Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 29 (2006) 3900–3912
computer communications
www.elsevier.com/locate/comcom

Multiple link failures survivability of optical networks with traffic grooming capability

Chadi Assi [a,*], Wei Huo [a], Abdallah Shami [b]

[a] Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, Que., Canada
[b] Department of Electrical and Computer Engineering, The Univeristy of Western Ontario, London, Ont., Canada

Received 30 April 2006; received in revised form 30 June 2006; accepted 3 July 2006
Available online 4 August 2006

Abstract

This paper investigates the problem of survivable traffic grooming (STG) in shared mesh optical networks and proposes different frameworks for improving the survivability of low speed demands against multiple near simultaneous failures. Spare capacity reprovisioning has recently been considered for improving the overall network restorability in the event of dual failures; here, after the recovery form the first failure, some connections in the network may become unprotected and exposed to new failures. Capacity reprovisioning then allocates protection resources to unprotected and vulnerable connections so that the network can withstand a future failure. In this paper, we propose two different reprovisioning schemes (lightpath level reprovisioning, LLR, and connection level reprovisioning, CLR); they differ in the granularity at which protection resources are reprovisioned. Further, each of these schemes is suitable for a different survivable grooming policy. While LLR provides collective reprovisioning of connections at the lightpath level, CLR reprovisions spare bandwidth for lower speed connections instead. We use simulation methods to study the performance of these schemes under two grooming policies (PAL and PAC), and we show that while CLR reprovisions substantially many more connections than LLR (i.e., potentially more management overhead) CLR yields a much better network robustness to simultaneous failures due to its superior flexibility in using network resources.

Keywords: Optical networks; Protection; Traffic grooming and routing; Simulations

1. Introduction

Over the past decade, research efforts have focused on improving the survivability of optical networks using either proactive (i.e., protection using preplanned resources) or reactive (i.e., dynamic discovery of alternate resources) mechanisms [1–3]. Recent research has also focused on improving the service availability of these networks against multiple simultaneous failures either through preplanned redundant capacity [4–6] or through capacity reprovisioning [7–11] or further using *p*-cycle reconfiguration[1] [12] in mesh networks. Most, if not all, of these efforts have assumed that every user demands a bandwidth equals to the full wavelength capacity. Currently, the transmission rate of a wavelength channel is STS-192 (10 Gbps) and expected to grow to STS-768 (40 Gbps) in the near future. Bandwidth requirement of a typical connection request varies, however, from full wavelength capacity to as low as STS-1 or lower. Hence, it is necessary to efficiently pack these lower speed demands (or connections) onto high capacity light channels (also known as lightpaths) in order to better utilize the network resources. This problem has emerged lately and is known as the traffic grooming problem [13–16]. Traffic grooming refers to the problem of efficiently packing low-speed connections onto high-capacity lightpaths in order to better utilize the network resources [13] and has been studied extensively over the past years both for SONET/WDM ring networks [17,18] as well as in optical mesh networks [13–16].

* Corresponding author. Tel.: +1 514 848 2424; fax: +1 514 848 3171.
E-mail addresses: assi@ciise.concordia.ca (C. Assi), w_huo@ciise.concordia.ca (W. Huo), ashami@eng.uwo.ca (A. Shami).

[1] In this paper, the terms "reprovisioning" and "reconfiguration" are used interchangeably. Reprovisioning does not mean a new capacity is placed into the network.

0140-3664/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2006.07.005

Now, how to efficiently groom such low-speed connections while satisfying their protection requirements is best known as survivable traffic grooming (STG) problem and presently is attracting some considerable research efforts [19,20]. The authors of [19] have proposed different frameworks for protecting low-speed connections against single link failures in optical mesh networks and have shown that providing collective protection of connections at lightpath level (PAL) achieves better performance than protecting at the connection level (PAC) while it also requires a smaller number of grooming ports. To make connections survivable under various failures, such as fiber cut and duct cut, the authors of [20] studied the static STG problem under the general shared risk link group diverse routing constraints where protection is provided at the lightpath level. In this paper, we revisit the STG problem in mesh networks and we study the survivability of connections against multiple concurrent failures where concurrent implies that the new failure occurs before the previous failure has been repaired. We focus on mesh networks that are only designed to withstand all single link failures either through lightpath level protection or through connection level protection with shared backup resources. To combat the effect of multiple failures, we propose to use capacity reconfiguration after the occurrence of the first failure in order to reprovision new protection capacity for unprotected or vulnerable demands. Namely, when a link failure occurs, all demands routed through that link will fail and are rerouted to their protection connections. These demands and other demands whose protection paths were originally routed through the failed link become unprotected. Moreover, when a demand is restored onto its protection, the protection capacity is now activated and can no longer be shared; hence other connections sharing the protection resources become vulnerable. Therefore, one needs to identify these unprotected and vulnerable connections/lightpaths and assigns/reprovisions for them new protection capacity in order to improve their robustness.

We present lightpath and connection level reprovisioning as complementary approaches for STG to achieve better service robustness against multiple failures. Note that when connections are protected at the lightpath level, the process of reprovisioning takes place at that level (thereafter referred to as lightpath level reprovisioning, LLR); in other words, only the lightpaths that become unprotected/vulnerable need to be reprovisioned. Alternatively, if connections are protected at the connection level, reprovisioning takes place at connection level (connection level reprovisioning, CLR). One critical difference between the two schemes is the granularity at which backup bandwidth is reprovisioned and the number of connections to be reprovisioned. While in LLR, lightpaths (whose number is substantially much smaller than the number of connections they carry) are reprovisioned, individual connections are reprovisioned in CLR. One drawback of LLR is that when a lightpath remains unprotected after reprovisioning, all the connections routed through this light-path are unprotected. Moreover, a lightpath is reprovisioned by requesting or searching for resources on the physical layer although protection resources may be available at the lightpath layer.[2] CLR on the other hand reprovisions unprotected/vulnerable connections first at the logical layer instead and then at the physical layer. We compare the performance of these two schemes and present some simulation results on the robustness of the network under both reprovisioning frameworks. Our results show that connection level reprovisioning substantially outperforms the lightpath level reprovisioning. The rest of the paper is organized as follows. Section 2 presents an overview of the survivable traffic grooming problem and we present some simple heuristics and compare their performance. Section 3 presents a detailed study of the reprovisioning approaches and we quantify their performances in Section 4. Finally, we conclude in Section 5.

2. Survivable traffic grooming

2.1. Background

Grooming connections while still satisfying their protection requirements is known as survivable traffic grooming, STG [19,20]. Different schemes have been proposed for protecting connections, namely protection at lightpath level (PAL), mixed, and separate protection at connection level (MPAC and SPAC). Under PAL, a connection is typically routed through a sequence of protected lightpaths (*p-lightpaths*), where a *p-lightpath* is a pair of working and link-disjoint backup lightpaths. A working lightpath consumes one grooming add port and one drop port and wavelengths along the route of a lightpath are reserved and configured. Resources for a protection lightpath, on the other hand, are only reserved and they are setup after the failure. Hence, a protection lightpath does not consume any grooming add/drop ports. Normally, a demand is routed over a multihop route (sequence of *p-lightpaths*) if there is no direct lightpath with enough capacity connecting the source and the destination of the demand. In case of a failure along the working lightpath, the carried traffic (i.e., the set of connections routed through this lightpath) is restored onto the protection lightpath; only the end (and intermediate) nodes of the lightpath are aware of the switching and the end nodes of the failed connections are oblivious to this protection switching.

Fig. 1a shows an illustrative example of 4 *p-lightpaths*. A demand, d_1, between nodes A and F can be routed through lightpaths (l_1–l_2), where l_1 and l_2 are protected by b_1 and b_2 respectively. Note that under PAL two *p-lightpaths* can share wavelengths along their protection

[2] A lightpath layer or logical layer is the set of lightpaths currently in the network.

