Monitoring Cycle Design for Fast Link Failure Localization in All-Optical Networks

Bin Wu, *Member, IEEE*, Kwan L. Yeung, *Senior Member, IEEE*, and Pin-Han Ho, *Member, IEEE*

***Abstract*—A monitoring cycle (m-cycle) is a preconfigured optical loop-back connection of supervisory wavelengths with a dedicated monitor. In an all-optical network (AON), if a link fails, the supervisory optical signals in a set of m-cycles covering this link will be disrupted. The link failure can be localized using the alarm code generated by the corresponding monitors. In this paper, we first formulate an optimal integer linear program (ILP) for m-cycle design. The objective is to minimize the monitoring cost which consists of the monitor cost and the bandwidth cost (i.e., supervisory wavelength-links). To reduce the ILP running time, a heuristic ILP is also formulated. To the best of our survey, this is the first effort in m-cycle design using ILP, and it leads to two contributions: 1) nonsimple m-cycles are considered; and 2) an efficient tradeoff is allowed between the monitor cost and the bandwidth cost. Numerical results show that our ILP-based approach outperforms the existing m-cycle design algorithms with a significant performance gain.**

***Index Terms*—AUTHOR, PLEASE SUPPLY YOUR OWN KEYWORDS OR SEND A BLANK E-MAIL TO KEYWORDS@IEEE.ORG TO RECEIVE A LIST OF SUGGESTED KEYWORDS.**

I. Introduction

As THE communication networks evolve towards all-optical networks (AONs) based on wavelength division multiplexing (WDM) technology, a single fiber is capable of carrying hundreds of wavelengths working in parallel, each at 10 Gb/s or higher data rate. As a result, a single link failure (such as a fiber-cut) can lead to a huge amount of data loss. To minimize the data loss, it is essential to have a fast link failure detection and localization scheme.

Link failure detection and localization can be implemented at different protocol layers. In general, implementations using upper layer protocols [1], [2] (or cross-layer designs [3]) require a much longer detection time than the typical 50 ms requirement for optical domain recovery. To achieve fast link failure detection and localization, optical/physical layer mechanisms are preferred [4]–[7]. At the optical layer, a link failure can be detected by measuring optical power, analyzing optical spectrum, using pilot tones or optical time domain reflectometry (OTDR) [4]–[7]. This is carried out by a special device called *monitor* [8]–[10]. As shown in Fig. 1, each monitor has a direct connection with the traffic manager (TM) which is responsible for traffic engineering. A failure in the network will be detected by a particular set of monitors, which immediately generate alarms to the TM. Then, the TM dynamically localizes the failure and reroutes the affected traffic to bypass the failure.

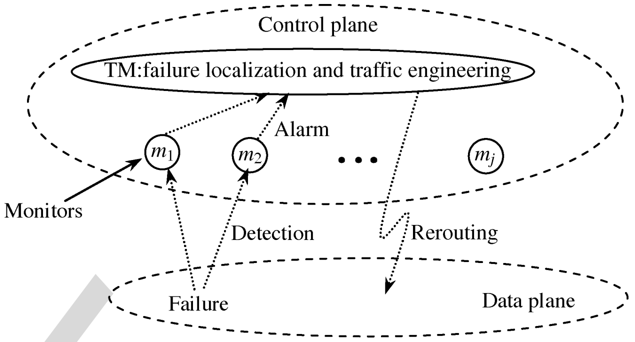


Fig. 1. Failure detection/localization and optical domain recovery.

A channel-based monitoring scheme requires one monitor for each wavelength channel, thereby consuming a very large number of monitors. A link-based monitoring scheme is more scalable, but still requires one monitor per link. It is important to reduce the required number of monitors for hardware cost reduction. More importantly, this minimizes the management efforts of the TM and thus makes the network more scalable.

To further reduce the number of required monitors, the concept of monitoring-cycle (m-cycle) [4]–[7] is introduced. An m-cycle is a preconfigured optical loop-back connection with a dedicated monitor. It is implemented using a supervisory wavelength on each link it covers/passes through. Assume that the bandwidth cost of an m-cycle is measured by its *length*, which is defined as the total number of links it covers. An m-cycle solution consists of a set of M m-cycles $\{c_0, c_1, \ldots, c_{M-1}\}$ covering every link in the network, and the *cover length* is the total length of all the M m-cycles. In other words, the total number of monitors required is M, and the total amount of supervisory bandwidth (i.e., supervisory wavelength-links) is the cover length.

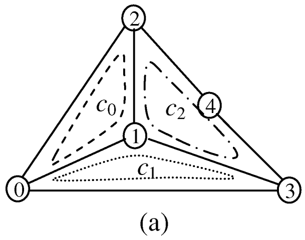
Assume there is at most a single link failure at a time. Upon a link failure, optical supervisory signals in all m-cycles covering the failed link will be disrupted, and the corresponding monitors will alarm. This generates an *alarm code* with the format of $[a_{M-1}, \ldots, a_1, a_0]$, where $a_j = 1$ means that the monitor on m-cycle c_j alarms and $a_j = 0$ otherwise. Fig. 2(a) shows an m-cycle solution consisting of three m-cycles $\{c_0, c_1, c_2\}$, as generated by the M^2-CYCLE algorithm proposed in [7]. If link

Manuscript received April 08, 2008; revised July 03, 2008 and July 28, 2008. This paper was presented in part at the IEEE GLOBECOM '07Washington D. C., USANov.2007.

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

K. 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).

Digital Object Identifier 10.1109/JLT.2008.2005068



Link	c_2	c_1	c_0	Decimal
(0,1)	0	1	1	3
(0,2)	0	0	1	1
(0,3)	0	1	0	2
(1,2)	1	0	1	5
(1,3)	1	1	0	6
(2,4)	1	0	0	4
(3,4)	1	0	0	4

(b)

Fig. 2. m-cycle solution based on M^2-CYCLE algorithm [7] $(\mathrm{D_L} = 1.167)$. (a) m-cycles. (b) Alarm code table.

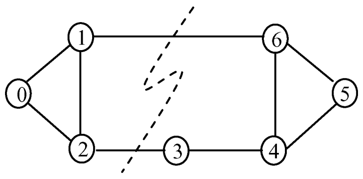
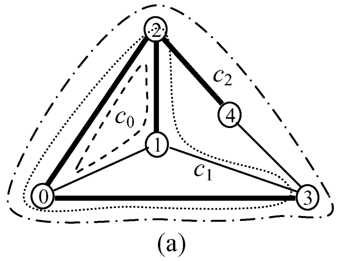


Fig. 3. Two-edge cut.

(0, 1) fails, the monitors on c_0 and c_1 will alarm to generate the alarm code [0, 1, 1]. Similarly, if link (0, 2) fails, the monitor on c_0 will alarm and the resulting alarm code is [0, 0, 1]. The alarm code table in Fig. 2(b) (as maintained by the traffic manager) lists all the possible alarm codes, where a particular link failure can be localized by the alarm code generated. Ideally, we should have a unique alarm code for every link failure. In practice, if two links form a cut [11] of the network topology (defined as a *two-edge cut* where the topology will be divided into two separate parts if the two edges/links are removed), any cycle-based monitoring scheme cannot distinguish the two link failures. This is because the two links must be covered by the same set of m-cycles. In Fig. 2(a), the two links (2, 4) and (3, 4) form a two-edge cut. Fig. 3 gives another example where two links (1, 6) and (2, 3) also form a two-edge cut. If we need to distinguish the two link failures in a two-edge cut, we can add extra link-based monitors [4]–[7]. For example, we can add an extra link-based monitor to either (2, 4) or (3, 4) in Fig. 2(a). This increases the total number of monitors from 3 to 4, but it is still less than 7 as required by a pure link-based monitoring scheme. In this paper, we focus on cycle-based monitoring schemes.

To measure the accuracy of the link failure localization, *localization degree* $D_L = \|\boldsymbol{E}\|/A$ is defined [5], where $\|\boldsymbol{E}\|$ is the total number of links in the network, and A is the size of the alarm code set. For a cycle-based monitoring scheme, we have $A \leq \|\boldsymbol{E}\|$ and thus $D_L \geq 1$. This is because each link failure can trigger only one alarm code, but failures in any two-edge cut will trigger the same alarm code. $D_L = 1$ means that we can accurately localize every link failure. In Figs. 2(b) and 4(b), A equals to 6 because the two links (2, 4) and (3, 4) share the same alarm code. Then, D_L is $7/6 = 1.167$ in both solutions, which is also the minimum localization degree achievable by a cycle-based monitoring scheme, or the *optimal localization degree*. In this paper, we focus on m-cycle design that not only yields the optimal localization degree, but also consumes the least amount of monitoring resources/cost. The monitoring cost consists of both the monitor cost (measured by the total number of monitors required) and the bandwidth cost (measured by the cover length). Note that the monitor cost counts not only the hardware cost, but also the network management cost associated with each monitor.



