Bridging the ring-mesh dichotomy with p-cycles

(invited paper)

W. D. Grover, D. Stamatelakis

TRLabs & Dept. of Electrical and Computer Engineering,
University of Alberta, Edmonton, T6G 2E1, Canada

Abstract -

p-cycles represent a new strategy for network restoration. Their main feature is that they retain ring-like switching characteristics but, under appropriate design optimization, yield networks that are virtually as efficient as span-restorable mesh networks. This remarkably desirable combination of attributes makes p-cycles and exciting topic for investigation. This paper attempts to summarize our work to date on p-cycle network concepts, design theory, and implementation methods for both connection-oriented (Sonet, WDM) and IP layer network restoration.

1. Introduction

The p-cycle networking concept is appears to break the long-standing quandary between the basic ring and mesh alternatives. The significance of the p-cycle concept is that it permits ring-like switching speeds (because only two nodes do any real-time actions) and yet optimized network capacity plans are virtually as efficient as a span-restorable mesh network. This definitely appears to be a case of getting the "best of two worlds". This paper is designed to provide a recap and unpublished extensions of the original p-cycle concept for span restoration in WDM or SONET environments (first published in [1]), and of more recent work towards both span and node protection in an IP (Internet Protocol) environment (previously unpublished work).

In the remainder of Sec.1 we briefly review the status-quo touching on the advantages and disadvantages of conventional ring and mesh alternatives. This may be safely skipped by those already familiar with the well-established ring versus mesh dichotomy.

The next major part of the paper (Sec.s 2-4) covers p-cycles as first conceived for span restoration in a discrete-capacity connection-oriented environment, as would apply for Sonet or WDM applications. It is in this section that the key intuitive understanding is developed of why p-cycles can be virtually as efficient as a mesh network while retaining operational characteristics and real-time switching behavior that is no more complicated than in a BLSR. Optimal p-cycle network design and self-organizing formation of p-cycles is also covered.

The final major part of the paper (Sec.s 5-7) is devoted to adaptation of the p-cycle concept to connectionless environments such as ATM or IP. The two main extensions here are the opportunity to plan p-cycles so as to exploit the stat-muxing nature of demand flows to reduce capacity requirements, and in response to the need to protect routers in the Internet against reportedly frequent node failures, the concept of node-encircling p-cycles. Operational details for IP node and span restoration are given and preliminary approaches towards optimal design of IP p-cycle networks to minimize the worst-case restoration-induced congestion loads imposed on surviving spans.

1.1. Ring and Mesh - “Two Solitudes?”

For nearly a decade there have been two basic, and quite separate, approaches to network restoration, generically denoted “ring” and “mesh”. Ring-based survivability involves the use of bi-directional line switched rings (BLSRs) or uni-directional path-switched rings (UPSRs) (or their optical versions: Optical Path Protection Ring (OPPR) and Optical Shared Protection Ring (OSPR)) as self-protecting transmission systems overlaid on the network topology. In the UPSR each point to point demand is duplicated at its origin and permanently routed both ways around the ring. This establishes a 1+1 receive selection situation for each node, equivalent to a virtual 1+1 diverse-routed protection arrangement for each tributary signal. In the BLSR, nodes adjacent to a span failure (or an intervening node failure) sense the signal loss, test the status of the protection channel, and (if free) switch their transmit signal to the protection channel in the reverse direction from the failure. Each of these nodes looks in the reverse direction on the protection channel to receive a replacement signal copy for their receiver. Since any two nodes can make similar use of the shared standby capacity around a BLSR, it is more efficient than the UPSR. The nodal elements of rings are relatively low-cost add-drop multiplexers (ADMs). An ADM has two optical line (e.g. OC-24, or OC-48 say) terminations and can originate (add) or terminate (drop) any of the tributary payload signals (e.g. STS-1 or DS3) from the line signal passing through it. See [2] for more on rings.

The important point is that rings use a simple switching mechanism which permits restoration in about 50-60 ms, but they require at least 100% redundancy. In complete multi-ring network designs the working fiber or channel groups themselves are usually not fully utilizable, so the overall installed-to-working

1. To avoid confusion we will reserve “IP” for Internet Protocol. For Integer Program design formulations, we will recognize that they are usually “mixed Integer Programs” and refer to them as MIPs, or generically as “math programming” formulations.

2. Note that “ring” generically also includes 1+1, 1:1 APS architectures.
capacity ratio can be 200-300%. Thus, rings are fast but not intrinsically capacity-efficient.

Mesh survivability is more capacity-efficient because each unit of spare capacity is reusable in more ways across the network as a whole. Mesh restoration is typically based on the use of digital cross-connect systems (DCS) embedded in a set of point-to-point transmission systems under centralized or distributed control. Signal units that traverse a failed span spread out as individuals or subgroups and may follow many diverse paths through smaller amounts of spare capacity than in ring networks. Performance very close to idealized maximum-flow routing efficiency can be achieved [3]. Because of the dynamic state-dependent routing mechanisms used, each unit of spare capacity in a mesh network is reusable in many different ways, depending on which failure occurs. (In a ring, the spare bandwidth protects only spans on the same ring). However, because of the more general nature of solving a discrete capacitated multiple-path re-routing problem, mesh restoration is not expected to be as fast as with rings[1], but it permits a major reduction in the capacity required to serve the same set of demands. While ring-based networks always require 100% or more redundancy (over 200% is not uncommon over complete well-planned networks), a span-restorable mesh network may be only 50 - 70% redundant, depending on network topology [5,6,7].

Thus, each of these long-standing contenders has strengths and weaknesses. Rings offer the fastest restoration times, with the simplest reconfiguration mechanisms and lower-cost terminal equipment. But design and operation of networks of many rings is quite complex and is increasingly seen to be lavish in the use of total capacity and to have problems in rapidly growing hard-to-forecast demand environments of “stranded capacity” and unanticipated ring exhaust. Mesh offers much greater efficiency, and is increasingly seen to be easier to optimize and grow flexibly in response to unforecast growth patterns. The mesh alternative is, however, based on more expensive DCS platforms. Consequently mesh restoration has its best economics in long-haul circumstances where cost is more dominated by the total bandwidth-distance product than in metro networks. Ring networks tend to be more cost-efficient in metro areas where cost is dominated by terminal equipment cost.

