

Extending the p -Cycle Concept to Path-Segment Protection

Wayne D. Grover, *Fellow IEEE*, Gangxiang Shen, *Student Member IEEE*
TRLabs, #800, 10611-98th Avenue, Edmonton, Alberta, Canada T5K 2P7
and Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Alberta
Contact: {grover, gshen}@trilabs.ca

Abstract – This work introduces a significant extension to the method of p -cycles for network protection. The main advance is the generalization of the p -cycle concept to protect multi-span segments of contiguous working flow, not only spans that lie on the cycle or directly straddle the p -cycle. This effectively extends the p -cycle technique to include path protection or protection of any flow segment along a path as well as the original span protecting use of p -cycles. It also gives an inherent means of transit flow protection against node loss. We develop a capacity optimization model for the new scheme and compare it to prior p -cycle designs and other types of efficient mesh-survivable networks. Results show that path-segment-protecting p -cycles (“flow p -cycles” for short) have capacity efficiency near that of a path-restorable network without stub release. An immediate practical impact of the work is to suggest the use of flow p -cycles to protect transparent optical express flows through a regional network.

I. INTRODUCTION

One of the most interesting recent developments in survivable network architecture is the method of p -cycles. p -Cycles, introduced in 1998 [1], are in a sense like BLSR rings, but with support for the protection of *straddling* span failures as well as the usual protection of spans on the ring itself. The most striking property of p -cycles is that they retain ring-like switching characteristics (only two nodes do any real time switching and are fully pre-planned for each failure) but can be designed with essentially the same capacity-efficiency as a span-restorable mesh network. This means p -cycle based networks can be 3 to 6 times more capacity-efficient than ring-based networks while still providing BLSR ring switching speed and simplicity. In fact for straddling span failures, the average protection path has half the number of hops of the corresponding ring, so it may even be faster on average. When a span failure happens, only the two end nodes of the span do any real-time switching; no switching actions are required at intermediate nodes of the cycles. This property, which is shared with rings, contrasts with other forms of efficient shared protection mesh restoration schemes in which restoration routes and spare capacity are pre-planned, but all the intermediate switching nodes on the restoration routes must act in real time.

Since 1998, the basic theory of p -cycles as pre-configured structures in the spare capacity of a mesh network has been developed [2,3], and there have been studies on self-organization of the p -cycle sets [4], application of p -cycles to the MPLS/IP layer [5], application to DWDM networking [6] and studies on joint optimization of working paths and spare capacity. Notably in [6] it was found that full survivability against any span cut could be achieved with as little as 39% total redundancy. This greatly motivates continuing work and practical applications of the p -cycle concept. A more

comprehensive review paper on p -cycles can provide more background [8].

Concurrent work on cycle covers [9] under the coincidentally similar name of “protection cycles” should not be confused with p -cycles. The fundamentally important difference is the aspect of straddling spans in p -cycles. Straddling spans on a p -cycle can each bear *two* units of working capacity per unit of p -cycle capacity and they require *no* associated protection capacity on the same spans. All forms of ring or cycle covers fundamentally involve equal (or greater) amounts of protection and working capacity on every span and *at best* (which is what *oriented* cycle double covers in [9] accomplishes) reach a 1-to-1 ratio between these, for 100% redundancy. In contrast p -cycles, due ultimately to the effectiveness of the straddling span protection aspects, yield fully restorable architectures at well under 100% redundancy. It is easy to show in fact that on a Hamiltonian network, a single p -cycle can protect all spans and reach a limiting redundancy of $1/(d-1)$ where d is the network average nodal degree [13]. The significance of this is that $1/(d-1)$ is the same lower bound on redundancy that characterizes a span-restorable mesh network.

p -Cycles, therefore, offer a unique and attractive combination of properties. Importantly, however, all studies so far on p -cycles have considered only what we call “span-protecting” p -cycles. Each such p -cycle protects only spans that are either part of it, or that directly straddle the respective p -cycle, such as span (5-0) in Figure 1(a).

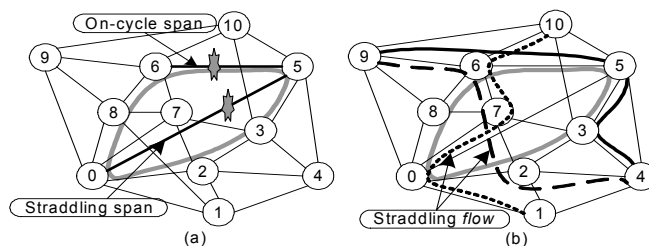


Figure 1. Examples of (a) a span protecting p -cycle and (b) the same cycle viewed as a flow-protecting p -cycle.

