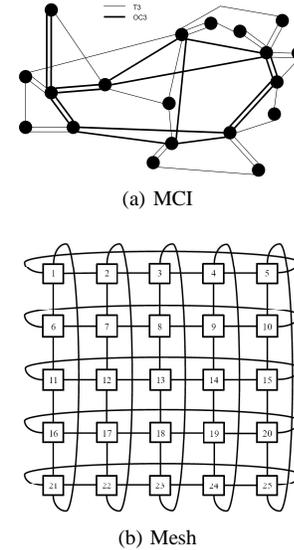


Fig. 5. Network Topologies.

heuristic with the ones we proposed in Section V-B, we can investigate the impact of the sorting steps. For every test case, we also provide a lower-bound for the optimal solution (*LB*). The lower-bound is computed by counting the occurrences of each network links in all destination trees. The maximum count among all the links is the lower bound for the minimum number of wavelengths we need. With this bound, we can estimate how well our heuristics can do in the experiments.

To simulate an optical network, we use a 25-node mesh-torus topology, a real world 19-node MCI network (Figure 5) and several randomly generated topologies. For randomly generated topologies, we set the out-degree of each node to be a random integer between 5 and 7. To ensure network connectivity, the random network has bidirectional links between nodes  $i$  and  $i + 1$  for every  $1 \leq i < n$ , where  $n$  is the number of nodes. Since the test results from MCI and Mesh topology are very similar to each other, in this paper we only present the results from MCI and Random topologies.

The traffic demands are also synthetically generated. Each request is described by a 3-tuple  $(s, d, BW)$ . We first identify the sets of source nodes and destination nodes from the all graph vertices  $V$ . In the experiments, we mark 40% of the vertices in  $V$  as source nodes and another 20% nodes as destination nodes. The remaining 40% nodes are served as communication nodes in the network. The process of marking nodes is totally random. The source  $s$  and destination  $d$  are then selected using a uniform random number generator from the respective sets so that the workload is distributed uniformly among different node pairs. The required flow size  $BW$  is generated using a chopped Normal Distribution  $\mathcal{N}(0.1, 2.5 \times 10^{-3})$ . Using this distribution, about 96% of the flows sizes are in the interval  $(0, 0.2)$ . Generated flow size are discarded if its value is outside the range  $(0, 1)$ . As the expectation of traffic demands is only 0.1, most admissible destination trees generated will comprise multiple light paths.

For each test case, the maximum number of traffic demands is bounded by the number of source-destination pairs. This number is denoted as *MaxLoad*. For example, in a 100-node

random network, if we mark 40% of the nodes as source nodes and 20% nodes as destination nodes, we will have at most 800 different source-destination pairs. That would be the maximum number of light paths that we need to handle in the test case. During the experiments, our workloads are varied from 20% of *MaxLoad* to 100% of *MaxLoad*.

## B. Evaluation Results

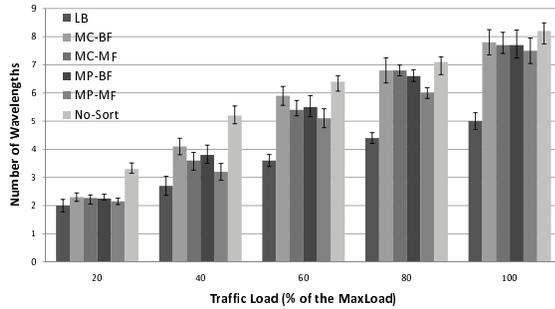


Fig. 6. The performances of wavelength assignment heuristics under different number of requests in MCI network.

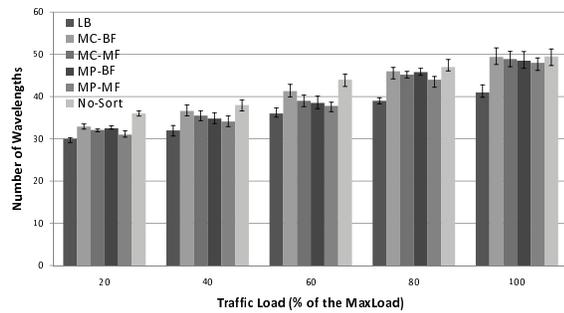


Fig. 7. The performances of wavelength assignment heuristics under different number of requests in 100-node random topologies.

