16 DESIGN OF SURVIVABLE NETWORKS BASED ON P-CYCLES

Wayne D. Grover, John Doucette, Adil Kodian,
Dion Leung, Anthony Sack, Matthieu Clouqueur and Gangxiang Shen

TRLabs and University of Alberta
Edmonton, Alberta, Canada.
grover@trlabs.ca

Abstract: *p*-Cycles are a recently discovered and promising new paradigm for survivable networking. *p*-Cycles simultaneously provide the switching speed and simplicity of rings with the much greater capacity-efficiency and flexibility for reconfiguration of a mesh network. *p*-Cycles also permit shortest-path routing of working paths (as opposed to ring-constrained working path routing), which adds further to network capacity efficiency. Operationally *p*-cycles are similar to BLSRs in that, upon failure, switching actions are required at only two nodes and both those nodes are fully pre-planned as to the actions that are required for any failure detected at their sites. With the optimization models in this chapter, entire survivable transport networks can be easily designed with essentially the same spare to working capacity (redundancy) ratios as optimized span-restorable mesh networks. *p*-Cycles thus bridge the ring versus mesh debate that dominated work in survivable networks through the 1990s and provide the best of both worlds: *the efficiency of mesh with the speed of rings*.
Keywords: *p*-Cycles, survivable networking, optimization, capacity design, node protection, Multiple Quality of Protection.

16.1 INTRODUCTION

16.1.1 Span-Protecting p-Cycles

The operation of a *p*-cycle is portrayed in Figure 16.1. A single unit-capacity *p*-cycle, as in Figure 16.1(a), is a closed path composed of one spare channel on each span it crosses. When a failure occurs on a span covered by the cycle, the *p*-cycle provides one protection path for the failed span, as shown in Figure 16.1(b). In this aspect, *p*-cycles operate like a unit-capacity bi-directional line switched ring (BLSR). But *p*-cycles also protect so-called *straddling spans*, which are spans that have end

nodes on the cycle but are not themselves on the cycle, as shown in Figure 16.1(c). The significance is that, because the p-cycle itself remains intact when a straddling span fails as in Figure 16.1(c), it provides *two* protection paths for each straddling span failure scenario, and straddling spans themselves require no spare capacity. This apparently minor difference actually has a great impact on the capacity requirements of p-cycle networks. p-Cycle network designs are in many cases exactly as efficient (or within a few percent) of the capacity efficiency of a span-restorable mesh network (Grover and Stamatelakis, 1998; Stamatelakis and Grover, 2000b).

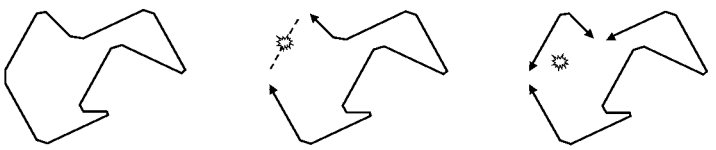

Figure 16.1 An illustration of basic p-cycle concepts. (Grover and Stamatelakis (1998))

16.1.2 Node-Encircling p-Cycles

For a span failure, it is intrinsically possible (and often the goal) to achieve 100% restoration of affected demands by network re-routing. But when a *node* fails, no method of network-level re-configuration or re-routing can restore the originating or terminating demands at the failed node. Thus "node restoration" is really a misnomer. It is only pre-failure transiting flows through a node that can be restored by any type of network-level response. Source-sink flows at the failed node itself are inherently unrestorable by network-level re-routing. Therefore span restoration and protection of transiting flows upon a node failure have inherently different target recovery goals. Perhaps counter-intuitively as a result, 100% of all transiting paths can be protected against node failures within the spare capacity required for 100% span-failure restorability because fewer affected demands are actually considered in the restorability target.

Node-encircling p-cycles (Stamatelakis and Grover, 2000a) provide an efficient strategy for protecting such transiting flows at every node, especially when applied in the MPLS layer to protect against router failure. In this context a set of virtual MPLS-defined node-encircling p-cycles can complement an underlying set of physical-layer span-protecting p-cycles using the same physical capacity. A node-encircling p-cycle provides an alternate path amongst all of the nodes that are adjacent to the failed node. To do this a node-encircling p-cycle must contain all of the nodes that were adjacent to the failed node, but not the failed node itself. This constitutes a kind of "perimeter fence" which is assured to be intersected at ingress and egress by all transiting flows that may be affected by the given node's failure. Such a p-cycle must therefore be constructed within the sub-graph that results when the protected node is itself removed from the network, and it may or may not be possible to form a simple cycle within the resulting sub-network. There is, however, always a logically encircling cycle for

each node in any two-connected (pre-failure) graph. Stamatelakis and Grover (2000a) provides details of the real time rerouting to and from node encircling p-cycles and how node failures are distinguished from span failures. Section 16.4 of this chapter treats node-encircling p-cycle network optimization.

16.1.3 Path-Segment or Flow-Protecting p-Cycles

A third type of p-cycle provides a path-oriented correspondent to basic p-cycles, which are span protecting. This is somewhat analogous to span-restorable and path-restorable mesh networks. So-called *path-segment* or *flow-protecting* p-cycles generalize straddling spans and on-cycle spans to the protection of contiguous flows of demand over path segments that may contain multiple-spans and nodes. The protected path segments do not have to be the whole path end-to-end as taken by a demand, but rather any portion of the entire path. Although they would yield even greater capacity efficiency, whole networks based on flow p-cycles would be fairly complex to operate. However, selective use of just one or a few flow p-cycles in conjunction with other protection schemes may be attractive. One such use is to support transparent optical transport of express flows through a regional network. Another is to use a flow p-cycle around the perimeter of an autonomous system domain to provide a single unified scheme for protecting all flows that completely transit the domain. The idea of using these two types of p-cycles in a network is presented in Shen and Grover (2003b;a) as well as the related design and operational theory for flow p-cycles, but we do not treat them further in this chapter.

16.1.4 Scope

In this chapter we emphasize the basic logical design problems for p-cycles. The extra complications on these models that pertain for optical networks in which precise wavelength assignment is explicitly modeled in the design problem is omitted. The models given thus pertain directly to a SONET-type transport network where full channel interchange is available at each node, or to an optical network with full wavelength conversion at each node (i.e., an OEO network), or to an optical network that has a sufficient number of wavelength conversion points so that the level of wavelength blocking is simply negligible. The fundamental problems of network architecture, design and evolution are issues of getting the capacity, topology, protection and working routing right. Given the right capacities and architecture in basic planning, the more operational problem of detailed wavelength assignment is essentially always solvable as a sub-problem with dozens of algorithms now available. Wavelength assignment without blocking due to "color mismatch" also fundamentally becomes easier as the number of channels available on each span increases (an implication of basic traffic theory). On the other hand, explicit representation of the decision variables for wavelength assignment would make most of the following ILP models virtually intractable above small sizes, unnecessarily removing one of the network planner and researchers most valuable tools (i.e. ILP). For these reasons, we treat the problems covered in this chapter as capacity allocation problems without individual wavelength assignment aspects.

For readers interested in aspects of p-cycles that are outside the current scope, Grover and Stamatelakis (1998) and Stamatelakis and Grover (2000b) are suggested as primary sources. In addition, Stamatelakis and Grover (2000a) presents the application of p-cycles in the IP or MPLS network layers and node-encircling p-cycles, and the use of p-cycles in WDM networks is addressed in Schupke et al. (2002). The chapter on p-cycles in Grover (2003) is also a comprehensive source. For an archive of all p-cycle literature known to us, we suggest our website A.Sack and W.D. Grover (2005).

16.1.5 Basic Notation and Models for p-Cycle Network Design

To define the basic p-cycle network design problem, we first introduce the following notation. This initial set of symbols and definitions applies for the remainder of the chapter, and will be supplemented with additional definitions as applicable for the more advanced design problems in subsequent sections.

- **Sets:**
  - S is the set of spans in the network, and is usually indexed by i for a failure span, and j for surviving spans.
  - M is the set of available modular transmission system capacities, and is indexed by m.
  - P is the set of (simple) cycles of the graph eligible for formation of p-cycles, and is usually indexed by k. This set of cycles is determined by a pre-processing method, and eligibility may be limited in some way (e.g., circumference).
- **Input Parameters:**
  - w_i is the number of working channels (or capacity units) on span i that require protection.
  - $x_{i,k} \in \{0, 1, 2\}$ encodes the number of protection relationships provided to span i by a unit-sized copy of p-cycle k. $x_{i,k} = 2$ if span i straddles cycle k, $x_{i,k} = 1$ if span i is on cycle k, and $x_{i,k} = 0$ in all other cases.
  - $\delta_{j,k} \in \{0, 1\}$ encodes the spans on a p-cycle. $\delta_{j,k} = 1$ if cycle k includes span j (i.e., if $x_{i,k} = 1$), and $\delta_{j,k} = 0$ if it does not (i.e., $x_{i,k} \neq 1$).
  - C_j is the cost of a unit of capacity (i.e., a single channel) placed on span j (in the non-modular or integer-capacity design context). This generally includes considerations of the actual length of the span, the technology employed etc.
  - Z^m is the number of channels provided by a modular transmission system of type m.
  - C_j^m is the cost to install one transmission system or module of type m on span j (also includes the same considerations as C_j, except that now it is per module on each span).

- **Decision Variables:**
  - $s_j \geq 0$ is the integer number of spare channels assigned to span j in the design.
  - $n_k \geq 0$ is the integer number of unit-capacity p-cycles placed on cycle k by the design.
  - t_j^m is the integer number of modules of type m placed on span j in the design.

The basic p-cycle design problem is modeled as an integer linear programming (ILP) formulation for *spare capacity allocation* (SCA). Working demands are already routed (typically via shortest path routing or perhaps some flow-leveling approach) and the problem is to find the optimal set of p-cycles that can fully protect all working capacities, w_i, while minimizing the total investment in spare capacity needed to support those p-cycles. The ILP is formulated as follows:

$$\min \quad \sum_{\forall j \in S} C_j s_j \tag{16.1}$$

$$\text{s.t.} \quad w_i \leq \sum_{\forall k \in P} x_{i,k} n_k \quad \forall i \in S \tag{16.2}$$

$$s_j \geq \sum_{\forall k \in P} \delta_{j,k} n_k \quad \forall j \in S \tag{16.3}$$

The objective we seek to minimize (16.1) is the total cost of spare capacity assignments. Constraint (16.2) asserts that the total number of protection relationships provided for span i by the set of p-cycles chosen must meet or exceed the number of working channels to be protected on span i. Constraints set (16.3) couples those requirements to the objective function by asserting that the spare capacity on each span must be sufficient to implement the set of p-cycles that provide this restorability to each span. Generically, these two constraint systems are often called the *restorability* and *spare capacity constraints*, respectively.

The SCA problem can easily be extended to recognize the modular and non-linear cost nature of the capacity increments of the actually available transmission systems, as in Doucette and Grover (2000). This is done by keeping constraints (16.2) and (16.3), changing the objective function to that in equation (16.4), and adding the new total modular capacity constraint in equation (16.5) as follows:

$$\min \quad \sum_{\forall j \in S} \sum_{\forall m \in M} C_j^m \cdot t_j^m \tag{16.4}$$

$$s.t. w_j + s_j \leq \sum_{\forall m \in M} t_j^m \cdot Z^m \quad \forall j \in S \tag{16.5}$$

It is useful to note that the cost parameters associated with the available set of modular transmission capacities, C_j^m, can follow a significantly non-linear progression versus the corresponding capacity vales, Z^m, allowing economy-of-scale benefits to be captured in the network design (Doucette and Grover, 2000).

16.1.6 *Efficiency of p-Cycles and p-Cycle Network Designs*

In other approaches to survivable network design, where the decision variables typically pertain to restoration path choices or flow assignments to eligible routes, it is hard to identify general figures of merit for individual candidate backup paths or eligible restoration routes. In contrast, p-cycles lend themselves rather handily to compact measures of the potential (or actual) efficiency of individual p-cycles in different network contexts, and we can easily define several efficiency metrics associated with each p-cycle. Grover and Doucette (2002) defines the *topological score* (TS_k) of a candidate cycle k as the number of protection relationships it is capable of providing as a unit capacity p-cycle, and the *a priori efficiency* (AE_k) of a cycle as the ratio of TS_k to the total cost of constructing a unit-capacity copy of a p-cycle on cycle k:

$$TS_k = \sum_{\forall i \in S} x_{i,k} \tag{16.6}$$

$$AE_k = \frac{TS_k}{\sum\limits_{\forall i \in S:\ x_{i,k}=1} C_i} = \frac{\sum\limits_{\forall i \in S} x_{i,k}}{\sum\limits_{\forall i \in S:\ x_{i,k}=1} C_i} = \frac{\sum\limits_{\forall i \in S} x_{i,k}}{\sum\limits_{\forall j \in S} \delta_{j,k} C_j} \tag{16.7}$$

We call AE_k an *a priori efficiency* because it reflects the best *potential* efficiency achievable by the cycle if all its straddlers are fully loaded with two working channels each and its on-cycle spans are loaded with one working channel. But this is calculated prior to obtaining any information about actual working loads, and this may not be the actual utilization efficiency of the p-cycle in any specific design. When the actual working channel quantities protected by the p-cycles are known, a closely related *working-weighted efficiency* measure can be defined (in Section 16.3.3).

The significance of p-cycles having such compact and highly characteristic efficiency measures is two-fold. First, they enhance intuitive understanding about what determines a good p-cycle and help to diagnose effects such as why an unconstrained optimum solution may tend to contain many large cycles, or even Hamiltonians. Second, these efficiency measures can fairly easily be built upon to realize some simple and effective heuristics, at least for the basic p-cycle design problem. In addition, we can use such measures to identify high merit cycles based on real insights about what makes a good p-cycles in the context of a given network design. The AE_k measure and concepts closely related to it are developed further in the discussion of algorithmic p-cycle design heuristics in Section 16.3.