However, even at the time of the initial work on span-protecting p -cycles in [1], it was natural to ask if there was a *path* protection equivalent to basic span-protecting p -cycles. As simple as basic p -cycles are, the latter question turned out to be difficult to address. Ultimately the difficulty is in how to handle the aspect of “mutual capacity” contention (which is intrinsic to any path oriented or multi-commodity flow type of recovery scheme) in formulating the design model under a

paradigm of cyclic spare capacity structures. The corresponding operational complexity in trying to coordinate which paths can access which p -cycles, for which failures was also beyond our reach at the time.

The significance of the present paper is that these difficulties have been essentially overcome with the concept, not specifically of end-to-end “path p -cycles” per se, but of *path segment-protecting p -cycles*, which are even more general in nature. We find, perhaps ironically, that a theoretical design model and an operational framework are more accessible within this actually more general framework than for strictly end-to-end path protection. Accordingly, the concept to be described actually includes pure span and pure path-protecting p -cycles as special cases.

We cannot judge whether path-segment-protecting p -cycles to follow (“flow” p -cycles for short) will be as practically-useful as span-protecting p -cycles are, but we think nonetheless, that flow p -cycles represent an important extension of the theory and understanding of p -cycles in general. It also constitutes a unique addition to the inventory of known schemes for path-oriented network protection in general. In contrast to other end-to-end path-based methods, including path-restoration, SBPP, etc, involving shared protection capacity, the flow p -cycle approach has a basic advantage in terms of restoration speed because there is no signaling or seizure-and-confirmation processes involved to access the protection paths. This alone motivates its study as a unique alternative within the family of path oriented schemes, in addition to our interest in how much more capacity-efficient it is relative to span-protecting p -cycles.

At the same time, while whole networks based on flow p -cycles may seem complex, once the concept is explained below, some more practical specific tactics become apparent involving *selective* use of just one or a few flow p -cycles in conjunction with other protection schemes. Two such concepts (to follow) are (i) to support transparent optical transport of express flows through a regional network and/or (ii) using flow p -cycles around the perimeter of an autonomous system domain to provide a single unified and out-of-mind scheme for protecting all transit flows through the domain. Another important observation is that flow p -cycles inherently also can protect transit flows against node loss, although capacity design and signaling mechanisms to exploit this property remain to be developed. Here we will concentrate on span failures.

The rest of the paper is organized as follows. In Section II, we introduce the basic concept of path-segment (or “flow”) protection by a p -cycle. In Section III, we develop the integer linear programming (ILP) formulation model for the flow p -cycle method. Section IV reports a first set of experimental results. In Section V we close with considerations of the real time phase of operation and adaptation of the general concept to the specific applications mentioned above.

II. CONCEPT OF FLOW P -CYCLES

Fig. 1(a) shows a conventional span-protecting p -cycle. The spans (0, 2), (2, 3), (3, 5), (5, 6), (6, 8), and (8, 0) are *on-cycle* spans of the p -cycle shown. The span (0, 5) is not on the cycle, but its two end nodes are, making it a *straddling* span. A p -cycle can offer one protection path for the failure of any one

cycle span and *two* protection paths for the failure of any (of possibly several¹) disjoint straddling spans (one failure at a time). For example, if span (5, 6) fails, the route (5, 3, 2, 0, 8, 6) on the cycle offers one protection path to restore the failed span, in exactly the way a BLSR operates. But if span (0, 5) fails, paths (0, 2, 3, 5) and (0, 8, 6, 5) are both (simultaneously) available to provide protection paths. In either type of failure only two nodes do any switching in real time, and it is effectively identical to a preplanned BLSR switching reaction. In contrast a ring (or any form of cycle cover [9]) only protects on-cycle spans.

As is now well understood, and validated by several other groups including [6], it is the admission of straddling span protection relationships that dramatically reduces the total design redundancy relative to ring-based networks, to the point of near-equivalence with a span-restorable mesh network. New reference results here again confirm this basic but remarkable property. We say “remarkable” because in a sense the addition of straddling spans to a BLSR seems like a minor technical difference. And yet its significance in terms of network efficiency is huge: one goes from ring-like redundancies of often 200% or more, right to mesh-like redundancies of often under 50 to 60% to support the same set of service demands. Put the other way around, within a given set of installed facilities, p -cycle based operation can typically support two to four-times growth in the entire demand served, *without adding new capacity and without giving up the protection speed of existing ring-based networks*.