To date, the choice between a ring or mesh-based network has been essentially black or white, one or the other proposition. There has been recognition that they can be used together in a ring-access / mesh-core division of responsibility [8,9] but in practice the ring and mesh components of such 'hybrids' continue to be designed and operated independently, along their respective principles. Prior to 1998 [1] no alternative had emerged that could offer the best advantages of both the ring and extremes. This well-entrenched dichotomy persisted for at least a decade from 1988-98 as evidenced by all operating networks and the either ring or mesh frameworks of virtually all research papers during that era. Obviously if there was a way to find a true melding of ring and mesh, the goal would be to get the speed of rings while keeping the efficiency of mesh.

At first blush, however, it seems reasonable that the simplicity of the switching in rings must in some way be inversely tied to the capacity needed for restoration, i.e., that to have minimal spare capacity you would unavoidably need a slower more complicated mechanism to exploit it. Or, in other words, that the only way rings or APS can be as fast as they are is by virtual of their 100% redundant nature. In initial work on spare-capacity pre-connection concepts for mesh networks [10] this is exactly the kind of trade-off we expected to manifest itself. It was therefore with great initial surprise that we discovered a ‘fully pre-connected’ structure that could use the same switching mechanism as a ring, but which gives up virtually nothing in terms of required extra spare capacity. This is the significance of p-cycles. They combine the speed of rings without compromising on the efficiency of mesh-based networking.

### 2. p-Cycles in a Sonet or WDM span-restoration context

The method of p-cycles for a SONET or WDM network[2] is based on the formation of closed paths (elementary cycles in graph theoretic terms), formed in the spare capacity only of a mesh restorable network. They are formed in advance of any failure, out of the previously unconnected spare capacity units of a restorable network. Despite similarity to rings in that both use a cycle on the network graph for their topology, p-cycles are unlike any ring-based systems that we know of to date[3] in that they protect both on-cycle and straddling failures, illustrated in Figure 1. This initially seems to

---

1. Although approaches involving distributed pre-planning with a mesh DRA could potentially be as fast as rings [4].
2. p-cycles apply in principle to any connection-oriented transport layer, including ATM using VP or VC constructs to create the p-cycles, but for simplicity the section is written in the language of a discrete-capacity SONET environment.
3. This includes 1+1 DP, BLSR- OSPR, UPSR- OPPR, FDDI rings

---

FIGURE 1. Use of p-cycles in restoration

a) A p-cycle, X  
b) A span on the cycle fails, p-cycle X contributes one res-  
c) A span off the p-  
d) A span off the p-cycle fails, p-cycle X contrib-
be a rather minor difference. When its implications are fully worked through, however, it turns out to be the key to obtaining mesh-like network efficiency from a ring-like protection structure.

Figure 1(a) shows an example of a pre-configured protection cycle (a "p-cycle"). In (b), a span on the cycle breaks and the surviving arc of the cycle is used for restoration. This action is functionally like a unit-capacity BLSR. In (c) and (d), however, the same p-cycle is accessed to support restoration of working paths that are not on the cycle. In fact, cases (c) and (d) are the more advantageous circumstances in general (and don’t apply for rings) because two restoration paths are available from each p-cycle for such failures. In contrast, any conventional ring provides at most one restoration path per unit of ring protection capacity and protect only against failures on the spans of the same ring, not on 'straddling’ spans.

This makes a very significant difference to the network restoration coverage provided by the same investment in spare capacity in a ring as opposed to in a p-cycle. To illustrate, further examination of the single p-cycle in Figure 1 shows that it (alone) can provide restoration in spare capacity in a ring as opposed to in a p-cycle. As shown in Figure 1, each p-cycle can contribute up to two paths to a wider range of restoration scenarios than a ring. Rings also have a structural association between the working demands which they protect and the protection band width in the same ring, while p-cycles are formed only within the spare capacity layer of the network, leaving the working paths to be routed freely on shortest paths, or any other route desired. In other words, the working demands may be provisioned freely as growth arises, as if in a failure-free point-to-point network; the p-cycles formed in the sparing layer adapt to suit the working path layer. A deployed p-cycle design may also be easily modified by the DCSs that form it, whereas SONET ring placements are essentially permanent structural commitments of both working and spare capacity, through which the routing of new working paths must conform. In contrast, with p-cycles, working paths go their own way: it is the p-cycle structures in the sparing pool that will be adapted accordingly, not vice-versa.

In other words, the working demands may be provisioned freely as growth arises, as if in a failure-free point-to-point network; the p-cycles formed in the sparing layer adapt to suit the working path layer. A deployed p-cycle design may also be easily modified by the DCSs that form it, whereas SONET ring placements are essentially permanent structural commitments of both working and spare capacity, through which the routing of new working paths must conform. In contrast, with p-cycles, working paths go their own way: it is the p-cycle structures in the sparing pool that will be adapted accordingly, not vice-versa.

Table 1 summarizes these and other aspects of the p-cycle concept, as distinct from rings. p-cycles are formed from individual spare links (or channels) of the point-to-point OC-n systems present, whereas SONET rings commit a whole OC-n module of working and spare capacity to the same cycle. As shown in Figure 1, each p-cycle can contribute up to two paths to a wider range of restoration scenarios than a ring. Rings also have a structural association between the working demands which they protect and the protection band width in the same ring, while p-cycles are formed only within the spare capacity layer of the network, leaving the working paths to be routed freely on shortest paths, or any other route desired. In other words, the working demands may be provisioned freely as growth arises, as if in a failure-free point-to-point network; the p-cycles formed in the sparing layer adapt to suit the working path layer. A deployed p-cycle design may also be easily modified by the DCSs that form it, whereas SONET ring placements are essentially permanent structural commitments of both working and spare capacity, through which the routing of new working paths must conform. In contrast, with p-cycles, working paths go their own way: it is the p-cycle structures in the sparing pool that will be adapted accordingly, not vice-versa.

Table 1: Comparison of the p-cycle and Ring Technologies

<table>
<thead>
<tr>
<th>Attribute</th>
<th>p-cycles</th>
<th>SONET rings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modularity</td>
<td>any add-drop or cross-connection signal unit</td>
<td>OC-n</td>
</tr>
<tr>
<td>Protection Yield</td>
<td>Up to two useful restoration paths per unit of p-cycle capacity</td>
<td>One restoration path per unit of ring protection capacity</td>
</tr>
<tr>
<td>Protection Flexibility</td>
<td>Restores failures on the cycle and on cycle-straddling spans</td>
<td>Each ring protects only spans contained in the same ring</td>
</tr>
<tr>
<td>Routing and provisioning of working paths</td>
<td>May proceed without regard to protection structures, with p-cycles adapted to suit.</td>
<td>Routing must conform to deployed (ring) structures and inter-ring transition restrictions</td>
</tr>
<tr>
<td>Network Redundancy</td>
<td>Essentially that of a span restorable mesh network</td>
<td>Significantly over 100% investment in spare capacity</td>
</tr>
</tbody>
</table>
trast, with $p$-cycles, working paths go their own way: it is the $p$-cycle structures in the sparing pool that will be adapted accordingly, not vice-versa.