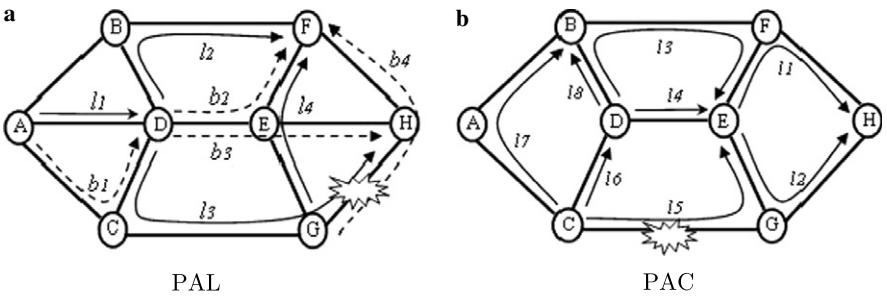

Fig. 1. Illustrative examples of STG.

lightpaths if their corresponding working lightpaths are link disjoint. For example, l_2 and l_3 can share the same protection wavelength along link (D–E) since they are link disjoints.

Alternatively, PAC provides end to end protection at the connection level and has two variants (SPAC and MPAC) that differ in the way connections are protected [19]. In SPAC, a connection is routed via a link-disjoint working and backup routes. The working traverses a sequence of lightpaths and the backup traverses a sequence of wavelength links, where each wavelength link consumes a pair of grooming ports, add and drop, at each end of the link. Under MPAC, a demand is routed via link-disjoint working and backup paths each traversing a sequence of lightpaths. Each lightpath consumes a pair of grooming ports, one add at the source and one drop at the destination. Every lightpath traversed by a working connection reserves a fixed amount of bandwidth to carry the traffic. A lightpath that is traversed by a backup connection correspondingly reserves a fixed amount of its capacity to protect against the failure of the working connection. In this paper, we will only use MPAC and we use the term PAC to refer to this grooming policy.

A working connection fails when any of the lightpaths that it traverses fails. Upon the failure, the source node of the failed demand switches traffic from the working into its corresponding backup. *Bandwidth sharing* is achieved under MPAC when two demands have their corresponding working connections physically end-to-end link-disjoint and their backup connections traverse the same lightpath(s). Fig. 1b shows an example of PAC grooming. The figure shows a set of existing lightpaths; when a new demand d_1 arrives (e.g., between nodes C and H and demanding a bandwidth of STS-12), it is routed through l_5 and l_2 and is protected by lightpaths l_6, l_4, and l_1. A demand, d_2, between nodes D and E and bandwidth $2\times$STS-12 can be routed through l_3 and protected by l_4. Here, d_1 and d_2 are both end-to-end link-disjoint and both share the same lightpath l_4, hence they can both share the protection bandwidth reserved along l_4 and the new protection bandwidth reserved along l_4 becomes $2\times$STS-12. When a link fails, the lightpaths routed through that link will fail and hence the connections routed through every failed lightpath will also fail.

2.2. Comparison between PAL and PAC

These schemes differ in the routing and the backup bandwidth sharing. In terms of routing, PAL provides end to end protection at the lightpath level whereas PAC provides end to end protection at the connection level. In PAL, after the failure of a link, the end nodes of a failed lightpath configures the protection lightpath and switch the traffic into it and the end nodes of the connections are unaware of this process. Alternatively, in PAC the end nodes of the failed connections configure their backup paths and restore the traffic. Under PAL, only working and protection paths of a *p-lightpath* must be link disjoint and a connection routed through a sequence of *p-lightpaths* is normally protected against any single link failure (that is the working and protection path of a demand need not be end to end link-disjoint). In PAC, however, the working and backup routes of a demand must be end to end link disjoint; when a demand spans multiple lightpaths, it becomes difficult to find a protection path.

With respect to backup sharing, protection wavelength links are the resources that can be shared in PAL. Namely, two *p-lightpaths* can share the same protection wavelength link if their working lightpaths are link disjoint and their protection lightpaths traverse through that same protection wavelength. Hence, all working connections traversing these *p-lightpaths* are said to be sharing that protection wavelength link. However, under PAC (i.e., MPAC) the sharing unit is a lightpath (or the backup bandwidth reserved in a light-path). Hence, two demands (d_1 and d_2) under PAC can share protection bandwidth in a lightpath l if (1) their corresponding working connections are end to end diversely routed and (2) their protection connections traverse lightpath l. Then the backup bandwidth required on l is $\max(bw_1, bw_2)$ where bw_1 and bw_2 are the bandwidth requirements of demands d_1 and d_2, respectively. Clearly, since a lightpath may traverse multiple physical links and a connection is routed through multiple lightpaths, it is less likely that conditions (1) and (2) are together satisfied and hence bandwidth sharing is hard to achieve. All these reasons make PAC algorithm less attractive than PAL; the authors of [19] have evaluated using simulations the performance differences between PAL, SPAC, and MPAC. Overall results showed that PAL achieves best performance when the

number of grooming ports is either small or moderate. Moreover, as mentioned earlier, under PAL only the working lightpath consumes add/drop grooming ports whereas under PAC every lightpath consumes add/drop grooming ports (note in SPAC every wavelength link along the protection route consumes one pair of add/drop ports). PAC, on the other hand, allows both working and protection connections of different demands to be routed through the same lightpath, a flexibility that does not exist under PAL.

2.3. STG grooming heuristics

Clearly, while PAL trades the flexibility in grooming for the freedom of backup sharing, PAC allows working and backup connections of different demands to be groomed on the same lightpath. However, one major drawback for PAC is the difficulty in sharing backup bandwidth between the demands. Unlike PAL where wavelength links are shared between *p-lightpaths*, in PAC two demands can share backup bandwidth if the two conditions of the previous section are both satisfied. Since, every lightpath traverses a sequence of wavelength links, these two conditions are difficult to meet due to the conflict in finding link disjoint working routes for demands whose protection routes share the same lightpath. This conflict is further exacerbated as the physical hop count of lightpaths gets larger. However, due to the flexibility of PAC, we propose some simple modifications to improve the efficiency of backup bandwidth sharing. We note first that if the physical hop count of every lightpath is limited to one, then in terms of backup sharing MPAC becomes similar to SPAC; further, since MPAC allows protection and working capacity to be groomed on the same lightpath, then this new MPAC-1 (1 means a lightpath is limited to one hop) achieves both high flexibility and better bandwidth sharing. However, the grooming capacity requirement could be excessive. Therefore, we propose to limit the length of a lightpath in order to improve the backup bandwidth sharing while still maintaining the flexibility of routing. This new version of PAC is referred to as PAC-HCL, where HCL refers to Hop count limit and HCL $\geqslant 1$.

Next, we explain the STG heuristic; in response to a new connection request, PAC-HCL first computes two link disjoint paths from source to destination using *Dijkstra* Algorithm in the existing logical topology. Every lightpath along the working path must have enough bandwidth to carry the new demand. Along the protection path, bandwidth sharing is used. Every lightpath (l) reserves backup bandwidth (v_l) to protect all the connections whose protection paths traverse through this lightpath. Note, $v_l = \max_{\forall e' \in E}\{v_l^{e'}\}$, where $v_l^{e'}$ is the amount of bandwidth reserved on lightpath l to protect against the failure of link e ($0 \leqslant v_l^{e'} \leqslant$ OC-192) and E is the set of all links in the network. When a new connection of bandwidth w is protected by l, the additional backup bandwidth reserved on l is α_l and is determined as follows: $\alpha_l = \max\{v_l^{\text{new}} - v_l, 0\}$, where $v_l = \max_{\forall e' \in E}\{v_l^{e',\text{new}}\}$ and $v_l^{e',\text{new}} = v_l^{e'} + w$ if the working connection of the new demand traverses through e', otherwise $v_l^{e',\text{new}} = v_l^{e'}$ In case there is no enough bandwidth to route and/or protect the demand on the logical layer, then new a lightpath(s) is setup on the physical layer for either the working or protection or both paths.

At the physical layer, the source node computes the shortest path route to the destination ($s{-}x_1{-}x_2{-}\cdots{-}x_n{-}d$). The source will select a node x_i from the shortest path that is HCL hops away and check whether there is a direct lightpath already setup with enough capacity (or with enough sharable capacity in case of a protection connection). If there is not, the source node checks for a node that is HCL-1 hops away from the source node and so on until a lightpath is found. If a node (x_i) is found, the same procedure as before is run again between x_i and the destination. Let $x_t (x_t = s$ if there is no direct lightpath between s and any node on its shortest path that is at most HCL hops away from s) be the node after which there is no outgoing lightpath(s) to any node along the shortest path to d. At this point, the algorithm tries to "setup" a new lightpath from x_t to a node that is HCL hops away. If that fails, then a node that is HCL-1 hops away is checked and so on until a lightpath is setup. If a lightpath is found, then the same procedure is repeated until a route is established all the way to the destination. At any step, if a lightpath could not be setup, the request is dropped and all allocated resources are released. With regards to PAL, the same algorithm as in [19] is implemented.