Link	c_2	c_1	c_0	Decimal
(0,1)	0	0	1	1
(0,2)	1	1	1	7
(0,3)	1	1	0	6
(1,2)	0	1	1	3
(1,3)	0	1	0	2
(2,4)	1	0	0	4
(3,4)	1	0	0	4

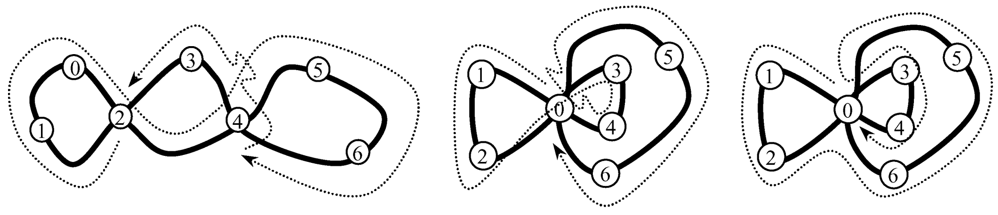
(b)

Fig. 4. m-cycle solution based on HST algorithm [5] $(\mathrm{D_L} = 1.167)$. (a) m-cycles. (b) Alarm code table.

Several algorithms [4]–[7] have been proposed for m-cycle design with similar goals. In particular, HST [5] adopts a spanning tree based approach. Fig. 4 shows a solution generated by HST. In Fig. 4(a), the spanning tree is denoted by the broad-brush links which are called *trunks*. Other links not in the spanning tree are called *chords*. HST generates an m-cycle from each chord where all other links on this m-cycle must be trunks. For example, the m-cycle generated from chord (1, 3) in Fig. 4(a) is 1-3-0-2-1. Another algorithm M^2-CYCLE [7] always generates m-cycles with minimum length, as shown by the example in Fig. 2. The length of c_1: 0-1-3-0 in Fig. 2(a) is 3, and it is smaller than 4 of c_1: 1-3-0-2-1 in Fig. 4(a). As a result, the cover length of the M^2-CYCLE solution in Fig. 2 is 10, in contrast to 11 of the HST solution in Fig. 4. It is proved [7] that M^2-CYCLE always outperforms HST.

The previously reported algorithms [4]–[7] are subject to a number of limitations. Firstly, they do not allow a tradeoff between the monitor cost and the bandwidth cost. In fact, they can only generate a single solution for a given network. Secondly, they only consider simple cycles, where a simple cycle can traverse any node at most once. As one of the contributions of this paper, we introduce the concept of *nonsimple m-cycle*. A nonsimple cycle [12]–[14] can traverse a node multiple times but a link at most once. Fig. 5 shows three nonsimple m-cycles. The dotted arrows indicate the possible connection patterns of the supervisory wavelengths. Define the links covered by an m-cycle as *on-cycle links*. If any on-cycle link of a nonsimple m-cycle fails, the associated monitor will alarm in the same way as in the simple m-cycle scenario. Note that the two nonsimple m-cycles in Fig. 5(b) and (c) have the same set of on-cycle links but different wavelength connection patterns. They will lead to the same monitoring result. Generally, nonsimple m-cycles can better exploit mesh connectivity of a network than simple m-cycles. This provides more flexibility in minimizing the monitoring cost.

Motivated by the above observations, in this paper we first formulate an optimal integer linear program (ILP) by considering both simple and nonsimple m-cycles. The cost ratio r of a monitor to a supervisory wavelength-link is introduced, and the overall monitoring cost is minimized. Since solving the optimal ILP generally needs a long running time, a heuristic ILP is then proposed, which provides an efficient tradeoff between the monitor cost and the bandwidth cost.



(a) (b) (c)

Fig. 5. Nonsimple m-cycles.

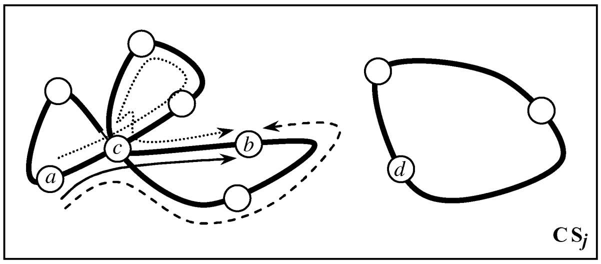


Fig. 6. Multiple disjoint cycles coexist in a cycle set $\mathbf{CS}_j$ (nodes and links not traversed by the cycles are omitted). Note that every node has an even number of on-cycle links incident on it.

The rest of the paper is organized as follows. Section II formulates the optimal ILP and the heuristic ILP. Section III presents numerical results by solving the ILPs. We conclude the paper in Section IV.

II. ILP Formulations

To formulate an ILP (Integer Linear Program) for m-cycle design, we first need to formulate cycles. In an ILP formulation, cycles can be defined by requiring each node in the network to have an even number (including zero) of on-cycle links incident on it. If this number is greater than 2 at some nodes, nonsimple cycles will be obtained. An issue with this definition is that multiple disjoint cycles (without any common node) may be generated, although we only intend to formulate a single cycle. Fig. 6 shows an example, where two disjoint cycles are generated based on the above definition. We call such a set of cycles as a *cycle set*. In our ILP, we need to predefine a value J which is the maximum number of cycle sets in the solution. A particular cycle set is denoted by $\mathbf{CS}_j$ with index $j \in \{0, 1, \ldots, J-1\}$. An on-cycle link of any cycle in $\mathbf{CS}_j$ is called an on-cycle link of $\mathbf{CS}_j$. If a node is traversed by any cycle in $\mathbf{CS}_j$, we say that this node is on $\mathbf{CS}_j$. However, we do not know the exact number of m-cycles required until a solution is obtained. To address this issue, we can set J sufficiently large whereas the ILP may return less cycle sets in the solution. Therefore, it is possible that some $\mathbf{CS}_j$ do not contain any on-cycle link, and we call them *empty cycle sets*.

We first consider networks without any two-edge cut, and thus each link failure must have a unique alarm code. By assuming that J is set large enough, we formulate the optimal ILP in Section II-A and the heuristic ILP in Section II-B. Section II-C considers networks with some two-edge cuts. Finally, Section II-D discusses how to determine the values for the pre-defined parameters (such as J) in our ILPs.

A. Optimal ILP

Without loss of generality, the monitoring cost of all the m-cycles in the network is defined as follows:

$$\begin{aligned}\text{Monitoring Cost} &= r \times \text{monitor cost} + \text{bandwidth cost}\\ &= r \times \text{number of monitors} + \text{cover length}\end{aligned} \tag{1}$$

The cost ratio r determines the relative importance between the monitor cost and the bandwidth cost. Its value is given as an input to the optimal ILP.

Since a cycle set may contain multiple disjoint cycles, the number of m-cycles and monitors in each $\mathbf{CS}_j$ remains unknown. This makes it impossible to accurately count the monitor cost in (1). To achieve optimal design of m-cycle solutions, this issue must be solved. Our approach is to ensure a single m-cycle in each nonempty $\mathbf{CS}_j$, such that the total number of required monitors can be obtained by just counting the number of nonempty cycle sets. Specifically, this is achieved by carrying out a *flow-based analysis* for all the node pairs in the network, as described below. Assume that the node pair $\{a, b\}$ in Fig. 6 is considered. Without loss of generality, we assume that a is the source and b is the sink. Source a can generate at most one flow but it does not receive any flow. Similarly, sink b can receive at most one flow but it does not generate or relay any flow. A flow must move along the on-cycle links of $\mathbf{CS}_j$. Except a and b, all other nodes in the network must obey flow conservation [15]. With the above constraints, if a flow exists between a and b (as shown by the solid arrow in Fig. 6), then nodes a and b must be on the same cycle. Otherwise, they are not on the same cycle. For example, if node pair $\{a, d\}$ in Fig. 6 is considered, no flow can exist between nodes a and d. Since both a and d are on $\mathbf{CS}_j$, this $\mathbf{CS}_j$ must contain multiple disjoint cycles and thus should be excluded from the solution space of the optimal ILP.

To reduce the number of variables and constraints in the ILP, we can use an undirected flow to replace the directed one in Fig. 6. If an on-cycle link incident on a node carries a flow, we say that there is a unit-flow incident on this node. Then, flow conservation can be formulated by requiring each node in the network (except source a and sink b) to have either 2 or 0 unit-flows incident on it, whereas source a or sink b can have at most 1 unit-flow incident on it. In our flow-based analysis, it is unnecessary to formulate the wavelength connection pattern of an m-cycle (refer to Fig. 5), and thus the ILP can be greatly simplified. For the nonsimple m-cycle in Fig. 6, even if the supervisory wavelength is preconfigured as indicated by the dashed or the dotted arrow, we can still confirm that nodes a and b are on the same cycle according to the (logical) flow indicated by the solid arrow.