The capacity-efficiency of 100% span-restorable p-cycle designs can be remarkably good in practice. In the special case of a particular class of semi-homogenous networks (probably most applicable for whole fiber-level protection), they can, by construction, actually *reach* the long-recognized lower bound on redundancy of $1/(\bar{d}-1)$ (see Doucette and Grover (2001)), where $\bar{d}$ is the network average nodal degree. A sample solution to SCA follows to illustrate the first claim. Figure 16.2 shows a 13-node, 23-span test network, with $\bar{d} = 3.54$. For simplicity, all span costs are $C_j = 1$, and the w_i values on each span in Figure 16.2(a) are from least-hop routing of a single demand unit between each node pair. In the specific design context of Figure 16.2(a) there are seven individual p-cycles constructed on five unique cycles, illustrated in

Figure 16.2(b)-(f). The design is 100% span-restorable with a logical (channel count) redundancy of 85/158 = 53.8%.

This is a highly efficient design solution that could be used for protection at the lightwave-channel level. But it is also possible with p-cycles to reach even greater efficiency. It was shown in Sack and Grover (2004) that if a single unit-capacity Hamiltonian cycle is used as a p-cycle in which all straddling spans are fully loaded, then we could achieve a 39.4% redundancy in that same network, which is exactly the $(1/(d-1))$ lower bound for generalized span-protection - $(1/(3.54-1) = 39.4\%)$. To see this, consider that in any such case we will have the ability to protect as many as

$$\sum_{\forall j \in S} w_j = |N| + 2(|S| - |N|) = |N| + 2\left(\frac{|N|\bar{d}}{2} - |N|\right) = |N|(\bar{d} - 1)$$

working channels (where N is the set of all nodes in the network), making the overall redundancy exactly equal to

$$\mathcal{R} = \sum_{\forall j \in S} s_j \Big/ \sum_{\forall j \in S} w_j = \frac{|N|}{|N|(\bar{d} - 1)} = \frac{1}{\bar{d} - 1}.$$

Clearly, these are motivating properties for a scheme that retains ring-like switching speed and structural simplicity. Heretofore, redundancies as low as 40% or so for 100% span survivability were only ever even approachable through end-to-end path restoration technologies that are far more complex to design and operate and much slower acting in practice.

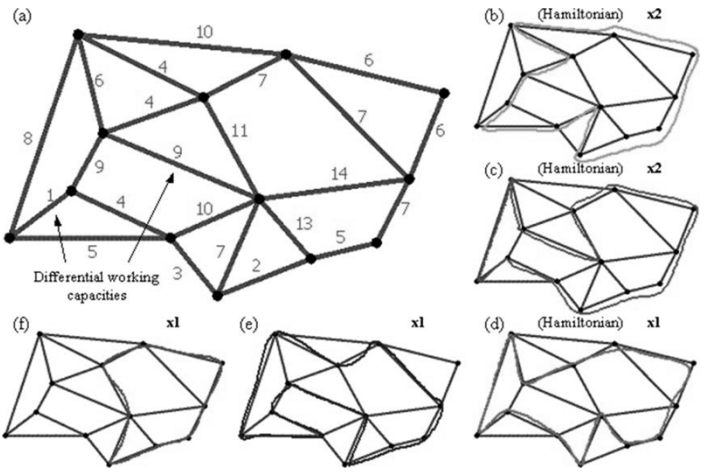

Figure 16.2 A fully restorable p-cycle network.(Sack and Grover (2004))

16.2 JOINT WORKING & SPARE CAPACITY DESIGN

To design a p-cycle network, one can first route the working demands via shortest paths (or by some other means) and then solve the minimum spare capacity allocation (SCA) problem. An alternate strategy is to optimize the choice of working routes in conjunction with the placement of p-cycles and spare capacity, with the goal of minimizing total (working plus spare) capacity. We call this the *joint* capacity allocation (JCA) problem.

The main advantage of joint optimization is that working routing is effectively allowed to deviate from shortest paths onto paths that allow a more efficient use of p-cycles. Allowing the optimization to consider alternate routings of working demands plays an important role in reducing total capacity requirements. One study in Iraschko et al. (1998) showed that total capacity requirements can be reduced by 4% to 27% in a span-restorable network and an average of 7% in a path-restorable network. Capacity savings in a p-cycle network are comparable; work in Grover and Doucette (2002) shows that for the COST 239 network (Schupke et al., 2002; Batchelor et al., 1999), capacity redundancy decreases by 25%, and translates to a 13% reduction in total capacity requirements. Studies on other networks show similar capacity reductions, attributable to JCA over SCA.

The SCA formulation in Section 16.1 can easily be modified to provide the JCA formulation. To do so, the prior w_i input parameters become decision variables, we modify the objective function, and we add new set, parameter, and variable notations as well as two new constraints to ensure the routing of working demands and adequate working capacity to support them. In addition to the notation from Section 16.1.5, we need the following additional notation for the joint problem:

- **Sets:**
  - D is the set of working lightpath demands, and is typically indexed by r.
  - Q^r is the set of all distinct pre-determined eligible routes available to carry the working paths for demand relation r, and is typically indexed by q.
- **Input Parameters:**
  - d^r is the number of lightpath demand units for demand relation r.
  - $\varsigma_i^{r,q} \in \{0,1\}$ is a parameter that encodes working routes. If $\varsigma_i^{r,q} = 1$, working route q used for demand relation r crosses span i. If $\varsigma_i^{r,q} = 0$, working route q used for demand relation r does not cross span i.
- **Decision Variables:**
  - $g^{r,q} \geq 0$ is the amount of working flow assigned to working route q used for demand relation r.
  - $w_i \geq 0$ is the integer amount of working capacity that is placed on span i (this is now a decision variable).

All previous notation from the SCA formulation of the problem in Section 16.1.5 still applies. In order to consider working capacities in addition to spare capacities, we

also make the following change to the objective function:

$$\min \sum_{\forall j \in S} C_j(s_j + w_j) \tag{16.8}$$

Finally, the restorability and spare capacity constraint sets in constraints (16.2) and (16.3) also still apply, as do the following two new constraint sets:

$$\sum_{\forall q \in Q^r} g^{r,q} = d^r \quad \forall r \in D \tag{16.9}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \varsigma_i^{r,q} g^{r,q} = w_i \quad \forall i \in S \tag{16.10}$$

Just as the SCA design problem chooses between eligible cycles, JCA also includes a choice between eligible routes for working demands. Constraint (16.9) is the working routing equivalent to equation (16.2). It ensures that the total working flow assigned to all eligible working routes for demand relation r is sufficient to fully route it. Note that as written, there is nothing in equation (16.9) that prevents a working flow bundle to be split (integrally) over multiple working routes. Equation (16.10) sizes the working capacities on each span in much the same way that equation (16.3) does so for spare capacity. The main structural difference between equations (16.10) and (16.3) is that in equation (16.10), the working flow for each demand relation is applied to each span *at the same time*, hence the double summation. In contrast, when sizing spare capacity, restoration flow is applied separately for each span failure scenario. Also, we use a strict equality here, rather than an inequality as in equation (16.3) for spare capacity. This is because working capacity assignment is equivalent to that required to carry all lightpath demands, which are all routed simultaneously, while spare capacity in equation (16.3) is sized to accommodate individual failure scenarios separately, some of which may require more or less spare capacity than others on any particular span. Note that the SCA design problem can actually be solved using this JCA formulation, by simply providing only a single eligible working route for each demand.

Analyses of working routing resulting from the JCA designs confirm that they are coordinating with p-cycle selections to provide a more balanced and efficient use of protection capacity. In the COST 239 network, working capacity costs increased by 5% in the JCA design relative to the SCA design, while spare capacity decreased by 43% (Grover and Doucette, 2002). This improvement was observed when we only provided all routes up to the 10th shortest eligible working route per O-D pair. Allowing working routes to deviate even further from their shortest paths rarely provides greater reductions in total capacity. Studies in Stidsen and Thomadsen (2004) show that joint optimization provided between 3% and 22% capacity savings versus the SCA designs on several test networks, and in Rajan and Atamturk (2002), capacity efficiency in JCA was found to increase by 30% versus SCA over a variety of test case networks.

16.3 HEURISTIC METHODS FOR THE P-CYCLE NETWORK DESIGN PROBLEM

The ILP methods developed above are quite effective at optimizing p-cycle network designs, and are an essential tool in many research activities, particularly when doing comparative analysis of new survivability mechanisms or design strategies. However, they generally require enumeration of a representative sample of all possible eligible cycles in a network graph (or the full set if strict optimality is required). This approach is suitable in practice for small or medium sized networks, but the number of possible cycles in a graph increases exponentially with the size of the graph. Providing a suitably large subset of eligible cycles for a large network, therefore, becomes difficult and time-consuming, and the number of eligible cycles directly increases the complexity of the ILP itself. We will therefore now address several techniques for dealing with these issues.

16.3.1 Eligible Cycle Pre-Selection Strategies

In Section 16.1.6, topological score, TS_k, and *a priori efficiency*, AE_k, were defined. In terms of these figures of merit, cycles with a large number of straddling spans relative to their size (or cost) are identified as having the highest potential efficiency as p-cycles. We can use this knowledge to pre-select high-merit cycles for population of the set of candidate cycles, P, when solving the SCA or JCA (or other) problems for large networks. Rather than providing all possible distinct cycles to the ILP problem, we can simply rank them by TS_k or AE_k and then provide only a limited number of the top-ranked cycles for representation in the model. Work in Grover and Doucette (2002) showed that this method of limiting eligible cycle sets in the COST 239 network was effective in reducing problem complexity and solution runtimes, while still reaching optimality in the resultant designs, using both SCA and JCA models. In the COST 239 network, there are a total of 3531 possible distinct cycles in the graph, but when given only the 250 top-ranked cycles by the AE_k metric, the SCA problem was able to be solved to within 1% of optimal in slightly more than 1/1000th of the time it took the same problem when all eligible cycles were provided. Similarly, when the JCA problem was provided with only 50 eligible cycles (again they were the top ranked by AE_k), it was solved to within 0.16% of optimal in 1/17th of the solution time of the complete problem. The cycle candidates provided by AE proved to be a more effective metric than TS, and allowed the eligible cycle set to be reduced significantly further while retaining optimality. The main drawback of this method is that it still requires enumeration of every cycle in the network prior to pre-selection in order to properly calculate AE_k or TS_k and rank them.

Work in Sack (2004) tested the hypothesis that using a candidate cycle set that "mimics" the statistical distribution of cycle lengths may permit even fewer cycles to be represented in the ILP problem for SCA or JCA. As an experiment to test this idea, following enumeration of the full set of cycles, the hop-length distribution observed for that set was scaled to reflect the number of cycles desired in the representative sample. Randomly selected cycles from the full set were then used to make up the sampled cycle set using a "bin-filling" approach - cycles of a certain length were only

added if needed to achieve the necessary distribution. The random selection algorithm is expressed by the following pseudo-code:

```
StatisticalMimicry(NumCyclesDesired) {
    Find FullCycleSet (with TotalFullCycles number of elements)
    Find FullNumCyclesLength[l] for each length l in FullCycleSet
    If TotalFullCycles < NumCyclesDesired
        NumCyclesDesired = TotalFullCycles
    For each length l
        StatNumCyclesLength[l] = RoundUpToInteger(FullNumCyclesLength[l]
        * (NumCyclesDesired / TotalFullCycles))
    Do {
        Pick a cycle at random from FullCycleSet (length
        is l)
        If StatNumCyclesLength[l] > 0
            Add cycle to StatCycleSet
            Set length l of that cycle in FullCycleSet to 0
                (so it cannot be used if radomly chosen again)
    } while any value of StatNumCyclesLength[l] >0
    Return StatCycleSet
}
```

Results showed that the average solution cost of several runs of the statistical sampling approach was lower than the cost with either AE_k or a set of short cycles, for three of four test networks where the same limited size of candidate cycle sets (as a percentage of the full set) was applied. Another test showed that the AE_k, shortest cycles, and statistical sampling methods were closer in performance at both very low and very high average nodal degree. Between these extremes, the average of the statistical sampling runs consistently produced lower-cost design solutions than either of the other two methods. Notably, in many cases, at least one of the statistically sampled test runs produced a design solution very close to the baseline solution cost with the full cycle set. Thus, even though the performance of a single run can be highly variable due to the random sampling approach, repeating the process several times can normally provide a very good result.

16.3.2 Other Candidate Cycle-Enumeration Strategies

The above methods are useful for reducing the eligible cycle set provided to the ILP model, but if the need for cycle-enumeration itself is a concern, there are methods to deal with that issue as well. The Straddling Link Algorithm (SLA) was proposed in Zhang and Yang (2002) as a means of enumerating only a very small subset of the possible cycles in a network. For each span in the network, SLA makes two calls to Dijkstra's shortest path algorithm to find the shortest route and the next shortest disjoint route connecting the span's end nodes, not using that span itself. When those two routes are combined end-to-end, the resultant *primary p-cycle* will have the span as a straddler. Although SLA is very fast (there are only $|S|$ such cycles to generate), the cycles produced tend to be quite inefficient and ill suited by themselves for overall capacitated network designs because they usually have only a single straddler and they are small cycles. While the fact that there are only at most $|S|$ primary p-cycles produced by SLA is the basis for its speed, such a small set of eligible cycles is far too limited for an optimization model to form efficient protection relationships.

