OptiComm 2003: Optical Networking and Communications, Dallas, Texas, USA, 13-17 October 2003, pp. 59-70

Exploiting Forcer Structure to Serve Uncertain Demands and Minimize Redundancy of *p*-Cycle Networks

Gangxiang Shen, Wayne D. Grover

TRLabs, 7th Floor, 9107-116St NW, Edmonton, Alberta, Canada T6G 2V4
and Department of Electrical and Computer Engineering, University of Alberta
Contact: {gshen, grover}@trlabs.ca

ABSTRACT

We study the forcer concept in the context of p-cycle based networks. A simple but efficient forcer analysis method is proposed specifically for span-restorable networks in general. Besides identifying forcers, the method is also capable of exploiting extra servable working channels given an initial network spare capacity budget designed for pre-existing working capacities. We find that a large number of extra working channels can be served with no increase in the pre-planned spare capacity budget. This attribute of p-cycle protected networks can be used to enhance their ability to serve unforeseen demand patterns or provide an expanded envelope of protected working capacity within which dynamic demand is servable without blocking due to exceeding the protected working capacity limits.

Keywords: optical networks, *p*-cycles, forcer structure, survivable network design, uncertain demand, capacity planning, redundancy

1 INTRODUCTION

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

This fortuitous combination of features (“ring-speed with mesh-efficiency”) in one scheme lay undiscovered through an entire decade of the ring *versus* mesh debate in the 1990s. Since 1998, however, the basic theory of *p*-cycles as pre-configured structures in the spare capacity of a mesh network has been developed [2], [3], and there have been studies on self-organization of the *p*-cycle sets [4], application of *p*-cycles to the MPLS/IP layer [5], application to DWDM networking [6], studies on joint optimization of working paths and spare capacity [7] and recently on *p*-cycle design with non-simple cycles [8]. In [6] it was found that full survivability against any span cut could be achieved with as little as 39% total redundancy in a well-connected European inter-city network model. Recently, an extension of the basic *p*-cycle concept to protect path segments—called “flow *p*-cycles,” offers even greater efficiency through path-segment oriented protection, but still with greater speed than ordinary path restoration schemes because the flow *p*-cycles remain fully preconnected spare capacity structures [9]. These benefits and advantages motivate yet further exploration of the basic *p*-cycle concept for survivable mesh networking.

In this paper we link the previously developed *forcer concept* from span-restorable mesh networks [10] to *p*-cycle networks. One advance is an improved, much-simplified, method of forcer analysis. With this method, we simultaneously identify and quantify the magnitude of the “forcer spans” of a *p*-cycle network. We then explain how this information gives the flexibility to serve additional or unforeseen demands without any additional investment in spare capacity. With knowledge of the forcer structure we can exploit the fact that although the spare capacity and *p*-cycles may have been designed for an initially forecast demand pattern, the forcer structure of the resultant protected network implies additional working channels can be used on certain spans without any increase in network spare capacity. Following that we show two ways that *p*-cycles can be chosen for a network so as to maximize the network-wide number of working channels that derive protection from the given finite amount of spare capacity. Theoretically, this provides a strategy for planning in the face of complete uncertainty about the actual demand pattern to be encountered. In practice, the approach can be used to cope with high, but not complete uncertainty, by adapting the "utility" coefficients in the model to reflect where servable capacity is most needed. The study is of relevance to network operators interested in designing networks with minimum redundancy and who are also interested in measures to cope with additional demands, or unforeseen demand patterns, beyond which the network was initially designed.

These studies are conducted using two different measures of protection-to-working capacity cost ratio or redundancy. Redundancy is of interest because theoretically the forcer-optimized survivable mesh designs we consider are expected in some cases to realize the *1/(d-1)* "lower bound" on logical redundancy of any span-oriented survivable network. In the logical redundancy capacity is "hop-weighted" or in effect we consider only working and spare channel counts—all spans have the same unit cost per channel. This is more applicable for metro-scale networks where most cost is related to per-channel filters and transponder costs. The other measure of design redundancy is the "distance–weighted" capacity cost which is a more characteristic measure for wide area transport networks—where cost is closer to being proportional to the span's physical distance. Network cost under this assumption is related to the physical distances of working and protection channels between OXC nodes and to the number of optical channels between nodes.

The last contribution of the paper has to do with these redundancy measures. In particular we revisit and refine some existing understandings about the nodal lower bounds for redundancy of span-restorable networks, which include *p*-cycles. This is necessary to explain some of our results, but also timely as there seems recently to be some confusion about the proper interpretation of *1/(d-1)* as a lower bound (where *d* is the average nodal degree). Here, the results for distance-weighted redundancy in some cases go below this value, so we include an explanation of how distance weighting actually allows any possible redundancy level, arbitrarily close to zero, to be achieved. The importance is that this clarifies that *1/(d-1)* needs to be understood (as it was initially proposed) as a lower bound on *logical* redundancy. While still a very practical guideline for real networks, it is not expected to hold in every case of distance-weighted redundancy, and this understanding is necessary to properly appreciate the results obtained here.

The rest of the paper is organized as follows. In Section 2, we review the forcer concept of span-restorable networks, and extend it to *p*-cycle networks. In Section 3, we develop a forcer identification method for *p*-cycle networks, which is general enough to be easily adapted to efficiently identify forcers for other types of span-based survivability networks as well. In Section 4, we show how forcer-manipulation ideas can be used to maximize the total number (or total utility) of protected working channels that a given investment in spare capacity can support, enhancing the ability to cope with uncertain demand patterns. The test methods and network topologies are described in Section 5. The paper is concluded in Section 6. Appendix A contains the considerations about the *1/(d-1)* lower bound on redundancy.