Assume the network contains no two-edge cut. To assign a unique alarm code to each link failure, we define *decimal alarm code* which is the decimal translation of the corresponding binary alarm code, as shown in Figs. 2(b) and 4(b). With decimal alarm codes, it is computationally easier to compare two alarm codes, as it does not involve bit-wise comparison of binary alarm codes. In the optimal ILP, J denotes the (estimated) maximum number of m-cycles (or cycle sets) in the solution, and each m-cycle matches one bit in the binary alarm codes [see Figs. 2(b) and 4(b)]. As a result, each binary alarm code has at most J bits, and the corresponding decimal alarm code is chosen from the candidate set $\{1, 2, \ldots, 2^J - 1\}$. To formulate the inequality among the chosen decimal alarm codes, our approach is to let each of them take one unique integer from $\{1, 2, \ldots, 2^J - 1\}$. With the notations defined in Fig. 7, the optimal ILP for m-cycle design is formulated in (2)–(13) below. It ensures not only the optimal/minimum localization degree, but also the optimal/minimum monitoring cost for a given network with a predefined cost ratio r

$$\text{Objective}:$$

$$\text{minimize}\left\{ r\sum_j m^j + \sum_j \sum_{(u,v)\in \boldsymbol{E}} c_{uv} e_{uv}^j \right\}; \qquad (2)$$

$$\text{Subject to}:$$

$$f_{uv}\Big|_{ab}^{j} \le e_{uv}^j,\ \forall a,b \in \boldsymbol{V}: a \ne b,\ \forall j,\ \forall (u,v) \in \boldsymbol{E}; \qquad (3)$$

$$\sum_{(u,v)\in\boldsymbol{E}} f_{uv}\Big|_{ab}^{j} = 2t_u\Big|_{ab}^{j},\ \forall a,b \in \boldsymbol{V}: a \ne b,$$
$$\forall j,\ \forall u \in \boldsymbol{V}: u \ne a, u \ne b; \qquad (4)$$

$$\sum_{(a,v)\in\boldsymbol{E}} f_{av}\Big|_{ab}^{j} \le 1,\ \sum_{(b,v)\in\boldsymbol{E}} f_{bv}\Big|_{ab}^{j} \le 1,$$
$$\forall a,b \in \boldsymbol{V}: a \ne b,\ \forall j; \qquad (5)$$

$$1 + \sum_{(a,v)\in\boldsymbol{E}} f_{av}\Big|_{ab}^{j} \ge x_a^j + x_b^j,$$
$$\forall a,b \in \boldsymbol{V}: a \ne b,\ \forall j; \qquad (6)$$

$$x_u^j \ge e_{uv}^j,\ x_v^j \ge e_{uv}^j,\ \forall (u,v) \in \boldsymbol{E},\ \forall j; \qquad (7)$$

$$m^j \ge x_u^j,\ \forall u \in \boldsymbol{V},\ \forall j; \qquad (8)$$

$$\sum_{(u,v)\in\boldsymbol{E}} e_{uv}^j = 2z_u^j,\ \forall u \in \boldsymbol{V},\ \forall j; \qquad (9)$$

$$\alpha_{uv} = \sum_j 2^j \times e_{uv}^j,\ \forall (u,v) \in \boldsymbol{E}; \qquad (10)$$

$$\alpha_{uv} = \sum_{k=1}^{2^J-1} k \times y_{uv}^k,\ \forall (u,v) \in \boldsymbol{E}; \qquad (11)$$

$$\sum_{k=1}^{2^J-1} y_{uv}^k = 1,\ \forall (u,v) \in \boldsymbol{E}; \qquad (12)$$

$$\sum_{(u,v)\in\boldsymbol{E}} y_{uv}^k \le 1,\ \forall k \in \{1, 2, \ldots, 2^J - 1\}; \qquad (13)$$

Objective (2) minimizes the monitoring cost (monitor cost plus bandwidth cost) by taking the cost ratio r into account. To ensure at most a single cycle in each $\mathbf{CS}_{\boldsymbol{j}}$, constraints (3)–(6) carry out the flow-based analysis by checking each source-sink pair $\{a, b\}$ in $\mathbf{CS}_{\boldsymbol{j}}$. In particular, constraint (3) confines the flow to move along the on-cycle links of $\mathbf{CS}_{\boldsymbol{j}}$. Constraint (4) formulates the flow conservation property at each node (except a and b). Constraint (5) indicates that a or b can have at most one unit-flow incident on it. With constraint (6), if there is a unit-flow incident on node a (i.e., a flow exists in $\mathbf{CS}_{\boldsymbol{j}}$), then both a and b are on the only cycle in $\mathbf{CS}_{\boldsymbol{j}}$. Otherwise, a and b cannot be on $\mathbf{CS}_{\boldsymbol{j}}$ at the same time. Constraint (7) says that, if a link is an on-cycle link of $\mathbf{CS}_{\boldsymbol{j}}$, then its both end nodes must be on $\mathbf{CS}_{\boldsymbol{j}}$. Constraint (8) means that a monitor is required by $\mathbf{CS}_{\boldsymbol{j}}$ if it is not an empty cycle set. Constraint (9) defines the cycle set $\mathbf{CS}_{\boldsymbol{j}}$ by requiring each node in the network to have an even number of on-cycle links incident on it. Binary alarm codes are converted into decimal codes by (10). A unique alarm code for each link failure is ensured by (11)–(13). Specifically, constraint (11) formulates a decimal alarm code into a combinatorial sum over the candidate set $\{1, 2, \ldots, 2^J - 1\}$. Constraint (12) ensures that each link takes one (and only one) decimal alarm code from the candidate set. Constraint (13) says that each alarm code can be assigned to at most one link, which ensures optimal localization degree.

Theoretically, if the value of J is set large enough, the ILP in (2)–(13) ensures the optimal m-cycle solution with the minimum monitoring cost. In this case, the solution contains less than J m-cycles (others are empty cycle sets). On the other hand, if J is set too small, the ILP may not be able to find a feasible solution, or the optimality of the solution cannot be ensured. How to set the value of J is an engineering concern and it is further discussed in Section II-D.

B. Heuristic ILP

The above optimal ILP involves the flow-based analysis to ensure a single cycle in each $\mathbf{CS}_{\boldsymbol{j}}$. Solving it generally requires a long running time. This is because the flow-based analysis [as formulated in (3)–(6)] introduces $O(\|\boldsymbol{V}\|^2\|\boldsymbol{E}\|J)$ variables and constraints to the ILP. To speed up the optimization process, we propose a heuristic ILP by removing the flow-based analysis. Instead of accurately counting the total number of monitors, our heuristic ILP uses the sum of all the decimal alarm codes as a heuristic measure for monitor cost. Though minimizing the sum of all the decimal alarm codes does not ensure that the monitor cost is always minimized, it can effectively suppress the required number of bits in the binary alarm codes. Since each bit corresponds to a cycle/cycle set, this in turn minimizes the total number of monitors and the monitor cost. As a side benefit, this also helps to minimize the total number of 1s in all binary alarm codes, i.e., the cover length. On the other hand, the bandwidth cost can still be accurately represented by the cover length.

Instead of using (2), the objective in the heuristic ILP is

$$\text{minimize}\left\{ \sum_{(u,v)\in\boldsymbol{E}} \alpha_{uv} + \gamma \sum_j \sum_{(u,v)\in\boldsymbol{E}} c_{uv} e_{uv}^j \right\}. \qquad (14)$$

Note that γ in (14) is different from the cost ratio r in (2). As a consequence of using the sum of all the decimal alarm codes (i.e., $\sum_{(u,v)\in\boldsymbol{E}} \alpha_{uv}$ in (14)) for the (heuristic) monitor cost, γ has lost the accurate physical meaning in the monitor/bandwith cost tradeoff. Nevertheless, the value of γ still determines the relative importance between the (heuristic) monitor cost and the bandwidth cost (though not as accurately as by the cost ratio r in the optimal ILP). Section II-D further discusses how to determine a suitable value of γ. Since no flow-based analysis is required, constraints (3)–(8) in the optimal ILP are removed. To summarize, the heuristic ILP consists of objective (14) and constraints (9)–(13) only. Though the optimal (minimum) monitoring cost is not ensured, the heuristic ILP can still achieve the optimal localization degree according to constraints (10)–(13).

Note that the heuristic ILP does not ensure a single cycle in each cycle set. Accordingly, we need to slightly modify the format of the alarm code table. Specifically, each bit of the binary alarm codes corresponds to a cycle set $\mathbf{CS}_{\boldsymbol{j}}$, instead of a single cycle c_j as in Figs. 2(b) and 4(b).