The SLA procedure was therefore used as a basis for more advanced cycle-generation algorithms in Doucette et al. (2003), where primary p-cycles of a network are transformed into a larger set of higher-efficiency cycles. Three key algorithms are called *SP-Add*, *Expand* and *Grow*. The *SP-Add* algorithm starts with an existing cycle (say a primary p-cycle), and for each of its on-cycle spans, a new cycle is created by replacing the span with the shortest path connecting its end nodes such that the path is node-disjoint from the original cycle as a result. The span that is replaced becomes a straddling span in the new resultant cycle, which will usually have higher AE_k value than the original cycle. The *SP-Add* operation can be expressed by the following pseudo-code:

```
SP-Add(OriginalCycleSet) {
   Initialize NewCycleSet
   For each cycle p in OriginalCycleSet {
      Mark all spans and nodes on cycle p
      For each span i on cycle p {
         Dijkstra(i, unmarked spans/nodes) -> r
         If Dijkstra() returns a route r {
            Let cycle x = cycle p
            Add route r to cycle x
            Remove span i from cycle x
            Add cycle x to NewCycleSet
         }
      }
      Unmark all spans and nodes
   }
   Return NewCycleSet
}
Dijkstra() {
   Find the shortest route r between the end nodes of i using
   unmarked spans and nodes only
   Return route r
}
```

When the *SP-Add* algorithm was applied to the set of primary p-cycles (obtained from SLA), average AE_k values increased 8%-16% and their average on-cycle span coverage increased 36%-44% on the two test case networks in Doucette et al. (2003). The number of cycles generated by *SP-Add* increased from 29 in SLA to 90 for one test network and from 50 to 190 for the other, thereby producing larger and more efficient eligible cycle sets for the SCA and JCA models but in no case enumerating all possible cycles in the network graph.

The *Expand* and *Grow* algorithms are somewhat more complicated than *SP-Add*. In *Expand*, the *SP-Add* algorithm is modified so that after a span has been converted into a straddler, we move on to the next spans in the original cycle seeking yet further route replacements for spans to create ever-expanding candidate cycles. This is done until every span in the original cycle has been visited. Each time a span is replaced, the replacement route must not only be node-disjoint from the original cycle but also from each previous route already added. *Expand* is expressed by the following pseudo-code:

```
Expand(OriginalCycleSet) {
   Initialize NewCycleSet
   For each cycle p in OriginalCycleSet {
      Let cycle p' = cycle p
      Mark all spans and nodes on cycle p
      For each span i on cycle p {
         Dijkstra(i, unmarked spans/nodes) -> r
         If Dijkstra() returns a route r {
            Add route r to cycle p'
            Remove span i from cycle p'
            Mark all spans and nodes on route r
            Add cycle p' to NewCycleSet
         }
      }
      Unmark all spans and nodes
   }
   Return NewCycleSet
}
```

It was found in Doucette et al. (2003) that the new cycles resulting from the *Expand* algorithm are 10%-18% more efficient (by AE_k) than the original cycle set from SLA, and the number of cycles increased from 29 to 96 for one network and from 50 to 197 for the other. The *Grow* algorithm is a further modification of the *Expand* algorithm, where we effectively reset the span counter, i, each time we create a new cycle p'. This is essentially the same as forcing the *Expand* algorithm to recursively call itself each time a new cycle is found and added to the set. The *Grow* algorithm improves average AE_k values of the original cycle set from SLA by 25% and 41% in the two test case networks in Doucette et al. (2003), and the number of cycles was increased to 839 and 2407 respectively. The *Grow* algorithm can be expressed by the following pseudo-code:

```
Grow(OriginalCycleSetA) {
   Initialize SPAddCycleSetB
   SPAddCycleSetB = SP-Add(OriginalCycleSetA)
   Initialize NewCycleSet
   For each cycle p in SPAddCycleSetB {
      Let cycle p' = cycle p
      For each span i on cycle p' {
         Mark all spans and nodes on cycle p'
         Dijkstra(i, unmarked spans/nodes) -> r
         If Dijkstra() returns a route r {
            Add route r to cycle p'
            Remove span i from cycle p'
            Add cycle p' to NewCycleSet
            Restart count of i to first span in p'
         }
         Unmark all spans and nodes
      }
   }
   Add OriginalCycleSetA to NewCycleSet
   Add SPAddCycleSetB to NewCycleSet
}
```

The increases in the number of eligible cycles generated and their average AE_k values are only indications that the cycle sets have the potential to provide low-redundancy protection in a capacitated network. Results in Doucette et al. (2003) showed that for one test network, if the set of 2407 cycles produced by the *Grow* algorithm were provided to an SCA design model, the gap to optimality was 0.5% relative to when a conventional eligible cycle set enumeration method was used to provide the 15000

shortest distinct cycles in the network. Additional strategies for improving the eligible cycle sets provided by *Grow* include supplementing the cycle set with the set of all faces in the graph, using other methods for combining pairs of cycles (see Grover (2003)), or even adding a small set of the shortest distinct cycles (for which efficient depth-first search algorithms can be applied).

16.3.3 Greedy Heuristic Methods

Let us now further consider the *AE* metric and outline a greedy iterative heuristic algorithm to approximate SCA solutions. AE_k represents a candidate cycle k's *potential* efficiency as a p-cycle. Why is it only a potential, however? The reason is that in an actual network design, any given candidate cycle may not find working capacity present on every span it is capable of protecting. Only if sufficient working capacity exists on all of those spans will a p-cycle actually realize *AE* efficiency. Even in the most efficiently designed networks, inspection of the chosen p-cycles will usually show that they exhibit a variety of AE_k values. It is also easy to demonstrate that for a given working capacity arrangement, a cycle with relatively low AE_k might be preferable to one with a higher AE_k (e.g., if one span is very heavily loaded, then a very small cycle with that span as a straddler would suffice). So in the complete problem in the presence of capacitated working demand, finding cycles of high AE_k is only a first step towards finding a combination of p-cycles to collectively protect the network in an efficient manner. In light of these insights, AE_k can be modified to approximate a cycle's *actual efficiency* or *working-weighted efficiency* (E_k^W) based in part on the working capacities of the spans it protects as in Doucette et al. (2003):

$$E_k^W = \frac{\sum_{\forall i \in S} w_i x_{i,k}}{\sum_{\forall i \in S:\ x_{i,k}=1} C_i} = \frac{\sum_{\forall i \in S} w_i x_{i,k}}{\sum_{\forall j \in S} \delta_{j,k} C_j} \tag{16.11}$$

An alternative definition, $E_k'^W$, also appears in Grover (2003):

$$E_k'^W = \frac{\sum_{\forall i \in S} \min(w_i, x_{i,k})}{\sum_{\forall i \in S:\ x_{i,k}=1} C_i} = \frac{\sum_{\forall i \in S} \min(w_i, x_{i,k})}{\sum_{\forall j \in S} \delta_{j,k} C_j} \tag{16.12}$$

This new quantity gives us an indication of a p-cycle's actual suitability in a specific environment of working capacity still remaining to be assigned to a p-cycle for protection. Using E_k^W (or $E_k'^W$) as a basis for identifying efficient cycles, a greedy iterative p-cycle network design algorithm can be easily developed. Given a set of eligible cycles (either through complete enumeration, pre-selection, or some other method), the idea is to iteratively test the placement of eligible cycles to determine their suitability for placement in the network. We first calculate E_k^W for each candidate cycle using the network's initial working capacity values, select the cycle with highest E_k^W, and place one copy of it in the design. Since this newly placed p-cycle provides protection for one unit of working capacity on each on-cycle span and two units on each straddling span, we subtract those amounts from the working capacities on all spans protected

by the cycle. The new working capacities represent the amount of protection still required on each span after placement of the first cycle. The process is then repeated by recalculating E_k^W for each eligible cycle, placing one copy of the cycle with the highest efficiency, and updating the working capacities on all spans protected by that cycle. The algorithm terminates when no further unprotected working capacity remains on any span in the network. We refer to the algorithm as the Capacitated Iterative Design Algorithm (CIDA), and it can be expressed by the following pseudo-code:

```
CIDA() {
   Initialize CycleSet, work[], and CycleUse[]
   CycleSet = EnumerateCycles()
   While work[i] > 0 for all spans i {
      BestCycle = 0
      For each cycle p in CycleSet {
         Calculate E_p^W
         If E_p^W > E_BestCycle^W {
            BestCycle = p
         }
      }
      CycleUse[BestCycle] = CycleUse[BestCycle] + 1
      For each on-cycle span i in BestCycle {
         work[i] = work[i] - 1
      }
      For each straddling span i in BestCycle {
         work[i] = work[i] - 2
      }
   }
   Return CycleSet and CycleUse
}
```

Analyses in Doucette et al. (2003) showed that CIDA produced p-cycle network designs that are within 5% or less of the optimal ILP solutions when given the same set of eligible cycles, but in a small fraction of the runtime of the ILP. Further studies suggest that various tactics such as placing multiple copies of the winning cycle at each iteration can increase the speed of CIDA even further with only minor impact on optimality.

16.3.4 Approaches with Little or No Cycle Enumeration

Another technique for solving the p-cycle design problem is the use of *column generation algorithms* as in Stidsen and Thomadsen (2004) and Rajan and Atamturk (2002) for JCA p-cycle design. Here, the algorithm allows solution of a relaxed ILP model while only *implicitly* also determining the p-cycles to use in the solution. In column generation, new constraints in the ILP model are only generated when needed to improve the objective function. At each iteration of the algorithm in Stidsen and Thomadsen (2004), the eligible working route with the lowest reduced cost is found for each O-D pair, as is the single eligible cycle with the lowest reduced cost. If any of the routes or the cycle has a negative reduced cost, those are added to the ILP. This continues until no such path or cycle with negative reduced costs can be added to the problem. Network design costs and runtimes reported in Stidsen and Thomadsen (2004) and Rajan and Atamturk (2002) with the column generation algorithms compare favorably to the ILP method presented above. Refer to Chvátal (1983) and Winston (1994) for more information on column generation techniques in general.

A fully distributed p-cycle forming process called the *distributed cycle preconfiguration* (DCPC) protocol is developed in Grover and Stamatelakis (1998) as a means of allowing a network to discover and organize its own set of p-cycles so that full protection is provided to every span. DCPC is essentially an adapted version of the SHN protocol Grover et al. (1991), and makes use of statelets that propagate through the network allowing nodes to identify potential cycles and collectively select the most efficient for configuration. The DCPC protocol was initially proposed as an in-situ on-line process to self-organize p-cycles in an existing network. However, it can also be simulated in which case it provides a polynomial time algorithm to find an efficient set of p-cycles.

An ILP model that does not require cycle enumeration is presented in Schupke (2004). Here, a transshipment-like set of constraints is developed so that p-cycles are actually computed within the ILP itself. The main drawback with this method is the increased complexity caused by the large number of variables, and its use is generally limited to networks of 25 nodes or less.

16.4 NODE-ENCIRCLING P-CYCLE DESIGN

Node-encircling p-cycles efficiently provide for rapid protection of transiting flows affected by a node failure. A node-encircling p-cycle must by its very nature, exhibit two important properties. First, it must cross through all nodes topologically adjacent to the node it is intended to protect; otherwise, it cannot capture all of the pre-failure flows that transit the failed node. Second, it cannot contain the node it is protecting; otherwise, the p-cycle itself will be disrupted when the node fails. Node-encircling p-cycles were initially designed for protection against node loss in the IP/MPLS layer, but can also offer protection against the loss of a logical link between a pair of adjacent routers, as well as node and span failures in an optical network. This is explained further in Grover (2003).In this section where we discuss the process of generating node-encircling p-cycles, then describe the node-encircling p-cycle network design problems for node and link failures. We close with an overview of ongoing research in node-encircling p-cycles and optical-layer p-cycles design for minimum over-subscription.

16.4.1 Types of Node-encircling p-Cycles

A node-encircling p-cycle is constructed within the subgraph that results when the protected node (and all its incident links) is removed from the network, as shown conceptually in Figure 16.3. In most cases, a *simple* cycle, one that crosses any node only once, can be formed within that resulting subgraph, as in Figure 16.3(a) and Figure 16.3(b). Such cycles visibly encircle the node they protect. In other cases this may not be possible. However, if the pre-failure network topology is at least two-connected, there will always be at least one *logically* encircling cycle possible if we allow *non-simple* node-encircling p-cycles that cross some node(s) and/or spans(s) more than once, as in Figure 16.3(c) and Figure 16.3(d). As shown in Figure 16.3(c), a cycle also does not necessarily have to physically encircle the protected node, so long as all nodes adjacent to that node are visited.

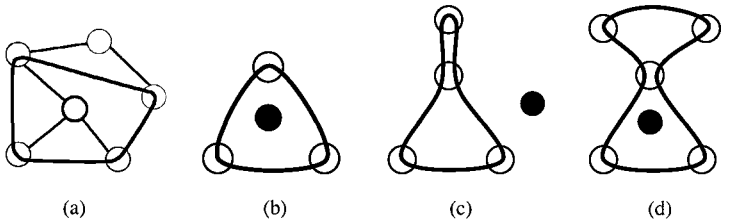

Figure 16.3 Illustrating (a) the basic (and simple) node-encircling p-cycle concept, (b) a simple node-encircling p-cycle, and (c)-(d) two non-simple node-encircling p-cycles. (Stamatelakis and Grover (2000a))

With a simple cycle, 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 visually apparent and easy to find. In Figure 16.3(c) however, removal of the protected node results in a subgraph with a *stub node* (i.e., a degree-1 node) that can only be included in the node-encircling p-cycle through a segment that the cycle must cross twice. The node-encircling p-cycle in Figure 16.3(d) results when removal of the protected node creates a subgraph with a *bridge node* (i.e., a node whose removal would disconnect the graph entirely). Here a logically encircling p-cycle still exists but takes on the form of a figure eight, as it is forced to visit the bridge node twice. In the worst case, a network of N nodes would be fully protected against any single node outage by establishment of N node-encircling p-cycles. Each node would have a logically encircling p-cycle established for protection of its transiting flows through the set of its immediately adjacent nodes.