Figures 6 and 7 present the evaluation results for our wavelength assignment heuristics under various traffic loads in MCI and random networks. In the experiments, we produce traffic loads that are 20%, 40%, 60%, 80% and 100% of the *MaxLoad*. From the experimental results, we make the following observations:

- 1) All 4 heuristics that we propose in Section V-B generate better assignments than the *no-sort* heuristic in all test scenarios. These heuristics outperform the *no-sort* heuristic with more obvious margins in the light traffic loads (less than 60%) than in heavy traffic loads. This shows that the sorting the trees and wavelengths provides more help to the wavelength assignment when the network is less occupied. When the networks links are saturated, rearrange the order of match will not be able to improve the scheduling much.
- 2) Among the four greed heuristics, *MP-MF* heuristic gives the best performance in all test cases. Regarding the sorting methods for the destination trees, the *MP* heuristics provides better assignments than the *MC* heuristics. This

means adjusting the tree order dynamically provides more reasonable matching orders during the tree-wavelength assignment. On the other hand, the *MF* heuristics outperform the *BF* heuristics when the workload is high (more than 80%). However, when the workload is less than 40%, the performances of *MF* heuristics and *BF* heuristics are comparable. This shows that when the traffic load is high, a more careful choice on the wavelength, like *MF*, is necessary to provide a better assignment. When there are plenty of resources available, a relatively crude sorting, like *BF*, is sufficient.

- 3) When the traffic load is light, the sorted heuristics provide a results close to the lower bound, i.e. a very good approximation on the optimal solution. When traffic load is high, the assignments from the heuristics are relatively far away from the lower bound. However, this does not necessarily mean that the heuristics cannot approximate the optimal solutions under high workloads, as the lower bounds may not tightly bound the optimal solutions when traffic load is high.
- 4) The number of wavelengths needed increased with the traffic load. In small networks like MCI and Mesh, the need for extra wavelengths increases faster than in large random networks. The reason is that it is less likely to find disjoint light paths for different source-destination pairs. When traffic load increases, conflicts are more frequent in small networks than in large networks.

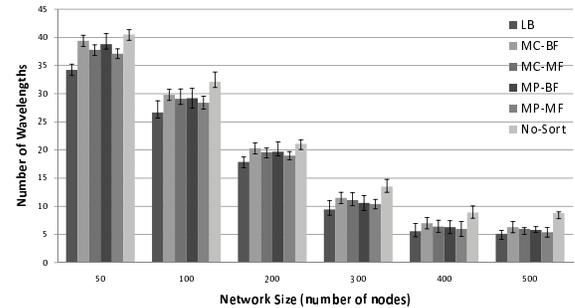


Fig. 8. The performances of wavelength assignment heuristics in random networks with various sizes.

Figure 8 presents the performance of the heuristics on random topologies of various sizes when the number of traffic demands is 800. We can see that with the increase of the network capacity, fewer wavelengths are required to accommodate the request set. However, when the network size is more than 400 nodes, the improvements are almost negligible. Recall that during the *tree construction* phase, multiple destination trees are built if the total traffic size exceeds the wavelength capacity. So when the network topology is large enough to resolve most conflicts in the tree topologies, the minimum number of wavelengths needed in such networks is heavily influenced by the maximum number of admissible trees that share the same destination, i.e. the capacity of the wavelength again becomes the main constraint.

Figure 9 presents the running time of our wavelength assignment heuristics under different workloads and Figure 10

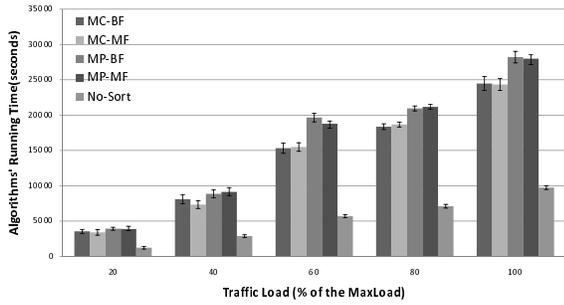


Fig. 9. The algorithm running time of wavelength assignment heuristics under different number of requests in 100 node networks.

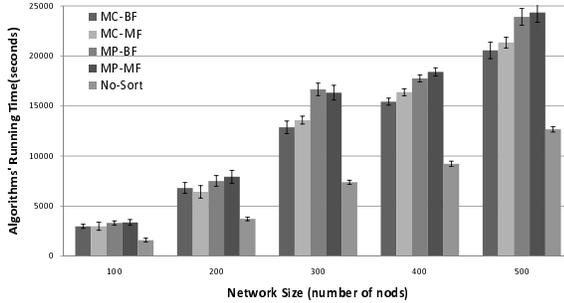


Fig. 10. The algorithm running time of wavelength assignment heuristics in random networks with various sizes.

gives the running time as a function of the network size. We see that the *no-sort* heuristic is always the fastest algorithms. The difference in the running time increase as the network size grows, as well as the traffic loads increase. Heuristics using the same tree sorting algorithms generally have the same running time, which means the overhead brought by the two wavelength sorting methods are similar to each other. For heuristics using different tree sorting scheme, we note that the *MC* sorting is faster than the *MP* sorting. However, their performance gap is much smaller than the gap with the *no-sort* heuristic. Although the sorted heuristics are relatively slow compared to the *no-sort* heuristics, their overall computational costs are still acceptable. In best cases, the average scheduling time for one request is less than 5 seconds. In the worst case, the average scheduling time is less than 30 seconds for the slowest heuristic.