C. ILPs for Networks With Two-Edge Cuts

If the network contains some two-edge cuts, the link failures in each two-edge cut cannot be distinguished by a cycle-based monitoring scheme. Let $\boldsymbol{S_{uv}}$ denote a set of links with at least two elements, where $\boldsymbol{S_{uv}}$ includes link (u,v) and all other links that can form a two-edge cut with (u,v). If no link can form a two-edge cut with (u,v), then $\boldsymbol{S_{uv}}$ is a null set $(\boldsymbol{\Phi})$. For example, in Fig. 2(a) we have $\boldsymbol{S_{24}} = \boldsymbol{S_{34}} = \{(2,4),(3,4)\}$, and $\boldsymbol{S_{uv}}$ for all other links (u,v) are null sets. All links in $\boldsymbol{S_{uv}}$ must have the same alarm code as (u,v). Further define $\boldsymbol{S}$ as a maximum set of *delegate links*, where each delegate link is an arbitrary link among a set of links with indistinguishable link failures. In other words, each link in $\boldsymbol{S}$ can form a two-edge cut with another link not in $\boldsymbol{S}$, but any two links in $\boldsymbol{S}$ cannot form a two-edge cut. In Fig. 2(a), $\boldsymbol{S}$ can be either $\{(2,4)\}$ or $\{(3,4)\}$, where the only link in $\boldsymbol{S}$ is regarded as the delegate of segment 2-4-3. For m-cycle design in a network with some two-edge cuts, our (optimal and heuristic) ILPs can be modified by replacing constraint (13) with (15)–(17) below.

$$\alpha_{uv} = \alpha_{mn},\ \forall (u,v), \quad (m,n) \in \boldsymbol{E} : \boldsymbol{S}_{uv} = \boldsymbol{S}_{mn} \neq \boldsymbol{\Phi}; \qquad (15)$$

$$\sum_{(u,v)\in \boldsymbol{S}} y_{uv}^{k} \leq 1,\ \forall k \in \{1,2,\ldots,2^{J}-1\}; \qquad (16)$$

$$\sum_{(u,v)\in \boldsymbol{E}} y_{uv}^{k} \leq 1 + \sum_{(u,v)\in \boldsymbol{S}} \left(\|\boldsymbol{S}_{uv}\| - 1\right) y_{uv}^{k}, \quad \forall k \in \{1,2,\ldots,2^{J}-1\}; \qquad (17)$$

Constraint (15) enforces the same alarm code for all the links in the same $\boldsymbol{S_{uv}}$. Constraint (16) ensures one alarm code (k) for each delegate link in $\boldsymbol{S}$. With constraint (17), if a decimal alarm code k is assigned to a link $(u,v) \in \boldsymbol{S}$, then at most $\|\boldsymbol{S_{uv}}\|$ links in the network can have the same alarm code k; otherwise k can be assigned to at most one link that cannot form a two-edge cut with any other link.

In fact, we can always use (15)–(17) to replace (13). If the network contains no two-edge cut, then $\boldsymbol{S}$ and $\boldsymbol{S_{uv}}$ will be null sets $(\boldsymbol{\Phi})$, and (15)–(17) will degenerate to (13).

D. Discussion

In our ILPs, the candidate alarm code set is $\{1,2,\ldots,2^{J}-1\}$. According to (11)–(13), the number of ILP variables and constraints soars exponentially as J increases. So, the ILP running time is sensitive to the value of J. By always setting J large enough and let the ILP returns less than J cycle sets in the solution, the optimal ILP always assures the optimality of the solution at the cost of a longer running time. In practice, we want to set J to be just-enough, such that the ILPs can return a proper solution faster. This is especially the case if the heuristic ILP is used. In the heuristic ILP, we actually focus on finding a good feasible solution, where the minimum monitoring cost cannot be ensured no matter how large the value of J is. Therefore, setting a large J is less critical in the heuristic ILP, but it still provides some extra flexibility in the monitor/bandwidth cost tradeoff. For example, if we want to have a smaller bandwidth cost, we may increase the number of m-cycles/monitors, which in turn requires a larger value of J (than just-enough). On the other hand, if J is too small, the ILPs may not be able to find a feasible solution. For the sake of engineering concerns, we need to properly set the value of J.

Let T be the total number of links that can form a two-edge cut with another link. We have

$$T = \left\| \bigcup_{(u,v)\in \boldsymbol{E}} \boldsymbol{S_{uv}} \right\|. \qquad (18)$$

To achieve the optimal localization degree (i.e., any distinguishable link failure has a unique alarm code), the size of the alarm code set must be

$$A = \|\boldsymbol{E}\| - T + \|\boldsymbol{S}\| \qquad (19)$$

and thus the optimal localization degree is

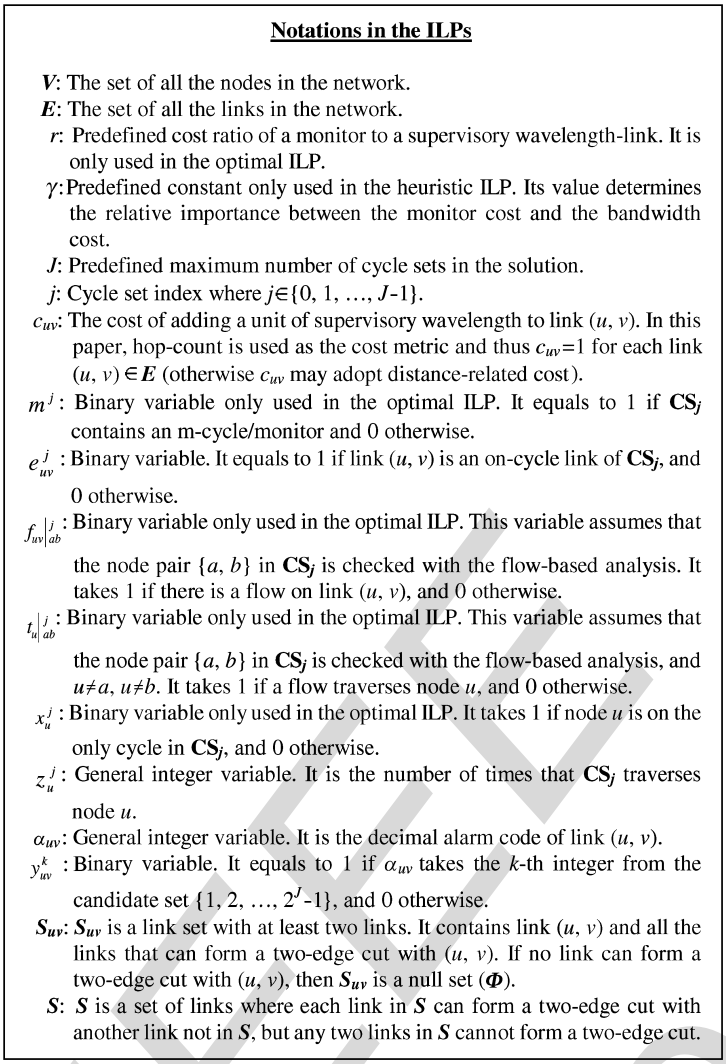
$$D_L = \frac{\|\boldsymbol{E}\|}{A} = \frac{\|\boldsymbol{E}\|}{\|\boldsymbol{E}\| - T + \|\boldsymbol{S}\|}. \qquad (20)$$

Let B be the lower bound of the required number of cycle sets (i.e., the number of bits in the binary alarm codes), we have

$$B = \lfloor \log_2 A \rfloor + 1 = \lfloor \log_2 \left(\|\boldsymbol{E}\| - T + \|\boldsymbol{S}\|\right) \rfloor + 1. \qquad (21)$$

Generally, the actual number of cycle sets in a solution tends to be close to the lower bound B in (21). This is because adding a single cycle set in the solution will double the size of the candidate alarm code set $\{1,2,\ldots,2^{J}-1\}$, and thus lead to much higher design flexibility. For example, if a network has $\|\boldsymbol{E}\| = 22$ links and no two-edge cut, then $B = \lfloor \log_2 \|\boldsymbol{E}\| - T + \|\boldsymbol{S}\| \rfloor + 1 = \lfloor \log_2 \|\boldsymbol{E}\| \rfloor + 1 = 5$. If we set $J = B + 4 = 9$, the size of the candidate alarm code set is $2^{J}-1 = 2^{9}-1 = 511$, which is much larger than $\|\boldsymbol{E}\| = 22$. As a result, the ILP has very high flexibility in choosing only 22 alarm codes from 511 candidates. Let δ_1 be a small positive integer. We can set the value of J according to

$$J = \lfloor \log_2 \left(\|\boldsymbol{E}\| - T + \|\boldsymbol{S}\|\right) \rfloor + \delta_1. \qquad (22)$$



Notations in the ILPs

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

$\boldsymbol{E}$: The set of all the links in the network.

r: Predefined cost ratio of a monitor to a supervisory wavelength-link. It is only used in the optimal ILP.

γ: Predefined constant only used in the heuristic ILP. Its value determines the relative importance between the monitor cost and the bandwidth cost.

J: Predefined maximum number of cycle sets in the solution.

j: Cycle set index where $j\in\{0, 1, \ldots, J\text{-}1\}$.

c_{uv}: The cost of adding a unit of supervisory wavelength to link (u, v). In this paper, hop-count is used as the cost metric and thus c_{uv}=1 for each link $(u, v)\in\boldsymbol{E}$ (otherwise c_{uv} may adopt distance-related cost).

m^j: Binary variable only used in the optimal ILP. It equals to 1 if $\mathbf{CS}_j$ contains an m-cycle/monitor and 0 otherwise.

e^j_{uv} : Binary variable. It equals to 1 if link (u, v) is an on-cycle link of $\mathbf{CS}_j$, and 0 otherwise.

$f_{uv}|^j_{ab}$: Binary variable only used in the optimal ILP. This variable assumes that the node pair $\{a, b\}$ in $\mathbf{CS}_j$ is checked with the flow-based analysis. It takes 1 if there is a flow on link (u, v), and 0 otherwise.

$t_u|^j_{ab}$: Binary variable only used in the optimal ILP. This variable assumes that the node pair $\{a, b\}$ in $\mathbf{CS}_j$ is checked with the flow-based analysis, and $u\neq a$, $u\neq b$. It takes 1 if a flow traverses node u, and 0 otherwise.

x^j_u : Binary variable only used in the optimal ILP. It takes 1 if node u is on the only cycle in $\mathbf{CS}_j$, and 0 otherwise.

z^j_u: General integer variable. It is the number of times that $\mathbf{CS}_j$ traverses node u.