16.4.2 Generating Node-Encircling p-Cycles

A simple method of generating a set of node-encircling p-cycles is based on decomposition of the network into its bi-connected components following removal of the protected node and its incident edges. A preliminary step is to generate a file of all distinct simple cycles of the graph. If the network remains as a single bi-connected component, at least one simple node-encircling p-cycle exists and can be found by filtering the file of simple cycles to find the least-hop cycle that includes all of the potentially failed node's neighbors. Thus, if the failed node n has the set of neighbors $N_n = \{x, y, z, \dots\}$, then the existence of a simple node-encircling p-cycle c'_n requires $N_n \in \{c'_n\} \in N - n$, after removing a specific node, where $\{c'_n\}$ is the set of nodes crossed by c'_n.

While removal of a node n cannot disconnect a bi-connected graph, G, it could result in multiple bi-connected components that are easily identified by an appropriate algorithm. In such a case, the bridge nodes or stub nodes are also identified. They will always be in N_n, and will connect the bi-connected components $G'_1, G'_2, \dots$. Within each bi-connected component, some least-hop cycle p_i can be found for each component G'_i that will cross all nodes k such that $k \in N_n \cap G'_i$. The node-encircling p-cycle c'_n can then be easily constructed as $\{c'_n\} = \bigcup_{\forall i} p_i \cup N_n$. In other words, it is the cy-

cle formed by merging the nodes $\{c'_n\}$ and bridge nodes and/or stub nodes p_i into a non-simple cycle.

This procedure can be easily implemented in an algorithm to determine at least one node-encircling p-cycle for each node in the network. These cycles can also be used in a version of CIDA that is modified to provide protection for node failures as well as span failures.

16.4.3 Designing Node-Encircling p-Cycle Networks

Although initially described in an IP-network context, the NEPC concept can easily be applied to optical networks, but there is reason for concern about cost if the NEPCs are formed as discrete wavelength channels separate and in addition to the span-protecting p-cycles. One observation, however, is that some span-protecting p-cycles will by chance also happen to have NEPC relationships with some nodes, in which case they can also be used to provide node-failure protection. However, the existence of such p-cycles will not be very widespread, so a deliberate design approach may be used to provide an assured level of node-failure protection. Therefore, a unified ILP design model to provide both span and node-failure protection is preferable. The main issue with such a model is that the ILP must contend with the many combinations of nodes, crossed with eligible NEPCs, crossed with working lightpaths, and then again with the spans on the NEPC so that we can efficiently reroute any lightpath one way or the other around the NEPC and minimize overall capacity requirements. In larger networks, this may result in a large number of potential input parameters, variables, and constraints, making the full ILP difficult to solve, so care must be taken when enumerating eligible cycle and route sets. In the future, algorithmic approaches similar to the CIDA algorithm may also be used to reduce the complexity of the problem when designing large networks. For present purposes, however, the joint working and spare capacity NEPC network design model is expressed as an ILP design model, which makes use of the following new notation:

- **Sets:**
  - N is the set of nodes in the network.
- **Input Parameters:**
  - $\phi_r^n \in \{0, 1\}$ encodes the end-nodes of each demand. $\phi_r^n = 1$ if node n is an end-node of demand r, and $\phi_r^n = 0$ otherwise.
  - $Z_n^{r,q} \in \{0, 1\}$ is a parameter that encodes working routes. If $Z_n^{r,q} = 1$, working route q used for demand relation r crosses node n. If $Z_n^{r,q} = 0$, working route q used for demand relation r does not cross node n.
  - $X_p^n \in \{0, 1\}$ encodes which cycles can act as NEPCs for which nodes. If $X_p^n = 1$, p-cycle p can be an NEPC for node n, and if $X_p^n = 0$, it cannot.
  - $Z_{n,p}^{r,q} \in \{0, 1\}$ further encodes working routes and their relationships to eligible cycles. $Z_{n,p}^{r,q} = 1$ if working route q used for demand relation r crosses node n, and can be protected by p-cycle p, and $Z_{n,p}^{r,q} = 0$ otherwise. $Z_{n,p}^{r,q}$ is explicitly defined as $Z_{n,p}^{r,q} = Z_n^{r,q} \cdot (1 - \phi_r^n) \cdot X_p^n$.

- $R1_{r,q}^{p,n,i} \in \{0,1\}$ defines one of the restoration routes around an NEPC. $R1_{r,q}^{p,n,i} = 1$ if span i is on cycle p and is crossed by the clockwise reroute of working route q for demand r in the event of the failure of node n, and $R1_{r,q}^{p,n,i} = 0$ otherwise.
- $R2_{r,q}^{p,n,i} \in \{0,1\}$ defines one of the restoration routes around an NEPC. $R2_{r,q}^{p,n,i} = 1$ if span i is on cycle p and is crossed by the counter-clockwise reroute of working route q for demand r in the event of the failure of node n, and $R2_{r,q}^{p,n,i} = 0$ otherwise.

- **Decision Variables:**
  - $r1_{r,q}^{p,n} \geq 0$ is the amount of restoration flow in the clockwise direction over p-cycle p used to restore working route q for demand r in the event of failure of node n.
  - $r2_{r,q}^{p,n} \geq 0$ is the amount of restoration flow in the counter-clockwise direction over p-cycle p used to restore working route q for demand r in the event of failure of node n.

All previous notation from earlier formulations still applies, and we now define an ILP formulation that selects a set of span-protecting and node-encircling p-cycles so that all span and node failures are fully protected and total capacity requirements are minimized. The ILP design model is formulated as follows:

$$\min \quad \sum_{\forall j \in S} C_j \cdot (s_j + w_j) \tag{16.13}$$

$$s.t. \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \quad \forall r \in D \tag{16.14}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \zeta_i^{r,q} \cdot g^{r,q} = w_i \quad \forall i \in S \tag{16.15}$$

$$w_i \leq \sum_{\forall p \in P} x_{i,p} n_p \quad \forall i \in S \tag{16.16}$$

$$g^{r,q} = \sum_{\forall p \in P} \left(r1_{r,q}^{p,n} + r2_{r,q}^{p,n} \right) \quad \forall n \in N \quad \forall r \in D \quad \forall q \in Q^r \tag{16.17}$$

$$n_p \geq \sum_{\forall r \in D} \sum_{\forall q \in Q^r} \left(r1_{r,q}^{p,n} \cdot R1_{r,q}^{p,n,i} + r2_{r,q}^{p,n} \cdot R2_{r,q}^{p,n,i} \right) \quad \forall n \in N \quad \forall p \in P \quad \forall j \in S \tag{16.18}$$

$$s_j = \sum_{\forall p \in P} \delta_{j,p} \cdot n_p \quad \forall j \in S \tag{16.19}$$

The objective function in (16.13) minimizes total cost of working and spare capacity. The constraints in equations (16.14) and (16.15) are as already seen in equations (16.9) and (16.10), respectively, and ensure that there is sufficient working flow to fully route all working demands, and there is enough working capacity to accommodate all working flows. The constraints in equation (16.16) assert sufficient restoration flow for all span failures and is the same constraint set as seen previously in equation

(16.2). Equation (16.17) ensures that there is enough restoration flow in either or both directions around all relevant NEPCs so that all working flow on working route q for demand relation r affected by failure of node n is fully restorable. The constraints in equation places enough copies of NEPC p to accommodate the restoration flow simultaneously routed in either direction over any span j on the NEPC for all affected working routes q for all demand relations r. Finally, constraint set (16.20) is identical to the one in equation (16.3) and ensures there is enough spare capacity to accommodate all p-cycles placed in the design. Note that n_p now considers the fact that an eligible cycle can be used as an NEPC in addition to a span-protecting p-cycle.

A further modification can be made to the model so that only a specified level of node-failure restorability is required. To do so, we define $0 \leq L_r \leq 1$ as the proportion of lightpaths in demand relation r that must be restorable in the event of any node failure, and change equation (16.17) to the following:

$$g^{r,q}L_r = \sum_{\forall p \in P} \left(r1_{r,q}^{p,n} + r2_{r,q}^{p,n}\right) \quad \forall n \in N \forall r \in D \forall q \in Q^r \tag{16.20}$$

Analyses in Doucette et al. (2005) and shown in Figure 16.4 illustrate that while providing full node-failure protection with NEPCs (the "NEPC 100%" curve) is quite costly relative to providing only span-failure protection (the "p-cycle" curve), limited node-failure protection can be quite cost-effective. For instance, providing node-failure protection to only 25% of the lightpaths for each demand relation requires an additional investment of only 2% to 8% additional capacity (the "NEPC 25%" curve). Note that in all cases shown in Figure 16.4, the test case network designs are 100% span-protected while at the same time providing the indicated levels of overall node failure protection. The test networks used are the 20-node network family from Doucette (2004).

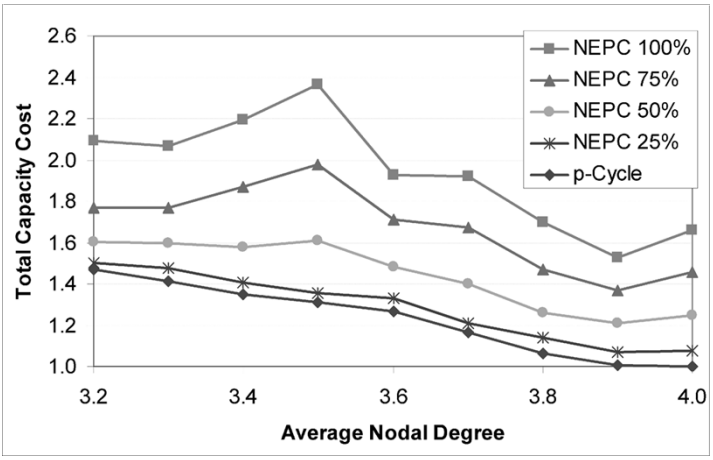

Figure 16.4 Normalized total capacity requirements for 25% to 100% node-failure restorable NEPC network design in the 20n40s1 network family.

16.4.4 Network Design using MPLS p-Cycles for Link Restoration

In a WDM-IP network, lightpaths in the WDM layer are used to set up logical connections between IP routers and/or label switched routers (LSRs). From the perspective of the IP network, these appear to be directional connections between routers, while in reality a single physical span may carry sections of multiple logical links protected between routers. Failure of a single physical span can then translate into the apparent failure of multiple links 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 with the same p-cycle.

In the IP network, restorability design also needs to consider the convergent flow effects arising from restoration. This is an aspect that does not exist for SONET or WDM where every working signal is either exactly replaced or not. In contrast, where packet or cell flows are being redirected upon restoration, one can take an over-subscription-based design to control the worst case simultaneously imposed flows on any link during any restoration scenario. Over-subscription is a comparative measure of actual capacity usage relative to pre-planned capacity allocations. Over-subscription-based capacity planning leads to assurance on the worst-case possible overloads, but it is a simpler and more practical planning framework than trying to plan capacity for protection while dealing directly with statistical traffic descriptions. Over-subscription under a restored or rerouted state in general is the ratio of the total of the pre-failure bandwidth allocations made to flows that now cross the link, relative to the actual capacity of the link.

In the MPLS context, it is technically meaningful to consider that the stochastic flows from more than one LSP may merge under protection rerouting to produce a combined load on a given link that somewhat exceeds the nominal utilization of that link, but still produces acceptable delay and cell loss probabilities in a restored-network state. In other words, we may plan to allow certain amounts of over-subscription of planned capacity during a failure. In the optical layer, however, over-subscription of capacity is clearly not an option because this layer requires exact matching of discrete working signals with corresponding signal paths for restoration. Over-subscription in the WDM layer would in effect correspond to simply not having enough protection lightpaths to cover a set of failed working lightpaths.

The over-subscription-based network design problem can be modeled as an ILP formulation, but we first define the following notation:

- **Set:**
  - S now refers to the set of IP links in the network. (This is generally more numerous than the set of physical layer spans.)
- **Input Parameters:**
  - M is the maximum number of p-cycles permitted in the design.
  - T_i is the total capacity allocated to link i.
  - w_i is the capacity allocation normally provided for the amount of working traffic on link i during normal operation.

- $\beta_{i,j,k} \in \{0, 0.5, 1\}$ is a pre-computed *imposed load ratio* corresponding to the $x_{i,k}$ coefficients for previous discrete-circuit p-cycles. Each $\beta_{i,j,k}$ is a constant that gives the fraction of the working flow from link i that is carried on link j if using cycle k for restoration. $\beta_{i,j,k} = 0$ if cycle k does not pass over link j, $\beta_{i,j,k} = 0.5$ if the p-cycle offers two restoration paths to link i and traverses link j (traffic is split), and $\beta_{i,j,k} = 1$ if p-cycle p offers only a single restoration path for link i and also crosses link j.

- **Decision Variables:**

  - $\delta_k \in \{0, 1\}$ is a binary decision variable, encoding selection of p-cycle k in the design. $\delta_k = 1$ if cycle k is formed into a p-cycle, and $\delta_k = 0$ if it is not.
  - $\alpha_{i,k} \in \{0, 1\}$ is a binary decision variable encoding whether a p-cycle k is assigned to carry restoration flow for failure of link i. $\alpha_{i,k} = 1$ if p-cycle k is assigned to the restoration of link i, and $\alpha_{i,k} = 0$ if it is not.
  - $X_{i,j} \geq 0$ is the over-subscription factor on link j during the restoration of link i (i.e., the total of the capacity allocations for normal flow that are now imposed on link j relative to the amount of its actual capacity.)
  - $\eta_M \geq 0$ is the maximum over-subscription ratio on any link during any restoration event.