2 FORCER CONCEPT APPLIED TO *p*-CYCLE NETWORKS

The forcer concept was first introduced for span-restorable networks in [10], and then applied to design of ring-mesh hybrid networks in [12]. An interesting observation made in [10] was that knowledge of the forcer structure of a span-restorable mesh network could guide the routing of new service paths in a way that would require no extra spare capacity at all to support survivability of the additional services. Indirectly, the ability to do routing in this way—in the "forcer shadows" as [10] called it is what we further develop and exploit for *p*-cycle based networks. In [10] the forcer concept is defined as follows:

> "*A forcer is any span for which an increase in network total spare capacity is required (to retain restorability) if the span's working capacity is increased.*"

As with span-restorable networks, the concept of forcers is also suitable to *span-protecting p*-cycles[1]. Most generally forcing is a relationship between the working capacity distribution of a network and the corresponding spare capacity distribution required for survivability. The precise relationship is mediated or defined by the details of some re-routing mechanism that applies for restoration. In the case of *p*-cycles, the choice of *p*-cycles employed determines the exact

[1] We say "span-protecting" to distinguish from recent "flow *p*-cycles" [9]. Span-protecting *p*-cycles are the original scheme of *p*-cycles, which carries out protection and restoration on a span-protecting basis. Flow *p*-cycles extend the concept to protect arbitrary segments of working paths, similar to path restoration or protection. In the rest of the paper, without explicitly pointing it out, the term "*p*-cycles" stands for "span-protecting *p*-cycles."

relationship between working capacity and "forced" spare capacity, and in the design models that follow the choice of *p*-cycles is itself a design variable, not a given. When we draw specific *p*-cycles for an example of forcer concepts, and how we can exploit them, we are therefore actually oversimplifying the concept—but doing so for the sake of an example to convey the general ideas.

Fig. 1 is such an example prepared to assist discussion of the forcer concept in the context of *p*-cycles. In Fig. 1(a), a *p*-cycle traversing nodes (0, 4, 8, 11, 12, 9, 7, 3) is configured with 3 spare channels and in (b) a *p*-cycle traversing nodes (1, 5, 10, 11, 8, 4) is configured with 4 spare channels. Both *p*-cycles thus have the same total spare capacity, i.e., 24 spare channels. In a *p*-cycle, the spare capacity can be forced either by the largest working capacity on any on-cycle span of the *p*-cycle or by half of the largest straddling span capacity; whichever is larger. If all we were interested in, however, was the total number of working channels that the *p*-cycle *could* protect, then this is the case where all spans associated with the *p*-cycle *equally co-force* its spare capacity. That is to say that every straddler would have w_i = 2 * `spare`(*p*-cycle) and every on cycle span would have w_i = `spare`(*p*-cycle) where `spare( )` is the number of spare channels on each span of the *p*-cycle. Because every span is an equal co-forcer of the spare capacity of the particular *p*-cycle considered, we say that this is the "forcer-leveled" condition. Its significance is that it represents the maximum working capacity that can be protected by the given investment in spare capacity. Note that this orientation does not assume any demand matrix or forecast at all.

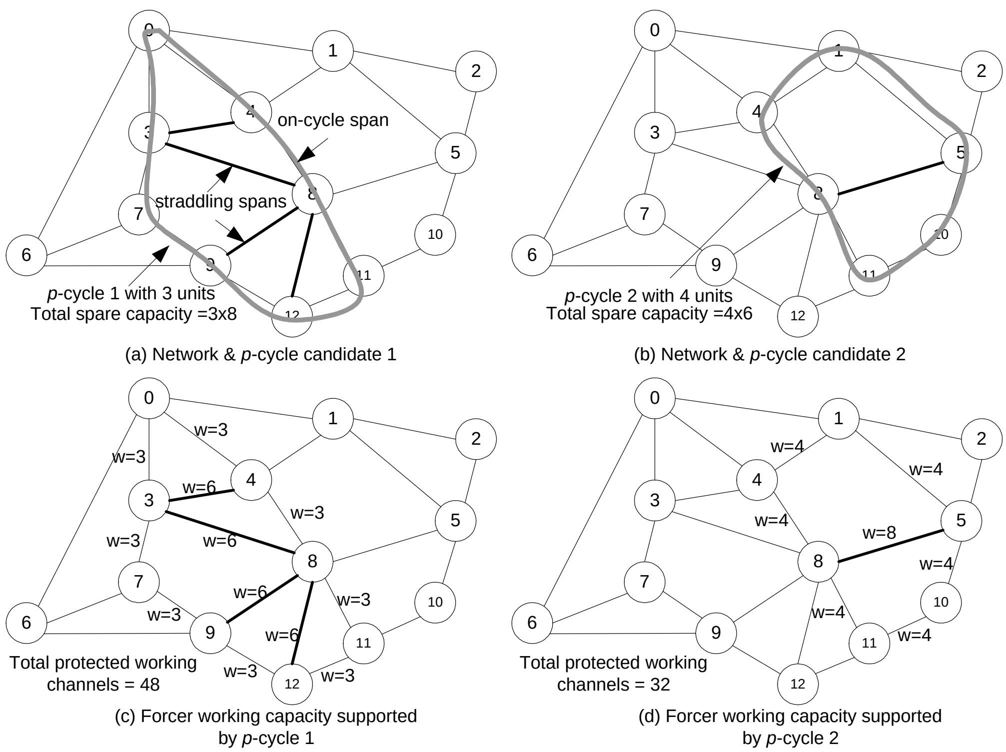

Fig. 1. Examples of the forcer-leveled working capacity distributions for two arbitrary *p*-cycles.

However, depending how a *p*-cycle of a given spare capacity is laid down upon the network, it will have a different ultimate ability to protect working channels, or conversely, different inherent redundancy limits. Thus, optimization will come into play to find a set of *p*-cycles that globally maximizes the protected working capacity potential for a given amount of spare capacity. In the example, we see how one cycle choice can be superior to the other. The maximum potential number of working channels protectable by *p*-cycles 1 and 2 is shown in (c) and (d), respectively. Although the two *p*-cycles consume the same amount of total spare capacity, they offer two different numbers and distributions of total protected working channels. *p*-Cycle 1 can protect up to 48 working channels—in the distribution shown—while *p*-cycle 2 can protect only 32 working channels, although depending on the way actual demand has evolved in the network, the second *p*-cycle's distribution of 32 working channels could possibly have more utility to the network operator than the first *p*-cycle's potential to support 48 turned-up working channels. Our work in the following sections will focus on how to select the *p*-cycles efficiently so as to maximize the total number working channels that are protected for the given investment in spare capacity. This is referred to as the "forcer-maximized" condition.