Finally, the remarkable implication of protecting straddling failures is that a $p$-cycle spare capacity design takes little or no more capacity than a corresponding span-restorable mesh network. Our next aim will be to show the optimization procedure under which this level of efficiency is attained.

3. Optimal Design with $p$-Cycles

Two MIP formulations were developed and tested in [11]. The first (called “$\text{max } Rp | \text{ spare}$”) produces a $p$-cycle plan with maximum restorability within a given amount and placement of spare capacity, such as from an existing mesh restorable design. Results with that formulation show restorability levels very close to 100% with $p$-cycles formed within the existing disposition of spare capacity for an optimal span-restorable mesh. For space reasons we present here only the second formulation, (min spare $| Rp=1$) which determines the set of $p$-cycles which minimize the total amount (or cost) of spare capacity required for strictly 100% $p$-cycle restorability. The high restorability achieved within a mesh capacity plan in $\text{max } Rp | \text{ spare}$ and conversely the minimal excess sparing above a mesh needed for $\text{min spare } | Rp=1$ show that optimal $p$-cycle assemblies and span-restorable meshes are very close cousins, but do not appear to be strictly identical twins of one another.

A preliminary step for (min spare $| Rp=1$) is to generate the set of all elemental cycles from the network graph, subject to a limit on the maximum cycle length. We used limits from 10 to 25 hops, depending on the test network. An adaptation of Johnson’s algorithm [12] for all cycles of a graph is implemented for this step. The MIP generates an optimal $p$-cycle plan by choosing the number of copies of each elemental cycle to be configured as a $p$-cycle.

We denote the set of cycles as $P$. Let $E$ be the set of all network spans. $S = |E|$ is the number of network spans. $s_j$ and $w_j$ are the number of spare and working links on span $j$, respectively. $n_i$ is the number of unit-capacity copies of cycle $i$ in the $p$-cycle design. $p_{ij}$ is the number of spare links required on span $j$ to build a copy of $p$-cycle $i$. $p_{ij}$ is 1 if cycle $i$ passes over span $j$; otherwise it is 0. $c_j$ is the cost or length of span $j$. $x_{ij}$ is the number of paths that a single copy of $p$-cycle $i$ provides for restoration of span $j$. The coefficients $x_{ij}$ and $p_{ij}$ are evaluated in advance for each cycle in $P$. $x_{ij}$ can be either 0, 1 or 2 for every $i,j$. It is zero if either or both of the end nodes of prospective failure span $j$ are not on the cycle. It is 1 if both of the span end nodes are on the cycle and are adjacent along the cycle. $x_{ij}$ is 2 if both of the failure span end-nodes are on the cycle but they are not adjacent to one another on the cycle, i.e., span $j$ has a straddling relationship to cycle $i$. Problem $\text{min spare } | Rp=1$ is then:

$$\text{minimize } \sum_{j=1}^{S} c_j s_j$$

subject to:

$$s_j = \sum_{i=1}^{P} p_{ij} \cdot n_i \quad \forall (j \in E)$$

$$w_j \leq \sum_{i=1}^{P} x_{ij} \cdot n_i \quad \forall (j \in E)$$

$$n_i \geq 0 \quad \forall i = 1, 2, \ldots |P|$$

This formulation was solved with CPLEX 3.0 in the test networks detailed in Table 2. In Table 3, the efficiency of optimal $p$-cycle spare capacity designs for these networks is compared to corresponding optimal spare capacity designs for span restorable mesh networks using the methods from [5]. “Excess Sparing” is the percentage of total spare capacity that the $p$-cycle design required above the mesh spare capacity design reference. Table 3 drives home the point that the $p$-cycle principle is essentially mesh-like in its efficiency.2 Thus, at least with an optimally designed set of $p$-cycles, we can put the network in a fully restorable state that requires only two cross-connect actions per restored signal, without significantly more spare capacity than in a conventional mesh-restorable network.

---

1. By “elementary” cycles we mean cycles that may cross over themselves span-wise but do not “figure- eight” through any node.

2. To help make the point: the corresponding entries in the “excess spare capacity” column for rings would be in the 100s of percent.
3.1. WDM capacity-slice nodal device for p-cycles

As so far described the p-cycle concept is amenable to realization in a DCS-based network environment. It is interesting, however, to look at the p-cycle equivalent to an ADM node. The basic case follows from Fig. 1 that it would be ADM-like in terms of having two optical line interfaces (each 50/50 working/spare) and a local access tributary add/drop interface. But unlike a ring ADM it would also have two additional line-rate working interfaces on its “south” or straddling span face. In fact, in general it may have 2, 4, 6 or more working ports on the straddling-failure side if these correspond to 1, 2, 3 physically diverse straddling spans (so they fail independently). The resulting generalized nodal device is portrayed in Figure 3. Failure may be sustained on any two such “straddling side” interfaces by switching their payload signals into the respective east and west direction spares. The “straddling” face is comprised of pairs of working line rate signals because the device provides shared protection access to the two halves of the respective p-cycle on which it resides. Typically the expected failures would be associated with each other on the same physical span which has undergone a failure, but strictly any two single working line rate failures, over the set of straddling spans, can be supported simultaneously.

This combination of properties puts the device in a unique middle ground in terms of a networking element architecture between ring ADMs and full-blown digital cross connect (DCS) systems. It is like an ADM in that it is a buy-as-you-need capacity-slice building block. This is one of the benefits of rings over DCS which are large complete switching systems interfacing all the transmission capacity arriving at a node. But unlike an ADM, they will have an application-dependent redundancy. In a nodal site of degree \( d = 2 \), a conventional ADM is the only choice. In higher degree sites, however, devices like this may support up to \((d-2)\) “straddling side” interfaces. Thus whereas an ADM has redundancy \( R = 100\% \), a p-cycle node device would have \( R = 2/(2+2k) = 1/(k+1) \) where \( k \) is the number of straddling physical spans interfaced at the site. A \( d = 5 \) site, for instance, could then have an individual nodal redundancy as low as 25% (i.e., \( k=3 \)).