All other notation from earlier formulations still applies. We can now define an ILP formulation that determines a set of IP p-cycles that minimizes the worst-case over-subscription factor on any link, over all failure scenarios. The design model for minimum peak oversubscription is formulated as follows:

$$\min \quad \eta_M \tag{16.21}$$

$$s.t. \quad \sum_{\forall k \in P} \delta_k \leq M \tag{16.22}$$

$$\alpha_{i,k} \leq \delta_k \quad \forall i \in S, \forall k \in P \tag{16.23}$$

$$\sum_{\forall i \in S} \alpha_{i,k} \geq \delta_k \quad \forall k \in P \tag{16.24}$$

$$\sum_{\forall k \in P} \alpha_{i,k} = 1 \quad \forall i \in S \tag{16.25}$$

$$X_{i,j} = \frac{w_j + \sum_{\forall k \in P} \beta_{i,j,k} w_i \alpha_{i,k}}{T_j} \quad \forall i \in S \, \forall j \in S | i \neq j \tag{16.26}$$

$$\eta_M \geq X_{i,j} \quad \forall i \in S \, \forall j \in S | i \neq j \tag{16.27}$$

The constraints in equation (16.22) restricts the total number of p-cycles to be no more than M. Equation (16.23) allows link i to use a cycle p for its restoration (as designated by $\alpha_{i,k} = 1$) only if that cycle is selected for use in the overall design (i.e., $\delta_k = 1$). In equation (16.24), a cycle k is permitted to be selected in a design only if it is used for restoration of at least one link. The constraints in equation (16.25) assert the usage of one and only one cycle for each failure scenario. The over-subscription factor on

link j for failure of span i is calculated in equation (16.26), as the ratio of the sum of the normal working flow on link j and all of the flows imposed on it by all p-cycles used to restore failure of link i relative to the total capacity on the link. (It is written this way to convey presence of the oversubscription ratio. Obviously T_j is a constant which can be moved to the left hand side.) Finally, equation (16.27) sets the maximum over-subscription ratio, η_M. Since the objective function in equation (16.21) seeks to minimize η_M, the interaction with the constraints in equation (16.27) will cause η_M to take the value of the largest $X_{i,j}$ on all links for all failure scenarios. In other words, η_M will be the worst-case over-subscription factor on all spans over all failure scenarios.

16.5 P-CYCLE DESIGN FOR ENHANCED AVAILABILITY

In this section, we present various formulations for capacity design of p-cycle networks in which single-failure restorability (referred to as R_1 demands or *Gold* quality of protection (QoP) class) is *not* the only protection option considered. We also consider an approach which improves the dual-failure restorability of selected demands with or without addition of any further capacity above that needed for the R_1 design above. These demands (of type *Gold-Plus*) are not strictly guaranteed dual-failure restorability but they will be more often restorable from dual failures and therefore will enjoy higher service availability. We then present an approach where selected demands (referred to as *Platinum* class) are provided with complete dual failure protection on all spans along their paths, a strategy that also provides extremely high availability. The following section then considers the simultaneous consideration of preemptible demands (type *Economy*), non-protected demands (*Bronze*), best-efforts restoration demands (*Silver*), single-failure restorable demands (*Gold*) and dual-failure restorable demands (*Platinum*).

Previous notation still applies, and we define the following new notation:

- **Sets:**
  - $D^g \subseteq D$ is the set of demand relations for the Gold protection class.
  - $D^{g+} \subseteq D$ is the set of demand relations for the Gold-*Plus* protection class.
  - $D^p \subseteq D$ is the set of demand relations for the Platinum protection class.
- **Input Parameters:**
  - $w_i^a \geq 0$ is the amount of working capacity allocated on span i for paths of protection class a, where $a \in \{g, g+, p\}$.

16.5.1 *Capacity Placement for Selectively Enhanced Availability (p-Cycle SEACP)*

The problem of Capacity Placement for Selectively Enhanced Availability jointly optimizes the routing of demands and the allocation of spare capacity to find the minimal cost capacity placement that allows us to serve all demands following the routing constraints described earlier and guarantee that each working channel is protected against

single span failures. It is modeled as follows:

$$\min \quad \sum_{\forall j \in S} \sum_{\forall m \in M} C_j^m t_j^m \tag{16.28}$$

$$s.t. \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \quad \forall r \in D^g \cup D^{g+} \tag{16.29}$$

$$w_i^g = \sum_{\forall r \in D^g} \sum_{\forall q \in Q^r} \varsigma_i^{r,q} g^{r,q} \quad \forall i \in S \tag{16.30}$$

$$w_i^{g+} = \sum_{\forall r \in D^{g+}} \sum_{\forall q \in Q^r} \varsigma_i^{r,q} g^{r,q} \quad \forall i \in S \tag{16.31}$$

$$w_i^g + w_i^{g+} \leq \sum_{\forall k \in P} x_{i,k} n_k \quad \forall i \in S \tag{16.32}$$

$$w_i^{g+} \leq \sum_{\forall k \in P} x_{i,k} (1 - \delta_{i,k}) n_k \quad \forall i \in S \tag{16.33}$$

$$s_j = \sum_{\forall k \in P} \delta_{j,k} n_k \quad \forall j \in S \tag{16.34}$$

$$w_j^g + w_j^{g+} + s_j \leq \sum_{\forall m \in M} t_j^m Z^m \quad \forall j \in S \tag{16.35}$$

The objective is to minimize the total cost of required modular capacity. Constraint set (16.29) ensures that for every demand relation r in either protection class, there is enough flow over all eligible working routes to fully serve that demand. Constraint sets (16.30) and (16.31) ensure that the right amount of working capacity is allocated on each span k for the Gold and Gold-*Plus* protection classes respectively. Constraint set (16.32) guarantees that there are enough p-cycles in the solution to protect all working capacity units on all spans. By routing demands such that all Gold-*Plus* working channels are protected as straddlers, it is guaranteed that at least one of the protection paths will survive in the event of a dual span failure. To enforce the requirement that paths in the Gold-*Plus* class have to be routed on straddling spans only, constraint set (16.33) ensures that for each span i, the number of p-cycles that span i straddles is enough to protect all working capacity units in the Gold-Plus class. (Note that through constraint set (16.32), paths in the Gold class will be protected either by p-cycles they straddle or by p-cycles to which they have an on-cycle relationship.) Constraint set (16.34) makes sure that there is enough protection capacity allocated on all spans to support all p-cycles in the solution. Finally, constraint set (16.35) ensures that there is enough capacity placed on all spans to allow the allocation of working and spare capacity as imposed by (16.30), (16.31), and (16.34). Other approaches to improving the dual-failure restorability of service paths in p-cycle based networks are presented in Schupke et al. (2004).

16.5.2 p-Cycle Multi-Restorability Capacity Placement (p-Cycle MRCP)

The approach proposed in this section is based on an evolution of the basic p-cycle principle in which p-cycles can either be used to offer two backup paths protecting two working channels on straddling spans, as in the normal p-cycle scheme, or two protection *options* for a single platinum channel on any straddler. The latter option can be

used for protection of service paths with ultra-high availability requirements. Based on studies in Clouqueur and Grover (2002) and Arci et al. (2003), it is known that offering two or more restoration options instead of just one leads to great improvements of the availability of service. This new class of service, that we call the *Platinum* class, is therefore expected to enjoy extremely high availability, and by virtue of design, full restorability to any dual span failures.

The multi restorability capacity placement (p-cycle MRCP) formulation is described below. It finds the optimal routing of demands and allocation of capacity that minimizes the required capacity placement subject to the routing constraints described earlier. The formulation is expressed as follows:

$$\min \quad \sum_{\forall j \in S} \sum_{\forall m \in M} C_j^m t_j^m \tag{16.36}$$

$$s.t. \quad \sum_{q \in Q^r} g^{r,q} = d^r \quad \forall r \in D^g \cup D^p \tag{16.37}$$

$$w_i^g = \sum_{\forall r \in D^g} \sum_{\forall q \in Q^r} \varsigma_i^{r,q} g^{r,q} \quad \forall i \in S \tag{16.38}$$

$$w_i^p = \sum_{\forall r \in D^p} \sum_{\forall q \in Q^r} \varsigma_i^{r,q} g^{r,q} \quad \forall i \in S \tag{16.39}$$

$$w_i^g + 2w_i^p \leq \sum_{\forall k \in P} x_{i,k} n_k \quad \forall i \in S \tag{16.40}$$

$$2w_i^p \leq \sum_{\forall k \in P} x_{i,k}(1 - \delta_{i,k}) n_k \quad \forall i \in S \tag{16.41}$$

$$s_j = \sum_{\forall k \in P} \delta_{j,k} n_k \quad \forall j \in S \tag{16.42}$$

$$w_j^g + w_j^p + s_j \leq \sum_{\forall m \in M} t_j^m Z^m \quad \forall j \in S \tag{16.43}$$

The objective function is the same as before. Constraint sets (16.37) to (16.39) are similar to constraint sets (16.29) to (16.31), with class Gold-*Plus* being replaced by class Platinum. Constraint set (16.40) is similar to (16.32), but unlike for the Gold-*Plus* protection class, the Platinum protection class requires that a p-cycle provide two protection options for a single working link in that class, therefore a factor 2 is added in front of the number of working links in the Platinum class. Constraint set (16.41) ensures that Platinum-class working links on any span i can only be protected by p-cycles that span i straddles. Finally, constraint sets (16.42) and (16.43) are similar to (16.34) and (16.35) in the previous formulation.

16.6 MULTIPLE QUALITY OF PROTECTION P-CYCLE NETWORK DESIGN (MULTI-QOP p-CYCLE DESIGN)

In this section, we consider incorporating (among others) all the service classes discussed in the previous section in p-cycle networks. Additionally this is done without requiring any dynamic post failure reconfiguration of p-cycles to effect dual failure survivability. Reconfiguring p-cycles after a span failure to protect against any subsequent span failure has been studied in (Schupke et al., 2004). It is, however, interesting

that with p-cycles, no reconfiguration is strictly necessary to effect guaranteed dual failure protection. Imposing the need for run-time reconfiguration requires that all or some of the nodes be full cross-connects capable of run-time switching at the line rate. Our interest here, however, is to design with a tradeoff in terms of increased capacity required to effect dual failure protection without assuming the flexibility of dynamic post failure reconfiguration in the network.

With p-cycles, to ensure that a particular platinum demand is protected end-to-end, it is sufficient that the two following conditions be met:

- (a) All platinum working capacity must be protected using straddling relationships only.

- (b) A whole unit of p-cycle capacity must be used to protect a unit of platinum capacity on the straddler. (As opposed to using half the p-cycle to protect a single unit of gold capacity on a straddler.)

Enforcing both conditions in the design ensures that a single p-cycle provides two disjoint routes (effectively 1+1+1 protection) to each unit of platinum capacity it protects. A second span failure along one of the protection routes (after failure of the straddling span) has no effect on survivability, as the end nodes of the failed span have only to switch over to the other still surviving and unused protection path. In the general case, it can also be shown that that p-cycles can be used to provide survivability solutions for an 'n' failure protected network, with a theoretical minimum redundancy η, given by $\eta = n/(\bar{d} - n)$. Thus, as an example, a network with $\bar{d} = 4$ requires a minimum of 100% redundancy to offer platinum class protection (i.e., $n = 2$) to all demands.

Some additional notation used in the multi-QoP p-cycle design formulation is:

- **Sets:**
  - $D^{sb} \subseteq D$ is the set of demand relations for the Silver and Bronze protection classes.
  - $D^{e} \subseteq D$ is the set of demand relations for the Economy protection class.

- **Input Parameters:**
  - $c_{i,j}^{k} \in \{0, 1\}$ encodes whether spans i and j are *both* on cycle k. $c_{i,j}^{k} = 1$ if spans i and j are both on the cycle k, and $c_{i,j}^{k} = 0$ if at least one is not. Mathematically, we set $c_{i,j}^{k} = \delta_{i,k}\delta_{j,k}$.

- **Decision Variables:**
  - $\eta_i^{k,g} \geq 0$ is the integer number of unit-capacity p-cycles on cycle k used to protect Gold class working capacity on span i.
  - $n_i^{k,p} \geq 0$ is the integer number of unit-capacity p-cycles on cycle k used to protect Platinum class working capacity on span i.