Note, however, in Fig.1(a) that spans (6-7) and (7-2), and several others, of the basic p -cycle are close to being straddling spans but cannot actually be span-protected by the cycle shown. However, a *path* that crosses both spans (6-7) and (7-2), as shown in Fig.1(b), *can* be considered to straddle the cycle shown when taking a path-level view. More specifically the *path segment* (6-7-2) can be considered to straddle the cycle shown, between the nodes 6 and 7. This simple observation is the starting point for the fully general concept. To develop this we must now take an explicit view of the service paths crossing the network and how they relate to the p -cycles we may use for protection. Fig. 1(b) shows this more highly resolved viewpoint in which we can design a flow p -cycle protected network. For convenience let us define a *flow* as any single contiguous segment of a working end-to-end path. Thus, the entire path, each span on the path, and any sequence of several spans along the path are flow segments that could be protected by flow-protecting p -cycles. Note this can refer to either a segment of a lightpath, an OC- n path, or an MPLS path.

Clearly flow p -cycle designs can access more opportunities for spare-capacity-sharing than the span p -cycle method. Any flow segment that intersects a flow p -cycle can be protected, not simply spans directly on or straddling the cycle. For example, if spans (2, 7) and (6, 7) in Fig. 1(a) incur failures, they cannot be restored by a span-protecting p -cycle. But, under the flow p -cycle shown in Fig. 1(b), the contiguous

¹ The number of opportunities for straddling span establishment on a given p -cycle depends on the relationship between the cycle and the graph topology. Straddling spans need not only be visually “inside” the cycle (see [8] for an example). A single Hamiltonian cycle, if it exists on a graph, will establish an on-cycle or straddling relationship to all network spans with a single p -cycle. But more generally we are considering p -cycle designs using multiple unit-capacity cycles to efficiently protect differential amounts of working capacity on spans and for greater availability, transmission performance, and capacity-efficiency than when protecting a whole network with a single p -cycle.

flows that traverse spans (2, 7) and (6, 7) can be restored by the cycle. Note, however that any demands being added or dropped at node 7 cannot be handled by the same flow p -cycle. The flow must be unchanged in its composition between the nodes where it intersects the flow p -cycle.

Theoretically, flow p -cycles can be categorized as the p -cycle equivalent to path restoration with failure specific stub-reuse. This is an intermediate between completely general stub release (as in [10,12]) and the complete prohibition against reuse of stub capacity that is built into the recent shared backup path protection scheme. In stub *re-use* the surviving path up to the protected segment is always part of the end to end path in the restored state. In other words the surviving stub path segments are always re-used in a failure specific way by the same path that used that capacity before the failure. Unlike stub release, which can permit the same or any other simultaneously failed path to exploit released stub capacity of failed paths, the stub re-use that is implicit in flow p -cycles requires no explicit real time actions to effect it.

A. Determining Flow-Protection Relationships

Given a cycle that is a candidate to be a flow p -cycle in a network design, the relation of any given path to the cycle can be either *intersecting* or *non-intersecting*. Only intersecting paths are relevant to the consideration of each candidate cycle. A path intersects a cycle if the two have at least two common nodes (which may include the source and destination nodes of the path). These are called *intersection nodes*. For example, the paths between nodes (4, 9) and (1, 10) in Fig. 1(b) both intersect the cycle shown and are relevant to the consideration of that cycle as a possible flow p -cycle. By inspection, that cycle would provide straddling-type protection to the two segments (6-7-2) and (0-7-6) and on-cycle protection to an example flow such as (6-5-3) should it exist. More generally, a path can intersect a cycle in a variety of more complex ways, involving more than two intersecting nodes.

Fig. 2(a) displays the simplest scenario, where the cycle and the relevant flow segment intersect at two nodes and the flow does not share any other spans or nodes with the cycle. Analogous to the concept of a straddling span in conventional p -cycles, it is natural to call this a *straddling flow* relationship. Similarly, Fig. 2(b) depicts a flow segment in an on-cycle relationship, which is called an *on-cycle flow*. From here, more complex relationships of flows to cycles need to be recognized that do not arise with span-protecting p -cycles. Fig. 2(c) shows a three-node intersection relationship of what is otherwise a plain straddling flow. Fig. 2 (d) shows an even more general intersecting flow relationship that produces two straddling portions and an on-cycle segment. Conceptually the flow-to-cycle intersection relationship can be arbitrarily complex, such as in Fig. 2 (e). But the following important observation can be made about any of these intersection relationships, no matter how apparently complex: With respect to a cycle j , and any span failure i on the intersecting flow segment, it is always possible to determine by simple inspection whether a unit copy of protection capacity on cycle j can provide either one or two protection paths for the affected flow. For example, in Figs 2(b) and (a) there are respectively one and two protection paths available from the cycle for any span failure on the corresponding intersecting flow segments. Fig. 2(c) is in this regard equivalent to Fig. 2(a). In Fig.2 (d) two paths are available if the failure span is in either of the straddling segments, but only one if a span fails in the on-cycle segment. Thus, a preprocessing program can