4. Self-organized p-cycle formation

The MIP design above would be suitable for a centrally configured p-cycle based network. But a self-organizing strategy for the autonomous deployment and continual adaptation of a network p-cycle state also seems possible. The Distributed Cycle PreConfiguration (DCPC) protocol is an adaptation of the statelet processing rules of the Selfhealing Network (SHN) protocol [13]. A statelet is embedded on each spare link and contains a number of semi-static information fields (like the K1-K2 bytes in the Sonet APS protocol). Each logical link (any managed capacity unit), as viewed by a node attached to it, has an incoming statelet and outgoing statelet. An incoming statelet arrives at a node on a link, and originates from the adjacent node connected through the link. As in the SHN, each outgoing statelet has an incoming statelet which forms its precursor and there is no nodal database of network topology nor is there any centralized co-ordination required other than long-term observation and capacity augmentation for growth.

The DCPC protocol is a set of event-driven rules for nodal interaction via such statelets. All interactions happen within the spare capacity of the network only and are all in non-critical time before any failure has arisen. At the large scale DCPC first allows each node to explore the network for p-cycle candidates that are discoverable by it from its location in the network. After completion of its exploratory role as “cycler node” (below), it hands off to the next node in order by a one-time “next-node hand-off” flood-notification. After all nodes have assumed the role of the cycler once, each “posts” its best-found cycle candidate in a distributed network-wide comparison of results. In this step all nodes hear the performance metric, and other details, of the globally best p-cycle candidate discovered by any of their peers. The competition flood expands through the network as each node locally relays the statelet with the best metric, or asserts its own best if and while the latter is superior to anything else it has yet received notice of. Eventually, the globally best cycle candidate dominates, everywhere. Upon thus learning of the winning candidate, the Cycler node who discovered this p-cycle candidate, goes on to trigger its formation. All nodes on the new p-cycle update their local s of restoration switching information to exploit the new p-cycle. The whole process repeats, spontaneously, adding one p-cycle per iteration until a complete deployment of near-optimal p-cycles is built. Thereafter, it continually adapts the p-cycle set to changes in the working capacity layer.

All statelet broadcasts originate at the cycler node. To initiate the cycle-exploration process, the cycler places an outgoing statelet on one spare link in each span at its site. Each of these primary statelets has a unique index number. After the primary statelet broadcast, the cycler node invests a pre-determined time in


1. There are patents pending on this device structure, the basic p-cycle concept, DCPC protocol, and node-encircling p-cycles for IP router restoration which follows.
sampling of the returning statelets. The Tandem node rules determine what p-cycle candidate the Cycler node will discover in a given round of global cycle comparison and formation. A Tandem node will broadcast each incoming statelet to the largest extent warranted by the statelet’s numpaths score within the context of the available outgoing link resources and other statelets currently present at the node. If all outgoing spare links on a span are occupied, a new incoming statelet can displace an outgoing statelet if it has a numpaths score better than the precursor with the lowest current score. Statelets on a given index can also only be forwarded to adjacent nodes which are not already present in the accumulating route of the corresponding precursor. The single exception to this rule is that a statelet may be broadcast from a Tandem to its own cycler node, which is present in all route fields. These rules, and a hop limit, restrict the process to consider only simple cycles of allowed sizes.

The emergent effect of the rules (more fully detailed in [1,14]) is that, shortly after triggering the process, a cycler node receives incoming statelets whose route fields trace out cycles which begin and terminate at the cycler node. As returning statelets arrive, the cycler maintains a record of the received statelet (and route) with the best score, defined as the ratio of accumulated protection relationship (useful paths embedded within the growing cycle) to links consumed. The cycler persists in observing the incoming statelets because a cycle tends to provide a number of highly useful paths as it is allowed to evolve under the collective interactions of the Tandem nodes. Usually it grows in size as it improves its score, until hopcount or network effects stabilize the pattern of the candidate cycle formed. Fig. 4 is an illustration, from simulation, of how one prospective p-cycle evolves improving its score with time.

1) Score = 1.0 2) Score = 1.8 3) Score = 2.0
4) Score = 2.33 5) Score = 2.43 6) Score = 2.71
7) Score = 3.22 8) Score = 3.4

**FIGURE 4. Evolution of the p-cycle candidate score.**

DCPC performance is measured in terms of the restorability level it achieves through self-organized p-cycle creation. As set of results obtained by OPNET simulation is given in Table 4. The test networks are efficiently modularized mesh spare capacity designs. Where “non-mod” is indicated, the restorability level is 100% prior to the modularization. In the other cases the DCPC-achieved restorability level was less than 100% (though over 90%) in the non-modular minimal spare capacity designs, then rose to the levels shown when the small extra amounts of unused capacity due to modularity were added1. Papers [1,14] provide greater detail on DCPC but the important point is that it seems possible to have the option of a self-planning network based on p-cycles. The DCPC process inherently also serves as an ongoing self-audit on the restorability margins within any network and can thereby generate the inputs needed for ongoing provisioning of new capacity as growth occurs, to always retain 100% restorability.

<table>
<thead>
<tr>
<th>Network</th>
<th>Modularity</th>
<th>DCPC p-cycle Restorability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net1</td>
<td>non-mod</td>
<td>100 %</td>
</tr>
<tr>
<td>Net2</td>
<td>non-mod</td>
<td>100 %</td>
</tr>
<tr>
<td>Net3</td>
<td>OC-24</td>
<td>91.5 %</td>
</tr>
<tr>
<td>Net4</td>
<td>OC-48</td>
<td>100 %</td>
</tr>
<tr>
<td>Net5</td>
<td>OC-12</td>
<td>95.1 %</td>
</tr>
</tbody>
</table>

5. IP-layer restoration with p-cycles

5.1. The Case for IP-layer restoration

The dominant characteristic of restoration in the physical layer, whether by ring, mesh, or p-cycles, is that pre-failure transmission capacity is directly replaced with equal-bandwidth transmission path substitutions. Because of this “direct bandwidth replacement” model (re-routing carrier signals as opposed to traffic flows), the effect on users and client networks is almost negligible; a very short transmission disruption, and an increase of milliseconds in propagation delay.

But the total capacity investment for restorability can still be expensive. Even with span-restorable mesh alternatives 60-80% physical redundancy levels are typical. In addition, node failures in a service layer can in general only be dealt with by actions of peer-level node elements. For example, no reconfiguration of underlying WDM light-paths in the physical layer can address the failure of a network interface card on a router. Affected IP traffic flows can only be restored by some form of dynamic routing action in router layer itself.