The multi-QoP p-cycle design formulation is expressed as follows:

$$\min \quad \sum_{\forall j \in S} \sum_{\forall m \in M} C_j^m t_j^m \tag{16.44}$$

$$s.t. \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \quad \forall r \in D^p \cup D^g \cup D^{sb} \cup D^e \tag{16.45}$$

$$\sum_{\forall r \in D^p} \sum_{\forall q \in Q^r} \varsigma_i^{r,q} g^{r,q} = w_i^a \quad \forall i \in S, \forall a \in \{p,g,s,b,e\} \tag{16.46}$$

$$\sum_{\forall k \in P} x_{i,k} n_i^{k,j} \geq w_i^g \quad \forall i \in S \tag{16.47}$$

$$\sum_{\forall k \in P} x_{i,k} n_i^{k,p} \geq w_i^p \quad \forall i \in S \tag{16.48}$$

$$n^k \geq n_i^{k,g} + n_i^{k,p} \quad \forall i \in S \, \forall k \in P \tag{16.49}$$

$$n^k \geq 2n_i^{k,p} \quad \forall i \in S \, \forall k \in P \tag{16.50}$$

$$n_i^{k,p} \leq \Delta(1 - c_{i,j}^k) \quad \forall i \in S \, \forall j \in S | i \neq j \, \forall k \in P \tag{16.51}$$

$$\sum_{\forall k \in P} (\delta_{i,k} n^k - w_i^e) \leq s_i \quad \forall i \in S \tag{16.52}$$

$$w_j^p + w_j^g + w_j^{sb} + w_j^e + s_j \leq \sum_{\forall m \in M} t_j^m Z^m \quad \forall j \in S \, \forall m \in M \tag{16.53}$$

The objective is to minimize the total cost of modular capacity of the design. Constraint (16.45) ensures that all demands (of all QoP classes) are routed. Constraint (16.46) to (16.48) ensures that there is sufficient working capacity placed on the network to accommodate them. The constraints in equations (16.49) to (16.51) place p-cycles in the network to fully protect all working capacity needing protection (Gold and Platinum). Constraints (16.50) and (16.51) ensure that Platinum capacity is protected using two disjoint protection paths over p-cycles, and that both rules (a) and (b) discussed earlier are obeyed. The constraint in (16.52) ensures that spare (and economy) capacity exists to build all the necessary p-cycles. Constraint set (16.53) introduces modularity (and economy of scale) into the problem. Initial results from Kodian et al. (2003) establish that, for certain networks, up to 30% of the total demand between every pair of nodes in a network can be offered platinum class protection without any extra capacity over the all-Gold design. (i.e. 30% of all services can have dual failure survivability in a network which has no more spare capacity in total than that required for a 100% single failure restorable network design.)

16.7 PROTECTION AGAINST SHARED RISK LINK GROUPS (SRLGS) WITH P-CYCLES

One special type of dual failures are called Shared Risk Link Groups (SRLGs). SRLGs are a special case where a single physical failure manifests itself as a logical dual span failure in the network (Doucette and Grover, 2002). (In general, SRLGs may also manifest as triple or quadruple simultaneous span failures, but currently we only

consider up to dual span failures as a network manifestation of a single SRLG-type failure.) For instance the fiber optical cables leaving a network node forming two logically distinct and nominally disjoint spans, may actually cross a bridge together. They are thus logically distinct spans that share one common-cause failure possibility, i.e. the bridge that they cross. A variation of the multi-QoP model from the previous section is used to protect selected demands against stipulated SRLG-type failures. The following new parameters and variables are added to the previous models:

- **Input Parameters:**
  - $\phi_{i,j} \in \{0,1\}$ encodes span pairs affected by an SRLG. $\phi_{i,j} = 1$ if spans i and j can have some common cause of failure against which we want to design the network to be fully restorable.
  - $b_{i,j}^k \in \{0,1\}$ encodes whether SRLG spans i and j are a straddler and on-cycle pair of cycle k. We set $b_{i,j}^k = 1$ if span i straddles cycle k, span j is an on-cycle span of cycle k, and $\phi_{i,j} = 1$, and $b_{i,j}^k = 0$ otherwise. Mathematically, we can say $b_{i,j}^k = \left(\frac{x_{i,k}}{2}(1 - \delta_{i,k})\right)\delta_{j,k}\phi_{i,j}$.

The entire multi-QoP p-cycle design problem in Section (16.6) applies, and we add one new constraint:

$$n_k \geq (2n_i^{k,p})b_{i,j}^k \quad \forall i \in S \, \forall k \in P \, \forall j \in S | i \neq j \tag{16.54}$$

Constraint (16.54)ensures that the only dual failure cases that are considered are the potential SRLG span pairs. Work in Doucette and Grover (2002) shows that depending on the exact set of spans chosen as SRLG-pairs, protection of SRLGs in a span restorable mesh network can increase the capacity requirements from as little as a few percent more than that required for single-failure protection, to as high as that required for full dual-failure protection.

16.8 DESIGN OF P-CYCLE NETWORKS IN THE FACE OF DEMAND UNCERTAINTY

All the p-cycle design problems discussed above assume a specific demand forecast to which one optimizes the routing and transport capacity assignment for a single target planning view. In this section, we look at the p-cycle design problem when the demand forecast is uncertain. The demand uncertainty can be modeled by a set of possible future scenarios, with each assigned a probability estimate. In practice these might correspond to a set of "what-if" scenarios used by a planner to individually test the sensitivity of a nominal design. Here, however, we discuss a capacity design formulation that inherently considers a range of possible futures all at once. The new model can produce a capacity plan that leads to significantly lower expected total lifetime costs than traditional p-cycle designs that target a single forecast.

The problem is formulated as a two-part problem involving a bi-criteria objective function. The first part considers the budget X to be invested at present and the second part represents the total expected corrective or "recourse" cost Y to be incurred in the future when uncertainty unfolds. Compared to the traditional approach where a

single demand forecast is assumed for a snapshot design, the two-part model better reflects the uncertainty arising from demand forecasting as well as the complete life-cycle investment costs associated with the capacity planning process. The two-part design also has the property of minimum expected total cost of the as-built current network plus future recourse actions as may be needed to cope with a range of different possible futures. Other work on capacity planning under demand uncertainty for span-restorable and/or path-protected mesh networks can be found in Kennington et al. (2003), Leung and Grover (2004b), and Leung and Grover (2004a).

16.8.1 Future-Proof p-Cycle Design

The goal of the future-proof p-cycle design (FPPC) problem is to minimize the total cost of the network that we commit to building in the present, as well as the expected value of future recourse actions to augment the design to serve possible future demand scenarios, where each scenario is associated with a probability estimate. To model the FPPC design problem, we must define new notation as follows:

- **Set:**
  - R_j is the recourse (or penalty) cost of placing an extra unit of capacity on span j to cope with future capacity additions. In the general case, these recourse costs are specific to each span and can reflect different practical considerations such as an abundance of dark fiber on some spans but not on others, the fact that one span may be nearing exhaust, the ability to lease capacity from another carrier, and so on. For comparative studies, we use a common recourse cost factor, R for all spans where $R_j = RC_j$.
  - $0 \leq P^u \leq 1$ is the probability estimate for demand scenario u.
  - $d^{r,u}$ is the number of lightpath demand units for relation r in demand scenario u.
- **Decision Variables:**
  - $y_j^u \geq 0$ is the integer number of additional working capacity units that would have to be placed on span j in the future to cope with scenario u now.(relative to the number required for the design solution to be built.)
  - $z_j^u \geq 0$ is the integer number of additional spare capacity units required on span j under future demand scenario u.
  - $g^{r,q,u} \geq 0$ is the amount of working flow assigned to working route q used for demand relation r in scenario u.
  - $n_k^u \geq 0$ is the integer number of unit-capacity p-cycles placed on cycle k in scenario u.

The FPPC problem is modeled as variation of the JCA problem in Section 16.2 where working routing and p-cycle design is performed jointly. The difference here is that this is done simultaneously for multiple demand scenarios so as to minimize the total

of the actual cost incurred at present to build the network plus the expected future cost to deal with inaccuracies in the forecast. The model is expressed as follows:

$$\min \quad \sum_{\forall j \in S} C_j(w_j + s_j) + \sum_{\forall j \in S} \sum_{\forall u \in U} P^u R_j(y_j^u + z_j^u) \tag{16.55}$$

$$s.t. \quad \sum_{\forall q \in Q^r} g^{r,q,u} = d^{r,u} \quad \forall r \in D \, \forall u \in U \tag{16.56}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \varsigma_j^{r,q} g^{r,q,u} = w_j + y_j^u \quad \forall j \in S \, \forall u \in U \tag{16.57}$$

$$w_i + y_i^u \leq \sum_{\forall k \in P} x_{i,k} n_k^u \quad \forall i \in S \, \forall u \in U \tag{16.58}$$

$$s_j + z_j^u \geq \sum_{\forall k \in P} \delta_{j,k} n_k^u \quad \forall j \in S \, \forall u \in U \tag{16.59}$$

$$y_j^u = 0 \quad u = 0 \, \forall j \in S \tag{16.60}$$

$$z_j^u = 0 \quad u = 0 \, \forall j \in S \tag{16.61}$$

For each scenario u, the constraints set in equation (16.56) allocates the demand flows $g^{r,q,u}$ of demand relation r onto working route q. Equation (16.57) determines the working capacity needed on each span to serve the demand flows. These two constraints sets are similar to those of equations (16.9) and (16.10) for the JCA formulation in Section 16.2, except that we now not only consider the *actual* working capacity, w_j, to be placed in the network, but also the *shortfall* working capacity $y_j^u \geq 0$ required to serve unexpected demands that may arise in future scenario u. Equations (16.58) and (16.59) correspond to the restorability and spare capacity constraint sets in equations (16.2) and (16.3), respectively, from the basic SCA design model. The final two constraint sets in equations (16.60) and (16.61) are needed to characterize a special scenario situation, $u = 0$, which represents any current existing demands that *must* be served and protected. These constraints ensure the condition by asserting that there can be no shortfall capacities for any demands in that scenario, forcing the design to contain adequate w_j and s_j capacities.

As a measure of the effectiveness of this model, tests were done in Leung and Grover (2004a). In that work we applied uniform random variations to an expected demand forecast to generate a set of plausible "what if" scenarios with total demand volumes ranging from 0.3 to 5.0 times the nominal forecasts. A recourse costs factor of $R = 5$ (i.e., $R_j = 5C_j$) allowed an FPPC network design with an expected cost reduction of 39% compared to a conventional single scenario JCA design on the COST 239 network. Note that in practice recourse costs could be either greater or less than unity (relative to present costs). It could be argued that technologically capacity is always less costly in the future. However recourse cost should include the total operational cost in the future of having to design, install and commission the capacity augmentation if it could have been included in the initial design.

16.9 PWCE DESIGN WITH P-CYCLES

The *protected working capacity envelope* (PWCE) is a new paradigm for provisioning dynamic survivable services, as described in Grover (2003), Grover (2004), Shen

and Grover (2004). In the PWCE model, protection may be performed between the nodes adjacent to the failure, so span restoration and p-cycles are both amenable to the PWCE concept. In PWCE, it is the bearer capacity itself that is protected and so any service connection routed exclusively over such capacity in a PWCE is inherently protected as well, without any explicit designation of a backup route. In comparison with the conventional Shared Backup Path Protection (SBPP)-based provisioning method, PWCE shows advantages of simple operation, fast restoration, and good blocking performance. A PWCE can be simply formed within the working capacity determined from a conventional fully restorable p-cycle network design. But larger "envelopes" can also be designed so as to maximize the envelope within a specified capacity budget or some other criteria. Designing a maximal envelope is an opportunity for a PWCE-based network, so we now discuss various possible envelope design models based on different assumptions and capacity budgets for the p-cycle network.

The PWCE concept is illustrated in Figure (16.5), where a set of spare channels defines a *reserve network* of spare capacities corresponding to the s_j budget and the maximal PWCE of $\langle w_i^0 \rangle$ values the reserve network can accommodate (the $\langle w_i^0 \rangle$ values correspond to the solution of the above ILP problem). The PWCE is then able to dynamically serve multiple service paths as they arrive (three are shown). There are no per-path protection arrangements to make because the channels used for provisioning in the working layer are themselves protected by the reserve network and some embedded restoration or protection mechanism. In other words, as long as the $\langle w_i^0 \rangle$ quantities will support working routing of the demand, it is inherently protected end-to-end and no further action to explicitly deal with protection is required. Under PWCE, nodes need not track individual channel states as they do in SBPP because sharing relationships are not defined precisely between individual paths and individual backup channels. The nodes need only know that individual spans have one or more provisionable channels available, which requires no signaling for state dissemination. Consequently, this greatly simplifies the network operation and service provisioning.

16.9.1 Simple PWCE Volume Maximizing Model

There are various methods to construct a PWCE within a p-cycle network, one of which is to use conventional SCA or JCA designs given in Sections 16.1 and 16.2. However, those design methods do not provide an optimal envelope (at least in the volume-maximized sense), and so will not fully exploit the spare capacity from a PWCE standpoint. To achieve a better capacity efficiency, we need volume maximization models. The volume maximization model places more working demand flow on non-forcer spans such that more working capacity is protected without any increase in spare capacity allocation. (Forcers are spans which, in the design of a restorable mesh network, have working capacity quantities and surrounding topological circumstances such that they fully use the spare capacity available on one or more other spans in the design. The significance is that if a forcer span has additional demand routed over it, then the total network spare capacity must increase. On the other spans this will not be so in general. See (Shen and Grover (2003a)) The term "volume" refers to the total number of protected working channels the design supports. Three such models are de-

veloped in Shen and Grover (2003b;a), where the forcer structure of the conventional designs is used to identify extra working capacity to exploit.

The simplest PWCE maximizing model we can formulate is one where a spare capacity budget is known (say to correspond to a pre-determined maximal set of p-cycles), and we seek to place working capacity so as to maximize the protection envelope over all spans. We express this model as follows:

$$\max \quad \sum_{\forall j \in S} C_j w_j \tag{16.62}$$

$$s.t. \quad \sum_{\forall k \in P} x_{i,k} n_k \geq w_i \quad \forall i \in S \tag{16.63}$$

$$s_j \geq \sum_{\forall k \in P} \delta_{j,k} n_k \quad \forall j \in S \tag{16.64}$$

In this model, s_j is an input parameter rather than a decision variable as in past models. Such a spare capacity budget can be obtained from the conventional survivable network design, with which the working capacity of a forecasted demand matrix is protected (and of course, other options are possible for the budget selection). The objective of the model is then to maximize the volume of the envelope of protected working capacity within that budget. The constraints in equation (16.63) ensure that working capacity allocated to each span is fully protected, and the constraints in equation (16.64) guarantee that the spare capacity budget is sufficient to form the p-cycles required.