Algorithm 1. Pseudo code of the provisioning in PAC-HCL

Input: A network represented as a directed graph $G = (V, E, \lambda, P)$, where V is a set of nodes, E is the set of unidirectional physical links, λ specifies the number of wavelengths on each link, and P specifies the number of grooming ports at each node. The logical topology of this network is represented as a graph $G' = (E, L)$, where L is the set of lightpaths.

Output: Link-disjoint working and backup paths, or NULL if fails.

1. Compute a pair of shortest and physically link-disjoint logical paths (i.e., a working path with enough bandwidth and a protection path with enough sharable bandwidth) from *src* to *des*; if successful, return the two paths, otherwise compute the shortest physical working path wroute = (src$-x_1-x_2-\cdots-x_n-$des). The source will select a node x_i and go to step 2.
2. The node s (s = wroute[0]) will check whether there is an existing direct lightpath to a node along the path which is HCL hops away and has enough capacity; if there is not, check for a lightpath with HCL-1 hops from the source node and so on until a lightpath is found. If there is such a lightpath (assume its destination node is x_t), then wroute $\leftarrow (x_t - x_{t+1} - \cdots -$ des) and repeat step 2 on *wroute*. If a working path has been found to the destination, go to step 4, otherwise, go to step 3.

3. Let x_t be the node after which there is no outgoing lightpath(s) to any node along *wroute*. Then, try to *establish* a new lightpath from x_t to a node that is HCL hops away. If this fails, then a node that is HCL-1 hops away is checked and so on until a new lightpath is setup. If a lightpath is setup and assume its destination node is $x_{t'}$, then wroute$\leftarrow (x_{t'} - x_{t'+1} - \cdots -$ des) and go to step 3. If a working path has been found to the destination, go to step 4; otherwise, return NULL.
4. Eliminate the working path in graph G, find the shortest physical path *broute* from *src* to *des*; repeat the same procedure in steps 3 and 4 on *broute* to provision the backup path (backup sharing as explained in Section 2.3 is considered). If the backup path can be provisioned successfully, return working and backup paths; otherwise, release the resources reserved along the working path and return NULL.

3. Connection and lightpath level reprovisioning

Various research efforts [4–6] have addressed the problem of routing connections under dual-failure assumptions, and findings show that designs offering complete dual-failure restorability require more than double the amount of spare capacity. In order to avoid this excessive deployment of extra spare capacity in the network, capacity reprovisioning or reconfiguration after the occurrence of and recovery from the first failure has been proposed [7–12]. This problem has recently been studied for mesh networks where every request is considered to consume the full wavelength capacity. After the occurrence of the first failure, the failed lightpaths are restored from their working paths into their protection paths. These recovered lightpaths and other lightpaths that were originally protected by protection resources on the failed link now become unprotected and exposed to a second failure. Moreover, protection wavelength links along recovered lightpaths are now active and hence can no longer be shared with other demands. As a result, all demands that were not directly affected by the failure but their protection paths can no longer share backup resources become vulnerable to new failures.

Spare capacity reconfiguration provides a mechanism by which one can find and allocate new protection capacities for these newly-unprotected lightpaths without a priori knowledge of the location of the second failure. We assume multiple near simultaneous link failures, where a second failure occurs after the first failure is recovered from, but before it is physically repaired. In this section we consider low-speed connection requests and propose two frameworks for improving their survivability against multiple failures. The first scheme is lightpath level reprovisioning (LLR) that relies on PAL and the second is connection level reprovisioning (CLR) which rather relies on PAC.

3.1. LLR

As mentioned before, under PAL a connection traverses a sequence of *p-lightpaths*. The working route of a connection traverses the sequence of working lightpaths and is protected by the sequence of corresponding protection lightpaths. Consider every link in the network to be associated with a conflict set to identify the sharing potential between protection lightpaths [19,21]. The conflict set v_e for link e can be represented as an integer set, $\{v_e^{e'} | \forall e' \in E, 0 \leqslant v_e^{e'} \leqslant \lambda(e)\}$, where $v_e^{e'}$ is the number of working lightpaths that traverse link e' and are protected by link e, E is the set of all links in the network, and $\lambda(e)$ is the number of wavelengths per link e. Then, the number of protection wavelengths reserved on link e to protect against the failure of any other link in the network is given by $v_e^* = \max_{\forall e' \in E}\{v_e^{e'}\}$.

When a link (e.g., f) fails, all lightpaths routed through that link also fails and accordingly all the connections carried by these lightpaths fail. The failed lightpaths are rerouted onto their corresponding protection lightpaths and consequently become unprotected and exposed to a new failure. For example, when link (G–H) in Fig. 1a fails, then lightpath l_3 fails and is restored into its protection lightpath b_3. All connections routed through l_3 will fail and will be restored to b_3. Note that b_3 and the restored connections are all exposed to a new failure. Moreover, all the demands that were originally protected by link f have lost their protection resources and become unprotected. For example, all connections traversing l_4 now become unprotected because the backup path b_4 has lost its protection wavelength on the failed link (G–H).

Upon the recovery of the failed lightpaths to their protection routes, some backup wavelengths on a link, say e, may be activated if at least one of these protection lightpaths traverses through link e. Hence, the number of new available protection wavelengths on link e is $v_e^a = v_e^* - v_e^f$ and the number of protection wavelengths on link e required to protect against a future link failure in the network is $v_e^{\text{new}} = \max_{\forall e' \in E-f}\{v_e^{e'}\}$. If $v_e^{\text{new}} > v_e^a$, then some of the existing lightpaths that were not directly affected by the failure are vulnerable to a new failure because link e does not have enough protection capacity. For example, before the failure, link (D–E) reserves only one wavelength ($v_{\text{DE}}^* = 1$) to protect l_2 and l_3. When l_3 is rerouted to b_3 after the failure, $v_{\text{DE}}^a = 0$ and $v_{\text{DE}}^{\text{new}} = 1$, hence l_2 is vulnerable to a new failure.

Let $e_1, e_2, \ldots, e_F$ be the set of links traversed by some active protection light-paths after the first failure; then the set of all vulnerable lightpaths can be identified. A connection that traverses an unprotected *p-lightpath* is unprotected and similarly a connection that traverses a vulnerable *p-lightpath* is also vulnerable. In LLR, the resources along the failed lightpaths are released, and every unprotected lightpath that is identified is reprovisioned by computing and allocating new protection capacity for this lightpath. On the other hand, some of the vulnerable

lightpaths need to be reprovisioned in order to reduce the vulnerability of the network to a second failure. When a vulnerable light-path is reprovisioned, some other vulnerable lightpaths may become protected [9,10] if they were originally contending with the reprovisioned lightpath for the same protection wavelength on a particular link. Hence, a vulnerable light-path l becomes protected after the reprovisioning of another lightpath, when for every link (e) along the backup of l we have $v_e^a \geqslant \max_{\forall e' \in E-f}\{v_e^{e'}\}$. Each time a new vulnerable lightpath is reprovisioned, the set of remaining vulnerable lightpaths is identified; this procedure continues until all vulnerable lightpaths are reprovisioned or no more reprovisioning is possible. Note that there are many policies [9] for selecting a vulnerable lightpath from the set, for simplicity we select a vulnerable lightpath randomly.

3.2. CLR

CLR is used when connections are protected at the connection level. Recall that a connection traverses a sequence of lightpaths and is also protected by a sequence of link disjoint lightpaths. The backup sharing between two connections is at the lightpath level, see Section 2. Let $A_l^{e'}$ be the set of all connections $c_i\,(i = 1, \ldots, M)$ each with bandwidth w_i traversing physical link e' and protected by lightpath l. The bandwidth reserved on lightpath l to protect against the failure of link e' is $v_l^{e'} = \sum_{i=1}^{M}(w_i), 0 \leqslant v_l^{e'} \leqslant$ STS-192. The total amount of backup bandwidth reserved on lightpath l to protect against the failure of any link e' in the network is $v_l^* = \max_{\forall e' \in E}\{v_l^{e'}\}$.