There are also network-operating contexts where physical layer restoration is not technically or economically an option. For instance, a wide-area or metro area Internet backbone operator may have a logical network comprised in part or wholly from carrier signals (STSn’s or wavelengths) leased from facilities based operators. Such an operator will tend to be naturally interested in restoration strategies that are within their

---

1. It is stressed that these test results are achieved within the theoretically minimum amounts of feasible spare capacity. A real survivable network would generally have more margin of spare capacity.
own sphere of control, i.e., their own logical IP transport network, either because control (or even knowledge of) the true physical layer is not available to them and / or because it may be more economic to restore in their own IP logical environment than lease signal paths with a premium for assured physical layer restorability.

A further advantage of restoration at the IP layer is (as with ATM [15,16]) that there need not be a distinction between working and ‘spare’ capacity. Extra capacity allocations still have to be engineered to ensure restorability while retaining congestion-delay performance assurances, but in non-failure times all such capacity is accessible for improved working service performance. And at the moment of a restoration event all unutilized capacity is available. This is in contrast to WDM or SONET restoration capacity planning where discrete separate assignments of working and spare capacity is required. Thus, there is a rationale for considering restoration techniques at both WDM and IP layers. (In particular, the combination of WDM layer restoration for span cuts, and IP layer restoration for node loss may be an effective combination for IP-over-WDM networks.)

5.2. p-Cycles and Existing IP-layer “Restoration”

Today’s IP networks are already restorable in the sense that OSPF, BGP or RIP routing protocols will, through dissemination of link-state and route advertisements, eventually update the routings network-wide to compensate for failure [17, 18, 19]. In terms of techniques usually considered for transport restoration, however, these are slow processes, and the methods have no direct regard for capacity congestion effects that may arise from the routing changes.

The initial approach to use p-cycle ideas for more rapid IP network restoration, covered in this section, is to use p-cycles in much the same logical way that they would be used for SONET or WDM networks, but adapted to the IP router environment. In the IP context of both this section and Section 3, the p-cycles are envisaged as the “fast” part of a “fast plus slow” overall recovery process. What we mean is that the ordinary routing update protocols (today OSPF, BGP for instance) will still proceed to develop routing updates as usual. But in the interval of real time before the update has completed, p-cycles will be in use to prevent the loss of packets. When the slower routing update process converges, the affected packet flows will return to normal routing procedures and traffic on the p-cycles will drop to zero. Thus the IP p-cycle mechanism can be viewed as an immediate fast-acting but temporary protective measure which secures network traffic while the routing s adapt to the failure.

5.3. Routing of IP Packets

Pure IP is inherently connectionless. A packet is forwarded from a router in response to an appropriate routing entry for the packet’s destination address. Based on the destination address or subnet an entry in the routing points to the local port of egress to the next router towards the destination. In this manner, packets are routed, from source to destination, on a hop by hop basis following a series of local router lookups. Each node’s routing is established by the network wide execution of a routing protocol.

For each router, the protocol determines a set of least “cost” routes from the router to each destination. The route cost is determined by the sum of the individual costs of the links used to form the route. The link costs may be determined by the link’s delay, the link’s monetary cost to use, or commonly, by the inverse of the link’s available capacity, or any other measure assigned by a sysadmin. Each entry, for a given destination, contains fields for the next router along the determined least cost route (or the connecting link to the next router), and the associated cost of the route from the router to the destination. By accessing its s, a router can rapidly determine the next router to which to forward a packet along its least cost route. This basic “forwarding engine” function of a router can be summarized as:

- inspect IP address,
- enter routing at match to this address (or subnet mask address),
- forward packet out port indicated in the routing.

It is within this fast -lookup forwarding environment that virtual p-cycles can be established with a small number of reserved IP addresses or any other kind of IP tunnel, tag, or label-switching technique.

5.4. p-Cycle Implementation in an IP Network

In an IP network there is no identifiable spare capacity that is so-designated and reserved for restoration. IP spans simply have a high or low utilization relative to their current installed “pipe” bandwidth. It is therefore not initially clear how to exploit the p-cycle concept with its pre-configured circuit-like logical structures of protection bandwidth. To do so we need to create some kind of virtual circuit construct within an IP router environment.

Although IP is far more widely used than ATM, ATM does have a number of useful properties and concepts which are being adapted (or independently reinvented) by the IP standards community. Primary amongst these are quality of service aspects (QoS) and, relevant to our proposal, point-to-point connections using virtual circuit-like IP tunneling or label switching methods. For example, multi-protocol label switching (MPLS) [20] is essentially a vehicle for forming VCs (in ATM parlance) within an IP environment, mainly for the purpose of shifting the routing of IP packets from the core of the network to the edge. Irrespective of the original motivation for MPLS or other tunnelling, label switching or tag switching proposals, it is possible to realize p-cycles using any of them. Even in pure IP, one could realize virtual p-cycles with a set of corresponding routing entries for a reserved or otherwise unused IP addresses, set aside to define the desired p-cycles through a form of IP tunneling. Thus, there is a range of technical means to effectively produce a virtual circuit (VC) construct in an IP environment. The importance of VCs to our problem is that once a packet is entered onto
a VC, its subsequent routing is completely pre-defined by the label switching sequence that defines the VC. In addition (as in ATM) bandwidth is consumed by a VC only when traffic is being carried. When there is no traffic for a VC it employs only logical resources such as tag/label allocations. Similarly a p-cycle will consume bandwidth only when used in failure recovery, since this is the only time that it carries traffic. The additional traffic, introduced by the p-cycles’ re-routing of packets during restoration must and can, however, be taken into consideration in the capacity design of the network so as to strictly limit (at least by design) the maximum congestion that could arise from restoration-related redirection of IP flows.

5.5. IP Span Restoration with p-Cycles

The first class of failure for which p-cycles can offer restoration is the loss of a logical IP span between a pair of adjacent routers. This could be caused by a failure at the physical/transport layer, such as a cable cut, or by the failure of an interface card. For physical layer failures, the failure will appear in the IP layer only if there is no physical layer restoration mechanism or its capability is exceeded for some reason. For example, a span failure occurring within an OSPR WDM ring should be restored in about 50 ms (i.e., at least as fast as a SONET BLSR). This time is far less than any IP routing protocol time outs, and so the failure and restoration event will not even be observed at the IP layer. In general therefore, WDM physical layer restoration of span cuts remains a very fast and effective line of defense against span failure and may be preferred in practise over router-based recovery against physical span cuts. Thus we intend our explanation of router-based restoration against span cuts to be seen as providing an additional option for IP network operators, rather than being necessarily advocated over physical layer restoration for span cuts if that option exists. (For node restoration, to follow, the logic is quite different however.)

