This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2009 proceedings.

CFP: Cooperative Fast Protection

Bin Wu, Pin-Han Ho, Kwan L. Yeung, Janos Tapolcai and Hussein T. Mouftah

***Abstract*—We introduce Cooperative Fast Protection (CFP) as a novel protection scheme in WDM networks. CFP achieves capacity-efficient fast protection with the features of node-autonomy and failure-independency. It differs from *p*-cycle by reusing the released working capacity of the disrupted lightpaths (i.e. stubs) in a cooperative manner. This is achieved by allowing all the failure-aware nodes to switch the traffic, such that the disrupted lightpaths can be protected even if the end nodes of the failed link are not on the protecting cycles. CFP also differs from FIPP *p*-cycle by not requiring the source node of the disrupted lightpath on the protecting cycle. By jointly optimizing both working and spare capacity placement, we formulate an ILP for CFP design. Numerical results show that CFP significantly outperforms *p*-cycle by achieving faster protection with much higher capacity efficiency.**

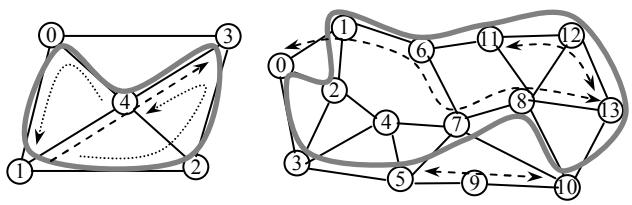
I. Introduction

Over the past decade *p*-cycle has been considered as the most capacity-efficient WDM protection scheme that can achieve the fastest optical recovery speed. In bidirectional WDM networks, a *p*-cycle can protect one unit of working capacity on each on-cycle link and two units on each straddling link [1]. High capacity efficiency is achieved by sharing the spare capacity to protect all the on-cycle and straddling links. Fast optical recovery is achieved because only the two end nodes of the failed link carry out real-time switching.

However, *p*-cycle has some intrinsic characteristics that limit the capacity efficiency and the recovery speed: 1) a *p*-cycle cannot protect those links with at least one end node off the cycle; 2) as a consequence of 1), a *p*-cycle tends to be large in size such that it can traverse or straddle more links for better capacity efficiency. This increases the length of the backup path, which decreases the optical recovery speed and promotes optical signal impairment en route. Though the size of each *p*-cycle can be limited, it implies more *p*-cycles required and suboptimal capacity efficiency; 3) each disrupted lightpath must be rerouted from the upstream end node of the failed link to the downstream one, instead of directly to the destination; and 4) the downstream released working capacity (defined as the *stub*) must be reused by the same lightpath instead of by others. Consider the lightpath $1\rightarrow4\rightarrow3$ in Fig. 1(a). If link (1, 4) fails, the lightpath must be rerouted to $1\rightarrow2\rightarrow3\rightarrow4$, and then reuse its own stub $4\rightarrow3$ to reach the destination. The rerouted lightpath passes through link (3, 4) twice in opposite directions. This is called the *backhaul problem*, where the rerouted traffic loops back to some nodes on the way. It decreases both the capacity efficiency and the optical recovery speed.

The *p*-cycle concept is also extended to path and segment protection. In particular, FIPP (Failure Independent Path Protection) *p*-cycle assumes bidirectional lightpaths on the same route. If a link or node fails, the end nodes of a disrupted lightpath will detect the failure, and then switch the traffic onto the FIPP *p*-cycle. As shown in Fig. 1(b), there are three types of relations between a lightpath and a FIPP *p*-cycle: pure straddling relationship ($5\leftrightarrow9\leftrightarrow10$), pure on-cycle relationship ($11\leftrightarrow12\leftrightarrow13$) and partially straddling/on-cycle relationship ($0\leftrightarrow1\leftrightarrow6\leftrightarrow7\leftrightarrow8\leftrightarrow13$). Protecting the first two is similar to that in link-based *p*-cycle, where only the two end nodes of the lightpath carry out failure detection and switching. For the partially straddling/on-cycle lightpath $0\leftrightarrow1\leftrightarrow6\leftrightarrow7\leftrightarrow8\leftrightarrow13$, the situation is more complex. It can be disrupted due to a failure at on-cycle link (1, 6) or (7, 8), or at another link or node on the lightpath. So, nodes 0 and 13 need to know whether the upper arm of the FIPP *p*-cycle or the lower arm is disrupted or not, and then switch the traffic to a viable arm accordingly. In FIPP *p*-cycle, this is treated as a trivial issue without violating failure-independency. It is explained that the switching nodes can detect not only the disruption of the lightpath, but also the direction from which the loss of light (LOL) or alarm indication signal (AIS) of the FIPP *p*-cycle arrives. Then, the spare capacity in the other direction of the FIPP *p*-cycle, or the predefined default direction if no LOL or AIS is observed on the cycle, can be used to reroute the lightpath.