Note in the discussion so far, and in what follows, the initially unusual reversal of what we consider to be the "givens" and the dependent variables of a typical survivable design problem. Overwhelmingly so far in the field we always look to a demand matrix or a set of span-working capacities to be given as a requirement. Then, the spare capacity, backup paths and/or any protection structures or preplans etc., are deduced for minimum added cost. But here we are reversing things in an important way. We will be saying, if the *spare* capacity is given (perhaps indeed as the result of a conventional design but whose forecast accuracy is now questioned), then what measures can at least support the maximum number of potentially needed working channels to face actual demand that might arise? In other words how can we best use an existing or limited investment in protection resources to maximize the number of working channels that we would be free to turn up to serve uncertain demand as needed? As stated, the answer to this question is always some form of *p*-cycle configuration whose forcer-leveled condition yields the maximum total number of working channels that can be protected. The significance of this approach is also that it represents a meaningful network planning strategy that can (if desired) be utterly devoid of any forecast requirements at all! Alternately, we will see how *a priori* notions or experience about where working capacity is most important can be introduced. Fig. 2 summarizes the difference and relationship between the regular spare capacity minimized design and the proposed forcer-maximized design.

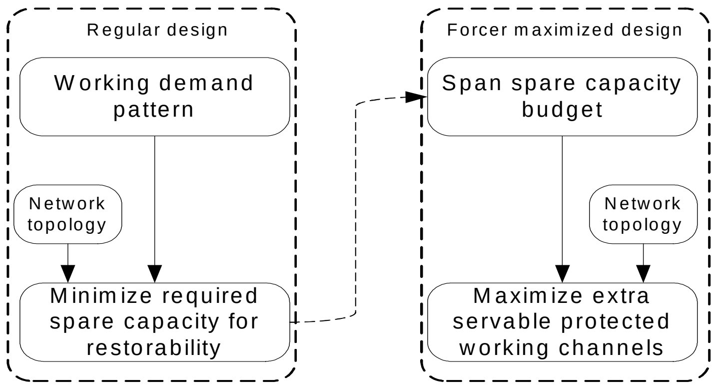

Fig. 2. Relationship between regular spare capacity minimized designs and forcer-maximized design for uncertainty.

3 EFFICIENT FORCER ANALYSIS METHOD

Our first step towards this will be to present an optimization-problem approach to identify the forcer spans of a conventionally-designed survivable network, and the magnitudes they have as forcers of the surrounding spare capacity distribution. Two options for forcer identification methods were previously proposed in [10], [12] for span-restorable mesh networks. One applied k-shortest paths routing to search all the forcers in a network [10]. The other employed repeated runs of a relaxed ILP model for spare capacity allocation solutions to identify the forcers [12]. The details of the second approach can be described as follows. Based on a given network topology and working capacity quantities on spans, an optimization model is run to find the minimal feasible total network spare capacity cost. One span is then selected from the network for a test run. A single extra working capacity unit is increased for the span, and the optimization process is rerun to see if there is any required increase in the total spare capacity cost. If there is any, then the tested span is as a forcer; otherwise, it is not. Following a return of the first test condition to normal, another span is selected from the network for the same test as described. The process is terminated when all the spans are tested. Based on this method, for a network with M spans, M such tests are to be performed. For a large-scale network, the above method can be time-consuming to run the ILP M times. Moreover, if we want to know how many extra channels can be added before a non-forcer becomes a forcer, it can be an even more tedious task with the ILP method. The simplest way is to add a single capacity unit one by one to the span until this span becomes a forcer. And a more efficient approach is to apply the method of binary section by adding an amount of channels, which is a half of that of the previous test. But for both of them, it is very difficult to predict when a span will become a forcer, so the total testing rounds of the whole network are also unpredictable, but at least we can foresee that it is much larger than M rounds for an M-span network.

In view of this, a more effective optimization-based approach for forcer identification is now proposed, which requires only one run of the optimization model to detect and quantify all the forcers associated with a given spare capacity distribution. This gives us full information to know how many extra working channels can actually be used for dynamic or unforeseen service provisioning uses on each span while strictly requiring zero extra protection capacity to retain full restorability. The parameters and variables of the optimization design model are as follows:

- $\boldsymbol{S}$ is the set of spans of a network.
- $\boldsymbol{P}$ is the set of all eligible cycles of the network.
- X_i^j takes the value of two if span *i* is a straddler on cycle *j*, one if span *i* is an on-cycle span, zero otherwise.
- P_k^j takes the value of one, if cycle *j* uses span *k*, zero otherwise.
- w_k is the number of existing working channels on span *k*.
- s_k is the number of assigned spare channels on span *k*.
- n_j is the number of copies of unit cycle *j* preconfigured to offer span failure protection.
- e_k is the number of extra working channels that can be served on span *k* without increasing the current spare capacity budget on any other spans.

The objective is to maximize the total number of extra channels (or distance-weighted mass of transmission capacity) that it is possible to serve without adding spare capacity. In the context of this problem it is important to note that both w_k and s_k are given as *inputs*. In other words, they are properties of some *existing* design on which we are performing the forcer analysis. The set of *p*-cycles associated with the initial design are disregarded, however, and new n_j values are solved for that correspond to the forcer-maximized condition for the given spare capacity environment. The *side-effect* of this maximization problem is the identification and quantification of the non-forcer spans (hence also the forcers).