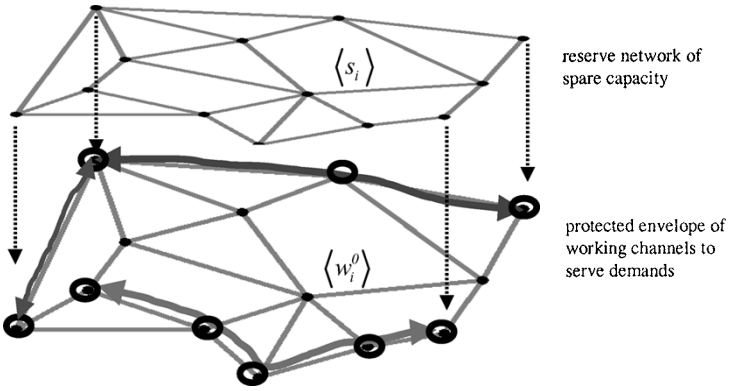

Figure 16.5 Three working paths routed within the PWCE of a p-cycle network (Grover (2003)).

16.9.2 Demand Target Pattern Matching

The above PWCE volume maximization model by itself simply creates the greatest total number of protected working channels over the entire network. However, they may not be in the best places in the network to be most effective. The problem can

thus be modified to accommodate the idea of *demand target pattern matching* to improve the performance of an envelope design by structuring the distribution of the protected working channels on spans to be reflective of some basic pattern of *relative* intensities expected of future demand. Mathematically this pattern specification can be identical in form to a static demand matrix, but its interpretation and meaning is no longer that of an assumed exact forecast. Rather, it is taken simply as a relative shaping template to help structure the PWCE as it is simultaneously maximized in volume. The philosophy is that even though demand is random and the future pattern has uncertainty in forecasting, it is still worthwhile to provide the solver a best view of the relative future intensities of demand on each node pair. This serves as a guideline for structuring the PWCE to generally fit plausible future demand scenarios better than if pure volume maximization is effected with no such guiding hand at all. For instance, compared to giving no shaping guidance at all to a volume maximization problem, in practice it would always seem reasonable to at least enter a shaping template that reflects the relative number of purely topological shortest paths of the graph that cross each span. Given a plausible forecast of relative loads between node pairs (a *network load template*) we can identify which spans are traversed by high relative loads (or simply traversed by many shortest-routes between all node pairs) and hence should preferably have a lot of working capacity assigned to them in the envelope volume maximization. Given a total design budget cost, we can then design a PWCE with a maximized envelope volume and a form of structuring to the distribution of working channels that is reflective of the relative load template.

This modified model requires the following additional notations:

- **Input Parameters:**
  - l_j is the target predicted relative load on span j, which can be computed say by shortest path routing of some demand forecasts.
- **Decision Variables:**
  - λ is a shape-asserting factor which structures the PWCE relative to the target load distribution.
  - α is a bi-criterion trade-off factor between structure shaping and volume maximization of PWCE.

The demand target pattern matching problem is formulated as follows:

$$\max \quad \lambda + \alpha \sum_{\forall j \in S} w_j \tag{16.65}$$

$$s.t. \quad \sum_{\forall k \in P} x_{i,k} n_k \geq w_i \quad \forall i \in S \tag{16.66}$$

$$s_j \geq \sum_{\forall k \in P} \delta_{j,k} n_k \quad \forall j \in S \tag{16.67}$$

$$w_j \geq \lambda l_j \quad \forall j \in S. \tag{16.68}$$

The problem is now in the form of a bi-criteria optimization, where there is some user-defined tradeoff between PWCE volume maximization and the shape factor, λ.

This shape factor is calculated by the constraints in equation (16.68), which expresses utility in the similarity between the PWCE and the target network load distribution. Equations (16.66) and (16.67) are repeated from the previous model. Generally, α is chosen so that the total volume term in the objective in (16.65) does not reduce from its single criteria optimal maximal value, but subject to this λ is forced as high as it can be.

16.9.3 Other Capacity Budget Scenarios

In the model in Section 16.9.1, spare capacity on each span has been pre-determined and specified as the available budget within which to construct a volume-maximized PWCE, however, there are other contexts for the capacity budget. Budgets can be span-based where a specified maximum number of spare channels is allowed on each span, or network-wide where the limit is applied on the total spare capacity of the network as a whole. By its very nature, a network-wide budget normally has more freedom in the PWCE construction than a span-based budget, and so more efficient designs are possible. Capacity budgets can also be defined on the total of working and spare capacity. Therefore, there are four possible types of budget scenarios to consider.

1. *Span-based spare capacity budget* – There is a specified maximum spare capacity on each span individually.

2. *Span-based total capacity budget* – There is a specified maximum total capacity on each span individually. In this scenario, there are no bounds on how much of that capacity must be working capacity and how much is spare.

3. *Network-wide spare capacity budget* – There is a specified collective maximum amount of spare capacity over all spans combined. In this scenario, there are no requirements on how this budget is distributed among the spans.

4. *Network-wide total capacity budget* – There is a specified collective maximum amount of combined working and spare capacity over all spans combined. In this scenario, there are no requirements on how much of that capacity must be working capacity and how much is spare or on how this budget is distributed among the spans.

These capacity budget scenarios can all be applied to the ILP models in Sections 16.9.1 and 16.9.2. We first introduce new notation as follows:

- **Input Parameters:**
  - T_j is the total number of deployed (or deployable) channels on span j, among which some channels will be assigned as the working capacity and with the remainder assigned as spare capacity.
  - B_S is the collective network-wide *spare* capacity budget.
  - B_T is the collective network-wide *total* capacity budget.

The ILP models in Sections 16.9.1 and 16.9.2 correspond to the span-based spare capacity budget (capacity budget scenario 1). If we wish to implement the second capacity budget scenario where we have a total working plus spare capacity budget for each span individually, we let the spare capacity values, s_j, be decision variables, and add the new constraints in equation (16.69) to either the volume maximization problem in Section 16.9.1 or the demand target pattern matching problem in Section 16.9.2:

$$T_j \geq w_j + s_j \quad \forall j \in S \tag{16.69}$$

This scenario closely models the situation where there is an existing deployed network of transmission systems upon which we wish to construct an efficient p-cycle network under the PWCE paradigm. The problem is then to determine how best to assign the installed capacity as either working or spare capacity so as to produce an optimal protection envelope. The new constraint set in equation (16.69) ensures that the combined working and spare capacity does not exceed the allowable budget on each span.

The third capacity budget scenario where we have a single network-wide collective spare capacity budget can be implemented by the adding the following constraint set to either the volume maximization problem or the demand target pattern matching problem:

$$B_S \geq \sum_{\forall j \in S} s_j \tag{16.70}$$

Here, equation (16.70) limits the total spare capacity allowed in the network. Again, the spare capacity values, s_j, are decision variables, rather than input parameters. In general, resultant protection envelopes tend to be large for this scenario than the first two, since we essentially allow greater flexibility in where the budget is allocated.

Finally, the fourth capacity budget scenario with a network-wide total capacity budget can be modeled by adding the constraint set in equation (16.71) to either the volume maximization problem or the demand target pattern matching problem:

$$B_T \geq \sum_{\forall j \in S} (w_j + s_j) \tag{16.71}$$

Equation (16.71) ensures that the total combined working and spare capacity allocated throughout the entire network is within the specified budget limit. This scenario allows for the most optimal PWCE formation, since it allows the most flexibility of all capacity budget scenarios, and corresponds to a green-fields design problem, where the goal is to build a completely new network from the ground up, subject to a limited capacity (or financial) budget.

Work in Shen and Grover (2003b;a), showed that over five different test networks, the volume maximization model allows for an additional 16% to 78% increase in the amount of working capacity capable of being protected under the optimal spare capacity allocation from the SCA design model in Section 16.1. It should be emphasized that all of these working capacity increases are free in the sense that no spare capacity increase is needed to retain 100% restorability. For a network operator, this is a good source of extra revenue (or an opportunity for savings) and an attractive option for provisioning growing services. Comparisons to the SBPP-based provisioning method also show that PWCE with p-cycle protection provides lower probability of blocking, when both models are provided with the same capacity distribution.

16.10 RING-MINING TO P-CYCLES

Migration from existing ring-based networks towards a future mesh-based architecture and operation is of considerable interest to network operators. In one migration technique called "ring mining," the line capacity and high speed interfaces of existing ring transport systems are reclaimed to support further ongoing growth of demand by converting to mesh based routing and protection operating under the span capacities of the prior rings (Clouqueur et al., 2001). p-Cycles are an interesting and natural alternative target architecture for ring-mining because like rings, they are cycle-oriented, but at the same time, unlike rings, they are mesh-like in capacity efficiency.

16.10.1 Ring Mining to p-Cycles Without Capacity Addition

One simple model for converting a ring network to a p-cycle network is to find the largest common multiplier, λ', that can be applied to every member of the demand matrix, while still keeping the network restorable using p-cycles, without adding any new capacity at all. Like the JCA model in Section 16.2, this model jointly optimizes working and spare capacity for a p-cycle based network, but now we must respect the existing ring fiber capacity limits (we also call this the pure ring mining problem). We add the following new notation to that used for previous models:

- **Input Parameters:**
  - a_j^m is the number of modules of type m available on span j from an already existing ring network.
- **Variables:**
  - $\lambda' \geq 1$ is the uniform least common demand growth multiplier, and is the maximum value by which we can multiply all demands while still being able to construct a fully-restorable p-cycle network that respects the capacity limits on each span.

The problem is formulated as the following ILP:

$$\max \quad \lambda' \tag{16.72}$$

$$s.t. \quad w_j \leq \sum_{\forall k \in P} x_{i,k} n_k \quad \forall i \in S \tag{16.73}$$

$$s_j = \sum_{\forall k \in P} \delta_{j,k} n_k \tag{16.74}$$

$$\sum_{\forall q \in Q^r} g^{r,q} = \lambda' d^r \quad \forall r \in D \tag{16.75}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \varsigma_i^{r,q} g^{r,q} = w_i \quad \forall i \in S \tag{16.76}$$

$$w_j + s_j \leq \sum_{\forall m \in M} a_j^m Z^m \quad \forall j \in S \tag{16.77}$$

The constraint sets in equations (16.73), (16.74), and (16.76) are identical to those for the JCA formulation in Section 16.2, while equation (16.75) replaces equation

(16.9). In equation (16.75), all working demands are routable even after the growth multiplier has been applied to each demand. Finally, equation (16.77) ensures that only the existing capacity mined from the prior ring network is used.

Work in Kodian et al. (2003) showed that for some networks, uniform demand growth multipliers as high as 2.75 were achievable, meaning without adding any new capacity, a ring network could be converted to a p-cycle network capable of fully routing and protecting all demands even if they are all increased to 275% of their original quantities.

From a practical view however, pure ring mining potential is just an indicator, and the use of a *uniform* demand growth multiplier can be very restrictive for the design since there are generally a few demands that act as upward forcers for the total network cost. Increasing these demands by a few units would soon exhaust capacity on some critical spans on all possible routes for the demand. Unused capacity may exist on other spans in the network, but it is effectively stranded because that capacity does not exist in a continuous fashion. These forcer demands then force the solver to choose a very low value for λ' so as to not go over the capacity bounds imposed by the prior ring networks, regardless of the fact that a good number of non-forcer demands may have supported higher growth multipliers individually.

There are two possible options at this stage. To start with, an accurate pattern of non-uniform growth multipliers could be specified per demand, which may result in a much better design. One possible solution is attempting to let the solver maximize all demands individually under the same capacity constraints. This would first require definition of new notation:

- **Variables:**
  - $\lambda'_r \geq 1$ is the maximum demand growth multiplier for demand r, and is the maximum value by which we multiply demand r in the new design while still being able to construct a fully-restorable p-cycle network that respects the capacity limits on each span.

The objective function then changes from the one in equation (16.72) to that in equation (16.78), below, and the constraints in equation (16.75) are replaced with those in equation (16.79).

$$\max \quad \sum_{\forall r \in D} \lambda'_r \tag{16.78}$$

$$s.t. \quad \sum_{\forall q \in Q^r} g^{r,q} = \lambda'_r d^r \quad \forall r \in D \tag{16.79}$$

All other constraints in equations (16.73), (16.74), (16.76), and (16.77) still apply. The result is that the solver first maximizes the demands between every adjacent node pair (subject to capacity availability), then the second adjacent node pair and so on. The second option is to add capacity selectively on spans that are very near exhaust, to effectively "unlock" the stranded capacity elsewhere in the network. We examine this in the next section.

16.10.2 Ring Mining with Selective Capacity Addition

In this model, we allow modular capacity to be selectively added while mining rings to p-cycles under an economy-of-scale module addition cost model. The uniform demand growth multiplier λ' now becomes a parameter to the optimization problem and is incremented in fixed steps, to calculate the minimal extra capacity-cost required to enable each step value of λ'. We define the following new notation:

- **Variables:**
  - $\xi_j^m \geq 0$ is the number of modules of type m strategically added to span j.

The objective function is now changed from that in equation (16.72) to the one in equation (16.80), which seeks to minimize the cost of additional capacity added to the network:

$$\min \sum_{\forall j \in S} \sum_{\forall m \in M} \xi_j^m C_j^m \tag{16.80}$$

All of the constraints with constraints in equations (16.73), (16.74), (16.75), and (16.76), are kept and the constraints in equation (16.77) are replaced with those in (16.81).

$$w_j + s_j \leq \sum_{\forall m \in M} (a_j^m + \xi_j^m) Z^m \quad \forall j \in S \tag{16.81}$$

This new constraint set limits the working and spare capacity total on each span to be under the span capacity available from the existing ring network, plus the added modular span capacity. Adding capacity increases the value of the objective, and thus ensures that the solver considers adding capacity only after fully exploiting the existing ring capacity. Minimizing the cost also ensures that the solver chooses the least costly construction of modular capacity. For example, adding 1 OC-48 module would be less expensive than 4 OC-12 modules on the same span. Again, if the solver has already added one module to a span, it will attempt to maximize the utilization of that module, even if some of the working/protection paths need to be longer, since using 1 or all 48 units of capacity on an added OC-48 module on a span does not affect the objective.