As a path protection scheme, FIPP *p*-cycle achieves much higher capacity efficiency than link-based *p*-cycle. Let the *length* of a cycle be the number of links it passes through. The length of FIPP *p*-cycles tends to be shorter than that of link-based *p*-cycles, because FIPP *p*-cycles do not need to straddle or pass through as many links as link-based *p*-cycles do. However, some intrinsic characteristics of FIPP *p*-cycle also limit its performance: 1) optical recovery is slower than that in



(a) *p*-cycle (b) FIPP *p*-cycle

Fig. 1. *p*-cycle and FIPP *p*-cycle protection.

Bin Wu and Pin-Han Ho are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, N2L 3G1 (e-mail: b7wu@uwaterloo.ca, pinhan@bbcr.uwaterloo.ca).

Kwan L. Yeung is with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Pokfulam, Hong Kong (e-mail: kyeung@eee. hku.hk).

Janos Tapolcai is with the Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Budapest, Hungary (e-mail: tapolcai@tmit.bme.hu).

Hussein T. Mouftah is with the School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, ON, Canada, K1N 6N5 (e-mail: mouftah@site.uottawa.ca).

link-based *p*-cycle. Not only the upstream on-the-way traffic ahead of the failure point will be lost, but also the switching nodes need to wait for the failure indication signal (such as LOL) before they can switch; 2) a FIPP *p*-cycle cannot protect any lightpath with an end node off the cycle; 3) the downstream released stub of each lightpath is not reused at all; 4) to keep the failure independent property, the spare capacity is generally underutilized. In Fig. 1(b), if lightpath $0\leftrightarrow1\leftrightarrow6\leftrightarrow7\leftrightarrow8\leftrightarrow13$ fails due to a failure at link (6, 7), the FIPP *p*-cycle can protect only one unit of traffic, though there are two usable backup paths on the cycle; 5) to protect a partially straddling/on-cycle lightpath, the switching nodes need signals from both the disrupted lightpath and the FIPP *p*-cycle; and 6) bidirectional traffic on the same route must be assumed. Otherwise it is difficult for the source node of a lightpath to detect the failure by receiving a loss of light (LOL) indication.

We consider directed WDM network which is more general. A novel scheme called Cooperative Fast Protection (CFP) is proposed to protect each lightpath against any single link failure. We observe that a link failure can be detected not only at the two end nodes of the failed link, but also at the destinations of all disrupted lightpaths. Though the destination of a disrupted lightpath cannot accurately localize the failure, it is still failure-aware. The key idea of CFP is to allow all failure-aware nodes to carry out protection switching, such that the stubs of the disrupted lightpaths can be reused in a cooperative manner to set up the backup paths. Our objective is to achieve even higher capacity efficiency than the path-based FIPP *p*-cycle, with even faster optical recovery speed than the link-based *p*-cycle. Meanwhile, each node must be fully autonomous and the protection must be strictly failure independent.

II. Cooperative Fast Protection (CFP)

A. Definition of Failure-Aware Nodes

Upon a link failure, the two end nodes of the failed link can detect the *adjacent failure* by loss of OSC (Optical Supervisory Channel) signal. Meanwhile, all the lightpaths passing through the failed link are disrupted. If the failed link is not incident on the destination of a disrupted lightpath, the destination can detect this *remote failure* by a loss of light (LOL) indication on the lightpath. In CFP, the two end nodes of the failed link and the destinations of all the disrupted lightpaths are identified as *failure-aware nodes*. Due to the transparency of the network, we assume that other nodes on the lightpath cannot sense the failure. All failure-aware nodes can initiate protection switching against the link failure without additional inter-node signalling. This has never been investigated in the previous studies.