This said, p-cycle restoration of logical span failures in an IP network is envisaged as follows: When a span failure has been detected, the router ports which terminate the failed span will be marked as dead (and the usual link-state advertisement update process will be triggered). Until a global routing update is effected, any packet whose next hop, as indicated by the normal routing entry for the packet’s destination address, would have been directed into the dead port, is, instead, deflected onto a p-cycle which has been assigned to protect the link. “Deflection onto a p-cycle” occurs by encapsulating the original IP packet in a ”p-cycle packet” and re-entering the same routing. When re-entering the routing the encapsulation IP address matches a surviving port where a virtual p-cycle has been previously established. The packet then travels through the p-cycle following either label switching or routing entries at other nodes, that have been pre-established to define the logical p-cycle, until it arrives at the router on the other side of the dead link. At this point, the original IP packet is removed from the encapsulating packet and is routed normally towards its final destination. The node that de-capsulates the detoured packet knows to do so because it has a local routing entry pointing to a non-failed outgoing port for the original IP address in the p-cycle packet. In the case of span failure specifically, a less general mechanism than this is to rely on knowledge of the neighbour node ID which was adjacent to it prior the span failure and a locally available list of p-cycle addresses that contain that node. In the latter approach just knowing: (a) lost link was to node xyz, and, (b) xyz is a peer with me on p-cycle R is enough information to detect that the node should do the decapsulation and continue the original IP packet on its normal routing.

An example of this process is given in Figure 5.

![Figure 5. P-Cycle Recovery of an IP Packet from a Span Failure](image)

The example shows a span failure, X, its associated p-cycle, and routers, A and B which were adjacent (pre-failure) through the span X. In Figure 5(a), an IP packet is arrives at router A with a next hop, indicated by the normal routing entry, that would direct it into the failed span X; In (b) router A, seeing that the normal routing entry now instructs it to route into a failed port, instead encapsulates the packet in a p-cycle packet with known IP address for the local p-cycle that accesses node B and re-enters the routing with the encapsulated packet. There a standing p-cycle routing entry points to the respective port for anything arriving on the p-cycle label or IP address. The p-cycle packets traverse all intervening routers along the p-cycle, as in (c), until, at (d), it arrives at the other end router, B, where the original IP packet is de-capsulated. Router B knows to do this since the packet bears the encapsulating address of a p-cycle on which node B is present and there is currently a local port alarm on the IP span to neighbour B. Finally, at event (e), the IP packet is routed normally towards its final destination.

There are, as mentioned, two classes of span failure which a p-cycle can restore: failures which occur on a
span of the \( p \)-cycle itself, and straddling failures which occur off the \( p \)-cycle, leaving it intact. The example of Figure 5 was for a straddling failure relationship. Figure 1 includes an on-cycle failure case as well. The local nodal action for re-routing into and from the \( p \)-cycle at the two end nodes is identical in both cases once the local routing \( s \) are set up to contain the appropriate additional \( p \)-cycle routing entries. Straddling failures, however, allow the restoration traffic to be split (by IP address or flows) in two directions around the failure. It is also possible to split traffic from a single failure among multiple available \( p \)-cycles by affected original IP address. Both these measures distribute the restoration load more evenly reducing possible congestion (or requiring less capacity in design.) Note, finally, that compared to the typical number of entries in a router (now often more than 100,000), \( p \)-cycles require at most 2\( d \) additional entries (one for each direction on the \( p \)-cycle) where \( d \) is the IP span degree of the node.

6. IP \( p \)-cycle Network Design

In a WDM - IP network, lightpaths (contiguous wavelength paths) in the WDM layer will be used to setup logical connections between IP nodes. From the perspective of the IP network, these appear to be direction connections between routers, while in reality a single physical span may carry sections of multiple logical connections between different IP routers. The consequence of this is that the failure of a single physical span can translate into the apparent failure of multiple simultaneous logical connections in the IP layer. This must be taken into consideration when designing a set of IP network \( p \)-cycles, so that any single physical span failure has a controlled or bounded maximum impact on the simultaneous failure of logical links within the same \( p \)-cycle. One approach is to take the mapping of physical to logical failures into account as multiple failure scenarios included directly the design formulation. Alternately one can design such that \( p \)-cycles are “orthogonal” with respect to physical cuts; that is the set of designed \( p \)-cycles are either not disrupted by the same physical failure, or that their simultaneous disruption is taken into account in limiting the restoration use of the respective \( p \)-cycle. The latter case takes into account that a \( p \)-cycle is still functional or useful for a given restoration problem if at least one path remains through the remaining portion(s) of the \( p \)-cycle between the end routers of the protected span. If the physical to logical fault mapping data is actually available it can be incorporated in the MIP which follows: where all physical span failures are enumerated, one adds all multiple failure scenarios to the constraints set.\(^1\) However, there is also a simple way to ensure the effectiveness of \( p \)-cycles under unknown physical to logical fault escalation. That is to restrict IP layer \( p \)-cycles to being formed only from IP spans which traverse a single physical span; i.e., form \( p \)-cycles only using hops between physically adjacent routers, not all possible logical links between routers. In other words in the graph of all IP layer links, one can formulate the design problem to form prospective \( p \)-cycles only out of logical links that have a 1:1 correspondence to a single underlying physical transmission span. Not only does this reduce computational complexity but it ensures that for any single physical span failure, any \( p \)-cycle would suffer, at most, a single logical link failure and, so, would continue to be able to offer at least one restoration path. Results to follow suggest that still very capacity efficient \( p \)-cycle designs are obtainable even with this restriction.

6.1. Capacitated Network Design using IP \( p \)-Cycles

(for Span Restoration)

In the IP context, restorability design needs to consider the convergent flow effects arising from restoration. This is an aspect of restoration design that does not exist for SONET or WDM. In the latter cases every working signal unit is either exactly replaced (100% restorability) or not (< 100% restorability.) In contrast, where stat-muxed packet or cell flows are being redirected upon restoration, one can take an “over-subscription” based design approach to control the worst-case simultaneously imposed flows on any span during a restoration scenario. In the following formulation we do this, drawing from prior work on over-subscription based ATM backup VP capacity planning [16]. The aim is to determine a set of IP \( p \)-cycles that minimizes the worst-case over-subscription factor on any span, over all failure scenarios. This allows the network designer to accept a slightly lowered (but an assured worst-case) QoS during a restored network state in return for economic savings, because less capacity is required to provision the network, compared to the corresponding 100% restorable SONET or WDM network. This is also an improvement over allowing ordinary IP routing protocols to be used solely for restoration since they do not directly take capacity / congestion effects into direct account at all.