Work in Kodian et al. (2003) finds that the capacity addition profile for ring-mining to p-cycles is very close to the profile obtained for span restorable mesh. But at the same time there is no rule as to why a particular network requires a specific number of modules added for demand growth. Two interesting effects are also observed. First, there is a deferral of cost until significant demand growth occurs, and secondly, there is a relatively low amount of extra capacity required to support a high uniform demand growth multiplier for some test networks. To a network operator this means that, depending on the network details, substantial demand growth can be supported without much capital investment.

16.10.3 Ring Mining Without Changes to Existing Working Routes

In all the models discussed so far, working routes are jointly optimized with the placement of p-cycles. This is acceptable for experimental studies with growth demands

and assumes that any implied rearrangement of existing paths may be acceptable if it is coordinated with customers and permitted by service contracts. But, based on industry feedback, this is not always an option. We now discuss a variation to the base model in Section 16.10.1 where we add a constraint restricting the rerouting of existing demand. We define the following new notation:

- **Input Parameters:**
  - $w_j^0 \geq 0$ is the previously existing working capacity on span j, obtained as part of the ring network data.

The objective function remains as that in equation (16.72), and we incorporate demand rerouting restrictions into the planning model by replacing the constraints in equations (16.75) and (16.77) by the constraints in equations (16.82) and (16.83), below, respectively:

$$\sum_{\forall q \in Q^r} g^{r,q} = (\lambda' - 1)d^r \quad \forall r \in D \tag{16.82}$$

$$w_j^0 + w_j + s_j \leq \sum_{\forall m \in M} a_j^m Z^m \quad \forall j \in S \tag{16.83}$$

The other constraints in the base model in Section 16.10.1 still apply. Constraint set (16.82) ensures that only the new growth demands are considered for routing decisions that are jointly coordinated with the spare capacity placement decisions. The existing working capacity on each span is left untouched because of the constraints in equation (16.83), and so existing working routes are not disturbed. In Kodian et al. (2003) it was shown that the specific test network could handle a uniform demand growth multiplier of $\lambda' = 1.5$ when working routes are allowed to be re-optimized, compared to $\lambda' = 1.33$ with no changes to existing working routing.

16.10.4 Ring Mining with Selective Capacity Addition and no Change to Existing Working Routes

For similar reasons as discussed in Section 16.10.2, selective capacity addition is considered along with the constraint on altering existing working routing, with λ' increased in steps. We consider the same model as in Section 16.10.3 with the constraints in equation (16.83) replaced by equation (16.84), below.

$$w_j + s_j + w_j^0 \leq \sum_{\forall m \in M} (a_j^m + \xi_j^m) Z^m \quad \forall j \in S \tag{16.84}$$

This new constraint set ensures that the existing working capacity is left untouched. The existing spare capacity is re-optimized for p-cycles, including the option to add new capacity in modular quantities. Of course, this assertion of fixed working routing may impose higher capacity requirements, as compared to the model in Section 16.10.2 where we do a global rerouting. But results in Kodian et al. (2003) for a real-world test network indicate that only a nominal amount of excess capacity is needed over the complete re-optimization design case.

16.11 ONGOING RESEARCH OUTLOOK AND ADDITIONAL RESOURCES

This chapter has recapped the p-cycle concept and collected together for reference a number of the now recognized problem models for design and further research in the topic area. Research in the area is still vigorous. As this chapter goes to press a couple of the most interesting ongoing developments include the extension of p-cycles into end-to-end path-protecting structures called Failure Independent Path Protecting (FIPP) p-cycles. (see Kodian and Grover (2005)). An important advantage of FIPP p-cycles is that, like span protecting p-cycles, the protection paths are fully pre-connected, end-to-end prior to failure. This gives added assurance of end-to-end optical transmission integrity of the protection paths compared to schemes of similar capacity efficiency for end-to-end protection in which optical backup paths have to be assembled "on the fly." This approach also inherently addresses node failures as well as span failures, and provides a simple end-node fault detection and protection activation paradigm. Another area under research is the combined or coordinated design of MPLS-layer node-encircling p-cycles and span-protecting optical layer p-cycles. Conventional span-protecting p-cycles with their fast protection switching and simple mechanism and operation are quite appealing in the optical layer but are not able to deal with node failures. On the other hand, oversubscription-based node-encircling p-cycles are capable of protecting node failures and potentially have high capacity efficiency because controlled oversubscription is allowed. This approach can also employ a "common pool" survivability concept to implement the node encircling MPLS layer p-cycles within the spare capacity used by span-protecting p-cycles, or at least re-using as much of that spare capacity as possible to deal with node failures (or even secondary span failures). Common pool survivability is based on reservation of resources in the optical layer to not only meet the needs of protection within that layer, but also as needed in the MPLS layer. The method allows saving capacity in the case of two-layer recovery procedures. Overall, p-cycles provide many new options for efficient survivable network architecture, as well as promising alternatives to dealing with new challenges such as highly differentiated availability guarantees to services in future networks, and the possibility of rapidly arriving and departing random transport path requirements and fundamental uncertainty in the future demand scenarios which a network design may have to face. To find out more about the ongoing research on p-cycles and the extent to which many aspects of p-cycle network design and operation have already been reduced to practice, see the p-cycles web site: A.Sack and W.D. Grover (2005). That web site will also serve as a forum for updates, addenda, or corrections as needed for this chapter.

Bibliography

D. Arci, G. Maier, A. Pattavina, D. Petecchi, and M. Tornatore. Availability models for protection techniques in WDM networks. In *Proceedings of the 4th International Workshop on Design of Reliable Communication Networks (DRCN 2003)*, pages 158–166, Banff, AB, October 2003.

A.Sack and http:netsys.edm.trlabs.cap-cycles W.D. Grover. Trlabs network systems group *p*-cycles website., June 2005. Online Reference.

P. Batchelor et al. *Ultra high capacity optical transmission networks, Final report of action COST 239*. Faculty of Electrical Engineering and Computing, Zagreb, Croatia, 1999.

V. Chvátal. *Linear Programming*. W. H. Freeman and Company, New York, NY, 1983.

M. Clouqueur and W. D. Grover. Availability analysis or span-restorable mesh networks. *IEEE Journal on Selected Areas in Communications*, 20(4):810–821, May 2002.

M. Clouqueur, W. D. Grover, D. Leung, and O. Shai. Mining the rings: Strategies for ring-to-mesh evolution. In *Proceedings 3rd International Workshop on the Design of Reliable Communication Networks (DRCN 2001)*, pages 113–120, Budapest, Hungary, October 2001.

J. Doucette. Advances on design and analysis of mesh-restorable networks. Master's thesis, University of Alberta, 2004.

J. Doucette, P. Giese, and W. D. Grover. Investigation of node protection strategies with node-encircling *p*-cycles. In *Design of Reliable Communication Networks(DRCN 2005)*, Ischia, Italy, October 2005.

J. Doucette and W. D. Grover. Influence of modularity and economy-of-scale effects on design of mesh-restorable DWDM networks. *IEEE Journal on Selected Areas in Communications*, 18(10):1912–1923, October 2000.

J. Doucette and W. D. Grover. Comparison of mesh protection and restoration schemes and the dependency on graph connectivity. In *Proceedings of 3rd International Workshop on Design of Reliable Communication Networks (DRCN 2001)*, pages 121–128, Budapest, Hungary, October 2001.

J. Doucette and W. D. Grover. Capacity design studies of span-restorable mesh networks with shared-risk link group (SRLG) effects. In *Optical Networking and Communications Conference (OptiComm 2002)*, pages 25–38, Boston, MA, July/August 2002.

J. Doucette, D. He, W. D. Grover, and O. Yang. Algorithmic approaches for efficient enumeration of candidate p-cycles and capacitated p-cycle network design. In *Proceedings of the 4th International Workshop on Design of Reliable Communication Networks (DRCN 2003)*, pages 212–220, October 2003.

W. D. Grover. *Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking*. W. H. Freeman and Company, Upper Saddle River, NJ, 2003.

W. D. Grover. The protected working capacity envelope concept: An alternative paradigm for automated service provisioning. *IEEE Communications Magazine*, 42(1):62–69, January 2004.

W. D. Grover and J. Doucette. Advances in optical network design with p-cycles: Joint optimization and pre-selection of candidate p-cycles. In *Proceedings of the IEEE-LEOS Summer Topical Meeting on All Optical Networking*, pages 49–50, Mont Tremblant, QC, July 2002.

W. D. Grover and D. Stamatelakis. Cycle-oriented distributed preconfiguration: Ring-like speed with mesh-like capacity for self-planning network restoration. In *Proceedings of IEEE International Conference on Communications (ICC 1998)*, pages 537–543, Atlanta, GA, June 1998.

W. D. Grover, B. D. Venables, M. H. MacGregor, and J. H. Sandham. Development and performance verification of a distributed asynchronous protocol for real-time network restoration. *IEEE Journal on Selected Areas in Communications*, 9(1): 112–125, January 1991.

R. R. Iraschko, M. H. MacGregor, and W. D. Grover. Optimal capacity placement for path restoration in STM or ATM mesh-survivable networks. *IEEE/ACM Transactions on Networking*, 6(3):325–336, June 1998.

J. Kennington, E. Olinick, K. Lewis, A. Ortynski, and G. Spiride. Robust solutions for the DWDM routing and provisioning problem: Models and algorithms. *Optical Networks Magazine*, 4(2):74–84, March/April 2003.

A. Kodian and W. D. Grover. Failure independent path protecting p-cycles: Efficient and simple fully pre-connected optical path protection, 2005. in press.

A. Kodian, W. D. Grover, J. Slevinsky, and D. Moore. Ring-mining to p-cycles as a target architecture: Riding demand growth into network efficiency. In *Proceedings of the 19th Annual National Fiber Optics Engineers Conference (NFOEC 2003)*, pages 1543–1552, Orlando, FL, September 2003.

D. Leung and W. D. Grover. Capacity planning of survivable mesh-based transport networks under demand uncertainty. *Journal of Photonic Network Communications*, June 2004a. Submitted.

D. Leung and W. D. Grover. Restorable mesh network design under demand uncertainty: Toward "future proof" transport investments. In *Optical Fiber Communication Conference (OFC 2004)*, Los Angeles, CA, February 2004b.

D. Rajan and A. Atamturk. Survivable network design: Routing of flows and slacks. In G. Anandalingam and S. Raghavan, editors, *Telecommunications Network Design and Management*, pages 65–82. Kluwer, 2002.

A. Sack. New techniques for p-cycle network design. Master's thesis, University of Alberta, 2004.

A. Sack and W. D. Grover. Hamiltonian p-cycles for fiber-level protection in homogeneous and semi-homogeneous optical networks. *IEEE Network, Special Issue on Protection, Restoration, and Disaster Recovery*, 18(2):49–56, March/April 2004.

D. A. Schupke. An ILP for optimal p-cycle selection without cycle enumeration. In *Proceedings of the 8th IFIP Working Conference on Optical Network Design and Modelling (ONDM)*, pages 2761–2765, Ghent, Belgium, February 2004.

D. A. Schupke, W. D. Grover, and M. Clouqueur. Strategies for enhanced dual-failure restorability with static or reconfigurable p-cycle networks. In *Proceedings of IEEE International Conference on Communications (ICC 2004)*, Paris, France, June 2004.

D. A. Schupke, C. G. Gruber, and A. Autenrieth. Optimal configuration of p-cycles in WDM networks. In *Proceedings of IEEE International Conference on Communications (ICC 2002)*, pages 2761–2765, New York, NY, April/May 2002.

G. Shen and W. D. Grover. Exploiting forcer structure to serve uncertain demands and minimize redundancy of p-cycle networks. In *Proceedings of 4th SPIE Optical Networking and Communications Conference (OptiComm 2003)*, pages 59–70, Dallas, TX, October 2003a.

G. Shen and W. D. Grover. Extending the p-cycle concept to path segment protection for span and node failure recovery. *IEEE Journal on Selected Areas in Communications, Optical Communications and Networking Series*, 21(8):1306–1319, October 2003b.

G. Shen and W.D. Grover. Design of protected working capacity envelopes based on p-cycles: An alternative framework for survivable automated lightpath provisioning. In A. Girard, B. Sanso, and F. Vazquez-Abad, editors, *Performance Evaluation*

and Planning Methods for the Next Generation Internet, pages 65–82. Kluwer Academic Publishers, 2004.

D. Stamatelakis and W. D. Grover. IP layer restoration and network planning based on virtual protection cycles. *IEEE Journal on Selected Areas in Communications*, 18(10):1938–1949, October 2000a.

D. Stamatelakis and W. D. Grover. Theoretical underpinnings for the efficiency of restorable networks using pre-configured cycles ("*p*-cycles"). *IEEE Transactions on Communications*, 48(8):1262–1265, August 2000b.

T. Stidsen and T. Thomadsen. Joint optimization of working and *p*-cycle protection capacity. Technical Report IMM Technical Report 2004-8, Informatics and Mathematics Modelling, Technical University of Denmark, May 2004.

W. L. Winston. *Operations research applications and algorithms*. Duxbury Press, Belmont, CA, 3rd edition, 1994.

H. Zhang and O. Yang. Finding protection cycles in DWDM networks. In *Proceedings of IEEE International Conference on Communications (ICC 2002)*, volume 5, pages 2756–2760, New York City, NY, April/May 2002.