A Wavelength Sharing and Assignment Heuristic to Minimize the Number of Wavelength Converters in Resilient WDM Networks

Shreejith Billenahalli*, Miguel Razo*, Wanjun Huang*, Arularasi Sivasankaran*, Limin Tang*, Hars Vardhan*, Paolo Monti†, Marco Tacca*, and Andrea Fumagalli*
* OpNeAR Lab, Erik Jonsson School of Engineering and Computer Science
The University of Texas at Dallas, Richardson, TX, USA
{sxb071100, mrazora, wxh063000, axs075200, lxt064000, hxv071000, mtacca, andreaf}@utdallas.edu
† NEGONET Group, School of Information and Communication Technology, ICT-FMI
The Royal Institute of Technology, Kista, Sweden
pmonti@kth.se

***Abstract*—With the successful introduction of reconfigurable optical add-drop multiplexers (ROADMs) and related technologies, WDM networks are now growing in the number of optical nodes, wavelengths, and lambda services supported. In addition, shared path protection mechanisms — whereby lambda services are allowed to share protection wavelength channels — are possible at the optical (WDM) layer. Efficient strategies must be devised to both determine the set of services that must share a common protection wavelength channel and assign wavelengths to every service. One objective of these strategies is to minimize the total number of wavelength converters (WCs), which are required every time the wavelength continuity constraint cannot be met.**

This paper presents a scalable and efficient heuristic, whose goal is to minimize the number of WCs in resilient WDM networks supporting static sets of shared path protection lambda services. The heuristic comprises a set of polynomial algorithms that are executed sequentially to obtain a sub-optimal solution. In small size instances of the problem, the heuristic is compared against the optimal solution obtained from ILP formulation. For large size instances — tens of thousands of lambda services and hundreds of nodes — the heuristic yields an average number of WCs that is close to be linear in the number of services, despite the fact that the wavelength sharing factor increases.

I. Introduction

End-to-end optical circuits can be established in optical transport networks to support lambda services without requiring electronic processing of the service traffic at the intermediate optical cross-connect nodes. In WDM networks, the optical circuit is obtained by reserving a wavelength channel from the source to the destination node, to constitute *the working path*. If the lambda service requires path protection switching, one additional wavelength channel is reserved to obtain a second end-to-end optical circuit — *the protection path*. In some solutions the protection path is activated only if the working path fails. Depending on the lambda service requirements, the protection path may be edge, node, or shared risk link group (SRLG) disjoint from the working path.

When fiber wavelengths are a scarce resource, protection wavelength channel must be shared among as many protection paths, as long as no more than one of these protection paths is activated at any given time[1]. In addition, *wavelength continuity* is preferred when assigning wavelengths to optical circuits due to the relative high cost of wavelength converters (WCs). When the wavelength continuity constraint cannot be satisfied, WCs are necessary along the path, to shift the optical signal from the wavelength reserved in one fiber to a different wavelength reserved in the next fiber.

The problem of assigning wavelengths (WA) to optical circuits (paths) in some optimal way has been long studied. The min-RWA (optimal way of routing and wavelength assignment) is NP-hard [1]. The WA problem alone is equivalent to the coloring problem, under the assumption that WCs are not used [2], [3]. Solutions are available to sub-optimally solve the RWA problem for unprotected lambda services, in order to minimize a given cost function [4], [5]. This cost function can be defined to minimize the number of necessary WCs for each lambda service, given the set of unreserved wavelengths. For (shared path) protected lambda services, algorithms designed to minimize the number of reserved wavelengths are available for different type of traffic and protection constraints [6], [7]. A number of additional papers addresses the problem of minimizing the blocking probability of dynamically created optical circuits, for a given set of existing WCs in the network [8]. A number of more recent solutions generalize both WA and RWA problems to account for optical transmission impairments [9], [10], [11].

With the successful advent of reconfigurable optical add-drop multiplexers (ROADMs) and related technologies, WDM networks are growing in the number of optical nodes, wavelengths, and lambda services supported. Efficient strategies must be devised to both determine the set of services that must share a common protection wavelength channel and assign wavelengths to every service. One objective of these

[1]In this paper it is assumed that the network can be affected by at most one single failure at a time.

978-1-4244-5048-0/09/$26.00

strategies is to minimize the total number of WCs, which are required when the wavelength continuity constraint cannot be met. This problem has not been addressed in the literature. For example, some solutions apply to unprotected services only and make use of an auxiliary graph whose number of vertices is proportional to the product between the number of network nodes and wavelengths [4], [12], [13]. Other solutions target the optimization of protected lambda services but do not address the problem of WC minimization [6], [7].