Forcer Analysis (FA): maximize $\left\{\sum_{k\in \boldsymbol{S}} e_k\right\}$ or maximize $\left\{\sum_{k\in \boldsymbol{S}} c_k \cdot e_k\right\}$

Constraints:

$$\sum_{j\in \boldsymbol{P}} \left(X_i^j \cdot n_j\right) \ge w_i + e_i \quad \forall i \in \boldsymbol{S} \tag{1}$$

$$s_k \ge \sum_{j\in \boldsymbol{P}} \left(P_k^j \cdot n_j\right) \quad \forall k \in \boldsymbol{S} \tag{2}$$

Constraint (1) says that all existing working capacity on span *i* upon a cut should be fully restored by *p*-cycles, in addition to any extra working channels that it would also be possible to restore. Constraint (2) guarantees that the *p*-cycle set chosen for (1) does not require more spare capacity than there actually is in the existing design on span *k*. From this optimization model, we discover how many extra working channels (e_k) can be served on each span. If, in the solution, e_k is zero, then the span is a forcer of the original network design. When not zero, e_k reflects the depth of the span below forcer status (we call this its non-forcer magnitude or forcing margin). In this case e_k is a direct measurement of the number of extra working channels that could be turned-up for service on span *k* before having any impact on the spare capacity requirement of the network as a whole (to retain 100% restorability).

4 MODELS FOR FORCER-RELATED MAXIMIZATION OF SERVING CAPACITY

Thus, we now see how a certain maximization problem can reveal the forcer structure of spans in an existing *p*-cycle based network. This is a useful tool in its own right, but we really want to use it just as a stepping-stone to the next concept. If we think about how the forcer analysis model works, we can see that it really is answering the question: by how much could the working capacity of each span be increased without requiring any increase in spare capacity (or new *p*-cycles) to remain survivable. But this is not that different from two questions that we imagine a network operator might pose in the face of uncertainty about where demand is going to arise. One question is: If I have an existing set of available spare capacities, which set of *p*-cycles built within this spare capacity will support the greatest number of protected working channels (with which to serve unpredictable demands)? We say this is a related question because the answer to the latter is, theoretically, to form the set of *p*-cycles where the forcer structure is the most “leveled” in the sense of [12] and further is maximal in the sense described when choosing cycle 1 in Fig. 1 over cycle 2. But in general we can do better than simply protecting the maximum number of working channels anywhere in the network—not all spans may be of equal importance. We can therefore target or structure the potential to support new working channels in the optimization by assigning a *utility* value associated with working channels on each span *k*. For generality, there could also be simple maximum and minimum requirements for working channels on each span. Therefore we define:

- u_k (where used) is the utility or extra value associated with having a working channel available for service on span *k*.
- w_k^{max} / w_k^{min} (where used) is an explicit limit to the maximum / minimum number of working channels to be protected for service on span *k*.

Let us now consider two models for planning that maximize the number (or utility) of working channels that can be turned up in a network without any increase beyond either an existing set of spare capacities or a total spare capacity budget. These are forcer-inspired optimization models to support the maximum number of working channels with which to serve dynamic demand, for a given amount of spare capacity. Conversely these models lead to realization of *p*-cycle network designs with the lowest possible ratio of spare to available working capacity. This is approached under two specific situations:

(1) Problem **FM-1**: Where a set of available spare capacities already exist on each span, and
(2) Problem **FM-2**: Where a budget limit is given on the total allowable spare capacity in the network.

In both situations, we determine the configuration of *p*-cycles so that the maximum number of protected working channels is provided for service provisioning over the network as a whole, or in a utility-weighted sense where planners can indicate the relative importance of being able to support additional demand on certain spans. This is seen as a possible strategy of "future-proof" design for uncertain demand patterns.

4.1 Maximum Working Channels Within Existing Spare Capacities on Spans

In this model we assume only a given set of spare channel counts on each span. These may have typically arisen from a nominal design to a nominal forecast, or they may simply be the unused capacities of existing spans. Within this spare capacity environment the problem is to form a set of *p*-cycles that then protects the largest possible total number of working channels on spans of the network as a whole. This would create a network where in the face of a basic belief about the expected demand pattern we still prepare the maximum amount of protected working capacity possible within which to try to serve dynamic or unforeseen demand patterns. With existing spare channel numbers s_k on span *k*, the model forms a set of *p*-cycles within this capacity which permits the largest possible number of (protected) working channels to be turned up over the network as a whole, without exceeding the spare capacity of each span. More generally, the objective is to maximize the total "demand serving utility" based on u_k coefficients representing the relative desirability of creating protected working capacity on each specific network span. For instance, recent congestion or blocking experience arising from previously mis-forecast conditions would tell an operator to place greater utility on certain spans than on others. In this way the following formulation could be used to "track" an uncertain evolving demand pattern, by changing the logical configuration of *p*-cycles within existing spare capacity, to support additional working channels where most needed in the network. There is no added investment in spare capacity and because the *p*-cycles employed are formed by cross-connections between spare channels, the "shape" of the protection pattern can be updated by re-running the model as needed. This *p*-cycle design model is:

Forcer Maximization-1 (FM-1): $\text{maximize}\left\{\sum_{k\in \boldsymbol{S}} u_k \cdot w_k\right\}$

Constraints:

$$\sum_{j\in \boldsymbol{P}} \left(X_i^j \cdot n_j\right) \geq w_i \quad \forall i \in \boldsymbol{S} \tag{3}$$

$$s_k \geq \sum_{j\in \boldsymbol{P}} \left(P_k^j \cdot n_j\right) \quad \forall k \in \boldsymbol{S} \tag{4}$$

$$w_i^{max} \geq w_i \geq w_i^{min} \quad \forall i \in \boldsymbol{S} \tag{5}$$

4.2 Maximum Working Channels Given a Budget Total on Spare Capacity

In this model we are allowed to place spare capacity (and commission related *p*-cycles) in any way desired up to a given total network spare capacity budget *B*. As above, there is no requirement to serve any specific demand matrix, but rather we are interested in creating simply the maximum network-wide total of protected working-channel volume for serving demand. This would create a network where—in the face of no forecast idea about the demand, we can nonetheless strictly prepare the maximum amount of protected working capacity possible within which to try to serve dynamic or unforeseen demand patterns. If we have a rough but uncertain idea of the demand pattern, we can take a characteristic example demand matrix, shortest-path route it over the graph and then use the relative span loads as the utility coefficients in the optimization. In this model the ILP solver directly identifies a pattern of working capacities that just fully loads the most efficient set of *p*-cycles that can be formed under the allowed total budget on spare capacity. This pattern and corresponding protection structures are of special interest in their own right because it represents minimum-redundancy structure. It also tells an operator that for a given total budget on spare capacity, and his given facility route topology, what pattern of demand would actually maximize his protected-capacity utilization for the given investment in total protection capacity.