The following MIP allows the relationship between restoration-induced oversubscription performance and corresponding capacity requirements of a network can be studied, dependent on the number of \( p \)-cycles the user allows in the design.

\[
\text{minimize} \quad \eta_M \quad \text{ (EQ 5)}
\]

\[
\text{s.t.} \quad \sum_{j=1}^{N_C} \delta_j \leq M \quad \text{ (EQ 6)}
\]

\[
\alpha_{i,j} \leq \delta_j \quad \forall i = 1 \ldots N_S, j = 1 \ldots N_C \quad \text{ (EQ 7)}
\]

\[
\sum_{i=1}^{N_S} \alpha_{i,j} \geq \delta_j \quad \forall j = 1 \ldots N_C \quad \text{ (EQ 8)}
\]

\[
\sum_{j=1}^{N_C} \alpha_{i,j} = 1 \quad \forall i = 1 \ldots N_S \quad \text{ (EQ 9)}
\]

---

\(^1\) The set of all physical single-span failures is in general a repeatable and defined basis on which to conduct research studies. There is always a “failure scenarios” constraint system to which real-world knowledge of the correlated multiple failure data can be added.
Table 5 for a test network with 15 nodes, shown in Fig-

The objective function, $\eta_M$, is the maximum over-

The number of spans in the network, $\delta_j$ is a decision

The number of cycles in the master set of

The maximum desirable number to administer.

The formulation is being challenged to minimize

For these test cases the network was provisioned

Table 5: Results with p-cycle network design MIP

<table>
<thead>
<tr>
<th># p-cycles allowed in design</th>
<th>Max restoration-induced over-subscription</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>infeasible</td>
</tr>
<tr>
<td>5</td>
<td>1.2</td>
</tr>
<tr>
<td>10</td>
<td>1.045</td>
</tr>
<tr>
<td>15</td>
<td>1.019</td>
</tr>
</tbody>
</table>

FIGURE 7. Optimal set of five p-cycles for 100% IP
Span Restorability with min peak over-subscription

For these test cases the network was provisioned

Table 5 shows the maximum oversubscription 

1. This parameter may be used to reflect an assessment of how many 

logical p-cycles is the maximum desirable number to administer.
7. Node-encircling p-cycles for Router Protection

Unlike most experience with SONET or WDM transmission networks IP networks reportedly suffer “node failures” as frequently or even more often than span outages. Router re-starts, due to the application of software patches/upgrades or crashes apparently generate the majority of router outages. In principle, existing IP routing protocols disseminate the news of the routers disappearance through link state advertisements by all adjacent nodes of the failed node, and the network as a whole converges to a revised routing plan. In practise, however, the volume of link state update flooding messages, the time required for complete reconvergence, and the resulting congestion effects could all be improved upon by a much faster and more localized response to router outage. Note of course that when we say “node restoration” it is really only recovery of pre-failure transiting flows that we are considering. Traffic terminating at the failed node itself is inherently unrecoverable by any type of network-level re-routing.

Adapting p-cycles to the restoration of IP routers involves extension of the basic concept to that of node-encircling p-cycles. In a node failure a surviving neighbour node knows only that the next surviving node on the original route of the affected packets, after the failed node, must have been one of the other surviving neighbours of the failed node. But it does not know which other neighbour nodes of the failed node the pre-failure route had transited. (To know this it would have to have access to the failed node’s routing table).

A node-encircling p-cycle can, however, cope with this aspect of a router failure by providing an alternate path amongst all of the routers which are adjacent to the failed router. Thus, each node encircling p-cycle can provide a readily available replacement detour path for up to n(n-1)/2 router-pairs, where n is the number of routers adjacent to any node to be considered as a prospective failure node. Thus, for a p-cycle to provide restoration for all of the pre-failure flows affected by a router failure, it must contain all of the routers that were adjacent to the failed router, but not the failed router itself.

The idea is that a node-encircling p-cycle constitutes a kind of perimeter highway or control volume which is assumed to be intersected by all (transiting) flows that may be affected by the given node’s failure. By virtue of this property it is also in a position to be used as a substituting detour between any or all node pairs that may have been exchanging pre-failure flows through the lost node. It must contain all adjacent routers as, otherwise, it cannot substitute routes for all of the possible pre-failure flows that traversed them via the failure node. And the p-cycle must not contain the router it is protecting, so that it is not itself disrupted when the router fails. These are the properties of what we call a “node-encircling p-cycle.”

7.1. Types of Node-Encircling p-cycles

A p-cycle, which protectively encircles a node, must be constructed within the sub-graph that results when the protected node (and all its incident links) is itself removed from the network. It may or may not be possible to form a simple cycle within the resulting sub-network (a simple cycle is defined as one which crosses each node only once.) There is however, always a logically encircling p-cycle construct that is possible if two special considerations are dealt with and, assuming only that all pre-failure graphs that are two-connected. Figure 8 gives an example of simple and non-simple p-cycles which can result when protecting routers. The first is a simple cycle, where the removal of the protected node does not disrupt the overall two-connectedness of the resulting network. In such cases the node encircling p-cycle is quite visually apparent and easy to find. In the second example, removal of the protected node results in a singly connected remaining network. The degree-1 node can only be included in the encircling p-cycle through a segment through which the cycle passes twice\(^1\). The last example results when removal of the protected node creates a sub-network with a bridge node; that is, a node whose removal would disconnect the graph. Here the logically encircling p-cycle is formed as a figure “8”, as it is forced to pass twice through the bridge node in the surviving graph. Note that in any of these cases a node-encircling p-cycle may have to visit nodes that are non-adjacent to the protected node, in order to form a cycle that does include all its adjacent nodes.

In the worst case, a network of N nodes would be made fully restorable against any single router outage by establishment of N p-cycles in the routing tables network-wide. Each node would have a logically encircling p-cycle established for its recovery through the set of its immediately adjacent nodes. There are several methods under study to establish and maintain this set of p-cycles as networks undergo growth and rearrangements, including self-organizing procedures. The immediate focus, however, is to explain how they operate in real time.

7.2. Node recovery p-cycle re-routing mechanism

First, let us briefly consider rapid detection of a router failure. Neighbour nodes may see only a disappearance of packet flows, rather than a local hardware alarm as they would for a span failure. Conventional it will take four missing “hello” packets (at 10 sec intervals) for neighbours to detect the router loss. Much

---

1. or closed at the bridge node with a relay interface to forward on packets to/from any nodes in the surviving linear segment.
faster strategies for neighbour node failure detection are easily conceived however; one is to opportunistically insert “idle-fill (hello)” packets in any outgoing port at any time the link is not in use with traffic, combined with detection of a negative spike in the first derivative of packet arrival rates. The composite fast detection scheme would then be to note either the sudden cessation of idle-fill (hello) packets during times of non-peak utilization, or, when link utilization is very high (starrying out opportunistic hellos), to note any sudden total cessation of packet arrivals. Using this method, far less than 40 seconds would be required to reliably detect a failed router by a neighbour node regardless of link utilization level.

