Efficient Design of *p*-Cycles for Survivability of WDM Networks Through Distributed Cycle Pre-Configuration (DCPC) Protocol

Hari Mohan Singh
Computer Science and I.T. *harimohansingh@gmail.com*
SHIATS
Allahabad-211007, India

Rama Shankar Yadav
Computer Science & Engineering *rsy@mnnit.ac.in*
MNNIT
Allahabad-211004, India

Raghav Yadav
Computer Science and I.T. *raghavyash@gmail.com*
SHIATS
Allahabad-211007, India

Abstract

The optical networks provide the backbone infrastructure for telecommunication networks. Because of the high-speed of optical networks, network failure such as a cable cut or node failure may result in a tremendous loss of data and hence revenue received. The *p*-cycle is a novel approach reported for design of survivable optical WDM networks. They are preconfigured protection structure, combining fast restoration speed of ring and mesh protection efficiency. The main issue in *p*-cycle network design is to find a set of *p*-cycles to protect a given working capacity distribution so that total spare capacity used by the *p*-cycles is minimized. An Integer Linear Programming (ILP) is the most efficient method reported in the literature for designing of optimal *p*-cycles. Where complexity of ILP increases as the size of network increases, i.e., it is not so efficient in case of large networks. Recently, a new, promising concept to support dynamic demand environments has been introduced by Grover namely, the distributed cycle pre-configuration (DCPC) protocol, which is an adaptation of the processing rule of the self-healing network (SHN). However, it is generally unable to provide 100% protection of the working capacity under Spare Capacity Optimization (SCO) design model. Therefore in this paper we have proposed enhancements in DCPC to increase its protection level under single failure scenario. The main idea behind the proposed enhancement is it to fix the span as a straddle span of a *p*-cycle where unprotected working capacity is more. From the simulation of test case networks, it is found that the proposed scheme significantly increases ratio of protection under the SCO design model.

Keywords: WDM, *p*-cycle, Integer Linear Programming (ILP), Distributed Cycle Pre-Configuration (DCPC) and Spare Capacity Optimization (SCO).

1. INTRODUCTION

Bandwidth demands are increasing continuously with explosive spread of networks deployments and emerging applications such as Voice over IP and e-commerce. A high demand of bandwidth is prompting ISPs to switch to Optical Networks. Optical networks based on WDM technology can potentially transfer several gigabytes per second of data on each fiber link in the network. A failure in a WDM network such as a cable cut or node failure may result in a tremendous loss of data and hence revenue received. This makes survivability a major concern for today's networks designers. The network survivability technology can be classified into two categories: Protection and Restoration. In protection, the dedicated backup paths are configured over spare capacity before the occurrence of failure. Since only two real times switching are required to obtain survivability of affected traffic and therefore recovery speed is very quick under such category of survivability schemes. In restoration category, backup paths for affected the traffic will be configured after occurrence of failure. In paper [20] authors conducted a study amongst dedicated protection paths and shared protection paths. The result of the study shows that shared backup path protection has significant capacity efficiency as compared to dedicated path protection. The basic protection schemes are APS, UPSR and BLSR whereas Shared Backup Path Protection (SBPP) is most popular scheme reported under restoration category [1-4].

The search for improving recovery switching time and reducing capacity redundancy leads to the discovery of preconfigured protection cycle (*p*-cycle), introduced by Grover et al. [4-7]. The *p*-cycles combine best part of ring protection schemes (UPSR, BLSR) and mesh restoration scheme (SBPP). It performs switching as fast as ring like restoration (50-60 msec.) and capacity efficient approximately like mesh restoration. The *p*-cycles are ring-like pre-configured structure formed over spare capacity available in the network. A unit *p*-cycle is composed of one spare channel of each span it crosses. The span of the network which is traversed by a *p*-cycle is referred as its on-cycle span whereas the span that has both the end nodes on the cycle but not themselves on the cycle is called straddle span of a *p*-cycle. A *p*-cycle provides one protection path for on-cycle span failure whereas in case of failure of straddle span it provides two protection paths. Hence, their efficiency can be as good as the efficiency of mesh survivable networks. The working of *p*-cycle can be observed from the diagram shown in Figure 1. Fig. 1(a), dark line shows an example of a *p*-cycle. In fig.1 (b), a span on the cycle breaks and the surviving portion of the *p*-cycle is used for restoration. Fig.1(c) shows two restoration paths under the failure of straddle span.