As above, the objective is to maximize the total utility of working capacity that it is possible to provide given an overall spare capacity budget of a network. Note that in this model the s_k values become variables, no longer an input to the problem. Now we also need to consider the cost of a unit capacity on each span to stay under the total budget, so we add to above:

- c_k is the unit capacity cost of span *k*.

The model is:

Forcer Maximization-2 (FM-2): $$\text{maximize}\left\{\sum_{k\in \mathbf{S}} u_k \cdot w_k\right\}$$

Constraints:

Subject to (3), (4) and optionally the limits (5) above, to which we add

$$\sum_{k\in \mathbf{S}} \left(c_k \cdot s_k\right) \le B \tag{6}$$

5 TEST METHOD AND RESULTS

We did experiments on five networks, shown in Fig. 3. These networks include ARPA2, NSFNET, SmallNet, COST239, and a topology from [15]. The numbers of nodes, spans, and elemental candidate cycles for each network are listed in Fig. 3. Here the term “cycles” denotes eligible cycle candidates from which the design models may select actual *p*-cycles for the design solutions. These are not the number of *p*-cycles in the designs themselves. The Euclidean length of each span is taken from the network topology as drawn and shown by the span. The span lengths are used in some contexts as the c_k values, in others as the utility weights, u_k. Lightpath demands were generated following a uniform random distribution on the range [1…20] for each node-pair. For the Level3 case only the largest 30% of demand pair volumes resulting from that initial assignment of demands are employed. A single shortest route is used for each working demand but the design models consider all elemental cycles for the initial (conventional) spare capacity planning problems and later forcer-maximization problems FM-1, FM-2. The design problems were all solved to complete optimality within half an hour with AMPL/CPLEX 7.1.

5.1 Forcer Analysis and Identification of Extra Servable Working Channels in Baseline Designs

Using the forcer analysis model **FA**, we identified the forcers of the five test networks. The forcer spans are highlighted in the figures with thicker lines. Among them, ARPA2 has eight forcers, NSFNET has ten, SmallNet has eight, COST239 has six, and Level3 has eleven. In addition, beside each non-forcer span we indicate how many extra working channels can be provided before that span would itself take on a forcer role. These values directly represent additional demand-serving capacity on these spans, which will have no requirement for extra *p*-cycles or spare capacity.

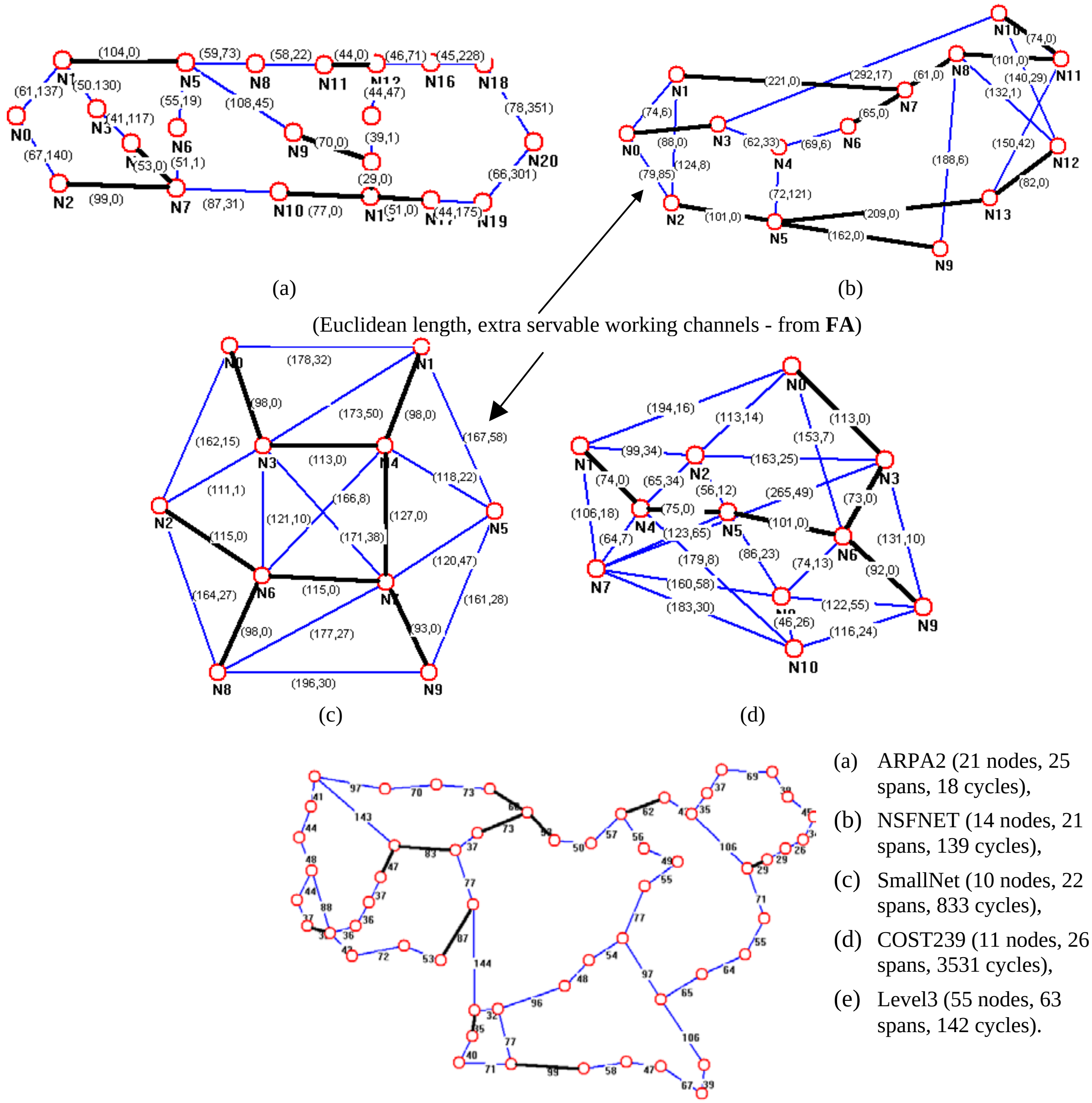

Fig. 3. The test network topologies: except for (e) the brackets by each span gives the length of the span and the number of extra working channels that can be served under the spare capacity of the basic designs (from forcer analysis model **FA**).