identify all the protection relationships between candidate cycles and the corresponding flow segments that it could protect. More formally this defines the topological parameters $\gamma_{i,j}^r \in (0,1,2)$, which give the number of protection relationship that a unit capacity on cycle j can (potentially) provide to the intersecting flow segment of the working path for end-node pair r , in the event of span failure i . It is zero if span i is not contained in a flow that intersects cycle j . It is one if span i is in an intersecting flow and is also a span of cycle j itself (i.e., an on-cycle relationship), and two otherwise (i.e., a straddling relationship).

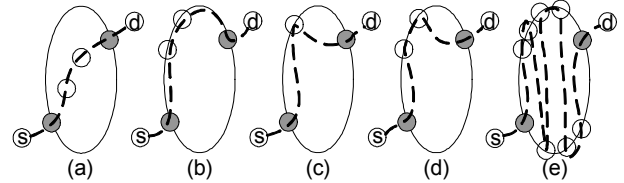


Figure 2. Various flow-to-cycle relationships.

B. Mutual Capacity Considerations

Now comes the most complicating aspect of the concept of path-oriented p -cycles. The reason we said “potentially provide” above is that, as the reader may have noted in Fig.1(b), the ability of the cycle to provide the assessed number of protection paths can depend on which other path failures occur at the same time. Unlike span-protecting p -cycles, when a span fails in the flow-oriented context, path segments with more than one set of end nodes are affected simultaneously. Because the affected paths (more specifically here, the affected path segments) all have different intersecting nodes on the flow p -cycle, we get into the problem of “mutual capacity.” This is the fundamentally complicating aspect of efficient multi node-pair re-routing problems (see [10] for elaboration). In the context of span restoration, there is only one “commodity” to be re-routed, between the two immediate end nodes of the failed span. This is at most $O(n^3)$ to perform optimally (i.e., via maximum flow) or in practice is well approximated by k -shortest paths in $O(n \log n)$ time. In contrast, as soon as more than one end-node pair is involved, the routing problem to make fullest use of available spare capacity is NP-hard. At the heart of it is the combinatorial issue of how to allocate the finite available capacity among the multiple end-node pairs that have mutual interest in using it.

Here we see the issue manifest itself in Fig. 1(b). If span (7,2) fails, then only the flow segment for path (4-9) needs protection and it can obtain two protection paths from the cycle shown. But if span (6,7) fails, then access to the same flow p -cycle must be coordinated between the segments for paths (9-4) and (10-1), which fail simultaneously. This failure-specific way in which a given p -cycle can be used in different ways leads requires a new family of intermediate variables, which is not needed in the design of networks using basic span p -cycles. These are the variables $n_{i,j}^r$ which denote number of unit-capacity copies of cycle j required to specifically restore the working flow for node pair r upon span failure i .

III. FLOW-PROTECTING DESIGN MODEL

We can now put together an integer linear programming (ILP) model for design of flow-protecting p -cycle networks.

Because we are presently only introducing the concept we will consider the “non-joint” design case where working demands are first routed via their shortest paths, followed by optimization of the spare capacity for flow p -cycle placement. Similarly, the present model assumes “oeo” OXC nodes, or transparent optical cross-connects with a suitable pool of wavelength converters, so that regeneration and/or wavelength conversion needs can be assumed to be met as required at any node. The cost of this assumed to be reflected in c_k below. (The significance of these considerations is that the design model does not need to be burdened with simultaneously solving for explicit wavelength assignments and need consider only capacity-related costs). The parameters of the model are: \mathcal{S} is the set of spans in the network. Index i denotes spans specifically in a failure context, k is used to index \mathcal{S} in other general contexts. \mathcal{D}_i is the set of end-node pairs of paths affected by failure of span i , index r . \mathcal{P} is set of all simple cycles of the graph that are candidates to become p -cycles, index j . In practice \mathcal{P} may be a limited set of candidate cycles that are eligible for use by virtue of limited circumference, hop count, or other engineering considerations. When needed to reduce computational difficulty, \mathcal{P} may also be limited to containing only elite or highly promising candidate cycles, along the lines of the work in [7]. c_k is the cost of adding a unit capacity (for instance an additional lightwave channel) to span k . c_k values are pre-computed constants which may include considerations such as the type of facility involved, distance-related costs for using amplifiers, plus the cost of any o/e interfaces at the span terminating OXCs, and/or an amortized allocation of the OXC core costs and wavelength converter pool cost.² d^r is the number of demand units (for instance, lightpath requirements) on node pair r . And as discussed above $\gamma_{i,j}^r \in (0,1,2)$ encodes the basic topological relationships between each span failure i with respect to the protection relationship cycle j provides for paths on demand pair r . Finally $\delta_{j,k} = 1$ if cycle j includes span k , otherwise, 0.