Algorithm 2. Pseudo code of LLR

1. Identify a set L^u composed by unprotected lightpaths $(l_1^u, l_2^u, \ldots, l_m^u)$, then release unavailable resources along these unprotected lightpaths and reprovision them by allocating new protection wavelength(s) in the physical topology, as explained before. If an unprotected lightpath l_i^u cannot be reprovisioned, move it to another set L_{after}^u.
2. Identify a set L^v composed by vulnerable lightpaths $(l_1^v, l_2^v, \ldots, l_m^v)$. For each vulnerable lightpath l_i, reprovision it using the same method as in step 1. If not successful, move l_i to another set L_{after}^v; otherwise, remove l_i from L^v and reidentify other vulnerable lightpaths in L^v and L_{after}^v, then repeat step 2 until there are no vulnerable lightpaths or no more reprovisioning is possible.

When a link f fails, all lightpaths traversing link f fail and accordingly all connections routed through these lightpaths will also fail. These connections will be rerouted onto their protection routes and become unprotected and hence exposed to new failures. For example, a connection c_1 of bandwidth $2 \times$ STS-1 between nodes C and H is routed through working path (l_5–l_2) and protected by (l_6–l_4–l_1), see Fig. 1b. When link (C–G) fails, the connection is restored to its end to end backup path and after recovery, the connection becomes exposed. Similarly, all connections that were originally protected by any light-path traversing link f also become unprotected. For example, a connection (c_2) between nodes C and E whose working is (l_6, l_4) can be protected by l_5. Hence, when (C–G) fails, l_5 fails and the connection c_2 loses its protection bandwidth and becomes unprotected. Now, when a connection is restored into its protection route, the backup bandwidth reserved on any lightpath along the protection route for this connection is activated and can no longer be shared. Hence, a lightpath l will have $v_l^a = v_l^* - v_l^f$ available protection capacity to protect against a new failure. The protection capacity required, however, on lightpath l to protect against the future failure of any link is $v_l^{\text{new}} = \max_{\forall e' \in E-f}\{v_l^{e'}\}$. For example, if a connection c_3 ($4 \times$ STS-l) between nodes D and E (Fig. 1b) has its working traversing l_3 and protected by l_4; this connection can share protection bandwidth along l_4 with connection c_1 since the working paths of these two connections are link disjoint. Hence, l_4 reserves max($2 \times$ STS-l, $4 \times$ STS-l) $= 4 \times$ STS-l to protect these two connections. When link (C–G) fails, the available backup bandwidth on l_4 becomes $4 \times$ STS-1 $- 2 \times$ STS-l $= 2 \times$ STS-l, which is not sufficient to protect c_3. Hence, c_3 becomes vulnerable to a new failure.

Let C_l be the total capacity of a lightpath, R_l be the residual capacity, A_l be the bandwidth used by working and *active* backup connections; hence, $R_l = C_l - A_l - v_l^a$. Let $\Delta = \{l_1, l_2, \ldots, l_1\}$ be the set of all lightpaths on which failed connections are rerouted. If for every $l_i (i = 1, \ldots, L)$, (1) $v_{l_i}^a \geqslant v_{l_i}^{\text{new}}$, then there will be no vulnerable connections in the network; or (2) $v_{l_i}^a + R_{l_i} \geqslant v_{l_i}^{\text{new}}$, then the lightpath l_i has enough available capacity (and should be reserved) that can protect against the failure of any link in the network. Hence, the backup capacity reserved along l_i becomes $v_{l_i}^{\text{new}} = \max_{\forall e' \in E-f}\{v_{l_i}^{e'}\}$. In this case, only unprotected connections as mentioned earlier need to be reprovisioned. Alternatively, when (1) and (2) are not satisfied for at least one lightpath l_i, then some connections in the network are vulnerable to a new failure. In this case, the set of all vulnerable connections is identified (as in LLR); a vulnerable connection is reprovisioned by allocating new protection capacity on its backup route. The set Δ is updated and conditions (1) and (2) are checked again for vulnerable connections. The same procedure is repeated until there are no more vulnerable connections or no more reprovisioning is possible.

Algorithm 3. Pseudo code of CLR

1. Identify a set U of unprotected connections $C_1^U, C_2^U, \ldots, C_m^U$.
2. Reprovision each connection C_i^U. If not successful, move C_i^U from U to another set U_{after} (which stores the unprotected connections after reprovisioning). Repeat step 2 until U is empty or no more reprovisioning is possible.

3. Initialize a set V composed by vulnerable connections $C_1^V, C_2^V, \ldots, C_m^V$, which are identified by the model described in Section 3.2.
4. Reprovision connection C_j^V. If successful, remove C_j^V from V; otherwise, move C_j^V to a set V_{after} and in both cases re-evaluate the remaining vulnerable connections in the network (i.e., two sets V and V_{after}) for vulnerability; those that become protected are removed from the two sets. Repeat step 4 until V is empty and exit. All connections in V_{after} are vulnerable and exposed for future failures.

3.3. LLR vs. CLR

Both of these schemes rely on spare capacity reprovisioning after the first failure in order to improve the network survivability against a new failure. However, the two schemes present some critical differences.

The first difference between the two schemes pertains to the granularity at which each scheme reprovisions protection bandwidth for its demands. In LLR, unprotected and vulnerable lightpaths are identified after a link failure; all unprotected lightpaths are reprovisioned and some of those that are vulnerable are reprovisioned in order to resolve the conflict between lightpaths sharing the same protection resource and reduce the network vulnerability to future failures. Hence, the end nodes of the demands are not aware of this re provisioning process. The source node of a failed *p-lightpath* reconfigures new backup resources without the intervention of end nodes of the connections it carries. Therefore, when an unprotected or vulnerable lightpath is successfully reprovisioned, then all demands traversing this lightpath becomes protected conversely, if the lightpath could not be reprovisioned, then all the demands it carries are unprotected. Thus, LLR provides collective reprovisioning for low speed connections. On the other hand, in CLR connections are reprovisioned at a smaller granularity. Here, the number of connections that are unprotected or vulnerable is substantially much more than the number of unprotected or vulnerable lightpaths (although the number of unprotected or vulnerable connections in both cases may be the same). In CLR, the end node of every unprotected/vulnerable connection needs to reprovision new protection capacity; hence, the management overhead may be excessive.

A second difference between LLR and CLR pertains to the method of reprovisioning. In LLR, all lightpaths are reprovisioned by setting up new protection resources on the physical layer. If resources are not available, then the reprovisioning fails and the lightpath remain unprotected. Alternatively, under CLR, unprotected connections are reprovisioned first on the logical layer; that is, the algorithm first attempts to allocate protection resources on already existing lightpaths. If this fails, then the physical layer is requested to setup new lightpaths. Hence, although the number of unprotected connections to be reprovisioned (under CLR) is much larger than the number of unprotected lightpaths (under LLR), CLR enjoys more flexibility for capacity reprovisioning. This is particularly advantageous when CLR is implemented with PAC-HCL since this latter has better flexibility and more efficient bandwidth sharing than PAC.

4. Performance evaluation

This section presents quantitative comparisons between PAL, PAC, and PAC-HCL (HCL = 3 throughout the simulations) presented in Section 2 and it also compares the performance of LLR and CLR presented in Section 3. We simulate a dynamic network environment where connection requests are uniformly distributed between all source–destination pairs and their arrival process is Poisson. The connection holding time of each connection follows a negative exponential distribution. The capacity of each wavelength is STS-192; the number of the connection requests follows the distribution: STS-1:STS-3c:STS-12c:STS-48c = 12:5:2:1. The load (in Erlangs) is defined as the arrival rate of connection requests times average holding time times a connections average bandwidth normalized in the unit of STS-192. The network we simulated consists of 24 nodes and 43 bi-directional links [9,19] and the number of wavelengths per link is $W = 8$.[3] Our experimental work is divided into two parts; the provisioning or grooming performance and the reprovisioning performance.

4.1. Provisioning results

We compare PAL, PAC, and PAC-HCL using the following metrics: bandwidth blocking probability (BBP), length of working and backup paths, impact of grooming capacity, and efficiency of backup sharing.