α_{uv}: General integer variable. It is the decimal alarm code of link (u, v).

y^k_{uv} : Binary variable. It equals to 1 if α_{uv} takes the k-th integer from the candidate set $\{1, 2, \ldots, 2^J\text{-}1\}$, and 0 otherwise.

$\boldsymbol{S}_{uv}$: $\boldsymbol{S}_{uv}$ is a link set with at least two links. It contains link (u, v) and all the links that can form a two-edge cut with (u, v). If no link can form a two-edge cut with (u, v), then $\boldsymbol{S}_{uv}$ is a null set ($\boldsymbol{\Phi}$).

$\boldsymbol{S}$: $\boldsymbol{S}$ is a set of links where each link in $\boldsymbol{S}$ can form a two-edge cut with another link not in $\boldsymbol{S}$, but any two links in $\boldsymbol{S}$ cannot form a two-edge cut.

Fig. 7. Notations in the ILPs.

From (22), the practical value of J is generally not too large. Consequently, the number of variables and constraints in our ILPs can be limited at an acceptable level.

In the objective (14) of the heuristic ILP, the monitor cost $\sum_{(u,v)\in\boldsymbol{E}}\alpha_{uv}$ and the bandwidth cost $\sum_j\sum_{(u,v)\in\boldsymbol{E}}c_{uv}e^j_{uv}$ are not on the same order of magnitudes. For example, adding a "1" at the $(J-1)$th bit in a binary alarm code will increase the bandwidth cost by 1, but the monitor cost by 2^{J-1}. To provide necessary balance between the two cost components in (14), we can set the value of γ based on the guideline formulated in (23), where δ_2 is an integer with a small absolute value $|\delta_2|$

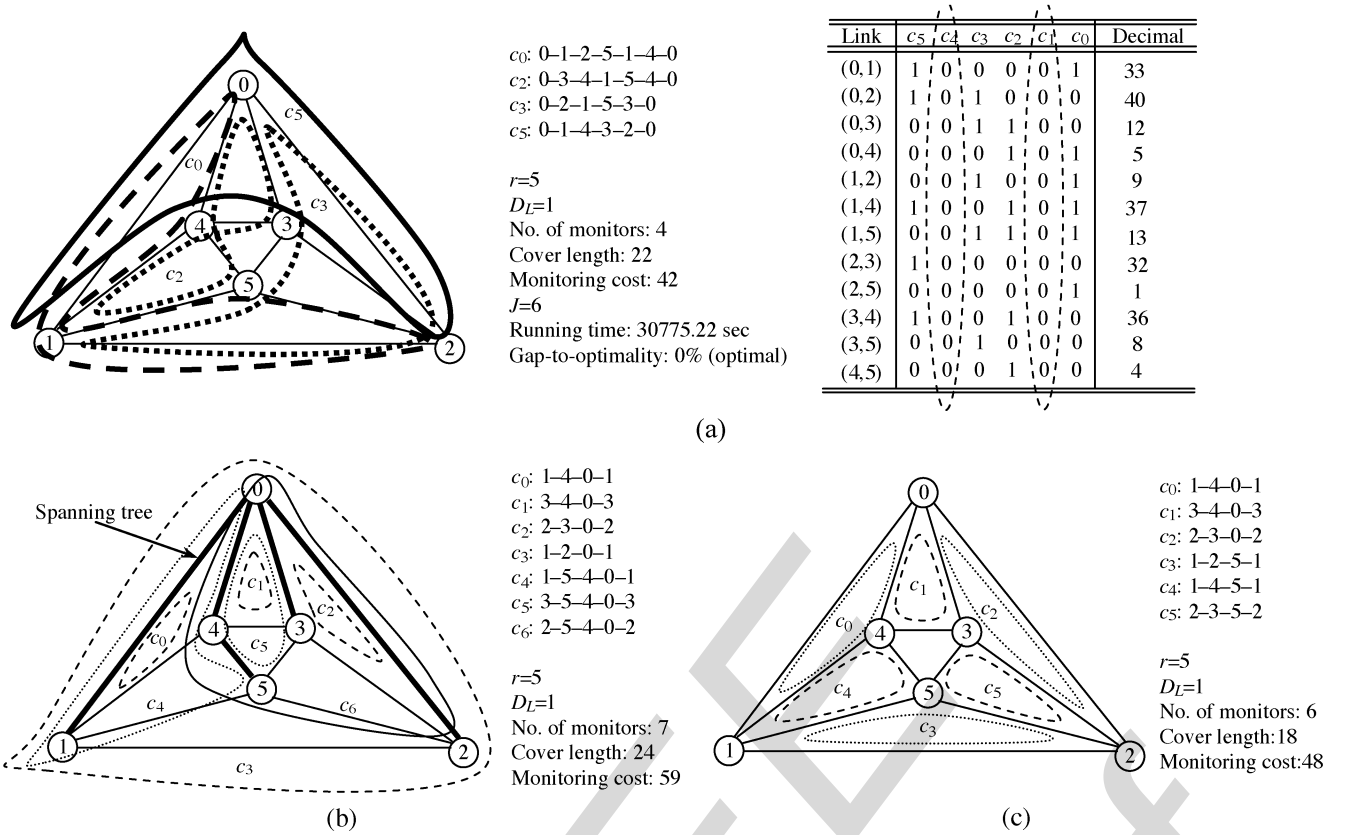
$$\gamma = 2^{\lfloor\log_2(\|\boldsymbol{E}\|-T+\|\boldsymbol{S}\|)\rfloor+\delta_2}. \tag{23}$$

Generally, varying the value of γ in the heuristic ILP provides an efficient tradeoff between the monitor cost and the bandwidth cost. If we emphasize on minimizing the number of monitors, we can set γ to a small value. Otherwise, a larger γ can be used in (14) to increase the weight of the cover length. Note that formula (23) only gives an engineering guideline in choosing a suitable value of γ. Theoretically, a feasible solution can always be generated no matter what is the value of γ (if J is large enough), because γ is used to provide a means for manipulating the monitor/bandwidth cost tradeoff. In practice, we hope that we can find a specific value of γ which can lead to a good-enough solution. Based on the guideline formulated in (23), this can be achieved according to some empirical engineering experiences. Besides, the high computation efficiency on solving the heuristic enables the possibility to run the heuristic multiple times with different values of γ, such that we can "calibrate" γ based on the knowledge of the cost ratio r and eventually select a proper value of γ for the targeted operational objectives.

III. Numerical Results

We use ILOG CPLEX 10.0 to implement the ILPs on a computer with a 3 GHz Intel Xeon CPU 5160. The CPLEX environment parameters are set according to (24) below

[1] [Online]. Available: http://www.ilog.com



c_0: 0–1–2–5–1–4–0
c_2: 0–3–4–1–5–4–0
c_3: 0–2–1–5–3–0
c_5: 0–1–4–3–2–0

r=5
D_L=1
No. of monitors: 4
Cover length: 22
Monitoring cost: 42
J=6
Running time: 30775.22 sec
Gap-to-optimality: 0% (optimal)

Link	c_5	c_4	c_3	c_2	c_1	c_0	Decimal
(0,1)	1	0	0	0	0	1	33
(0,2)	1	0	1	0	0	0	40
(0,3)	0	0	1	1	0	0	12
(0,4)	0	0	0	1	0	1	5
(1,2)	0	0	1	0	0	1	9
(1,4)	1	0	0	1	0	1	37
(1,5)	0	0	1	1	0	1	13
(2,3)	1	0	0	0	0	0	32
(2,5)	0	0	0	0	0	1	1
(3,4)	1	0	0	1	0	0	36
(3,5)	0	0	1	0	0	0	8
(4,5)	0	0	0	1	0	0	4

(a)

c_0: 1–4–0–1
c_1: 3–4–0–3
c_2: 2–3–0–2
c_3: 1–2–0–1
c_4: 1–5–4–0–1
c_5: 3–5–4–0–3
c_6: 2–5–4–0–2

r=5
D_L=1
No. of monitors: 7
Cover length: 24
Monitoring cost: 59

(b)

c_0: 1–4–0–1
c_1: 3–4–0–3
c_2: 2–3–0–2
c_3: 1–2–5–1
c_4: 1–4–5–1
c_5: 2–3–5–2

r=5
D_L=1
No. of monitors: 6
Cover length:18
Monitoring cost:48

(c)

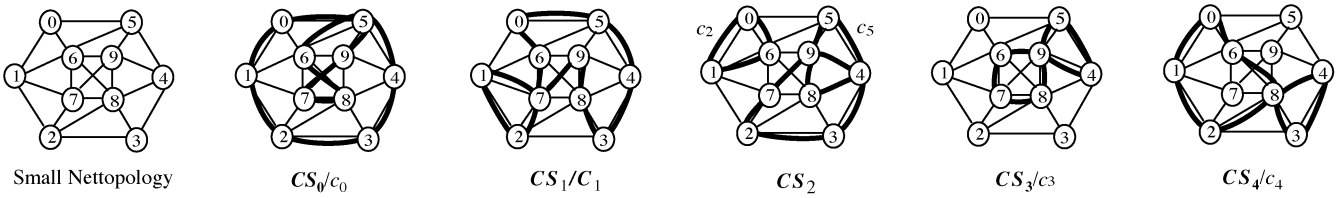
Fig. 8. m-cycle solutions for a small-size network with $\|\mathrm{V}\| = 6$, $\|\mathrm{E}\| = 12$ and $\mathrm{r} = 5$ ($\mathrm{D_L} = 1$). (a) Solution returned by the optimal ILP. (b) Solution returned by HST [5]. (c) Solution returned by M^2-CYCLE [7].