The variables to be solved for are as follows: n_j is the number of unit-capacity copies of cycle j to build. s_k is the number of spare capacity units required on span k to support the set of flow p -cycles used and, as explained, $n_{i,j}^r$ is a family of intermediate variables that consider the number of copies of cycle j that are needed specifically for protection of path r against failure i . All variables are non-negative integers, although it can be seen from the problem structure that spare capacity can be relaxed without loss of integrality. The model is:

$$\text{Flow } p\text{-cycles: } \text{Minimize } \sum_{k \in \mathcal{S}} c_k s_k \quad (1)$$

Subject to:

$$\sum_{j \in \mathcal{P}} \gamma_{i,j}^r \cdot n_{i,j}^r \geq d^r \quad \forall i \in \mathcal{S}; \forall r \in \mathcal{D}_i \quad (2)$$

² Note that node costs can also be referred into the link costs. Any dedicated per-channel nodal equipment is naturally a direct part of c_k . “Common equipment” costs such as for a wavelength converter pool to avoid blocking, and for DWDM mux / demux filters and the OXC core can be charged to each capacity unit added to incident spans based on the ultimate channel or link capacities of the core nodal equipment.

$$n_j \geq \sum_{r \in \mathcal{D}_i} n_{i,j}^r \quad \forall i \in \mathcal{S}; \forall j \in \mathcal{P} \quad (3)$$

$$s_k \geq \sum_{j \in \mathcal{P}} n_j \cdot \delta_{j,k} \quad \forall k \in \mathcal{S} \quad (4)$$

The first constraint system asserts that affected working flows must be fully restored. The second says that the number of copies of cycle j to build is set by the largest failure-specific simultaneous use of unit copies of cycle j . This is like setting the spare capacity of a conventional span or path restorable mesh network to satisfy the largest simultaneously imposed set of restoration flows over the set of all non-simultaneous failure scenarios. The final constraint system says that the spare capacity on span k must be enough to support the number of copies of each p -cycle that overlies the span.

An important simplifying aspect to note about this model (relative to prior attempts at strictly path protecting p -cycles) is that it avoids explicit enumeration of the specific paths or path segments to be protected. Rather, the decisions about which protected path segments are defined come out *implicitly* from the choice of p -cycles that the solution employs. But since every span failure scenario is considered in the model, all spans (and hence all paths) wind up being implicitly protected end to end, but over one or more protected flow segments on each end-to-end path. This is the key sense in which the more general nature of the flow p -cycle paradigm overcomes the previous problems we mentioned in attempts to directly formulate end-to-end p -cycle path protection.

Finally, a word in passing about the role of ILP-based design models such as above: The primary reason to use an ILP formulation in research is so that we can have an unobscured look at what is fundamentally true and possible about various basic architectures. In seeking basic insights and understanding, and repeatable comparisons of the inherent properties of different architectures, computational time is not a primary concern. If a given architecture or networking idea looks highly promising then any number of possibly speedier and more practical heuristics can be developed to exploit the opportunity. A common confusion seems, however, to be the assumption that a given network architecture “requires ILP and is therefore not scalable,” just because ILP tools *were used to study it*. But this is scientifically illogical. ILP methods can be applied to the design of any basic architecture, and any basic architecture can also be approached with heuristics if desired.

IV. TEST METHODS AND RESULTS

Four test networks were used to obtain samples of flow p -cycle -based networks for analysis and evaluation. The networks are summarized in Fig. 3.