Fig. 2 shows the BBP for the different grooming schemes; the number of grooming ports is 16 (16 add and 16 drop) per node. The BBP is defined as the amount of bandwidth blocked over the amount of bandwidth requested. The figure shows that PAL and PAC-HCL have comparable performance with PAC-HCL slightly outperforming PAL; while PAC on the other hand is exhibiting worse performance than the other schemes. The reasons are as follows. First, under PAC, a connection traverses a sequence of lightpaths and is protected by another physically disjoint sequence of lightpaths. When a lightpath traverses more hops (i.e., is longer), finding two sets of lightpaths that are end to end physically disjoint becomes more difficult (physical disjoint constraint). Second, sharing of backup bandwidth under PAC is end to end; that is, as mentioned in Section 2, two connections must be link-disjoint themselves and

[3] This flat capacity networks is artificial and is unrealistic of real networks. It is only used as a suitable test case for research purposes.

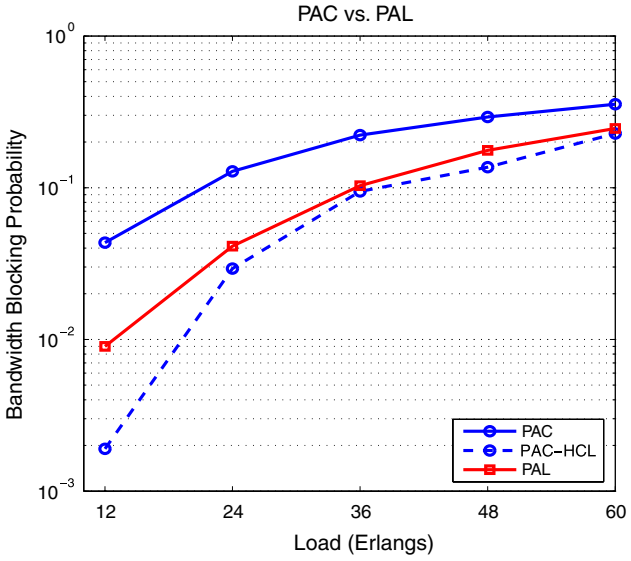

Fig. 2. Bandwidth blocking probability.

should have their protection paths traversing the same lightpath in order for them to share protection bandwidth on that lightpath. This is difficult to achieve under PAC due to the physical disjoint constraint particularly when lightpaths may traverse more physical hops. Third, in PAC although the bandwidth on the existing lightpaths may be available for carrying new connections, the physical disjoint constraint prevents some of these connections from being routed on the logical topology and instead they are routed on the physical topology by setting up new lightpaths and therefore consuming new wavelengths. And fourth, since a connection when successfully routed may traverse more physical hops, it essentially consumes more bandwidth resources and therefore increases the bandwidth blocking probability.

Alternatively, in PAL, a connection traversing *p-lightpaths* does not necessarily have to be end to end disjoint; only the working and protection lightpaths of a *p-lightpath* need to be. Moreover, backup sharing is not end to end between connections and it is at the *p-lightpath* level.[4] That is, two connections can share protection bandwidth although they are not end to end link-disjoint themselves. It is sufficient that they both traverse a pair of *p-lightpaths* where the working of these *p-lightpaths* are disjoint and their protection share a common wavelength link. PAC-HCL, on the other hand, outperforms PAL since it allows the grooming of protection and working bandwidths on the same lightpath. It also outperforms PAC since restricting the hop count of a lightpath yields better flexibility in finding disjoint routes on the logical topology and furthermore backup bandwidth sharing is better exploited.

[4] In a sense, PAL behaves like link protection of a mesh network whereas PAC behaves like path protection.

Fig. 3 shows the physical hop count for working and protection connections in all three schemes. We have two findings here. First, connections under PAL are routed through longer routes than the connections under PAC, PAC-HCL. Note that PAL allows this because "backup sharing" condition and "physically link-disjoint" condition are not end-to-end and rather they are only at the lightpath level. Second, as the load increases, physical hops of working/backup paths in all schemes decrease because longer lightpaths are blocked and connections tend to traverse shorter routes under higher loads. As expected, by limiting the hop count of lightpaths in PAC-HCL connections tend to traverse shorter hops and hence consume less bandwidth resources; this is one of the reasons that PAC-HCL outperforms other schemes.

So far, we have neglected the effects of the network grooming capacity on the performance of the grooming schemes. Fig. 4 shows the BBP vs. grooming capacity when the network load is 24 Erlangs. The figure shows when the number of grooming add/drop ports is smaller, PAL outperforms PAC and PAC-HCL. That is expected since a protection lightpath in a *p-lightpath* under PAL does not consume any grooming ports. Unlike PAL, all lightpaths are setup under PAC and PAC-HCL and each consumes one pair of add/drop grooming ports. Therefore, when this number is small, the poor performance of both schemes of PAC is evident (64–73% blocking). However, as the number of grooming ports increases, the performance gradually improves. One notable observation is that PAC-HCL outperforms PAC, which is different than one would expect; that is, the shorter is the lightpath, the more grooming ports one needs to consume. The reason PAC-HCL shows better performance than PAC is due to the four reasons mentioned before. The bandwidth in the logical topology is more judiciously used due to the increased routing flexibility and better bandwidth sharing. Therefore, fewer lightpaths are setup and hence fewer grooming ports are consumed. Note that when the number of grooming ports is increased to 16, PAC-HCL slightly outperforms PAL. This is consistent with Fig. 2 and can be explained by similar reasons. It is important to mention here that the average nodal degree of the network studied is 3.74.

Next, we study the sharing efficiency of backup bandwidth under PAC and PAC-HCL. We do not consider PAL, since sharing is not end to end and is done between *p-lightpaths* at the wavelength level (i.e., PAL normally has better sharing). We measure the total amount of backup bandwidth reserved at a particular load (e.g., 48 Erlangs) throughout the simulation time for both schemes. The base of comparison is the dedicated protection, in which no bandwidth sharing is allowed. Here, if the set of connections protected by a particular lightpath is $c_i (i = 1, \ldots, n)$ with w_i bandwidth per demand, then the amount of backup bandwidth reserved on that lightpath to protect those demands is $\sum_{i=1}^{n}(w_i)$ If sharing is allowed, then the backup bandwidth reserved on a lightpath l is $v_l^* = \max_{\forall e' \in E}\{v_l^{e'}\}$, where $v_l^{e'}$ is the bandwidth reserved on

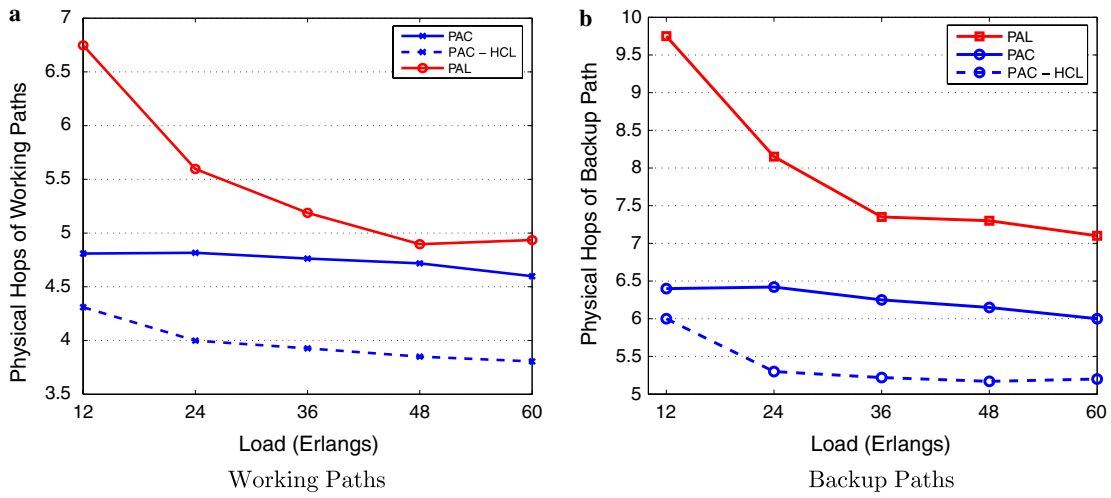

Fig. 3. Average hop count of connections.