This paper presents an efficient heuristic, whose goal is to minimize the number of WCs in resilient WDM networks supporting static sets of shared path protection lambda services. The heuristic comprises a set of polynomial algorithms that are executed sequentially to obtain a sub-optimal solution. In the former part of the heuristic, protection paths are assigned to *shared wavelength channels* while aiming for a contained number of WCs that may result from the assignment. In the latter part of the heuristic, *wavelengths are assigned* to both services and protection wavelength channels in order to minimize the number of required WCs. The proposed solution is referred to as the *wavelength sharing and assignment* (WSA) heuristic.

The polynomial algorithms in the WSA heuristic are designed to be run-time and memory efficient, and applicable to large size instances of the problem. They are successfully applied to networks with 750 nodes and above, 40 wavelengths and above, 20,000 lambda services and above, requiring only 1GB of memory. The optimality of the overall solution computed by the algorithms is however difficult to assess, as this problem has not been addressed before and benchmark solutions are not available. In Section III, the WSA heuristic is compared against the solution obtained with an ILP solver for small size instances of the problem. The results indicate that the number of WCs is almost linearly proportional to the number of services, and the heuristic result is on average 40% worse when compared to the optimum found by the ILP solver. In large size instances of the problem the wavelength sharing factor — the average number of protection paths that share one wavelength channel — increases, as more options exist that may lead to wavelength sharing. Despite the fact that an increased sharing factor exacerbates the wavelength continuity constraint, the number of WCs computed by the WSA heuristic remains close to linear in the number of services.

II. WSA Heuristic

Let the optical transport network be modeled as an undirected graph $G(N, E)$, where N is the set of network nodes, and E is the set of edges. Each edge represents a pair of fibers, one for each direction of propagation. Let each fiber carry up to W wavelengths. Let the set of lambda services contain a number of bidirectional (shared path) protected services. Assume that every protected service requires two disjoint paths (edge, node, or SRLG), i.e., a working path and a protection path, to tolerate one single network element failure. Assume that protection switching is achieved at the optical layer, i.e., the signal is generated by the transmitter (laser) and optically switched to either the working or protection path. The reverse procedure is used at the receiver, to collect the received signal from either the working or the protection path. Assume that the transmitter and receiver are both non-tunable. Then the wavelength assigned to both the working and the protection path must be the same, i.e., wavelength continuity constraint, unless wavelength conversion is allowed. Assume that both the working and protection paths for every service are given. For each conduit (fiber pair or edge) the information of all the working and protection paths which pass through it, is given.

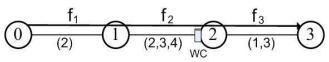

Fig. 1. Wavelength conversion along a path.

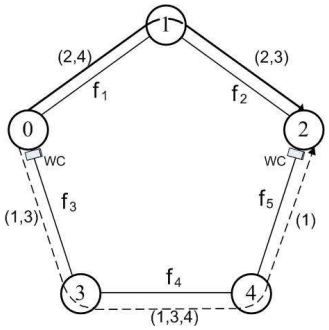

Fig. 2. Wavelength conversion at source/destination of a protected service.

The set of algorithms (WSA) described in this section aims to determine (1) which services share protection resources with one another (to form a *Group of Sharing* or GS), and (2) which wavelength is assigned to each service (WA), subject to minimizing the number of wavelength converters that are required in the network to support the services. Wavelength converters may be necessary for two reasons. First, due to the finite value of W, for some services, it may not be possible to assign a single wavelength all across to both its working and protection path due to the *finite number of wavelengths* (FNW) problem [4]. The FNW problem is illustrated in Fig. 1 and Fig. 2. In Fig. 1, a single wavelength from node 0 to node 3 cannot be found (the wavelengths available in each edge are in brackets). In Fig. 2, a single wavelength from node 0 to node 2 is available for either the working or protection lambda services, but as they use different wavelengths, wavelength converters are required at the source and destination. Second, services that are allowed to share protection wavelengths — i.e., services in the same GS — may require more than one wavelength due to the working-protection interference (WPI) problem. The WPI problem is illustrated in Fig. 3. Two services have disjoint working paths (W_1 and W_2) and are allowed to share the same protection wavelength over edges (0,4) and (1,5). However, protection path P_2 and working path W_1 both contain edge (0,1), thus each service must be assigned

a distinct wavelength on that edge. The protection wavelength channels on edges (0,4) and (1,5) are said to be *undecided*, as their chosen wavelength may match the wavelength of either W_1 or P_2. While it is possible to avoid the WPI problem by not allowing the two services to share the same wavelength on edges (0,4) and (1,5), the objective of this paper is to allow this type of sharing to minimize the total number of reserved wavelengths in the network.