B. Working Principles of CFP

CFP organizes the spare capacity into pre-cross-connected cycles. We use Fig. 2 to illustrate how CFP works. In Fig. 2, a failure at link (0, 1) is detected as an adjacent failure by nodes 0 and 1, and a remote failure by 3 and 5 due to the disruption of lightpaths $0\rightarrow1\rightarrow2\rightarrow3$ and $2\rightarrow1\rightarrow0\rightarrow5$. Meanwhile, the working capacity $1\rightarrow2\rightarrow3$ on the first lightpath and $0\rightarrow5$ on the second are released as stubs. Since the two lightpaths pass through (0, 1) in opposite directions, the stub of one lightpath can be reused by the other. By utilizing both the stubs and the spare capacity on the solid (directed) cycle C_1, the backup path for $0\rightarrow1\rightarrow2\rightarrow3$ is $0\rightarrow5\rightarrow6\rightarrow4\rightarrow3$, and that for $2\rightarrow1\rightarrow0\rightarrow5$ is $2\rightarrow1\rightarrow2\rightarrow3\rightarrow5$, where the set of failure-aware nodes {0, 1, 3, 5} perform switching for backup path setup. For simplicity, if a lightpath is protected against all possible link failures using the spare capacity on a cycle, we say that it is protected by this cycle, although the protection may be assisted by some stubs. In CFP, each lightpath is protected by a single cycle, and each cycle can only protect those lightpaths with an on-cycle destination.

If the backup path of lightpath l_1 reuses the stub of lightpath l_2, we call l_2 the *partner* of l_1 at the failed link, where l_2 must pass through this link in the opposite direction of l_1 and its destination must be on the protecting cycle of l_1. However, l_1 may not be the partner of l_2 at the same time. In Fig. 2, lightpaths $0\rightarrow1\rightarrow2\rightarrow3$ and $2\rightarrow1\rightarrow0\rightarrow5$ are partners of each other at link (1, 2). If (1, 2) fails, the two lightpaths can be protected in a similar way, but the set of switching nodes is {1, 2, 3, 5}. Note that the switching at nodes 3 and 5 is independent of the failure location. No matter link (0, 1) or (1, 2) fails, nodes 3 and 5 carry out the same switching. They always receive the restored traffic from the viable arm on C_1, and connect the corresponding stub to the other arm of the cycle. However, the situation is slightly different if the failure is adjacent to the destination of the lightpath. For example, if link (2, 3) fails, lightpath $0\rightarrow1\rightarrow2\rightarrow3$ is rerouted to $0\rightarrow1\rightarrow2\rightarrow4\rightarrow3$, because it has a partner $3\rightarrow2\rightarrow4$ at link (2, 3), and nodes {2, 3, 4} are failure-aware to make the proper switching. Node 3 receives the restored traffic from $4\rightarrow3$ on C_1 and switches the disrupted traffic of $3\rightarrow2\rightarrow4$ (instead of any stub) onto $3\rightarrow5$, where the backup path for $3\rightarrow2\rightarrow4$ is $3\rightarrow5\rightarrow6\rightarrow4$ on C_1. Since node 3 detects an adjacent failure, it does not connect any stub to C_1. The switching at node 3 is still failure-independent against any remote failure.

The above example shows that $0\rightarrow1\rightarrow2\rightarrow3$ can be protected by C_1 against every link failure, though its source 0 and at least one end node of the failed link are not on C_1. The key points are: 1) a partner of $0\rightarrow1\rightarrow2\rightarrow3$ exists at every link on the lightpath, which provides a stub to bridge the disrupted traffic onto C_1; 2) all the failure-aware nodes can properly switch to set up the desired backup paths; and 3) the protection is node-autonomous and failure-independent, where each failure-aware node switches based on the locally observed OSC and LOL signals.