As in [10], all the forcers make up a "skeleton" network, and all other spans exist in the "forcer shadow" of this skeleton. The forcer shadow is a subset of the network on which you can serve more working channels with the spare capacity for protection being provided for "free"—it is an already built-in side-effect of providing protection for the forcer skeleton spans alone. It is therefore preferable when possible to satisfy new path requests fully or as much as possible within the forcer shadow. For example, in ARPA2, if there is a new connection request between N0 and N3, we may select the route N0-N1-N3 to serve it, and on that route, up to 130 extra demands can be accommodated without any impact on spare capacity requirement.

Table 1 summarizes the **FA** results for the networks as a whole indicating the overall percentage of extra working channels that can be served without increase in spare capacity. Results are presented for both objective function options shown for **FA**, i.e., simple channel or hop counts and distance-weighted transmission totals. Results show the ability to turn up from 16% to 78% more working capacity in any of these networks above and beyond the initial demand matrices for which they were designed. These working capacity increases are supported for free in the sense that no spare capacity increase is needed to retain this 100% restorability.

Table 1. Percentages of extra working channels that can be served without any increase in total spare capacity budget

Networks	ARPA2	NSFNET	SmallNet	COST 239	Level3
Distance	25.7%	16.2%	70.8%	78.0%	47.8%
Hop	20.4%	32.6%	49.4%	21.9%	36.7%

5.2 Spare Capacity Redundancy of FM-1 and FM-2 Forcer-Maximized Designs

We now study the spare capacity redundancy for *p*-cycle networks under FM-1 and FM-2 design and compare with other popular network survivability techniques. These include span-restorable networks [13], path restoration with and without stub release [14], and flow *p*-cycles [9]. For each these techniques, we carried out the same investigations under the scenarios and assumptions similar to those of span *p*-cycles. Table 2 reports the redundancies of various survivability techniques under various scenarios for the three networks, NSFNET, ARPA2, and SmallNet, where numbers from “1” to “4” correspond to the design scenarios as described. Column 1 corresponds to the conventional survivable design models without any change from [1], [12], [8], [13] respectively. In other words these are the reference designs produced in the usual way starting with a specific demand matrices mentioned above. They serve only to define the span-wise and total spare capacity budget references for FM-1 and FM-2 cases, respectively. Column 2 gives results of the FM-1 forcer-maximized designs with specific working and spare capacities given on each span from the designs of Column 1. Column 3 gives the corresponding FM-1 results where the spare capacities are again as given from column 1 designs, but the working channel counts from column 1 are not enforced as minimums. Column 4 shows the results of the FM-2 forcer-maximized models which have the largest freedom to create servable working channels—there are no per-span constraints, only a total network spare capacity budget, taken from the designs of column 1. In addition, for comparison we note *1/(d-1)* for each of the three test networks in the table. Each FM-1, FM-2 design is conducted under the two contexts: one where the utility is 1 for every channel (“hop”), the other is where the utility is taken as the span distances (“distance”).

Table 2. Spare capacity redundancies of various survivable architectures under forcer-maximized design

Networks		NSFNET (1/(d-1)=0.5)				ARPA2 (1/(d-1)=0.724)				SmallNet (1/(d-1)=0.294)			
		1	2	3	4	1	2	3	4	1	2	3	4
Span *p*-cycles	Hop	0.722	0.545	0.545	0.501	1.170	0.972	0.972	0.875	0.480	0.321	0.321	0.296
	Distance	0.918	0.790	0.752	0.398	1.252	0.995	0.995	0.937	0.620	0.363	0.363	0.254
Span restoration	Hop	0.708	0.585	0.585	0.5004	1.118	0.993	0.993	0.834	0.475	0.313	0.313	0.294
	Distance	0.899	0.706	0.706	0.400	1.159	1.020	1.020	0.789	0.618	0.380	0.380	0.253
Flow *p*-cycles	Hop	0.522	0.380	0.355	0.260	1.022	0.525	0.512	0.448	0.358	0.245	0.240	0.224
	Distance	0.689	0.538	0.508	0.348	1.093	0.577	0.565	0.472	0.523	0.304	0.276	0.245
Path restoration (no stub release)	Hop	0.500	0.381	0.323	0.260	0.862	0.568	0.522	0.421	0.352	0.237	0.232	0.208
	Distance	0.662	0.520	0.492	0.341	0.909	0.631	0.595	0.473	0.497	0.330	0.304	0.240
Path restoration (stub release)	Hop	0.404	0.304	0.288	0.224	0.832	0.578	0.517	0.413	0.330	0.220	0.216	0.195
	Distance	0.549	0.444	0.437	0.326	0.844	0.697	0.613	0.475	0.445	0.270	0.256	0.210

From design scenario 1 to 4, the redundancy becomes progressively lower when more constraints are released. As expected, FM-2 (column 4) designs shows the lowest redundancy among all cases. For span-based survivability techniques, which include span *p*-cycles and span-restorable mesh networks, the logical redundancy never goes below *1/(d-1)*. For example, NSFNET’s *1/(d-1)* is 50%. The redundancies of all the design scenarios are never smaller than this. Similarly for the other networks. However, the logical redundancy bound would not necessarily be expected to apply in all cases when the network design is optimized and characterized by distance-weighted measures, and we see this in Table 2. Some of the distance-weighted redundancies of both span *p*-cycles and span restorable network are below *1/(d-1)* for both NSFNET and SmallNet. The reason this can be expected is treated in Appendix A.