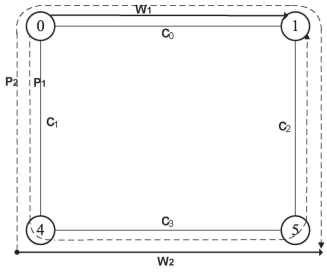

Fig. 3. Building graph for wavelength assignment.

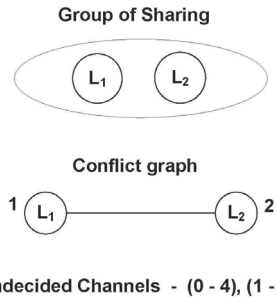

Fig. 4. Building graph for wavelength assignment.

Before presenting the set of WSA algorithms, one last observation is in order. The study in this paper makes the assumption that the protection mechanism for every service must not be sensitive to the actual location of the fault. The example in Fig. 5 illustrates the implication of this assumption using three services, each requiring edge disjoint protection. On edge (4,5), one could reserve only 2 protection wavelengths to service all three services, as any single edge failure can disrupt at most 2 services. However, if only 2 protection wavelengths were reserved on edge (4,5), at least one of the services would be required to use either wavelengths. For example, P_2 is assigned λ_1 and P_3 is assigned λ_2. Then P_1 is assigned either λ_1 if edge (1,3) is faulty, or λ_2 if edge (1,2) is faulty. To avoid this problem, three protection wavelengths must be reserved on edge (4,5).

The WSA set contains a number of sequential algorithms, that are used to break the overall problem into smaller subproblems, which can be solved individually requiring less computation. This "divide and conquer" approach does not guarantee optimality. In addition, the solutions proposed for the subproblems may not be optimal, as most of the subproblems are still NP. The subproblems are: (1) to compute the

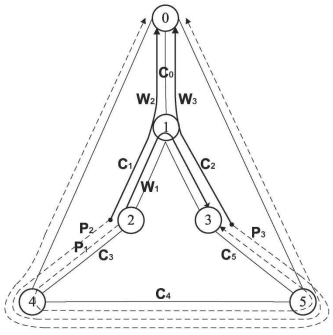

Fig. 5. Computing sharing and number of channels required.

number of shared wavelengths to be reserved on each edge (fiber) and, compute the set of services that must share the same wavelength channel on a single edge [Algorithm 1, 2], (2) to compute the GS (set of services that must share some protection wavelengths) [Algorithm 3, 4], (3) to assign a single lambda to as many GSs as possible [Algorithm 5], (4) to assign two or more lambdas to the remaining GSs [Algorithm 6]. Subproblem (1) is solved using a solution similar to the one described in [14]. The other steps of the proposed heuristic are not presented in [14].

A. Compute Number of Shared Wavelengths

The protection paths of multiple services may share one channel if the working paths of the corresponding services do not share any edge[2]. A simple data structure is associated to each edge as follows.

One edge is considered at a time. Sharing information consists of a matrix $M(i, j)$, where i represents the edges that are part of the working path of the lambda services whose protection path is in the edge being considered and j represents all the lambda services whose working path consists of the edge i. Each edge in the sharing information has the list of all such lambda services.

To determine sharing information in every edge, consider each of the protection lambda services through it. Get the working path of protection lambda service. For each edge i of the working path, add the lambda service j to the sharing information $M(i, j)$. Each edge in the sharing information will contain all the lambda services that have the protection path in the edge being considered.

Every shared wavelength channel must be associated with one or more services using the sharing information $M(i, j)$. For each edge in G, only the services whose protection path contains this edge are considered. An auxiliary graph of such services is created by representing each sharing lambda service as a node. An edge is added between the nodes of the auxiliary graph if the services are present in the sharing information $M(i, j)$ for each edge, i.e., if the working path of the lambda services are present in the same edge in the original topology graph. Then the nodes of the auxiliary graph are colored.

[2]Node and SRLG case are handled similarly.

The node is assigned a color among all the colors which satisfies the coloring condition by considering 2 factors in order. First, a color which minimizes the WPI. Second, a color which maximizes the number of edges shared between the lambda services. The former has the highest priority and the latter factor is used only if there is no WPI. Nodes with same color represent a set of services that share the same protection channel.

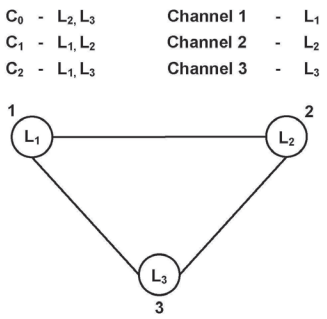

Fig. 6. Graph for the determination of sharing.