In addition to $0\rightarrow1\rightarrow2\rightarrow3$, both $2\rightarrow1\rightarrow0\rightarrow5$ and $3\rightarrow2\rightarrow4$ in Fig. 2 can also be protected by C_1 in a similar way. For example,

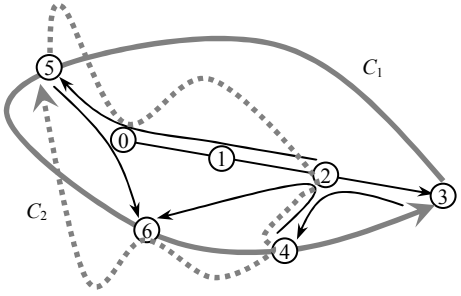


Fig. 2. Cooperative Fast Protection (CFP).

the partner of 2→1→0→5 is 0→1→2→3 at (0, 1) and (1, 2), and 5→0→6 at (0, 5). But 5→0→6 cannot be protected by C_1. To keep the failure-independency feature, each lightpath must be protected by a single cycle, and its destination must respond identically to any possible remote failure. Although a backup path 5→6 on C_1 can be found for 5→0→6 against a failure at (0, 5), the lightpath cannot be protected by C_1 against another failure at (0, 6) due to the lack of a partner. In fact, lightpath 5→0→6 is protected by the dotted cycle C_2 against a failure at (0, 5) or (0, 6). If either link fails, its upstream end node switches the traffic onto C_2, whereas the destination node 6 always receives the restored traffic from 4→6 on C_2. Consider a failure at (0, 5) with two disrupted lightpaths 2→1→0→5 and 5→0→6. The set of failure-aware nodes is {0, 5, 6}. Nodes 0 and 5 detect an adjacent failure and node 6 detects a remote one. The switching at node 0 allows the backup path of 2→1→0→5 to reuse stub 0→6 of 5→0→6. Node 6 connects stub 0→6 to C_1 but receives the restored traffic of 5→0→6 from C_2. Meanwhile, node 5 switches the disrupted lightpath 5→0→6 onto C_2 but receives the restored traffic of 2→1→0→5 from C_1. Accordingly, the backup path for 5→0→6 is 5→0→2→4→6 on C_2, and that for 2→1→0→5 is 2→1→0→6→4→3→5 on C_1. This example shows how the failure-aware nodes, stubs and spare capacity on the cycles work in a cooperative manner to protect all the disrupted lightpaths. With similar analysis, it is easy to see that 3→2→4 is protected by C_1 but 4→2→6 by C_2. Note that CFP cycles are directed. Lightpath 4→2→6 can be protected by C_2 against a failure at (2, 4) because it passes through the link in the opposite direction of C_2.

The switching policy is summarized below: 1) if the destination of a lightpath detects a failure, it receives the restored traffic from the protecting cycle of this lightpath; 2) if a node detects an adjacent failure, it switches each lightpath bounding for the failed link to the stub of its partner. If no partner, the traffic is switched onto the protecting cycle of this lightpath; and 3) if the destination of a lightpath detects a remote failure and the lightpath is the partner of other lightpaths, the destination node connects the stub to a single cycle that can protect all those lightpaths (ensured by ILP). Note that the protection is node-autonomous and strictly failure-independent.

C. Realization of Fast Protection

Now we show how CFP achieves even faster optical recovery than link-based *p*-cycle. If link (1, 2) in Fig. 2 fails, the backup path for 0→1→2→3 is 0→1→0→5→6→4→3, where stub 1→0→5 is all-optically connected. Node 5 switches slightly later than node 1, because the arrival time of LOL at node 5 is slightly deferred due to the optical transmission in the stub. However, the restored traffic reuses the same stub and also suffers the same optical transmission delay. Assume the switching time is the same at each node. When the restored traffic arrives, node 5 should have finished switching. So, the switching at node 5 is transparent to the restored traffic, and optical recovery can be as fast as in link-based *p*-cycle. Moreover, CFP is even faster because 1) the backup path directly goes to the lightpath destination along the cycle (instead of the downstream end node of the failed link), and thus the backhaul problem is effectively suppressed; and 2) CFP cycles do not need to traverse or straddle as many links as in link-based *p*-cycle, and thus tend to have a much shorter cycle length.

D. Capacity Efficiency

CFP is more capacity-efficient than link-based *p*-cycle, because it can fight against a link failure even if the end nodes of the failed link are off the CFP cycle. The fact that CFP suffers less from the backhaul problem also supports its higher capacity efficiency than link-based *p*-cycle. In fact, the capacity efficiency in CFP is even higher than that in FIPP *p*-cycle. Those pure on-cycle and straddling lightpaths in FIPP *p*-cycle can be protected in CFP with the same capacity efficiency but much faster optical recovery due to a different mechanism. For a partially straddling/on-cycle lightpath, the FIPP *p*-cycle can protect only one unit of traffic (see the example in Fig. 1 with a failure at (6, 7)), but there is no such a constraint in CFP. Besides, CFP removes the assumption of bidirectional traffic on the same route, and thus is more general in WDM networks.