Comparing the redundancies of span-restorable networks and *p*-cycle networks, we find that in most cases the former has lower redundancy but in some cases, the opposite is true. We ascribe this “abnormality” to the different spare capacity budgets employed by the two methods for use in FM-2. Given an initial demand matrix and corresponding conventional design, a span-restorable network should require the same or somewhat less spare capacity than a *p*-cycle network. With a smaller spare capacity budget, a span-restorable network may then only support a smaller number of working channels, which therefore can result in a higher redundancy of the forcer-maximized design. In contrast if the initial *p*-cycle design needs more spare capacity in the Column 1 design, then using this spare capacity total as a design budget for FM-1 and FM-2 problems, it is possible that more working channels may be supported in total.

If we compare the redundancy of the span-based and the path-based survivability techniques, we know that path restoration with stub release should serve as a lower bound on redundancy for all the survivability techniques. Indeed, in the data, the path-based techniques always outperform the span-based techniques under logical redundancy with an almost unchanged improvement in redundancy from column 1 to 4. Under distance-weighted measures, however, the difference varies more. For example, for NSFNET, the difference in logical redundancy between span *p*-cycles and path restoration with stub release is about 32% in the initial designs and about 28% in FM-2 designs. But under distance-weighting the difference is about 37% to start with and only about 7% for FM-2 designs. The reason is attributed to *1/(d-1)* effects. The logical redundancy of a span-based restoration scheme cannot get below this limit, but under length–based measures the redundancy can go lower to continue to compete with the path-oriented schemes. Additionally, for distance-weighted FM-2 designs (column 4) we observe that under certain conditions a span-based technique may achieve redundancy very close to that of the path-based techniques. For example, in SmallNet the maximum distance-weighted redundancy difference between span-based and path-based survivability techniques is only 4.4% (i.e., span *p*-cycles versus path restoration with stub release). A similar observation can be made for the NSFNET network as well.

6 CONCLUDING DISCUSSION

In this paper, we exploited the forcer concept as it applies to *p*-cycle protected networks. A simple but efficient solution to an optimization problem identifies and quantifies the forcer structure of an existing *p*-cycle network. This indicates directly how many extra channels could actually be used to serve dynamic traffic or unforeseen demands over each span, without requiring any revision of the spare capacity design or protection arrangements whatsoever. Results show that typically quite significant numbers of extra working channels can be used with no increase in spare capacity budget. Two other forcer-inspired design models produce new ways to look at planning networks for unforeseen or dynamic demands. In the first (FM-1), a set of span spare capacities are given, but we use them to prepare a set of *p*-cycles under which the network-wide maximum number of working channels are protected, creating an inherently protected working channel pool within which to serve dynamic and/or uncertain demand. The last model (FM-2) does the same but also decides where to place spare capacity, and what *p*-cycles to form so that the total pool of protected working channels is even further increased under only a total network wide budget limit on spare capacity. Both of the models can either maximize the pool of working channels available in a simple bulk basis over the network as a whole or be guided by utility values and/or minimums and maximums associated with different spans. The FM-2 designs are of special theoretical interest as well because the implied demand pattern that just fully loads this structure is of amazingly high efficiency and could suggests various business-planning targets for certain network operators. We also considered the spare capacity redundancy of the forcer-maximized designs and explained why the distance-weighted redundancy can go under the *1/(d-1)* lower bound on logical redundancy.

7 REFERENCES

[1] W. D. Grover, D. Stamatelakis, "Cycle-oriented distributed preconfiguration: Ring-like speed with mesh-like capacity for self-planning network restoration," *ICC'98*, 1998, pp. 537-543.

[2] D. Stamatelakis, Theory & Algorithms for Preconfiguration of Spare Capacity in Mesh Restorable Networks, M.Sc. Thesis, Univ. Alberta, Canada, 1997.

[3] D. Stamatelakis, W.D. Grover, "Theoretical Underpinnings for the efficiency of restorable networks using pre-configured cycles ("*p*-cycles")" *IEEE Trans. Comm*, vol.48, no.8, Aug. 2000, pp. 1262-65.

[4] D. Stamatelakis, W.D. Grover, "OPNET Simulation of Self-organizing Restorable SONET Mesh Transport Networks", in *Proc. OPNETWORKS '98 Conference* (CD-ROM), Washington, D.C., April 24-25 1998, paper 04.

[5] D. Stamatelakis, W. D. Grover, "IP layer restoration and network planning based on virtual protection cycles," *IEEE JSAC*, vol.18, no.10, Oct. 2000, pp. 1938 - 1949.

[6] D. A. Schupke, C. G. Gruber, and A. Autenrieth, "Optimal configuration of *p*-cycles in WDM networks," in *Proc. of IEEE ICC'02*, NY City, Apr. 28-May 2, 2002.

[7] W. D. Grover, J. E. Doucette, "Advances in optical network design with *p*-cycles: Joint optimization and pre-selection of candidate *p*-cycles," *Proc. IEEE-LEOS Topical Meetings*, Quebec, July 15-17, 2002, paper WA2 pp.49-50.

[8] C. G. Gruber, "Resilient networks with non-simple *p*-Cycles," in Proc. *International Conference on Telecommunications (ICT 2003)*, Papeete, Tahiti, French Polynesia, Feb. 23 - Mar. 1, 2003.

[9] W. D. Grover, G. Shen, "Extending the *p*-cycle concept to path-segment protection," *ICC'03*, Anchorage, May 10-15, 2003, pp. 1314-1319.

[10] W. D. Grover, D. Y. Li, "The forcer concept and express route planning in mesh-survivable networks," *Journal of Network and Systems Management*, vol. 7, no. 2, Jun. 1999, pp. 199-223.

[11] W.D. Grover, J. Doucette, "Topological design of survivable mesh-based transport networks," *Annals of Operations Research*: *Topological Network Design in Telecommunication Systems*, vol. 106 (2001), September 2001, pp.79-125.

[12] W. D. Grover, R. G. Martens, "Optimized Design of Ring-Mesh Hybrid Networks," in *Proc. IEEE / VDE Proc. Design of Reliable Communication Networks (DRCN 2000)*, Munich, Germany, April 2000, pp. 291-297.

[13] M. Herzberg, S. Bye, "An optimal spare-capacity assignment model for survivable network with hop limits," *Proc. of IEEE GLOBECOM'94*, pp. 1601-1607.