The example in Fig. 5 is used to build the auxiliary graph for determining sharing. The protection lambda services are represented as nodes as shown in Fig. 6 and the nodes are colored with distinct color since there is a conflict between every pair of nodes, hence correctly requires 3 protection channels.

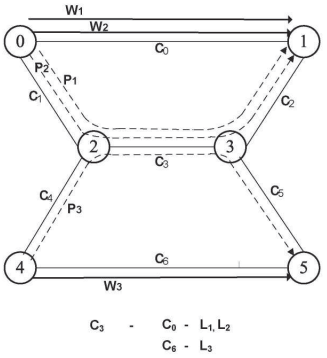

Fig. 7. Building sharing information.

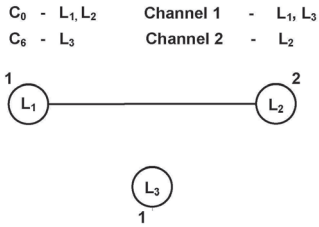

Fig. 8. Building shared channels from sharing information.

Fig. 7 shows 3 lambda service requests in a network. L_1 and L_2 are from 0 to 1. The working path takes the route 0-1 and the protection path takes the route 0-2-3-1. L_3 is from 4 to 5. The working path takes the route 4-5 and the protection path take the route 4-2-3-5. When the sharing is computed in C_3, C_0 has the working paths of L_1 and L_2 and C_6 has the working paths of L_3. Fig. 8 shows the conflict graph obtained by using this information. Lambda services L_1 and L_3 are assigned color 1 and L_2 is assigned color 2. Hence we need 2 channels one for L_1 and L_3, and the other for L_2.

Pseudo code given in Algorithm 1 and 2 describes building shared channels in an edge. An auxiliary graph, SharingGraph is created with each node representing the protection lambda service in the edge.

The node also contains the information about neighbors list and the color assigned to the node and this information is not assigned initially. Each edge i in $M(i,j)$ is considered. An edge is added between the nodes for every pair of lambda services in the edge and the neighbor list of the nodes is updated.

The edge will now have the information of the number of channels required, which is the sum of the shared channels required (as determined above) and the number of working lambda services through the edge.

Algorithm 1 Building Shared Channels A

```
1: SharingGraphNode: Node representing the protection lambda service in the edge
2: SharingGraph: vector of SharingGraphNode
3: for each protection lambda service in the edge do
4:     Create a node of type SharingGraphNode
5:     SharingGraphNode.color := -1
6:     Add SharingGraphNode to SharingGraph
7: end for
8: for each edge in M(i,j) do
9:     for every pair of the lambda services in the edge do
10:        Add edge between the 2 SharingGraphNodes which represent these lambda services. The neighbors of the SharingGraphNodes is updated
11:    end for
12: end for
13: Sort the nodes of SharingGraph in decreasing order of nodal degree
14: for every pair of SharingGraphNodes do
15:    Compute the number of WPI
16:    Compute the number of shared edges between every pair of protection lambda services
17: end for
```

B. Compute GS

All the shared protected services which share some channel have to be grouped together, since all of these have to be assigned a common wavelength. This collection of shared protected services is called *Group of Sharing*(GS). GS is built by scanning through the edges. All the services which are in the same channel of the edge will form the initial group. If any of the services is part of a group already formed, the group being considered will be merged with the existing group. Thus GS is formed by globally considering all the edges. Due to the WPI problem, the services that belong to the same GS

Algorithm 2 Building Shared Channels B

```
1: for each node of SharingGraph in the sorted order do
2:    for each color from 0 to the maximum number of nodes in SharingGraph do
3:       if none of the neighbors colored till now has this color then
4:          for each of the lambda service which have been assigned the same color do
5:             if WPI is minimum or if no WPI, then if sharing between this lambda service and the lambda service being colored is maximum then
6:                The lambda service being colored has the affinity towards this color
7:             end if
8:          end for
9:          if there are no lambda services with this color already then
10:            The lambda service being colored has the affinity towards this color
11:         end if
12:      end if
13:   end for
14:   Assign the node with the color with which the lambda service has the affinity
15: end for
16: channelToLambdaServices: map of channel to the lambda services assigned to the channel
17: for each node in the graph, SharingGraph do
18:    color := color assigned to the node
19:    Add the lambda service representing the node to the map channelToLambdaServices with key as color
20: end for
```

may not be assigned one single wavelength. In this case, GS is partitioned to make sure that each partition does not face the WPI problem. Partitioning of GS is done by coloring as follows. Each lambda service in GS is represented as a node in an auxiliary graph. An edge is added between two nodes if (a) both lambda service working paths contain a common edge in G, or (b) the working path of one service and the protection path of the other service contain a common edge in G. The edge indicates that the two lambda services must use different wavelengths on at least one fiber. The auxiliary graph is colored, and the lambda services with the same color form a sub-group (partition) of GS. For each sub-group, a node is created in the coloring graph for wavelength assignment. As explained in Figs. 3 and 4 some of the shared channels may support lambda services that belong to two or more sub-groups. These channels are termed undecided channels.