As a summary, the sorting schemes provide considerable benefits to the TWIN wavelength assignments. The improvement is more with relatively low workload. However, the sort heuristics' running time is affected by the extra overhead brought by the sorting process. Nevertheless, the running times are still reasonable even in the worst case. The *no-sort* heuristic is competitive when the networks are large and traffic demands are heavy. It provides much faster scheduling speed while yielding little in the assignment optimality.

## VII. CONCLUSION

In this paper, we discuss the wavelength assignment for TWIN networks. We propose a 2-step process to compute the wavelength assignment for a given set of the traffic demands. The goal of our scheduling algorithm is to find the

assignment that use the minimum number of wavelengths. We show that this wavelength assignment problem is NP-Complete by reducing the *Graph Coloring* problem to it. Four greedy heuristics are presented to compute an approximated solution within reasonable time. Extensive experiments are performed to evaluate the optimality and the running time of the heuristics. The results show that sorting the destination trees by their degree of conflicts and sorting the wavelengths according to their available resources are effective approaches to improve the heuristics performance. However, certain overheads on the Heuristics' running times are introduced by the sorting process. We also notice that in some extreme scenarios, the *no-sort* heuristic can provide competitive results using much smaller running time.

## REFERENCES

- [1] I. Widjaja, I. Saniee, R. Giles, and D. Mitra, "Light core and intelligent edge for a flexible, thin-layered, and cost-effective optical transport network," 2003, vol. 41, no. 5, pp. 530–536.
- [2] C. Nuzman and I. Widjaja, "Time-domain wavelength interleaved networking with wavelength reuse," in *INFOCOM*, 2006.
- [3] K. Ross, N. Bambos, K. Kumaran, I. Saniee, and I. Widjaja, "Scheduling bursts in time-domain wavelength interleaved networks," 2003, vol. 21, no. 9, pp. 1441–1451.
- [4] J. S. Turner, "Terabit burst switching," 1999, vol. 8, no. 1, pp. 3–16.
- [5] K. SHIMOMURA and Y. KAWAKITA, "Wavelength selective switch using arrayed waveguides with linearly varying refractive index distribution," in *Photonics Based on Wavelength Integration and Manipulation*, 2005, pp. 341–354.
- [6] M. Kauer, "Terabit burst switching," 2002, p. 3.3.3.
- [7] H. Ishio, J. Minowa, and K. Nosu, "Review and status of wavelength-division-multiplexing technology and its application," 1984, vol. 2, pp. 448 – 463.
- [8] R. S. Tucker, "Optical packet switching: A reality check," *Optical Switching and Networking*, vol. 5, no. 1, pp. 2–9, 2008.
- [9] Y. Chen, C. Qiao, and X. Yu, "Optical burst switching: a new area in optical networking research," *IEEE Network*, vol. 18, no. 3, pp. 16–23, 2004.
- [10] I. Saniee, I. Widjaja, and J. Morrison, "Performance of a distributed scheduling protocol for twin," *SIGMETRICS Performance Evaluation Review*, vol. 32, no. 2, pp. 38–40, 2004.
- [11] D. Xue, Y. Qin, and C. K. Siew, "Performance analysis of a novel traffic scheduling algorithm in slotted optical networks," *Computer Communications*, vol. 30, no. 18, pp. 3559–3571, 2007.
- [12] "Binpacking problem," <http://mathworld.wolfram.com/BinPacking.html>.
- [13] "Graph coloring problem," <http://mathworld.wolfram.com/GraphColoring.html>.
- [14] E. Jung, Y. Li, S. Ranka, and S. Sahni, "Performance evaluation of routing and wavelength assignment algorithms for optical networks," in *13th IEEE Symposium on Computers and Communications*, 2008.
- [15] Q. Ma and P. Steenkiste, "On path selection for traffic with bandwidth guarantees," in *5th Intl. Conf. on Network Protocols (ICNP)*, 1997, pp. 191–204.
- [16] R. Guerin and A. Orda, "Networks with advance reservations: The routing perspective," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM*, 2000, pp. 118–127.
- [17] Y. Li, S. Ranka, and S. Sahni, "In-advance first-slot scheduling with wavelength conversion for e-science applications," in *In Proceedings of The IEEE Symposium on Signal Processing and Information Technology*, 2010.
- [18] A. Gadkar and S. Subramaniam, "Wavelength-reuse in optical time-slotted networks," *Optical Switching and Networking*, vol. 7, no. 4, pp. 153–164, 2010.