[14] R. R. Iraschko, M.H. MacGregor, and W.D. Grover, "Optimal capacity placement for path restoration in STM or ATM mesh survivable networks," *IEEE Trans. Net.*, vol. 6, no. 3, June 1998, pp. 325-336.

[15] Network map available at www.level3.com

Appendix A: On the Nodal Lower Bound on Logical Redundancy

Under the measure of logical capacity redundancy, *1/(d-1)* is a well-known lower bound on spare capacity redundancy for span-based survivability networks [11]. However, for networks where we may be more interested in the distance–weighted redundancy the above measure may still be quite characteristic in practice, but not strictly an unbreakable lower bound. In this section, we explain.

To begin with let us revisit the basic result of *1/(d-1)* to see the considerations from whence it arises. Fig. A.1 illustrates a general node model for the analysis of the isolated nodal redundancy lower bound. The nodal degree is *d*, and the working channels of each span around the node are listed by each span. When there is a span cut, say span 1, the affected working channels w1 are restored by some alternate routes that start from the node, passing through the other incident spans, excluding the failed span. For any span failure *i*, the following relationship must be satisfied in order to guarantee 100% span failure restoration. In other words, this is a necessary but not by itself sufficient condition for restorability and hence forms a lower bound on the total spare capacity at each node.

$$\sum_{j=1,\, j\neq i}^{d} s_j \geq w_i \tag{A.1}$$

This relationship means that there must be enough total spare capacity on the remaining $d-1$ neighboring spans, which are able to restore the affected working capacity on the failed span.

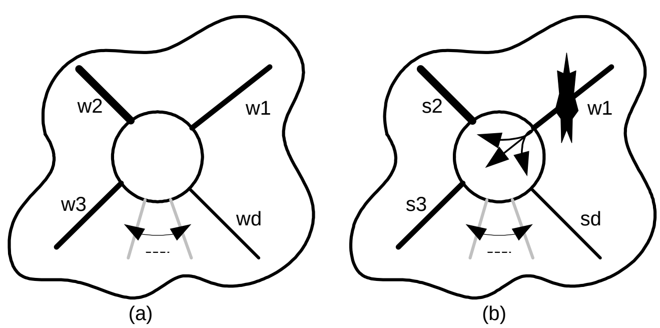

Fig. A.1 Isolated node model for considerations of logical redundancy [11]

From this it follows that:

$$R_1 = \sum_{j=1}^{d} s_j \Big/ \sum_{j=1}^{d} w_j \geq \sum_{j=1}^{d} s_j \Big/ \sum_{j=1}^{d} \sum_{i=1,i\neq j}^{d} s_i = \sum_{j=1}^{d} s_j \Big/ \left((d-1) \cdot \sum_{j=1}^{d} s_j \right) = 1 \big/ (d-1) \tag{A.2}$$

and this lower bound is effective for any w_j combination for a network with hop-weighted cost, in other words the logical channel redundancy. The derivation allows each span to have a different number of working channels; no prerequisite like equal working channels on all the *d* spans is required, but the lower bound arrives when all are equal.

For a network with distance-weighted span costs, the basic expression of redundancy becomes:

$$R_2 = \sum_{j=1}^{d} c_j \cdot s_j \Big/ \sum_{j=1}^{d} c_j \cdot w_j \tag{A.3}$$

where c_k is the distance or cost of the span. Now, by substituting (A.1) for the denominator in (A.3) we obtain

$$R_2 = \sum_{j=1}^{d} c_j \cdot s_j \Big/ \sum_{j=1}^{d} c_j \cdot w_j \geq \sum_{j=1}^{d} c_j \cdot s_j \Big/ \sum_{j=1}^{d} \left(c_j \cdot \sum_{i=1,i\neq j}^{d} s_i \right) \tag{A.4}$$

Because s_j is a variable, which can be infinite or zero, as an extreme case we assume that only s_k is non-zero, but all the other spans have zero (or very small) $s_j, j \neq k$. In that case we can have

$$R_2 \geq \sum_{j=1}^{d} c_j \cdot s_j \Big/ \sum_{j=1}^{d} \left(c_j \cdot \sum_{i=1,i\neq j}^{d} s_i \right) = (c_k \cdot s_k) \Big/ \sum_{j=1,j\neq k}^{d} (c_j \cdot s_k) = c_k \Big/ \sum_{j=1,j\neq k}^{d} c_j \tag{A.5}$$

c_j is also a cost parameter which can be infinite or close to zero. As an extreme case if we assume c_k is very small and/or one of $c_j, j \neq k$ is very large, then the term $c_k \Big/ \sum_{j=1,j\neq k}^{d} c_j$ goes arbitrarily close to zero. Therefore, we can conclude that under the measure of distance-weighted span unit cost, the redundancy R_2 can approach a limit of zero. This is not hard to appreciate, however. It arises simply from assuming arbitrary cost weights applied to situations where the logical redundancy is still above *1/(d-1)*. Fig. A.2 illustrates how node redundancy with distance-weighted cost can be below 1*/(d-1)* while the logical redundancy is above it still. The node has three incident spans. The distance-weighted span cost, working channels, and spare channels of each span are indicated. The distance-weighted redundancy of the node is only about 1%, much smaller than 50%, which is *1/(d-1)* here, while the logical channel redundancy is still very much above *1/(d-1)*, in fact it is 98%.

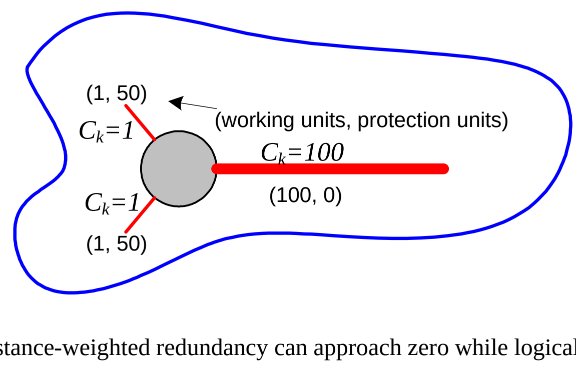

Fig. A.2 Example showing how distance-weighted redundancy can approach zero while logical redundancy remains above *1/(d-1)*.