The undecided channel is assigned to the partition which has fewer number of edges of graph G and the channel is removed from all other partitions. The color assigned to the partition (to which undecided channel was assigned) is assigned to the undecided channel and the same color is used by the lambda services which are in the other partitions and wavelength converters are required at both the ends of the undecided channel. Pseudo code in Algorithm 3 and 4 describes building the graph for wavelength assignment.

After the nodes for the coloring graph are determined, the edges are determined in the following way. An edge is

Algorithm 3 Building Graph for Wavelength Assignment A

```
1: for each edge in the topology do
2:    for each shared channel in the edge do
3:       for each lambda service in the channel do
4:          for each GS already formed do
5:             if GS contains the lambda service then
6:                if this is the first GS to contain the lambda service then
7:                   Augment the GS with all the lambda services in the shared channel
8:                else
9:                   Copy all the lambda services in this GS to the first found GS
10:                  Remove this GS
11:               end if
12:            end if
13:         end for
14:      end for
15:      if there is no GS which contains this lambda service then
16:         Create a new GS with all the lambda services in the shared channel
17:      end if
18:   end for
19: end for
20: for each each GS do
21:    Create an auxiliary Graph with each lambda service in the GS as nodes
22:    for every pair of lambda services in GS do
23:       if pair of lambda services share a fiber then
24:          if fiber is shared between, working paths of the lambda services or working path and protection path of the pairs or protection path of lambda services and the protection paths are assigned different channels then
25:             Add edges between the nodes of the auxiliary graph
26:          end if
27:       end if
28:   end for
29: end for
```

Algorithm 4 Building Graph for Wavelength Assignment B

```
1: Color the conflict graph to determine color for the nodes
2: All the nodes with same color form a sub-group (partition) which form the node of the graph for wavelength assignment
3: undecidedChannelToServices: Map of undecided channel to the lambda services in the channel
4: for every pair of sub-groups of GS do
5:    if pair of lambda services across sub-groups share a channel then
6:       Add the lambda services to undecidedChannelToServices with channel Id as key
7:    end if
8: end for
9: for each undecided channel in undecidedChannelToServices do
10:   Assign the undecided channel to the lambda service in the sub-group which contains minimum number of physical services
11:   Remove the undecided channel from all other lambda services in other sub-groups
12:   for each sub-group from which undecided channel is removed do
13:      Add the lambda service and the channel Id, to be processed later for assigning the color assigned to the decided channel
14:   end for
15: end for
```

added between two nodes of the conflict graph if the nodes share at least one edge in G. Each node in the coloring graph contains all the edges of the lambda services which are grouped together to form a node in the coloring graph.

The WA problem is formulated as follows: Assign each service one or more dedicated wavelength values along its path(s), in order to minimize the total number of WCs, which are required in the network. This is done in two steps as described in sections II-C and II-D.

C. Assign Single Wavelength to GS

In the first step, wavelengths are assigned to all the services, which can be assigned a single wavelength value. This is done by coloring a conflict graph with a technique similar to the one proposed in [2]. Each lambda service is associated with a node in the conflict graph and each node represents a GS.

Once the conflict graph is built, the coloring is computed using any polynomial graph coloring algorithm. For example, the nodes are ordered in decreasing order of their nodal degree in the conflict graph. Each node is examined in this order and is assigned the color with the lowest identifier, which is not already assigned to any of its neighbors. The number of required colors is found at the end of this procedure. If no more than W colors are required, no further steps are necessary, as every node is colored with an identifier that is within the W wavelengths. If more than W colors are required, colors are sorted by decreasing number of channels used by all the nodes which are assigned the color. The maximum number of nodes being assigned the color is used to break the tie. The nodes colored with any of the top ranked W colors are permanently colored with the rank of the color. The other nodes are left uncolored. The color of the node is then turned into the corresponding wavelength, which is assigned to the service corresponding to the node. The colored services meet the wavelength continuity constraint, and do not require any WCs. Nodes (services) left uncolored require some WCs and are dealt with in step 2. A pseudo code description is given in Algorithm 5.