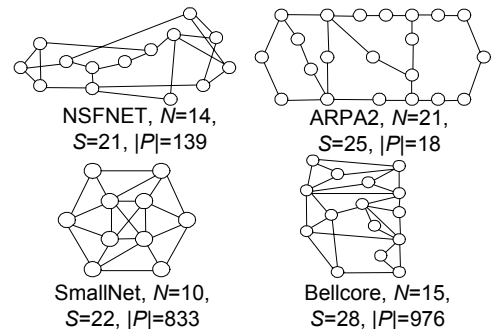


Figure 3. Test Networks

Below each network graph above, N is its number of nodes, S its number of spans and $|P|$ is the complete number of distinct simple cycles on the graph. The network average nodal degree, $d = 2S/N$. Each network has span distances assigned in proportion to the Euclidean distances of the span in the networks as drawn above. Except for Bellcore, demands were generated for all node pairs from a uniform random distribution on $[1..20]$ and the design problems were solved to optimality in under 2 hours with AMPL/CPLEX 7.1 on an Ultrasparc Sun Server at 450MHz with 4GB of RAM. The same demand generating procedure was used to create the Bellcore test case but only 50% of node pairs exchange non-zero demand quantities. The Bellcore solution required about a day and was solved to within 2% gap to optimality.

For comparative reference on each test network, optimal span-protection (SP) p -cycle designs, span-restorable (SR) mesh network designs as well as path-restoration designs with and without stub release were also produced for comparison. Notably the SP p -cycle reference designs here were actually produced with the flow p -cycles model by simply defining and equivalent demand matrix having non-zero values only between directly connected node pairs. The span-restorable mesh reference designs were produced by the method of [11] with no hop limit. The path restorable designs were produced with the methods referenced in [12]. The reference problems all solved in seconds or minutes at most.

Table 1 summarizes aspects of the results. For each test network, column 1 describes the span-protecting (SP) p -cycle designs, column 2 details the SR mesh designs and column 3 is for the flow p -cycle designs. Col.'s 4 and 5 are results for path restorable designs (PR) without, and with, stub release, respectively. (The sequence reflects the theoretically expected ranking in terms of increasing efficiency). All designs for each network have the identical working capacity and demand routing. The first row records the total spare capacity relative to the SP p -cycle design needed for 100% restorability against all single span failures. The next row shows the total design capacity cost relative to the SP p -cycle design. The third row shows the corresponding redundancy ratio of the overall network designs. This is followed by the number of distinct cycles on which p -cycles were formed for the p -cycle designs. Below this is a characteristic measure for each scheme of the average number of reconfiguration actions involved for each span failure. For SR and PR mesh designs it is the average number of restoration paths per failure. For SP p -cycles, it is the average number of p -cycles switching per failure, and for flow p -cycles it is the average number of flow segments deviated onto p -cycles to protect against each span failure. The last two rows record the numbers of constraints and variables for each network design.

TABLE I
SUMMARY OF RESULTS

	NSFNET					ARPA2					SmallNet					Bellcore				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
Spare capacity	1	96.9%	75.0%	71.4%	59.7%	1	92.6%	87.3%	72.6%	67.4%	1	99.7%	84.3%	80.0%	71.8%	1	96.5%	85.9%	76.9%	71.8%
Norm. total cost	1	99.0%	88.0%	86.3%	80.7%	1	95.9%	93.0%	84.8%	81.9%	1	99.9%	94.0%	92.3%	89.2%	1	98.5%	94.0%	90.2%	88.1%
Redundancy	91.8%	89.9%	68.9%	65.6%	54.8%	125.2%	115.9%	109.3%	90.9%	84.4%	62.0%	61.8%	52.3%	49.6%	44.5%	73.6%	71.0%	63.2%	56.6%	52.8%
Num. of p -cycles chosen	11	—	24	—	—	12	—	8	—	—	9	—	15	—	—	12	—	29	—	—
Ave. num. of paths/cycles/ segs. per failure	6	4	21	13	13	5	3	32	31	31	5	5	8	8	8	5	4	17	8	7
Constraints	42	441	2502	615		50	625	1202	1325		44	484	18419	533		55	757	18047	853	
Variables	160	1354	17590	11213		43	219	13129	7134		855	6382	59998	20042		1004	9591	84692	49801	

V. CONCLUSION DISCUSSION

A. Interpretation of Results and Significance

Results show that the flow p -cycle method can yield significant reductions in spare capacity requirement relative to the SP p -cycle architectures, ranging from 12 to 25%. In these results it is hard to generalize about the difference between SP and flow p -cycles in terms of dependency on d , because significant dependencies also exist on the demand patterns, which differ across the networks tested. In comparing flow p -cycles to path restoration without stub release, however, we see the difference is always under ~10% (except to ARPA2) and tends to be smaller in the higher degree networks. Other results indicate that in general a design based on flow p -cycles may have more distinct p -cycle structures than in SP p -cycle designs, although one test case (ARPA2) required only eight flow p -cycles whereas twelve SP p -cycles were otherwise required.