$$\begin{cases} 1\rightarrow\text{emphasis mip} \\ 2\rightarrow\text{mip strategy probe} \\ 3\rightarrow\text{mip strategy rins} \\ 3\rightarrow\text{mip strategy heuristicfreq} \\ 2\rightarrow\text{mip cuts all} \\ 3\rightarrow\text{mip strategy dive} \\ 3\rightarrow\text{preprocessing symmetry} \end{cases} \tag{24}$$

The optimal ILP is first used for m-cycle design in a small-size network as shown in Fig. 8(a), with $r = 5$ and $J = 6$. Fig. 8(a) shows an optimal solution with 4 m-cycles and a monitoring cost of 42. In the alarm code table, c_1 and c_4 do not cover any link and thus are not m-cycles (i.e., empty cycle sets). As we have discussed earlier, the optimality of the solution (with the optimal minimum monitoring cost) is ensured because the solution contains less than $J = 6$ m-cycles. We can see that nonsimple m-cycles are enabled in our ILP-based approach. Instead, HST [5] and M^2-CYCLE [7] algorithms can only generate simple m-cycles, as shown by the solutions in Fig. 8(b) and (c). Note that HST and M^2-CYCLE always enforce the same solution for different values of r, and no tradeoff is allowed between the monitor cost and the bandwidth cost. From Fig. 8, we can see that the solution in Fig. 8(a) cuts down the monitoring cost by 28.81% over the HST solution in Fig. 8(b), and 12.5% over the M^2-CYCLE solution in Fig. 8(c).

The optimal solution in Fig. 8(a) is found in 60.52 seconds, but CPLEX takes 30775.22 seconds to prove the optimality. If the heuristic ILP is used with $\gamma = 1$ and $J = 6$, another optimal solution with the same monitoring cost of 42 can be found in only 0.64 second. As the network size increases, generally the optimal ILP needs a very long running time. In what follows, we focus on the heuristic ILP, which can achieve the optimal localization degree but without the optimal monitoring cost guarantee.

Fig. 9 shows a solution for SmallNet. It is returned by the heuristic ILP with $J = 8$ and $\gamma = 0$. With $\gamma = 0$, we aim at minimizing the monitor cost only. The cycle sets obtained are shown in Fig. 9(a), with the "alarm code table" in Fig. 9(b). Unlike Fig. 8(a), the "alarm code table" in Fig. 9(b) is based on cycle set $\boldsymbol{CS_j}$ instead of m-cycle c_j. Though it may not have a proper physical meaning (due to multiple m-cycles in $\boldsymbol{CS_j}$), it is the best way to record the original data directly returned by CPLEX. In fact, the "alarm code table" in Fig. 9(b) can be easily translated into a true alarm code table based on m-cycle c_j. For example, Fig. 9(a) shows 5 cycle sets $\boldsymbol{CS}_0 - \boldsymbol{CS}_4$ with 6 m-cycles $c_0 - c_5$, where $\boldsymbol{CS}_2$ consists of a simple m-cycle c_2 and a nonsimple m-cycle c_5. Except $\boldsymbol{CS}_2$, each cycle set $\boldsymbol{CS_j}$ ($0 \leq j \leq 4$) matches an m-cycle c_j. For link (2, 3), its $\boldsymbol{CS_j}$ based "alarm code" $[\boldsymbol{CS}_7, \boldsymbol{CS}_6, \boldsymbol{CS}_5, \boldsymbol{CS}_4, \boldsymbol{CS}_3, \boldsymbol{CS}_2, \boldsymbol{CS}_1, \boldsymbol{CS}_0]$= $[0, 0, 0, 0, 0, 1, 0, 1] = 5$ can be translated into a true alarm code $[c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0] = [0, 0, 1, 0, 0, 0, 0, 1] = 33$. This is achieved by translating $\boldsymbol{CS}_2$ in the former to c_5 in the latter, because (2, 3) is covered by c_5 in $\boldsymbol{CS}_2$. On the other



(a)

Link	CS_7	CS_6	CS_5	CS_4	CS_3	CS_2	CS_1	CS_0	Decimal	Link	CS_7	CS_6	CS_5	CS_4	CS_3	CS_2	CS_1	CS_0	Decimal
(0,1)	0	0	0	1	0	1	0	1	21	(4,5)	0	0	0	0	1	1	1	1	15
(0,5)	0	0	0	0	0	0	1	1	3	(4,8)	0	0	0	1	0	1	0	0	20
(0,6)	0	0	0	1	0	1	1	0	22	(4,9)	0	0	0	0	1	1	0	0	12
(1,2)	0	0	0	1	0	0	1	1	19	(5,6)	0	0	0	0	0	0	0	1	1
(1,6)	0	0	0	0	0	1	0	0	4	(5,9)	0	0	0	0	1	1	0	1	13
(1,7)	0	0	0	0	0	0	1	0	2	(6,7)	0	0	0	0	1	0	1	0	10
(2,3)	0	0	0	0	0	1	0	1	5	(6,8)	0	0	0	1	0	0	0	1	17
(2,7)	0	0	0	0	0	1	1	0	6	(6,9)	0	0	0	0	1	0	0	0	8
(2,8)	0	0	0	1	0	0	0	0	16	(7,8)	0	0	0	0	1	0	0	1	9
(3,4)	0	0	0	1	0	1	1	1	23	(7,9)	0	0	0	0	0	1	1	1	7
(3,8)	0	0	0	1	0	0	1	0	18	(8,9)	0	0	0	0	1	1	1	0	14

(b)

Heuristic ILP:

J: 8
γ: 0
Running time: 152.14 sec
Gap-to-“optimality”:4.53%
No. of monitors: 6
Cover length: 49

HST [5]:

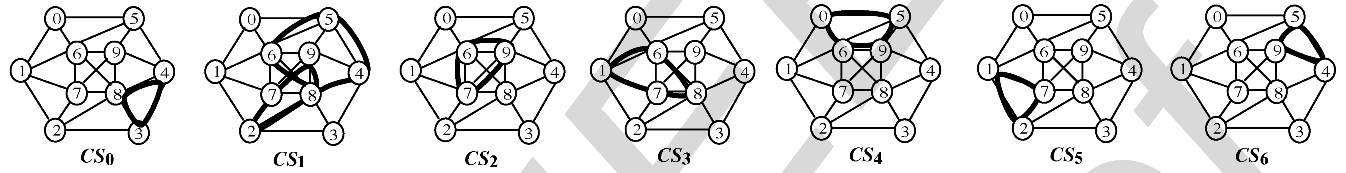
No. of monitors: 13
Cover length: 43

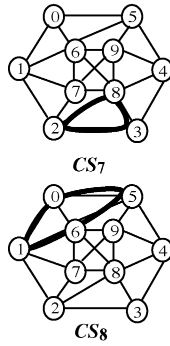
M^2-CYCLE [7]:

No. of monitors:12
Cover length: 36

(c)

Fig. 9. m-cycle design based on the heuristic ILP for SmallNet with $\mathrm{J} = 8$ and $\gamma = 0$ $(\mathrm{D_L} = 1)$. (a) SmallNet topology (with 10 nodes and 22 links) and m-cycles/cycle sets. (b) Alarm code table based on $\mathrm{CS_j}$. (c) Comparison with existing algorithms.





Heuristic ILP model:

J:9
γ: 1024
Running time:1795.15 sec
Gap-to-“optimality”: 2.88%
No. of monitors:9
Cover length: 35

M^2-CYCLE [7]:

No. of monitors: 12
Cover length: 36

Link	CS_8	CS_7	CS_6	CS_5	CS_4	CS_3	CS_2	CS_1	CS_0	Decimal	Link	CS_8	CS_7	CS_6	CS_5	CS_4	CS_3	CS_2	CS_1	CS_0	Decimal
(0,1)	1	0	0	0	0	0	0	0	0	256	(4,5)	0	0	1	0	0	0	0	1	0	66
(0,5)	1	0	0	0	1	0	0	0	0	272	(4,8)	0	0	0	0	0	0	0	1	1	3
(0,6)	0	0	0	0	1	0	0	0	0	16	(4,9)	0	0	1	0	0	0	0	0	0	64
(1,2)	0	0	0	1	0	0	0	0	0	32	(5,6)	1	0	0	0	0	0	0	1	0	258
(1,6)	1	0	0	0	0	1	0	0	0	264	(5,9)	0	0	1	0	1	0	0	0	0	80
(1,7)	0	0	0	1	0	1	0	0	0	40	(6,7)	0	0	0	0	0	0	1	0	0	4
(2,3)	0	1	0	0	0	0	0	0	0	128	(6,8)	0	0	0	0	0	1	0	1	0	10
(2,7)	0	0	0	1	0	0	0	1	0	34	(6,9)	0	0	0	0	1	0	1	0	0	20
(2,8)	0	1	0	0	0	0	0	10		130	(7,8)	0	0	0	0	0	1	0	0	0	8
(3,4)	0	0	0	0	0	0	0	0	1	1	(7,9)	0	0	0	0	0	0	1	1	0	6
(3,8)	0	1	0	0	0	0	0	0	1	129	(8,9)	0	0	0	0	0	0	0	1	0	2

Fig. 10. m-cycle design based on the heuristic ILP for SmallNet with $\mathrm{J} = 9$ and $\gamma = 1024$ $(\mathrm{D_L} = 1)$.

hand, since link (0, 1) is covered by c_2 in $\boldsymbol{CS}_2$, its alarm code $[0, 0, 0, 1, 0, 1, 0, 1] = 21$ is the same in both scenarios. Such a translation will change neither the m-cycles nor the localization degree. It only rearranges the value of the alarm codes.