D. Assign Multiple Wavelengths to GS

For every GS left uncolored after running step 1, the following algorithm is used to assign two or more wavelength values, i.e., wavelength converter, to the services in the GS. Consider one uncolored GS at a time. Create a subgraph of G, which is formed by only the edges and the nodes that belong to the service paths in the uncolored GS. Sort the edges by increasing number of unreserved wavelengths. Sort the wavelength values by decreasing wavelength popularity, i.e., the popularity of a wavelength is defined as the number of edges in the subgraph of G, which have that wavelength value unreserved. Every edge in the subgraph is uncolored and it is then colored using one of its unreserved wavelengths as follows. Start with the uncolored edge, which has the smallest number of unreserved wavelengths. Color that edge with the most popular wavelength value that is available on that edge. Let this wavelength value be the temporary default wavelength.

Algorithm 5 Wavelength Assignment Using Graph Coloring

```
1:  Build the graph with service requests as nodes
2:  Add edges between nodes if the service request share a fiber
3:  Sort the nodes in the non increasing order of nodal degree
4:  Initialize all the nodes to be uncolored
5:  for each node in the graph in sorted order do
6:      for each color from 0 to number of nodes - 1 do
7:          if none of the neighbors colored till now has this color
            then
8:              Assign this color to the node
9:              Break the loop
10:         end if
11:     end for
12: end for
13: Group all the nodes which are assigned the same color
14: for each color that has been assigned in the previous step do
15:     Calculate the number of fibers being used by all the nodes
        which are assigned the color and call it points
16: end for
17: Rank the colors based on maximum number of points
18: Break the tie of colors with same points by minimum number
    of nodes which as assigned the color
19: Nodes which are assigned the top W colors are permanently
    assigned a color equal to the rank of color
20: All the other nodes are left uncolored
```

This colored edge forms a fragment of the subgraph of G. The fragment is augmented by adding only edges which can be colored using the default wavelength, as follows. Every edge which is adjacent to the fragment is added to the fragment if the default wavelength is unreserved on the edge. Every edge added to the fragment is colored with the default wavelength. The algorithm keeps adding edges to the fragment till no additional edge can be added. The following iterative steps are then performed until all edges in the subgraph of G are colored. Create a new fragment with the uncolored edge, which has the smallest number of unreserved wavelengths. Color that edge with the most popular wavelength value that is available on that edge. This wavelength value is the new default wavelength, and this fragment is augmented as described earlier, till no more edges can be added. A pseudo code description of step 2 is given in Algorithm 6.

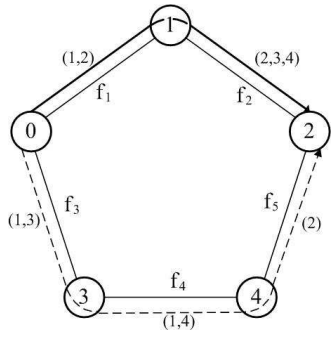

Fig. 9. Step 2: Example using a protected lambda service.

The execution of step 2 is illustrated with the help of Fig. 9, which shows the subgraph of G obtained for a protection

Algorithm 6 Coloring lambda services using multiple colors

```
for each node in the G that is uncolored in Step 1 do
    Rank colors based on the availability in the fibers of the node (The color available in most fibers is top ranked, the one which is least available is ranked lowest)
    Sort the fibers of the node which need to be colored in increasing order of available colors
    for each fiber in the sorted order which has not assigned a color do
        Get the top ranked available color in the fiber
        Assign this color to the fiber
        adjacentFibers := All the fibers adjacent to this fiber
        while adjacentFibers is not empty do
            if the color is available in the fiber then
                Assign this color to the fiber
                Add all the fibers adjacent to this fiber to adjacentFibers, if they have not assigned a color yet
            else
                Remove the fiber from adjacentFibers
            end if
        end while
    end for
end for
```