III. ILP Formulation

We consider a joint optimization on both working and spare capacity placement with a given traffic matrix. The ILP organizes its constraints into three parts: cycle formulation, routing and protection. Cycle formulation is based on a *Cycle Exclusion* technique [1]. We skip this part due to page limit and interested readers can refer to [1]. The routing part is based on flow conservation of each lighpath. Each lightpath starts at its source and terminates at its destination, whereas all other nodes in the network must obey flow conservation for this lightpath.

The protection part formulates how each lighpath is protected against each possible link failure. In particular, a lightpath can be protected by a cycle only if its destination is on this cycle. If multiple cycles pass through the destination node of a disrupted lightpath, the stub can be connected to at most one cycle, which may not be the one that protects this lightpath. Define the lightpaths passing through the failed link in the same direction as *peers*. Among all the peers, at most one can have its stub connected to a specific cycle. As a result, a cycle can protect a lightpath and its partner at the same time, but not two peers. To keep the feature of failure-independency, the stubs resulting from different link failures on a lightpath must be connected to the same cycle (defined as the *consistency constraint*). If lightpath l is protected by a cycle C_j but its stub is connected to another cycle, then the stub of any other peer of l cannot be connected to C_j (defined as the *sovereignty constraint*). Consider 5→0→6 protected by C_2 in Fig. 2. If (0, 5) fails, stub 0→6 is connected to C_1 at node 6. Suppose there is a peer 5→0→1→2 across (0, 5) with stub 0→1→2 connected to C_2 at node 2. Then, the sovereignty constraint is violated. Due to the switching at node 2, the disrupted lightpath 5→0→6 cannot be properly restored using its backup path 5→0→2→4→6 on C_2.

General Notations:

J: The maximum number of cycles allowed in the solution.

j: CFP cycle index where $j \in \{1, 2, \ldots, J\}$.

$\boldsymbol{V}$: The set of all the nodes in the network.

$\boldsymbol{E}$: The set of all the directed links in the network, where two directed links (u, v) and (v, u) pass through the same

physical link in opposite directions.

c_{uv}: The cost of adding one unit of working or spare capacity to link (u, v) and $c_{uv}=c_{vu}$. If hop-count is used as the cost metric, then $c_{uv}=1$ for each link (u, v). Otherwise c_{uv} may include distance-related cost.

$\boldsymbol{L}$: A given traffic matrix. An entry L_{sd} in $\boldsymbol{L}$ denotes L_{sd} distinct lightpaths between source s and destination d. For simplicity, we use $l \in \boldsymbol{L}$ to denote a lightpath.

λ: A predefined positive fraction where $1/\|\boldsymbol{E}\| \geq \lambda > 0$ (see [1]).

$s(l)$: The source node of lightpath l.

$d(l)$: The destination node of lightpath l.

Decision Variables:

e_{uv}^{j}: Binary variable. It takes 1 if cycle C_j passes through link (u, v), and 0 otherwise.

w_l^{uv}: Binary variable. It takes 1 if lightpath l passes through link (u, v), and 0 otherwise.

r_u^j: Binary variable. It takes 1 if node u is the reversal node (see [1]) in formulating a cycle C_j, and 0 otherwise.

z_u^j: Binary variable. It takes 1 if cycle C_j passes through node u, and 0 otherwise.

q_{uv}^{j}: Fractional variable. It is the voltage of the vector on (u, v) in formulating cycle C_j. It takes 0 if there is no vector on (u, v) (see [1] for definitions of voltage and vectors).

y_l^j: Binary variable. It takes 1 if lightpath l can be protected by cycle C_j, and 0 otherwise.

x_{luv}^{j}: Binary variable. It takes 1 if the stub of lightpath l is connected to cycle C_j upon a remote failure at link (u, v), and 0 otherwise.

h_l^j: Binary variable. It takes 1 if the stub of lightpath l is connected to cycle C_j upon any remote failure on l, and 0 otherwise.

p_{luv}^{j}: Binary variable. It takes 1 if lightpath l passes through link (u, v), and is protected by cycle C_j. Otherwise it is 0.