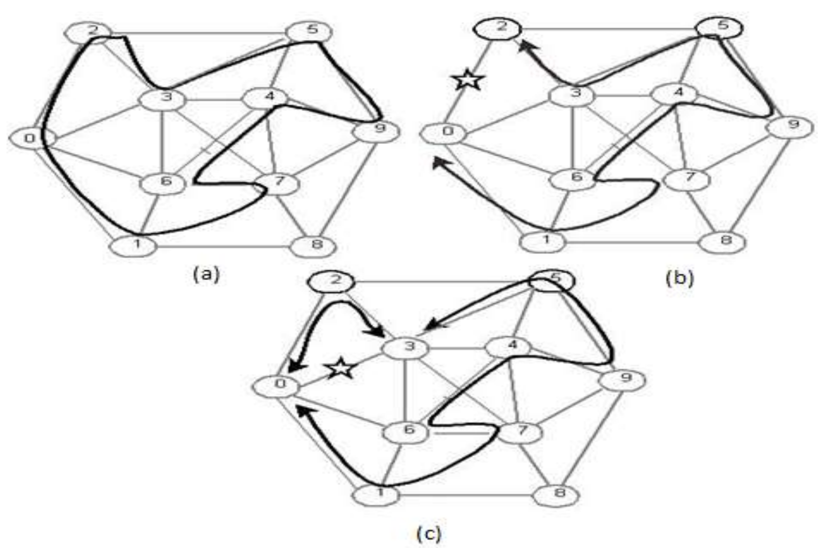


FIGURE 1: *p*-Cycle protection (a) a *p*-cycle (b) on-span protection (c) straddle-span protection.

Because of its outstanding performance on both the recovery speed and capacity efficiency, *p*-cycle has attracted extensive research interests, particularly in the field of designing of *p*-cycles. The objective of *p*-cycles design is to minimize the spare capacity requirements and covering all the demands on a network graph. In the literature, various methods are presented to design the survivable network with *p*-cycles [6-17]. Efficient *p*-cycles can be obtained either by centralized network management system or distributed self organization. The Integer Linear Programming (ILP) is the well-known centralized approach to design *p*-cycles in WDM networks. ILP works in two steps; first the set of all distinct cycles are generated from the network topology [18] and in the second step it generates an optimal *p*-cycle plan by choosing the number of copies of each elemental cycle on the network graph, to be configured as a *p*-cycle [5-6][9-10]. However, in order to generate the optimal set of *p*-cycles, all cycles in the network should be taken into the account. The elementary cycles in the network exponentially increases as the size of network increases. Such ILP formulation is however impractical in large scale or dense networks because the number of candidates are too large. More importantly, a large candidate set incurs a huge number of variables in the ILP, even when dealing with moderate size networks. This slows down the optimization process. To speed up the optimization process and to avoid dependence on centralized control for the deployment and maintenance of *p*-cycle state for a network, a distributed self-planning protocol called Distributed Cycle Pre-Configuration (DCPC) protocol was introduced by D. Stamatelakis and W.D. Grover [8]. Since the DCPC protocol is a self-organizing approximation to theoretically minimal spare capacity design, it does not always guarantee 100% restorability [7]. However, when compared to centralized optimal *p*-cycle design, its restorability levels are quite satisfactory.

As mentioned above, the common *p*-cycle designing method ILP is not efficient in case of large networks. On the other hand DCPC is unable to provide 100% survivability of working capacities if the spare capacity is deployed as per the enumeration of Spare Capacity Optimization (SCO) model. In this paper, our contribution is to enhance survivability level of DCPC protocol by incorporating straddle score during selection of *p*-cycle amongst other available *p*-cycles.