lambda service. The protection service has source node 0 and destination node 2. The working path (solid arrow) takes route 0-1-2 and the protection path (dashed arrow) takes route 0-3-4-2. The unreserved wavelengths (colors) on each edge of the subgraph are shown in brackets, next to the edge. The wavelength continuity constraint cannot be met for this service, and step 2 is therefore required. f_1, f_2, f_3, f_4 and f_5 represent the fibers in the corresponding edges. The algorithm first sorts the wavelengths based on their decreasing popularity. Here wavelength 1 is available in 3 fibers, 2 is available in 3 fibers, and so on. The wavelengths are then sorted as 1, 2, 3 and 4. The edges (fibers) are sorted with increasing number of unreserved wavelengths: f_1 has 2 unreserved wavelengths, f_2 has 3, and so on. Edges are thus sorted as follows: f_5, f_1, f_3, f_4 and f_2. f_5 is considered first and is assigned the only unreserved wavelength, i.e., 2. f_5 constitutes the first fragment. All edges which are adjacent to the fragment (f_2 and f_4) are considered next. f_2 can be added to the fragment as it can be assigned wavelength 2, too. f_4 cannot be added to the fragment, as wavelength 2 is not unreserved on that edge. The fragment now contains both f_5 and f_2, and f_1 is an adjacent edge to the fragment. f_1 is added to the fragment and assigned wavelength 2. At this point the fragment cannot be further augmented. The algorithm then creates a new fragment by choosing the uncolored edge with the smallest number of unreserved wavelengths. In this case, both f_3 and f_4 have each 2 unreserved wavelengths. Let f_3 be the chosen edge. f_3 is assigned wavelength 1, which is more popular than wavelength 3. The new fragment is augmented by adding f_4, which is assigned wavelength 1, too. All the edges in the subgraph are now colored and the algorithm stops. In summary, this lambda service is assigned two wavelength values: wavelength 2 over edges f_1, f_2 and f_5 and wavelength 1 over edges f_3 and f_4. Two bidirectional WCs are then required: one at node 4, along the protection path; the other at node 0[3].

After assigning colors to all the nodes, the undecided channels which were formed while building the graph for coloring need to be handled. All the undecided channels which were removed before coloring are added to the lambda services and are assigned the color assigned to the retained channel which was decided to go with one node. Pseudo code in Algorithm 7 describes the handling of undecided channels.

Algorithm 7 Handling Undecided Channels

```
for each node in graph for coloring that has undecided channels do
    for each lambda service that has undecided channel do
        for each undecided channel in the lambda service do
            Add the channel back to the lambda service which was removed earlier
            Get the lambda service with the corresponding decided channel
            Get the color assigned to the decided channel
            Assign this color to the undecided channel
        end for
    end for
end for
```

III. Results

In this section, both the optimality and scalability of the WSA heuristic are assessed.

Benchmarking. The WSA heuristic optimality is investigated first. As already mentioned, no other solution is available in the literature that can be used as benchmark. An ILP formulation [15] is therefore used for small instances of the problem. A network with a number of nodes $N = 5$, a number of fiber links (edges) $L = 8$ and a number of wavelengths per fiber $W = 8$ is used. The number of protected services[4] is varied from 5 to 12. For each number of lambda services, 20 experiments are run. The results are averaged. For each experiment, the ILP formulation and the WSA heuristic are solved using the same set of input lambda services where, for each service the same routing is used. Results in TABLE I show the comparison. The average number of required channels is the same for both the ILP and the WSA heuristic. The table also shows the average number of required WCs. The WSA heuristic requires a number of WCs that is 27% to 48% larger than the optimal number obtained with the ILP.

Scalability. The scalability of the WSA heuristic is assessed next. TABLE II shows results obtained by the WSA heuristic when the network size — i.e., the number of nodes, fiber links (edges), and lambda services — is increased. The number of wavelengths per fiber is set to $W = 40$. The table shows the number of nodes, fiber links, WCs required, sharing factor (i.e., average number of services sharing one protection

[3]The transmitter at node 0 can be assigned wavelength 2 (1), and the WC is placed between the transmitter and the multiplexer of f_3 (f_1)

[4]The ILP solver [16] requires several hours to compute the optimal solution for the ILP formulation when the number of lambda services is larger than 12.

TABLE I
ILP VS. WSA HEURISTIC.

	ILP		WSA	
Req.	WC	Channels	WC	Channels
5	1.3	14.1	1.65	14.1
6	1.75	16.6	2.25	16.6
7	2.35	18.9	3.45	18.9
8	3.15	20.85	4.65	20.85
9	4.05	22.85	5.7	22.85
10	4.85	24.95	6.8	24.95
11	5.8	27.25	8.3	27.25
12	6.65	29.45	9.4	29.45

channel), wavelength channels used, and the time (in seconds) taken by the WSA heuristic, respectively. The run time is reasonable, even for large networks with hundreds of nodes and tens of thousands of lambda services.

TABLE II
SCALABILITY OF THE WSA HEURISTIC.

N	F	Req.	WC	Sharing	Channels	Time [s]
30	133	2194	8361	5.02763	9516	78
40	179	2499	10125	5.34633	11560	104
50	223	2995	12671	5.94234	14484	157
75	335	3500	15318	6.5274	18160	201
100	446	5974	29839	7.63172	32406	674
150	666	7989	41556	8.08658	46740	1245
200	890	9998	53918	8.59914	61034	2052
300	1337	14927	86273	9.40011	96536	5215
400	1789	17994	104883	9.64224	121598	7729
500	2231	21985	131515	9.92145	154760	12492
750	3351	26000	157735	9.66434	195634	18938

IV. SUMMARY

This paper presented the polynomial WSA heuristic, whose goal is to minimize the number of WCs in resilient WDM networks supporting lambda services protected with shared path protection. The WSA heuristic (i) computes the set of protection paths that can share the same wavelength channel, and (ii) assigns wavelengths to both services and protection wavelength channels, given a pre-computed set of routes for every service's working and protection path.