Note that most m-cycles in Fig. 9(a) are nonsimple m-cycles. For comparison, Fig. 9(c) shows the results obtained from HST [5] and M^2-CYCLE [7] algorithms with $D_L = 1$. It is proved in [7] that M^2-CYCLE always outperforms HST on both the monitor cost and the bandwidth cost. So we only compare our ILP-based approach with M^2-CYCLE. From Fig. 9(c), our heuristic ILP saves 50% monitors but requires a larger cover length when $\gamma = 0$.

To provide tradeoff between the monitor cost and the bandwidth cost, we vary γ from 0 to 256, 512 and 1024 while keeping $J = 8$. The heuristic ILP returns solutions consisting of 10, 8 and 9 m-cycles, but all with the same cover length of 36 (Note that M^2-CYCLE requires 12 m-cycles and the same cover length of 36). Compared with the heuristic ILP result $(\gamma = 0)$ in Fig. 9, as expected the bandwidth cost is reduced and the monitor cost is increased. But increasing γ from 256 to 1024 does not decrease the cover length 36. This is because $J = 8$ is not large enough and it limits the flexibility in the tradeoff. With $J = 9$ and $\gamma = 1024$, another solution with 9 m-cycles and a cover length of 35 is obtained in 1795.15 seconds, as shown in Fig. 10. Compared to M^2-CYCLE, this gives a 25% cut on the monitor cost with one less supervisory wavelength-link. Fig. 10 also indicates that a larger γ value favors more simple m-cycles in the solution, as more emphasis is put on reducing the bandwidth cost.

The above example shows that, varying the value of γ can provide an efficient tradeoff between the monitor cost and the bandwidth cost, though this is not as accurate as using the optimal ILP (in term of reflecting the physical meaning based on the knowledge of the cost ratio r). Note that such a tradeoff is due to varying the value of γ instead of J, though we have increased the value of J by 1 in Fig. 10. As we have discussed in Section II-D, setting a proper value of J is an engineering concern for saving the ILP running time. If J is set large enough (at a cost of a longer running time), the solution obtained from both the optimal and the heuristic ILPs will not be affected by the specific value of J. For example, if we increase the value of J from 6 to 10 in Fig. 8(a), the same optimal solution will be obtained. With a large enough value of J, r in the optimal ILP or γ in the heuristic ILP becomes the only factor for determining the tradeoff. On the other hand, a small value of J may limit the flexibility in the tradeoff, as shown by the examples in Figs. 9 and 10. With $J = 8$ in Fig. 9, efficient tradeoffs can be achieved by varying γ from 0 to some large values, because $J = 8$ (cycle sets) is sufficiently large to accommodate the m-cycles generated. As we try to further decrease the cover length, the number of m-cycles has to be increased and $J = 8$ is no longer large enough. As a result, we need to increase J to 9 to accommodate the 9 m-cycles required (as shown in Fig. 10). Because the heuristic ILP does not ensure the optimal/minimum monitoring cost, a just-enough value of J will allow the tradeoff but still keep a short ILP running time. Of course, if running time is not an issue, we should always set J large enough to completely remove its adverse impact on the flexibility of the tradeoff.

Note that the networks in Figs. 8–10 do not contain any two-edge cut. As a result, the localization degree $D_L = 1$ can be achieved and each link failure has a unique alarm code. We now consider three typical networks ARPA2, NSFNET and BellCore [4], [5], [7] in Fig. 11(a), each of which contains some segments (any two links on the same segment form a two-edge cut of the network topology). Though our ILPs can be directly applied to the original topologies in Fig. 11(a), the ILP problem size can be reduced by simplifying the topologies first. This is achieved by treating each segment in Fig. 11(a) as a "link" with a cost equal to the sum of all link costs on it. For example, segment 5-6-7 in ARPA2 is treated as a "link" (5, 7) with a cost of $c_{57} = 2$. However, both segments 5-13-11 and 4-14-5 in BellCore are not involved in this treatment, and segment 0-1-2-5 in ARPA2 is treated as a "link" (0, 2) with $c_{02} = 2$ plus a link (2, 5) with $c_{25} = 1$. This is to avoid multiple "links" between two neighboring nodes. For example, if segment 4-14-5 in BellCore is treated as a "link", two "links" will exist between nodes 4 and 5. For simplicity, we avoid this case although our ILPs can be extended to handle it. Based on the simplified topologies, the solutions obtained with $\gamma = 1024$ are shown in Fig. 11(b)–(d) and are compared to M^2-CYCLE solutions. For NSFNET, CPLEX runs for a long time without further improving the solution in Fig. 11(c), which has a gap-to-"optimality" of 11.08%. In our heuristic ILP, this gap is not the true gap-to-optimality as in the optimal ILP. For NSFNET, Fig. 11(c) indicates that our heuristic ILP generates a solution with one less m-cycles and a 7.69% cut on the cover length. For BellCore, our heuristic ILP achieves the same cover length as M^2-CYCLE, but saves 21.43% monitor cost [see Fig. 11(d)]. However, the same solution as in M^2-CYCLE is generated for ARPA2, as shown in Fig. 11(b). Though we separately count the monitor cost and the cover length in Figs. 9–11, the monitoring cost can be calculated by (1) if the cost ratio r is given.

IV. Conclusion

The m-cycle provides an efficient way to achieve fast link failure localization in AONs. Compared to a channel-based or link-based scheme, an m-cycle based monitoring scheme can greatly reduce the required number of monitors. Previously reported m-cycle design algorithms cannot achieve an optimal design, and they do not allow a tradeoff between the monitor cost and the bandwidth cost. In this paper, we introduced nonsimple m-cycles and proposed an ILP-based approach to minimize the monitoring cost (i.e., monitor cost plus bandwidth cost). An optimal ILP and a heuristic ILP were formulated. The former provides optimal m-cycle design, but generally needs a long running time. The latter provides an efficient tradeoff between the monitor cost and the bandwidth cost with a shorter running time. Compared to the previously reported m-cycle design algorithms, our ILP-based approach can significantly reduce the monitoring cost while ensuring the optimal localization degree.

References

[1] Y. G. Wen, V. W. S. Chan, and L. Z. Zheng, "Efficient fault diagnosis algorithms for all-optical WDM networks with probabilistic link failures (invited paper)," *J. Lightw. Technol.*, vol. 23, no. 10, pp. 3358–3371, Oct. 2005.

[2] M. Goyal, K. K. Ramakrishnan, and W.-C. Feng, "Achieving faster failure detection in OSPF networks," in *Proc. IEEE ICC '03*, May 2003, vol. 1, pp. 296–300.

[3] C. Assi, Y. Ye, A. Shami, S. Dixit, and M. Ali, "A hybrid distributed fault-management protocol for combating single-fiber failures in mesh-based DWDM optical networks," in *Proc. IEEE GLOBECOM '02*, Nov. 2002, vol. 3, pp. 2676–2680.

[4] H. Zeng, C. Huang, A. Vukovic, and M. Savoie, "Fault detection and path performance monitoring in meshed all-optical networks," in *Proc. IEEE GLOBECOM '04*, Dec. 2004, vol. 3, pp. 2014–2018.

[5] H. Zeng, C. Huang, and A. Vukovic, "A novel fault detection and localization scheme for mesh all-optical networks based on monitoring-cycles," *Photon. Netw. Commun.*, vol. 11, no. 3, pp. 277–286, May 2006.

[6] H. Zeng and A. Vukovic, "The variant cycle-cover problem in fault detection and localization for mesh all-optical networks," *Photon. Netw. Commun.*, vol. 14, no. 2, pp. 111–122, 2007.

[7] B. Wu and K. L. Yeung, " M^2-CYCLE: An optical layer algorithm for fast link failure detection in all-optical mesh networks," in *Proc. IEEE GLOBECOM '06*, Dec. 2006.

[8] Y. Hamazumi, M. Koga, K. Kawai, H. Ichino, and K. Sato, "Optical path fault management in layered networks," in *Proc. IEEE GLOBECOM '98*, Nov. 1998, vol. 4, pp. 2309–2314.

[9] C.-S. Li and R. Ramaswami, "Automatic fault detection, isolation, and recovery in transparent all-optical networks," *J. Lightw. Technol.*, vol. 15, no. 10, pp. 1784–1793, Oct. 1997.