g_{uv}^{j}: Binary variable. It takes 1 if cycle C_j passes through node u but C_j does not pass through link (u, v) from node u to node v, and 0 otherwise.

$$\text{minimize}\left\{\sum_{j}\sum_{(u,v)\in \boldsymbol{E}} c_{uv}e_{uv}^{j} + \sum_{l\in \boldsymbol{L}}\sum_{(u,v)\in \boldsymbol{E}} c_{uv}w_l^{uv}\right\} \quad (1)$$

$$\sum_{u\in \boldsymbol{V}} r_u^j \leq 1, \qquad \forall j\ ; \quad (2)$$

$$e_{uv}^{j} + e_{vu}^{j} \leq 1, \qquad \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (3)$$

$$\sum_{(u,v)\in \boldsymbol{E}} e_{uv}^{j} = \sum_{(v,u)\in \boldsymbol{E}} e_{vu}^{j}, \qquad \forall u\in \boldsymbol{V},\ \forall j\ ; \quad (4)$$

$$z_u^j = \sum_{(u,v)\in \boldsymbol{E}} e_{uv}^{j}, \qquad \forall u\in \boldsymbol{V},\ \forall j\ ; \quad (5)$$

$$q_{uv}^{j} \leq e_{uv}^{j}, \qquad \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (6)$$

$$r_u^j + \sum_{(u,v)\in \boldsymbol{E}} q_{uv}^{j} - \sum_{(v,u)\in \boldsymbol{E}} q_{vu}^{j} \geq \lambda z_u^j, \qquad \forall u\in \boldsymbol{V},\ \forall j\ ; \quad (7)$$

$$w_l^{uv} + w_l^{vu} \leq 1, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E}\ ; \quad (8)$$

$$\sum_{(s(l),v)\in \boldsymbol{E}} w_l^{s(l)v} = 1, \qquad \forall l\in \boldsymbol{L}\ ; \quad (9)$$

$$\sum_{(u,d(l))\in \boldsymbol{E}} w_l^{ud(l)} = 1, \qquad \forall l\in \boldsymbol{L}\ ; \quad (10)$$

$$\sum_{(u,f)\in \boldsymbol{E}} w_l^{uf} = \sum_{(f,v)\in \boldsymbol{E}} w_l^{fv}, \qquad \forall l\in \boldsymbol{L},\ \forall f\in \boldsymbol{V}: f\neq s(l), f\neq d(l)\ ; \quad (11)$$

$$y_l^j \leq z_{d(l)}^{j}, \qquad \forall l\in \boldsymbol{L},\ \forall j\ ; \quad (12)$$

$$x_{luv}^{j} \leq \frac{1}{2}\left(w_l^{uv} + z_{d(l)}^{j}\right), \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (13)$$

$$\sum_{j} x_{luv}^{j} \leq 1, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E}\ ; \quad (14)$$

$$\sum_{l\in \boldsymbol{L}} x_{luv}^{j} \leq 1, \qquad \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (15)$$

$$\left(1 - w_l^{uv}\right) + x_{luv}^{j} \geq h_l^j, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (16)$$

$$h_l^j \geq x_{luv}^{j}, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (17)$$

$$p_{luv}^{j} \leq \frac{1}{2}\left(y_l^j + w_l^{uv}\right), \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (18)$$

$$\left(1 - w_l^{uv}\right) + p_{luv}^{j} \geq y_l^j, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (19)$$

$$\left(1 - p_{luv}^{j}\right) + h_l^j \geq \sum_{l\in \boldsymbol{L}} x_{luv}^{j}, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (20)$$

$$\sum_{l\in \boldsymbol{L}} p_{luv}^{j} \leq 1, \qquad \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (21)$$

$$y_l^j + w_l^{uv} + e_{uv}^{j} \leq 2, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (22)$$

$$g_{uv}^{j} \leq \frac{1}{2}\left(z_u^j + (1 - e_{uv}^{j})\right), \qquad \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (23)$$

$$y_l^j \leq \left(1 - w_l^{uv}\right) + \sum_{l\in \boldsymbol{L}} x_{lvu}^{j} + g_{uv}^{j}, \qquad \forall l\in \boldsymbol{L},\ \forall (u,v)\in \boldsymbol{E},\ \forall j\ ; \quad (24)$$