The WSA heuristic is designed to be time and memory efficient, and applicable to large size instances of the problem. It is successfully applied to networks with 750 nodes and above, 40 wavelengths and above, 20,000 lambda services and above, requiring only 1GB of memory. The optimality of the overall solution computed by the algorithms is however difficult to assess. A comparison with an ILP formulation was carried out in small size problem instances. In large size instances of the problem, where the wavelength sharing factor increases and the wavelength continuity constraint is exacerbated, the number of WCs computed by the WSA heuristic is close to linear with the number of services.

REFERENCES

[1] T. Erlebach, K. Jansen, and C. Elvezia, "The complexity of path coloring and call scheduling," *Theoretical Computer Science*, vol. 255, p. 2001, 2001.

[2] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath communications: An approach to high bandwidth optical wans," *IEEE Transaction on Communications*, vol. 40, no. 7, pp. 1171–1182, July 1992.

[3] G. Li and R. Simha, "The partition coloring problem and its application to wavelength routing and assignment," in *Proceedings of the First Workshop on Optical Networks*, 2000.

[4] I. Chlamtac, A. Farago, and T. Zhang, "Lightpath (wavelength) routing in large wdm networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 5, pp. 909–913, Jun 1996.

[5] D. Banerjee and B. Mukherjee, "A practical approach for routing and wavelength assignment in large wavelength-routed optical networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 903–908, 1996.

[6] A. Jaekel and T. Khan, "Routing and wavelength assignment in optical mesh networks with wavelength conversion," in *Proceedings of 25th IEEE International Performance, Computing, and Communications Conference (IPCCC 2006)*, April 2006, pp. 273–280.

[7] H. Zang, C. Ou, and B. Mukherjee, "Path-protection routing and wavelength assignment (rwa) in wdm mesh networks under duct-layer constraints," *IEEE/ACM Transactions on Networking*, vol. 11, no. 2, pp. 248–258, Apr 2003.

[8] X. Chu, B. Li, and Z. Zhang, "A dynamic rwa algorithm in a wavelength-routed all-optical network with wavelength converters," in *In Proceedings of IEEE INFOCOM 2003*, March-April 2003, pp. 1795–1804.

[9] R. Martinez, F. Cugini, N. Andriolli, L. Valcarenghi, P. Castoldi, L. Wosinska, J. Comellas, and G. Junyent, "Challenges and requirements for introducing impairment-awareness into the management and control planes of ason/gmpls wdm networks," *IEEE Communications Magazine*, vol. 44, no. 12, p. 76, 2006.

[10] Y. Huang, J. P. Heritage, and B. Mukherjee, "Connection provisioning with transmission impairment consideration in optical wdm networks with high-speed channels," *Journal of Lightwave Technology*, vol. 23, no. 3, p. 982, 2005.

[11] A. Jirattigalachote, K. Katrinis, A. Tzanakaki, L. Wosinska, and P. Monti, "Quantifying the benefit of ber-based differentiated path provisioning in wdm optical networks," in *Proceedings 11th International Conference on Transparent Optical Networks (ICTON 09)*, July 2009.

[12] S. Gowda and K. M. Sivalingam, "Protection mechanisms for optical wdm networks based on wavelength converter multiplexing and backup path relocation techniques," in *Proceedings of IEEE INFOCOM 2003*, March-April 2003.

[13] Y. Wang, L. Li, and S. Wang, "A new algorithm of design protection for wavelength-routed networks and efficient wavelength converter placement," in *Proceedings of IEEE International Conference on Communications (ICC 2001)*, vol. 6, 2001, pp. 1807–1811.

[14] J.-F. Labourdette, E. Bouillet, R. Ramamurthy, and G. Ellinas, *Path Routing in Mesh Optical Networks*. John Wiley & Sons, 2006.

[15] W. Huang *et al.*, "Shared Protection ILP Formulation," The University of Texas at Dallas, Tech. Rep., May 2009. [Online]. Available: http://opnear.utdallas.edu/publications/reports/UTD-EE-03-2009.pdf

[16] Ilog, Inc., "Solver cplex," 2009, http://www.ilog.com/products/cplex/. [Online]. Available: http://www.ilog.com/products/cplex/