Clearly, however, there is greater complexity associated with flow p -cycle designs than with SP p -cycles in terms of the design and operational measures in the last 3 rows of Table 1. The number of variables and constraints in the ILP design formulation is much larger than for either SP p -cycles or SR mesh, indeed suggesting a role for heuristics in further design

and research work. The complexity of the failed protected reconfigured state with flow p -cycles (as measured by the average number of re-routed segments) is also higher than corresponding measures of reconfiguration extent in SP p -cycles or SR mesh.

B. Operational Aspects

Let us now address the operation of flow p -cycle protection in terms of providing a real time mechanism of reaction to a failure. We assume that p -cycles would be formed and sustained by OXC nodes under central control to connect the required spare wavelength channels together to create the desired set of p -cycles. At the time a p -cycle j is established through a node x , a list of the corresponding protected flow segments that intersect the cycle at that node is recorded in association with the p -cycle. The Signal_ID of the working path of which the p -cycle protects a segment is also recorded at the node for each span (as a possible failure) on the flow segment. In effect this data sets up matching conditions for node x to know locally which working signals (if any) it should switch into p -cycle j , depending on which span fails on any of the flow segments passing through it.

Upon failure, node x is either adjacent to the failure, in which case it sees LoS, or AIS is inserted downstream by the

two nodes adjacent to the failure. All working signals bear a unique Signal_ID in their overheads and any time a node inserts AIS, it appends the ID of the incident span that has failed. Thus, at propagation speeds, the failure indication data {AIS, Signal_ID=Z, Span_ID=k} passes through all nodes on the failed path. But only node x will have been “pre-wired” with the matching conditions to associate Signal_ID=Z with locally accessible p -cycle j if an indication of its failure arrives, arising from span k . Thus a logical matching rule can be applied at any node seeing an AIS indication to quickly determine if it has a custodial responsibility to do protection switching for the failed signal. Fig.4 summarizes the principle concept.

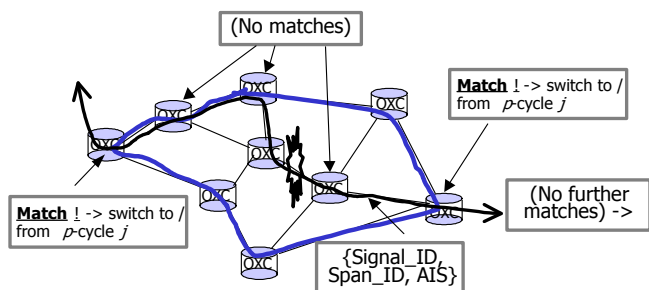


Figure 4. How the network is pre-planned to activate flow-protecting p -cycles at the time of failure.

C. Simplified Applications of the General Concept

While it is a useful advance to understand the fully general model of flow-protecting p -cycles, the complexity may be judged higher than desired from a near-term operational standpoint. But an appreciation of the concept actually suggests some simpler specific adaptations of flow p -cycles that may be more practically manageable and useful. One of these is to only consider the use of flow p -cycles only for the important express flows through a network region. Conceptually one can picture an overall network design comprised in part of a set of simple fast-acting “local” span-protecting p -cycles. Logically overlaid on this is a select set of “express flow” protecting flow p -cycles. A particular economic advantage of this selective use of a few flow p -cycles is that the express flows they protect may take advantage of long-reach optical technology for optical bypass of all intermediate nodes on the protected flow segment. This allows the express flow signals to remain in the optical domain through the entire region, but remaining protected, saving considerably over the alternative of terminating on each OXC en route.

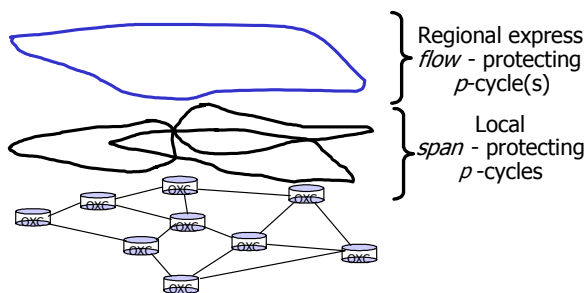


Figure 5. Using a mixture of flow-protecting and span-protecting p -cycles for optical bypass on express flow.