$$\sum_{j} y_l^j = 1, \qquad \forall l\in \boldsymbol{L}\ ; \quad (25)$$

Objective (1) minimizes the total working and spare capacity. The set of constraints (2)-(7) is for cycle formulation (see [1] for details). The set of constraints (8)-(11) formulates the routing of each lightpath. Specifically, constraint (8) prevents a lightpath to pass through any link twice. Constraints (9)-(10) stipulate that the lightpath emanates at the source and terminates at the destination. Constraint (11) requires all other nodes to obey flow conservation. The protection part is formulated in (12)-(25). By (12), a lightpath can be protected by a cycle only if its destination is on the cycle. By (13), if the stub of lightpath l can be connected to cycle C_j upon a failure at link (u, v), then l must pass through (u, v) and its destination must be on C_j. Constraint (14) requires the stub of each lightpath to be connected to at most one cycle. Constraint (15) means that only one lightpath among all the peers can have its stub connected to a specific cycle. The consistency constraint is formulated in (16)-(17). According to (16), if lightpath l passes through link (u, v) but its stub is not connected to cycle C_j upon a failure at (u, v), then the stub resulting from any other link failure on l cannot be connected to C_j. Otherwise, the stub resulting from each possible link failure on l must be connected to the same cycle C_j, as formulated in (17). Constraints (18)-(19) define p_{luv}^{j}. The sovereignty constraint is formulated in (20). Constraint (21) means that a cycle cannot protect two or more peers against a link failure. Constraint (22) indicates that a cycle cannot protect a lightpath if both of them

pass through any on-cycle link in the same direction. Constraint (23) defines g_{uv}^{j}. Note that $g_{uv}^{j}=1$ does not prevent cycle C_j to pass through (v, u) from v to u. Constraint (24) says that, if lightpath l passes through link (u, v) and it can be protected by C_j against a failure at (u, v), then it must find a partner at (u, v), or g_{uv}^{j} must be 1 (i.e. the end node u of the failed link must be on C_j, and C_j does not pass through (u, v) in the same direction as l). Finally, constraint (25) ensures that every lightpath is protected by a cycle.

IV. Numerical Results

We consider the SmallNet topology in Fig. 3, where the traffic matrix $\boldsymbol{L}$ includes sixteen lightpaths. The ILP is implemented using ILOG CPLEX 11.0 on a workstation with 3GHz Intel Xeon CPU 5160. Hop-count is used as the cost metric and J=3, λ=0.01. We compare the optimal CFP solution in Fig. 3(a) with the optimal link-based p-cycle solution in Fig. 3(b), which is obtained from an ILP modified based on the Cycle Exclusion approach in [1] for directed networks. For fair comparison, we also carry out a joint design of working and spare capacity placement in the link-based p-cycle scenario. FIPP p-cycle is not compared because it assumes bidirectional traffic on the same route. As we have analyzed in Section II.D, theoretically CFP is more capacity-efficient and more general than FIPP p-cycle, with much faster recovery speed.