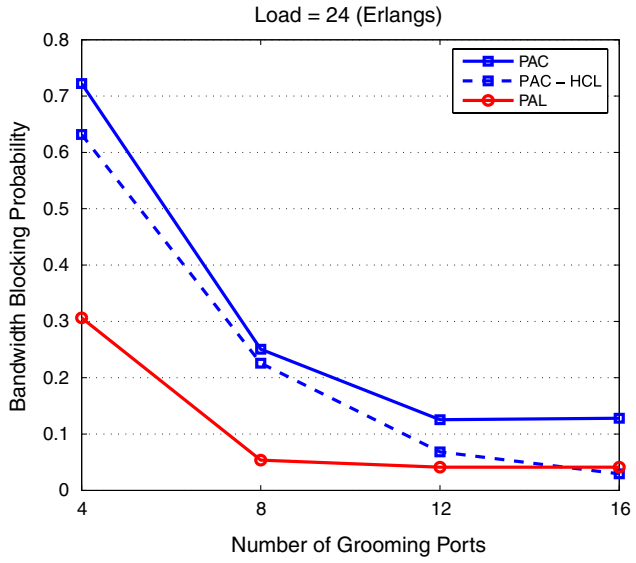

Fig. 4. BBP vs. grooming capacity.

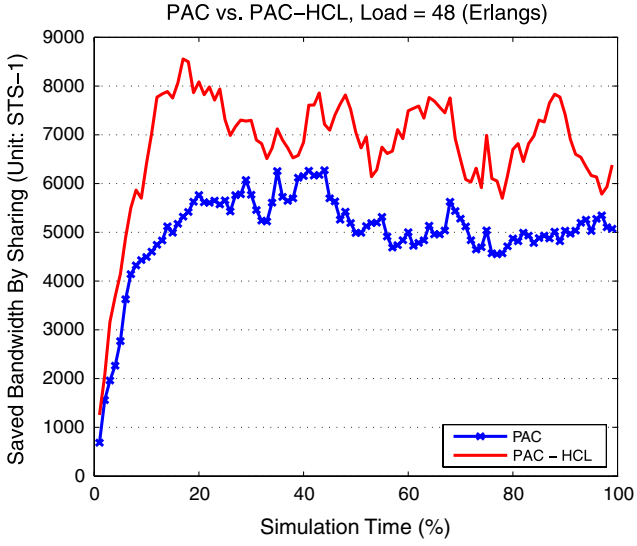

Fig. 5. Backup bandwidth sharing performance.

l to protect against the failure of link e' in the network; $v_l^{e'} = \sum_{i=1}^{M}(w_i), 0 \leqslant v_l^{e'} \leqslant$ STS-192.

We calculate the saving that bandwidth sharing yields over the dedicated protection case under both schemes; e.g., the saving per lightpath l is $\sum_{i=1}^{n}(w_i) - v_l^*$. Clearly, as Fig. 5 shows, the saving under PAC-HCL is more than that of PAC which means that backup bandwidth sharing is more efficient under PAC-HCL. The figure shows a maximum bandwidth of almost 3000 STS-1 that PAC-HCL can additionally save over the savings achieved by PAC. On average this additional saving is 1772 STS-1. We should also note here that under PAC-HCL, more demands are admitted into the network (11.57% more than PAC) and that the bandwidth reserved to protect the connections is on average 463 STS-1 less than that of PAC. So, compared with PAC, PAC-HCL protects more demands by using less resources.

4.2. Reprovisioning results

In this section, we compare the performance of LLR and CLR in improving the network robustness against multiple failures. We simulate the failure of one unidirectional link and we calculate the percentage of unprotected/vulnerable connections in the network before and after reprovisioning for the two schemes.

Fig. 6 shows the percentage of unprotected connections.[5] Clearly, under PAL the percentage of unprotected connections before reprovisioning is more than PAC and PAC-HCL. To understand the reason, we note here that the set of unprotected connections include (1) connections that directly fail and (2) connections that become unpro-

[5] The total number of connections in the network when the link fails in PAC is smaller than PAC-HCL and the latter is slightly smaller than that under PAL. Example, 2191 vs. 2340 vs. 2390 at 60 Erlangs loads.

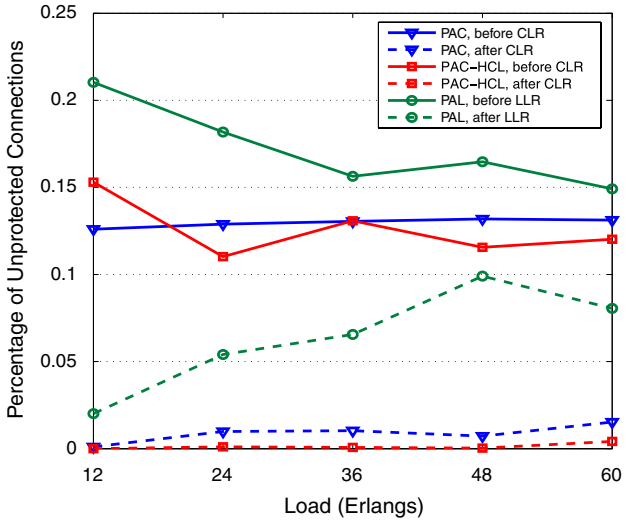

Fig. 6. Unprotected connections.

tected because they lost their protection connections. Since, generally more connections are admitted into the network under PAL, normally more connections in category (1) become unprotected. However, our simulation results show that there is a slight higher number of admitted connections to the network in PAL than PAC and PAC-HCL. Therefore, this higher percentage shown in Fig. 6 is mainly coming from category (2). To elaborate, note the fact that PAL has a better backup bandwidth sharing than PAC and PAC-HCL; hence, a large number of connections directly lose their protection paths when a link fails and this explains the higher percentage of unprotected connections. Further, note that under PAL the percentage decreases as the load increases (e.g., from 21% to 15% as the load varies between 12 Erlangs and 60 Erlangs). Although the percentage of unprotected connections decreases, it does not necessarily mean that the number of unprotected connections at higher load is lower in the network. The reason is that at a higher load, the total number of connections admitted into the network becomes large and hence when a link fails, the fraction of unprotected demands is higher in comparison with the fraction at lower loads although the percentage is lower. PAC and PAC-HCL on the other hand shows similar results; the percentage of unprotected connections under PAC-HCL is slightly smaller, however the total number of unprotected connections between the two schemes is very close.

Now after reprovisioning, LLR yields a large number of unprotected connections by comparison with CLR. The reason that LLR does not have good performance is due to the granularity at which LLR reprovisions connections; here, only unprotected lightpaths are reprovisioned, instead of unprotected connections, by requesting resources from the physical layer. This means, although resources may be available at the lightpath layer, they cannot be exploited. Moreover, when LLR fails to reprovision a lightpath, all connections traversing that lightpath remain exposed to a new failure. Alternatively, CLR reprovisions connections at a finer granularity than LLR; every unprotected connection is identified and an attempt is made to protect that connection. CLR exploits resources at the logical (or lightpath) layer to find sufficient protection resources. When this fails, it requests resources from the physical layer to setup new lightpaths in order to protect exposed connections. Accordingly, CLR shows a much better performance than LLR. Our simulation showed that more than 80% of the unprotected connections are successfully reprovisioned using CLR at the logical layer whereas only less than 20% of connections are reprovisioned by setting new lightpaths at the physical layer. Fig. 6 also shows that although PAC and PAC-HCL both use CLR, PAC-HCL yields a slightly lower percentage of unprotected connections after reprovisioning. This is due to the fact that under PAC, a connection traverses a longer path (i.e., larger physical hop count) and hence consumes more network resources than PAC-HCL. Moreover, when PAC-HCL is used, the physical layer has more resources than PAC. Tables 1 and 2 show our simulation results that present the numbers of connections to be reprovisioned in physical/logical topology under PAC and PAC-HCL; columns A–D represent the number of unprotected and vulnerable connections before reprovisioning, the number of connections successfully reprovisioned in the logical topology, the number of connections successfully reprovisioned in the physical topology and the number of unprotected connections left after reprovisioning (note that, a vulnerable connection becomes unprotected if it cannot be reprovisioned) respectively. Clearly, the results show that using CLR, 80% of the connections (i.e., column C) which are left to be reprovisioned at the physical layer (i.e., columns C and D) can be successfully reprovisioned under PAC-HCL and this is mainly due to higher resource availability

Table 1
PAC-HCL

Loads	A	B	C	D
12	85	85	0	0
24	170	160	9	1
36	342	309	30	3
48	374	305	65	4
60	461	360	79	22

Table 2
PAC

Loads	A	B	C	D
12	115	114	1	0
24	230	215	5	10
36	319	290	12	17
48	516	431	56	29
60	594	473	55	66

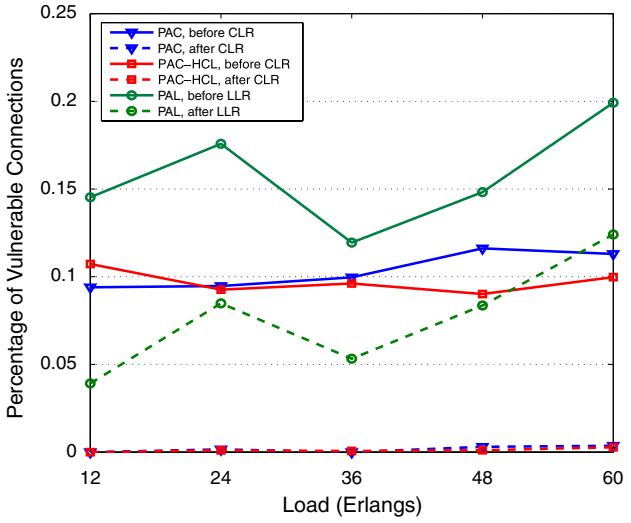

Fig. 7. Vulnerable connections.

at the physical layer. Whereas, under PAC, over 50% gets blocked and hence they remain unprotected.