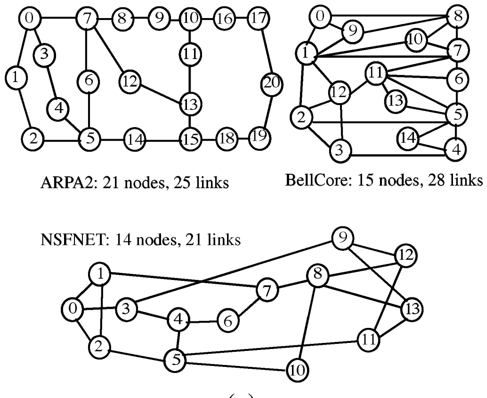
[10] S. Stanic, S. Subramaniam, H. Choi, G. Sahin, and H.-A. Choi, "On monitoring transparent optical networks," in *Proc. Int. Conf. Parallel Process. Workshops*, Aug. 2002, pp. 217–223.

[11] R. Diestel, *Graph Theory*, 2nd ed. New York: Spring-Verlag, 2000.

[12] C. G. Gruber, "Resilient networks with nonsimple p-cycles," in *Proc. IEEE ICT '03*, Feb. 2003, vol. 2, pp. 1027–1032.

[13] D. A. Schupke, C. G. Gruber, and A. Autenrieth, "Optimal configuration of p-cycles in WDM network," in *Proc. IEEE ICC '02*, May 2002, vol. 5, pp. 2761–2765.

[14] ***[Page numbers?]*** B. Wu, K. L. Yeung, and S. Z. Xu, "ILP formulation for p-cycle construction based on flow conservation," in *Proc. IEEE GLOBECOM '07*, Dec. 2007.


ARPA2: 21 nodes, 25 links
BellCore: 15 nodes, 28 links
NSFNET: 14 nodes, 21 links

(a)

Links	CS_5	CS_4	CS_3	CS_2	CS_1	CS_0	Decimal
(0,1), (1,2), (2,5)	0	1	0	0	0	0	16
(0,3), (3,4), (4,5)	0	1	1	0	0	0	24
(0,7)	0	0	1	0	0	0	8
(5,6), (6,7)	0	0	1	0	0	1	9
(5,14), (14,15)	0	0	0	0	0	1	1
(7,8), (8,9), (9,10)	0	0	0	1	0	0	4
(7,12), (12,13)	0	0	0	1	0	1	5
(10,11), (11,13)	0	0	0	1	1	0	6
(10,16), (16,17) (17,20), (19,20) (18,19), (15,18)	0	0	0	0	1	0	2
(13,15)	0	0	0	0	1	1	3

J: 6
γ: 1024
Time: 55.03 sec
Gap: 0%

No. of monitors: 5
Cover length: 35
No. of monitors in M^2-CYCLE: 5
Cover length in M^2-CYCLE: 35

(b)

Link	CS_7	CS_6	CS_5	CS_4	CS_3	CS_2	CS_1	CS_0	Decimal
(0,1)	0	0	1	0	0	1	0	0	36
(0,2)	0	0	1	0	0	0	0	0	32
(0,3)	0	0	0	0	0	1	0	0	4
(1,2)	0	0	1	0	0	0	0	1	33
(1,7)	0	0	0	0	0	1	0	1	5
(2,5)	0	0	0	0	0	0	0	1	1
(3,4)	0	0	0	0	1	1	0	0	12
(3,9)	0	0	0	0	1	0	0	0	8
(4,5)	0	0	0	1	1	0	0	0	24
(4,6)	0	0	0	1	0	1	0	0	20
(5,10)	0	0	0	1	0	0	0	0	16
(5,11)	0	0	0	0	1	0	0	1	9
(6,7)	0	0	0	1	0	1	0	0	20
(7,8)	0	0	0	1	0	0	0	1	17
(8,10)	0	0	0	1	0	0	0	0	16
(8,12)	0	0	0	0	0	0	1	0	2
(8,13)	0	0	0	0	0	0	1	1	3
(9,12)	0	1	0	0	1	0	1	0	74
(9,13)	0	1	0	0	0	0	1	0	66
(11,12)	0	1	0	0	1	0	0	0	72
(11,13)	0	1	0	0	0	0	0	1	65

J: 8
γ: 1024
Time: 1498.45 sec
Gap: 11.08%
No. of monitors: 7
Cover length: 36
No. of monitors in M^2-CYCLE: 8
Cover length in M^2-CYCLE: 39

(c)

Link	CS_8	CS_7	CS_6	CS_5	CS_4	CS_3	CS_2	CS_1	CS_0	Decimal
(0,1)	0	0	0	1	1	0	0	0	0	48
(0,8)	0	0	0	1	0	1	0	0	0	40
(0,9)	0	0	0	0	1	1	0	0	0	24
(1,2)	0	1	0	0	0	0	0	0	0	128
(1,7)	1	0	0	1	0	0	0	0	0	288
(1,9)	1	0	0	0	1	0	0	0	0	272
(1,10)	0	0	0	0	0	0	1	0	0	4
(1,12)	0	1	0	0	0	0	1	0	0	132
(2,3)	0	0	0	0	0	1	0	1	0	10
(2,5)	0	0	0	0	0	1	0	1	1	11
(2,12)	0	1	0	0	0	0	0	0	1	129
(3,4)	0	0	0	0	0	0	0	1	1	3
(3,12)	0	0	0	0	0	1	0	0	1	9
(4,5)	0	0	0	0	0	0	0	1	0	2
(4,14)	0	0	0	0	0	0	0	0	1	1
(5,6)	0	0	1	0	0	1	0	0	0	72
(5,11)	0	0	1	0	1	0	0	0	0	80
(5,13)	0	0	0	0	1	0	0	0	0	16
(5,14)	0	0	0	0	0	0	0	0	1	1
(6,7)	0	0	1	0	0	0	0	0	0	64
(6,11)	0	0	0	0	0	1	0	0	0	8
(7,8)	0	0	0	1	0	0	0	0	0	32
(7,10)	1	0	0	0	0	0	1	0	0	260
(7,11)	0	0	1	0	0	0	1	0	0	68
(8,9)	1	0	0	0	0	1	0	0	0	264
(8,10)	1	0	0	0	0	0	0	0	0	256
(11,12)	0	0	0	0	0	1	1	0	0	12
(11,13)	0	0	0	0	1	0	0	0	0	16

J: 9
γ: 1024
Time: 7948.64 sec
Gap: 4.69%

No. of monitors: 11
Cover length: 46
No. of monitors in M^2-CYCLE: 14
Cover length in M^2-CYCLE: 46

(d)

Fig. 11. m-cycle design based on the heuristic ILP for ARPA2, NSFNET and BellCore. (a) Three typical topologies taken from [4], [5], [7]. (b) Solution for ARPA2 $(\mathrm{D_L} = 2.500)$. (c) Solution for NSFNET $(\mathrm{D_L} = 1.105)$. (d) Solution for BellCore $(\mathrm{D_L} = 1.077)$.

[15] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ: Prentice-Hall, 1993.

Bin Wu (M'XX) received the B.Eng. degree from Zhe Jiang University, Hangzhou, China, in 1993, the M.Eng. degree from University of Electronic Science and Technology of China, Chengdu, China, in 1996, and the Ph.D. degree from the University of Hong Kong, Hong Kong, in 2007.

During 1997–2001, he served as the Department Manager of TI-Huawei DSP Co-Lab in Huawei Tech. Co. Ltd, Shenzhen, China. He is currently a Postdoctoral Research Fellow at the University of Waterloo, Waterloo, ON, Canada.

Kwan L. Yeung (SM'XX)was born in 1969. He received the B.Eng. and Ph.D. degrees in information engineering from The Chinese University of Hong Kong in 1992 and 1995, respectively.

He joined the Department of Electrical and Electronic Engineering, The University of Hong Kong in July 2000, where he is currently an Associate Professor, and the Information Engineering Program Co-Director. Before that, he has spent five years in the Department of Electronic Engineering, City University of Hong Kong as an Assistant Professor. During the summer of 1993, Dr. Yeung served with the Performance Analysis Department, AT&T Bell Laboratories (now Bell Labs, Lucent Technologies), Holmdel, NJ, as a Member of Technical Staff. His research interests include next-generation Internet, packet switch/router design, all-optical networks and wireless data networks. He has obtained two patents and published over 120 papers in international journals and conferences since 1993.

Pin-Han Ho (M'XX)received the B.Sc., M.Sc., and Ph.D. degrees from the Electrical and Computer Engineering Department, National Taiwan University, Taiwan, in 1993, 1995 and 2002, respectively.

He joined the E&CE Department, University of Waterloo, Waterloo, ON, Canada,where he is currently an Associate Professor. He is the author/co-author of more than 150 refereed technical papers and several book chapters, and the coauthor of a book on optical networking and survivability. His current research interests cover a wide range of topics in broadband networks, including survivable network design, wireless communications such as video multicasting and network security.

Prof. Ho is the recipient of Distinguished Research Excellent Award and Early Researcher Award in 2005, Best Paper Award in SPECTS'02, ICC'05, and ICC'07, and the Outstanding Paper Award in HPSR'02.