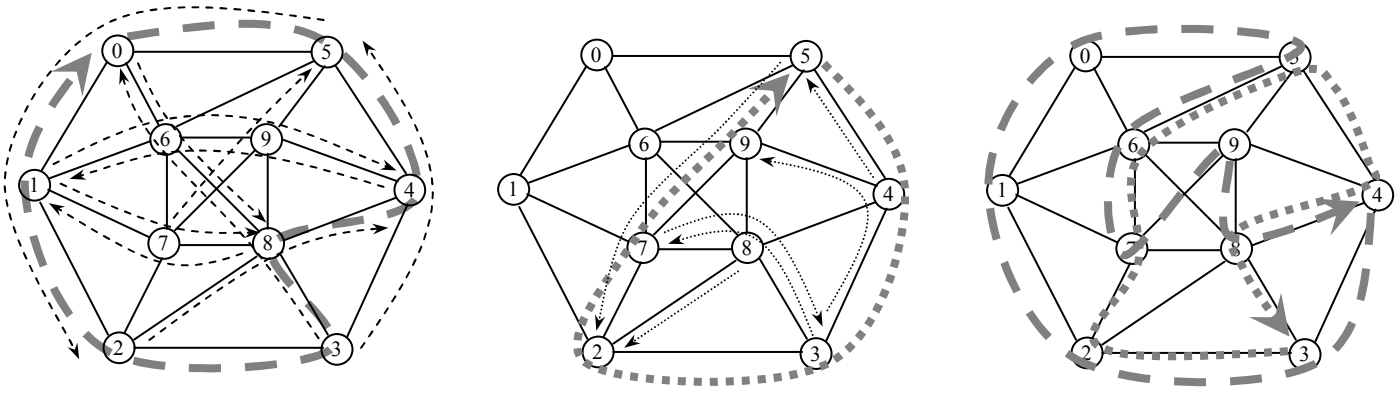
For clarity, in Fig. 3(a) we separate the two CFP cycles and the lightpaths protected by each CFP cycle. By comparing the cycle length of both the dashed and the dotted cycles between the CFP and p-cycle solutions, we can see that CFP cycles tend to have a smaller cycle length. Note that none of the CFP cycles in Fig. 3(a) passes through node 6, which has to be traversed by both p-cycles in Fig. 3(b). In Fig. 3(a), the dashed lightpath 7→9→5 is protected by the dashed CFP cycle, because its partner at links (7, 9) and (5, 9) is the dotted lightpath 5→9→7→2 which is protected by the dotted CFP cycle. Similarly, the partner of the dotted lightpath 8→2 at link (2, 8) is the dashed lightpath 2→8→4. Though lightpaths 8→2 and 2→8→4 are protected by different CFP cycles, the former can reuse the stub released from the latter against a link failure at (2, 8). Compared with the CFP solution in Fig. 3(a), the p-cycle solution in Fig. 3(b) increases the total capacity by 8.33%, and the spare capacity by 30.77%. Besides, the average end-to-end hop-count of the backup paths is 4.89 in the CFP solution, in contrast to 7.03 in the p-cycle solution.

V. Conclusion

We proposed a novel protection scheme called Cooperative Fast Protection (CFP) in WDM (Wavelength Division Multiplexing) networks to protect each lightpath against any single link failure. Based on the observation that a link failure can be detected not only by the two end nodes of the failed link but also by the destination nodes of all the disrupted lightpaths, CFP allows all those failure-aware nodes to carry out protection switching in a node-autonomous and failure-independent manner. Another distinct feature of CFP is that it enables cooperative stub reuse among different lightpaths, such that the backup paths can be set up using both the stubs and the pre-cross-connected spare capacity on the CFP cycles. Upon a link failure, CFP reroutes each disrupted lightpath directly to its destination along the CFP cycle and thus the backhaul problem can be effectively mitigated. The unique features of CFP also allow each lightpath to be properly protected even if the two end nodes of the failed link and the source node of the lightpath are not on the CFP cycle. Compared with link-based p-cycles, CFP cycles tend to have a shorter cycle length, and a CFP solution tends to include less number of cycles. We formulated an ILP for CFP design to jointly optimize both working and spare capacity placement. Numerical results and analysis showed that CFP outperforms those p-cycle based schemes by achieving faster protection with higher capacity efficiency.

References

[1] B. Wu, K. L. Yeung and P.-H. Ho, "ILP formulations for p-cycle design without candidate cycle enumeration," submitted to *IEEE/ACM Trans. Netw.*, http://www.eee.hku.hk/research/doc/tr/TR2008001_IFDCC.pdf.



(a) Optimal CFP solution consisting of two CFP cycles and their protected lightpaths (total cost: 48, spare capacity required: 13, running time: 39004.59 seconds).

(b) Optimal p-cycle solution consisting of two cycles (total cost: 52, spare capacity required: 17, running time: 17.22 seconds).

Fig. 3. An example in SmallNet with 10 nodes and 22 links. The traffic matrix includes 16 lightpaths where $\boldsymbol{L}=\{L_{sd}\}=\{L_{08}=1, L_{14}=1, L_{18}=1, L_{24}=1, L_{30}=1, L_{35}=1, L_{37}=1, L_{39}=1, L_{41}=1, L_{45}=1, L_{52}=2, L_{73}=1, L_{75}=1, L_{81}=1, L_{82}=1\}$. The working paths in the p-cycle solution are the same as those in the CFP solution except for L_{41} and L_{75}, where L_{41} takes 4→8→6→1 and L_{75} takes 7→6→5. For simplicity, working paths are not shown in the p-cycle solution.