Fig. 7 shows the connection vulnerability before and after reprovisioning. Clearly, the higher is the shareability of protection resources, the more would be the vulnerability of connections after the first failure. The figure shows that PAL has always higher vulnerability before and after reprovisioning. We also observe that the vulnerability increases as the load increases which is due to the fact that the sharing potential gets higher at higher loads. Similar to before, since the granularity of LLR is a lightpath, when a vulnerable light-path fails to be reprovisioned, all connections carried by this lightpath remain vulnerable (and hence unprotected). Therefore, PAL shows higher connection vulnerability after LLR reprovisioning. CLR, on the other hand, reduces the vulnerability of connections groomed using either PAC or PAC-HCL due to its finer granularity substantially. We similarly notice that the majority of vulnerable connections are reprovisioned at the lightpath layer.

Network robustness is another important performance metric used to compare LLR and CLR. Robustness is defined as the capability of the network to maintain high restorability[6] of its connections (e.g., 95% or above) when a pair of links randomly fail (one after the other) [10]. We measure the robustness before and after reprovisioning and for different add/drop grooming ports per node (8 or 16). Our evaluation is based upon measuring the percentage of links in the network that yields higher dual failure restorability after the first failure. In other words, the robustness is measured by first taking a link down and then measuring the restorability of the connections when another link fails from the remaining links. We then measure the percentage of links that result in a particular restorability value. This experiment is repeated for all links in the network and then we average all the results. Hence, the larger the fraction of network links that yield higher connection restorability, the better is the overall robustness. That is, given equal failure probability on all links, if dual failure restorability is kept at a desirable level for the majority of these links, then the network is said to be more robust. Fig. 8 shows a comparison of the network robustness before and after reprovisioning at a load of 24 Erlangs. It shows 10 different intervals for the restorability ranging from 0% to 100%. Namely, one large interval is chosen to cover a relatively low restorability range 0–55% and the remaining intervals are chosen in increments of 5% to cover higher ranges above 55%. The figure shows the robustness of the network as the probability of having the restorability (R) within a certain interval. For example as Fig. 8a shows, the 90% restorability of PAC is defined as $\Pr(R \geqslant 90\%) = \Pr(R \in [95\% \sim 100\%]) + Pr(R \in [90\% \sim 95\%]) = 0.22$. After reprovisioning, this value increases to 0.96 (see Fig. 8b).

First, LLR improves the robustness of PAL; before reprovisioning $\Pr(R \geqslant 90\%) = 0.35$ (Fig. 8(a)) and after reprovisioning this value becomes 0.6 (Fig. 8b). This is justified from Figs. 6 and 7, where we showed that the percentage of unprotected connections drops from 18% to 6% (at a load of 24 Erlangs) and the percentage of vulnerable connections drops from 18% to 8% before and after reprovisioning correspondingly. Alternatively, the robustness (e.g., $\Pr(R \geqslant 90\%)$) of PAC (PAC-HCL) improves from 22% (48%) before reprovisioning to almost 96% using CLR (Fig. 8a and b). This shows a substantial improvement of CLR over LLR; this is clearly explained in the previous discussions and from Figs. 6 and 7 where after reprovisioning only a very small percentage of unprotected and vulnerable connections exist in the network. CLR performance is equal for PAC and PAC-HCL (e.g., at higher restorability) due to the small percentage of vulnerable and unprotected connections remaining in the network.

Another observation is with regards to the impact of grooming capacity on the network robustness. We study the robustness when the grooming capacity is 8 and 16 add/drop ports per node. As mentioned earlier, the grooming capacity has minor impact on PAL (see Fig. 4) and hence on LLR. This is due to the fact that under PAL protection lightpaths do not consume any add/drop ports. However, the grooming capacity has direct effect on PAC and PAC-HCL and hence on CLR. For example, before reprovisioning when we increase the grooming capacity from 8 (Fig. 8a) to 16 (Fig. 8c), the robustness (e.g., $\Pr(R \geqslant 90\%)$) changes from 22% and 48% to 43% and 51% for PAC and PAC-HCL correspondingly. After reprovisioning, the robustness of PAC and PAC-HCL changes from around 96% to almost 100% after reprovisioning (Fig. 8b–d). We find that when the grooming capacity increases, small gain is achieved by the reprovisioning

[6] The restorability, $R(i,j)$,of a double failure $(i,j)$is defined as the portion of all working paths $w_i + w_j$ on links i and j that are simultaneously affected and survive the failures [5,22].

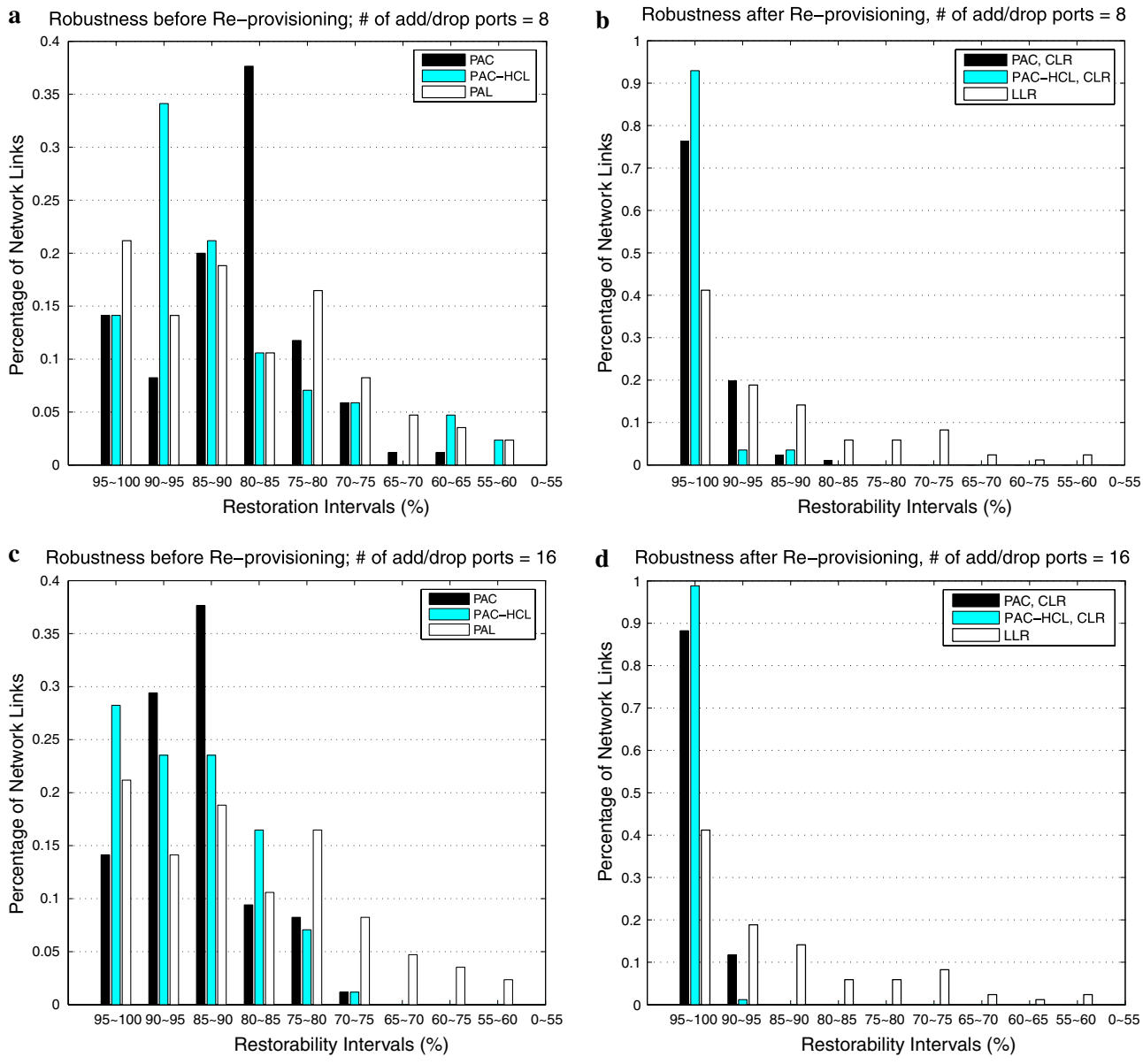

Fig. 8. Network robustness.

algorithm (in terms of robustness). Although as shown in Fig. 4, the BBP reduces substantially and hence more connections are admitted into the network.