Closely related to this idea is the even more specific proposal of an area-boundary flow-protecting p -cycle. This is

in the context of multi-domain optical networking. The idea is that within a domain, any local protection schemes could be used, but flows traveling entirely through the domain are protected by a domain perimeter flow p -cycle. The primary advantages of this would again be optical bypass savings and the simplicity of separating all transit flow considerations from the protection of intra-domain flows. The concept is summarized in Fig. 5. In a long-haul network the “diameter” of the express flow-protecting p -cycles may match the transparent optical reach limit of the path segments involved so wavelength conversion and regeneration can be provided at the flow p -cycle nodes only, with all-optical transmission on the protected path segments.

D. Further Work

In ongoing work, we are investigating joint working and spare capacity assignment for the flow p -cycle method as well as heuristics to reduce the computational complexity by p -cycle preselection strategies, along the lines of [7]. We are also interested in understanding more thoroughly how network average nodal degree and the hop-limit of allowable p -cycles affects performance and in characterizing the level of transiting flow restorability in the event of node failures, when capacity is optimally designed only for span restorability. Other lines of research seek to optimize the selective use of flow p -cycles with long-reach optics for express network flows or regional transiting flows and to determine how much extra spare capacity is needed to allow for 100% transit flow recovery from node loss if desired.

REFERENCES

- [1] W. D. Grover, D. Stamatelakis, “Cycle-oriented distributed preconfiguration: Ring-like speed with mesh-like capacity for self-planning network restoration,” in *Proc. ICC'98*, 1998, pp. 537-543.
- [2] D. Stamatelakis, Theory & Algorithms for Preconfiguration of Spare Capacity in Mesh Restorable Networks, *M.Sc. Thesis*, Univ. Alberta, Canada, 1997.
- [3] D. Stamatelakis, W.D. Grover, “Theoretical Underpinnings for the efficiency of restorable networks using pre-configured cycles (“ p -cycles”),” *IEEE Trans. Comm*, vol.48, no.8, Aug. 2000, pp. 1262-1265.
- [4] D. Stamatelakis, W.D. Grover, “OPNET Simulation of Self-organizing Restorable SONET Mesh Transport Networks”, in *Proc. OPNETWORKS '98 Conference (CD-ROM)*, Washington, D.C., April 24-25, 1998, paper 04.
- [5] D. Stamatelakis, W. D. Grover, “IP layer restoration and network planning based on virtual protection cycles,” *IEEE JSAC*, vol.18, no.10, Oct. 2000, pp. 1938 - 1949.
- [6] D. A. Schupke, C. G. Gruber, and A. Autenrieth, “Optimal configuration of p -cycles in WDM networks,” in *Proc. of IEEE ICC'02*, NY City, Apr.28-May2, 2002.
- [7] W. D. Grover, J. E. Doucette, “Advances in optical network design with p -cycles: Joint optimization and pre-selection of candidate p -cycles,” in *Proc. IEEE-LEOS Topical Meetings*, Quebec, July 15-17, 2002.
- [8] W. Grover, D Stamatelakis, “Bridging the ring-mesh dichotomy with p -cycles”, in *Proc. IEEE / VDE DRCN 2000*, Munich, Germany, April 2000, pp. 92-104.
- [9] G. Ellinas, A. G. Hailemariam, T.E. Stern, "Protection Cycles in Mesh WDM Networks" *IEEE JSAC*, vol.18, no.10, Oct.2000, pp. 1924-1937.
- [10] R. R. Iraschko, W.D. Grover, “A highly efficient path-restoration protocol for management of optical network transport integrity,” *IEEE JSAC*, vol.18, no.5, May 2000, pp. 779- 793.
- [11] M. Herzberg, S. Bye, “An optimal spare-capacity assignment model for survivable network with hop limits,” in *Proc. of IEEE GLOBECOM '94*, pp. 1601-1607.
- [12] R. R. Iraschko, M.H. MacGregor, and W.D. Grover, “Optimal capacity placement for path restoration in STM or ATM mesh survivable networks,” *IEEE Trans. Net.*, vol. 6, no. 3, June 1998, pp. 325-336.
- [13] A. Sack, W.D. Grover, “Hamiltonian p -cycles for fiber-level protection in homogeneous and semi-homogeneous optical networks,” submitted to *IEEE Network, Special Issue on Protection, Restoration and Disaster Recovery*, February 2003.