Using this, or other fast failure detection schemes, to detect a dead neighbour, each adjacent surviving router will mark the port to the failed router as dead. Subsequently, packet arrivals which would normally be sent toward the failed router are forwarded within the local node to a p-cycle handler which encapsulates the original IP packet in a p-cycle packet, as described above for span restoration. Re-entering the routing with the encapsulated packet has the effect of injecting the encapsulated packet into the respective p-cycle. To support node recovery the p-cycle packet contains a P-Cycle ID field, an Original Route Cost field, a Destination Address field, and, as payload, the original IP packet (Figure 9). The P-Cycle ID field contains a unique identifier for the p-cycle to which the packet belongs, the Destination Address field contains the IP address of the packet’s destination, and the Original Route Cost (ORC) contains the cost of the route the packet would have used prior to failure, as indicated by the local routing table entry for its destination address. As will be explained, the cost field is necessary to determine when it is “safe” (i.e., assured to be loop-free) to remove the packet from the p-cycle, and continue routing it normally to its destination.

As the encapsulating packet travels along the p-cycle, each router tests the packet (to be described) to determine if it should remove it from the p-cycle, decapsulate it, and continue it along the remainder of its normal route. If the test fails, the router forwards the packet along the p-cycle (this is the normal routing entry for packets arriving on the p-cycle address); if the test is met, the original packet is extracted from the p-cycle packet, and forwarded normally from that node using the local routing entry for the original IP address of the decapsulated packet.

The test which each router applies to any packet arriving with a p-cycle IP address is as follows: the router enters its local routing table using the Destination Address field of the p-cycle packet. If there is no router entry for that address, it does nothing and relays the packet on along the p-cycle. If there is a route entry that points to a currently intact port, the router then considers the route cost in the local routing table, against the ORC in the p-cycle packet. If the local route cost entry is less than the ORC it will decapsulate the original IP packet and forward it accordingly. If the local route cost is greater than or equal to the ORC, or the port indicated by the entry is non-functional, the router continues relaying the encapsulated packet along the p-cycle.

These rules ensure the packet will only be returned to normal IP forwarding on an existing routing that is closer to its destination than its entry point into the p-cycle. Without this mechanism to detect the first appropriate de-capsulation node, a packet could get in a loop where it would continuously be introduced into a p-cycle at one point, be removed from the p-cycle at another point, and then be routed normally back to the first point where it could re-enter the p-cycle. In other words we ensure that the IP packet is re-introduced to the network only at a point which is closer to its destination (in the sense of the network wide routing tables established by OSPF or BGP) than where it originally entered the p-cycle. After this, the packet will continue to move away from the failure point as it is routed towards its destination, and there will be no danger of it re-entering the p-cycle.

Figure 10 gives an example of an IP packet being automatically detoured around a router failure, then restored to its original (or in general a preferred) continuing route. For the example, all spans have a cost of 1, except two links which have a cost of 10 to permit illustration of the points above. In Fig. 10(a) a packet follows its normal route to its destination. The packet’s pre-failure route has a cost of 3. After being disrupted by a router failure, in (b), the IP packet is encapsulated within a p-cycle packet (with original route cost of 3) and is injected into the p-cycle. The p-cycle packet is relayed along the p-cycle until it reaches a router at (c). The router compares the original route cost in the p-cycle packet (3) to the route cost from its routing (11) and since the local cost is not less than the original cost it allows the p-cycle packet to continue. At (d), the p-cycle packet reaches another router where the same cost comparison is performed. However, here the local cost of 1 is less than the original cost of 3; therefore, the router unencapsulates the IP packet and routes it normally. In (e), the packet arrives at its destination.

7.3. Network Design for Node-encircling p-Cycles

This section describes a preliminary design heuristic for node-recovery based on node-encircling virtual p-cycles. The heuristic itself is relatively simple and generates a protecting p-cycle for each network node. It

<table>
<thead>
<tr>
<th>P-Cycle ID</th>
<th>Destination Address</th>
<th>Original Route Cost</th>
<th>IP Packet</th>
</tr>
</thead>
</table>

FIGURE 9. P-Cycle Packet Format

FIGURE 10. p-cycle re-direction of flows upon router failure
operators by splitting the network into sub-regions within which simple cycles can be are possible and merging these together to form a final, possibly complex, cycle which has the desired p-cycle properties; it covers the protected nodes adjacent nodes, and does not touch on the protected node itself.

For each node to be protected, the first step is to mark each node which is adjacent to the protected node. Next, in the sub-network that remains when the protected node and all incident links are removed, all bridge nodes are discovered along with the associated sub-graphs that are “hinged” at the bridge nodes. The procedure in the next paragraph is performed within the entire sub-network if there are no bridge nodes, otherwise in each of the subnetworks arising from the cut at any bridge node.

A cycle is then found within each of these sub-graphs such that the cycle traverses all of the previously marked nodes and sub-graph boundary bridge nodes which the sub-graph may contain. All the cycles found will be simple cycles, as in Figure 8, except for the case where the resulting subgraph is a simple segment; for this case, the cycle will be “flattened” and pass through the segment twice.

The cycles from all the sub-graphs (in the case of bridge nodes) are then merged to form the node-protecting p-cycle; if the network did turn out to have bridge nodes after failure node removal, the resulting p-cycle will not be simple.

Figure 11 shows an example set of node-encircling p-cycles which result when this algorithm is run in the network from the previous sections.

FIGURE 11. The first five node-encircling p-Cycles for the 15-node test network.

8. Concluding Discussion

This paper has recapped the concept of p-cycles as so far developed by the authors and proposed extensions into the IP domain to provide for faster restoration of both logical span and router (node) failures. The discrete connection-oriented environment of Sonet or WDM was used initially to demonstrate and explain that for span restoration p-cycle based networks can be as efficient as span-restorable mesh networks, but retain ring-like switching speeds because only two nodes have any real-time cross-connection work to do and, as in rings, exactly which switchovers to effect is known before the onset of failure. Two types of virtual p-cycles, were then considered for IP span restoration or IP node restoration via node-encircling p-cycles. These can be formed as cyclic virtual circuits which are quickly and directly accessible as additional routing entries to provide alternate routings to packets in the event of failure.

References