5. Conclusion

In this paper, we considered the problem of protecting low speed connections in optical networks against multiple near simultaneous failures. These low speed connections are groomed together either using PAL or using PAC survivable grooming policies. To improve the survivability of these connections, we proposed to use spare capacity reprovisioning after the first failure in order to allocate protection resources and protect exposed and vulnerable connections. We proposed two different reprovisioning schemes, LLR and CLR, and studied their performances. The two schemes differ in the granularity at which they reprovision spare resources and which grooming policy they each require. For example, LLR uses PAL and operates at the lightpath level; on the other hand, CLR uses PAC and operates at a finer granularity (connection level). We have shown that CLR substantially outperforms LLR due to the increased flexibility that it enjoys. In addition, CLR reuses the available capacity at the lightpath level to protect exposed or vulnerable connections before requesting resources from the physical layer. Our results have shown that 80% of the unprotected/vulnerable connections are accommodated at the lightpath layer. LLR on the other hand only reprovisions at the physical layer although resources may be available in the existing lightpaths. We have measured the robustness of the network against dual failures and have shown that a network

deploying CLR with PAC as a grooming policy achieves a very high robustness by comparison to LLR under PAL.

Since, CLR deals with a larger number of connections, the management overhead may be excessive as opposed to the smaller number of lightpaths that LLR deals with. Hence, we intend in the future to assess the overhead resulting from CLR and how this could impact the robustness of the network when CLR is implemented in a distributed environment.

References

[1] W. Grover, Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET and ATM Networking, Prentice Hall, Englewood Cliffs, NJ, 2003.

[2] J. Labourdette, Shared Mesh Restoration in Optical Networks, Proc. OFC'04, February 2004.

[3] S. Ramamurthy, B. Mukherjee, Survivable WDM mesh networks, Part II – Restoration, IEEE ICC 1999.

[4] H. Choi, S. Subramaniam, H. Choi, On double-link failure recovery in WDM optical networks, IEEE INFOCOM, 2002.

[5] M. Clouqueur, W.D. Grover, Availability analysis of span-restorable mesh networks, IEEE JSAC 20 (4) (2002) 810–821.

[6] W. He, A. Somani, Path-based protection for surviving double-link failures in mesh-restorable optical networks, Proc. IEEE Globecom 2003 (2003).

[7] S. Kim, S. Lumetta, Evaluation of Protection Reconfiguration for Multiple Failures in WDM Mesh Networks, Proc. OFC03, March 2003.

[8] R. Ramamurthy, A. Akyamac, J.-F. Labourdette, S. Chaudhuri, Pre-Emptive Reprovisioning in Mesh Optical Networks, Proc. OFC03, March 2003.

[9] J. Zhang, K. Zhu, B. Mukherjee, A Comprehensive Study on Backup Reprovisioning to Remedy the Effect of Multiple-Link Failures in WDM Mesh Networks, ICC04, Paris, June 20–24, 2004.

[10] C. Assi, W. Huo, A. Shami, N. Ghani, Analysis of capacity reprovisioning in optical mesh networks, IEEE Comm. Lett. 9 (7) (2005) 658–660.

[11] D. Schupke, R. Prinz, Performance of Path Protection and Rerouting for WDM Networks Subject to Dual Failures, Proc. OFC03, March 2003.

[12] D.A. Schupke, W.D. Grover, M. Clouqueur, Strategies for Enhanced Dual Failure Restorability with Static or Reconfigurable *p*-Cycle Networks, ICC'04, Paris, June 20–24, 2004.

[13] K. Zhu, B. Mukherjee, Traffic grooming in an optical WDM mesh network, IEEE JSAC 20 (2002) 122133.

[14] E. Modiano, P.J. Lin, Traffic grooming in WDM networks, IEEE Commun. Mag. 39 (2001) 124129.

[15] B. Mukherjee, C. Ou, H. Zhu, K. Zhu, N. Singhal, S. Yao, Traffic grooming for mesh optical networks, OFC 2004, 2004.

[16] Hongyue Zhu, Hui Zang, Keyao Zhu, Biswanath Mukherjee, A. Novel, Generic graph model for traffic grooming in heterogeneous WDM mesh networks, IEEE/ACM Trans. Networking 11 (2) (2003) 285–299.

[17] O. Gerstel, R. Ramaswami, G.H. Sasaki, Cost-effective traffic grooming in WDM rings, IEEE/ACM Trans. Networking 8 (2000) 618630.

[18] X. Zhang, C. Qiao, An effective and comprehensive approach for traffic grooming and wavelength assignment in SONET/WDM rings, IEEE/ACM Trans. Networking 8 (2000) 608617.

[19] Canhui Ou, Keyao Zhu, Hui Zang, Laxman H. Sahasrabuddhe, Biswanath Mukherjee, Traffic grooming for survivable WDM networks – shared protection, IEEE JSAC 21 (9) (2003) 1367–1383.

[20] Wang Yao, Byrav Ramamurthy, Survivable Traffic Grooming in WDM Mesh Networks Under SRLG Constraints, ICC05, Seoul, Korea, 2005.

[21] D. Xu, C. Qiao, Y. Xiong, An Ultra-fast Shared Path Protection Scheme – Distributed Partial Information Management, Part II, ICNP'02, Paris, 2002, pp. 344–353.

[22] M. Clouqueur, W.D. Grover, Mesh-Restorable Networks with Complete Dual Failure Restorability and with Selectively Enhanced Dual-Failure Restorability Properties, SPIE OPTICOMM, Boston, MA, 2002.

Chadi M. Assi received the B.S. degree in engineering from the Lebanese University, Beirut, Lebanon, in 1997 and the Ph.D. degree from the Graduate Center, City University of New York, New York, in April 2003. He was a Visiting Researcher at Nokia Research Center, Boston, MA, from September 2002 to August 2003, working on quality-of-service in optical access networks. He joined the Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, QC, Canada, in August 2003 as an Assistant Professor. Dr. Assi received the Mina Rees Dissertation Award from the City University of New York in August 2002 for his research on wavelength-division-multiplexing optical networks. His current research interests are in the areas of provisioning and restoration of optical networks, wireless and ad hoc networks, and security.

Wei Huo received his Bachelor and Master degrees of Computer Science from Xi′an Jiaotong University, Xi'an (China) in 1997 and 2000 respectively. He also finished his Master's in the Department of Electrical and Computer Engineering in 2005 from Concordia University, Montreal, Canada. From year 2000 to 2003, he was a software engineer at Lucent Technologies in Shanghai, China. Currently, he is working as a Software Quality Assurance engineer in Montreal.

Abdallah Shami received the B.E. degree in electrical and computer engineering from the Lebanese University, Beirut, Lebanon, in 1997, and the Ph.D. degree in electrical engineering from the Graduate School and University Center, City University of New York, New York, in September 2002. In September 2002, he joined the Department of Electrical Engineering at Lakehead University, ON, Canada, as an Assistant Professor. Since July 2004, he has been with the University of Western Ontario, London, ON, Canada, where he is currently an Assistant Professor in the Department of Electrical and Computer Engineering. His current research interests are in the area of wireless/optical networking, EPON, WIMAX, and WLANs. Dr. Shami held the Irving Hochberg Dissertation Fellowship Award at the City University of New York and a GTF Teaching Fellowship.