In the next section, background and related works are briefly explained. Overview and contribution of the work discussed in section-3, and section-4 presents proposed modifications in DCPC protocol: Incremental and cumulative approaches. Section-5 discusses the simulation and result and conclusion is given in section-6. Finally related future scope suggested in section-7.

2. BACKGROUND AND RELATED WORK

The idea of optimal spare capacity design for *p*-cycle was initially formulated using Integer Linear Programming. Two different ILP models have been used for optimization. In first model only spare capacity is optimized and it is referred as Spare Capacity Optimization (SCO) model [5]. In case of second model the working and spare both the capacities are optimized jointly. In SCO ILP model, first the shortest working routes are determined in advance for given traffic demand. Then optimal spare capacity is determined for 100% protection of these working capacities. Optimal ILP design required to enumerate all eligible cycles in the network to form a candidate set and then use an ILP model to find optimal set of *p*-cycles from the candidate set. However, the number of possible cycles in a network grows exponentially with the network size. This makes optimization as NP-hard problem. In paper [9], authors have given an alternative approach to just consider a limited number of promising cycles as a candidate set. Heuristics have been proposed in the literature for pre-selecting the most promising eligible cycles in the large sized network. However, limiting the size of candidate set adversely affect the quality of optimization. Pure Heuristic algorithms [11-14] were proposed to design *p*-cycles without candidate cycle enumeration and ILP. However, heuristics design methods requires 5-7% more spare capacity as compared to optimal design. In paper [16-17], authors formulated ILPs to construct *p*-cycles without candidate cycle enumeration and pre-selection. However, pure ILPs are not much effective in case of large networks.

In paper [16], authors have given another kind of approach for optimal design of *p*-cycle referred as distributed cycle pre-configuration protocol (DCPC). This protocol is a self-organizing strategy for the autonomous deployment and continual adaption of the network cycle configuration. It is an adaption of the statelet processing roles of the self-healing network (SHN) [17]. The statelets are small packets containing index number, hop count, cycler node, and number of paths that gets protection and the route of the statelet. A statelet is embedded on each spare link of the network. Each logical link has an incoming statelet and outgoing statelet. An incoming statelet arrives at a node on a link and originates from the adjacent node connected through the link.

There are only two node roles in the DCPC; a combined sender / chooser role called a “Cycler” and a Tandem node. The Cycler sources and later receives parts of the statelet broadcast pattern it initiates. When not in the cycler role, node plays a Tandem-node role which mediates the statelet broadcast competition, as in the SHN, but with a new decision criterion. The DCPC first allows each node to explore the network for *p*-cycle candidates that are discoverable by it. After completion of its exploratory role as cycler, it hands off to the next node in order by a simple flood-notification. After all nodes have assumed the role of the cycler once, each “posts” its best found cycle in a distributed network-wide comparison of results. Eventually, the globally best cycle candidate dominates everywhere. Upon thus learning of the winning candidate, the Cycler node that discovers this *p*-cycle goes on to trigger its formation as a *p*-cycle. All nodes on the *p*-cycle update their local tables of restoration switching pre-plans to exploit the new *p*-cycle. The whole process then repeats, spontaneously without any central control, adding one *p*-cycle per iteration until a complete deployment of near-optimal *p*-cycles are built.

In paper [7] author observed that the *p*-cycles generated by DCPC, many of the *p*-cycles have multiple copies. However, existing DCPC obtains only one *p*-cycle per iteration. The number of iterations required by the DCPC is equal to the number of copies of the *p*-cycle. This becomes more severe in case of heavily loaded or large networks. Therefore author proposes the modified DCPC (MDCPC) where all the copies of same *p*-cycles are identified and deployed at single iteration. In MDCPC, the main idea is to aggregate all the copies of the *p*-cycle and indicate the number of copies with capacity of the *p*-cycle. This will decrease overall running time of an algorithm as well as fabric requirement at each corresponding OXC’s.

3. OVERVIEW AND CONTRIBUTION

As mentioned in section-2, the DCPC searches for the available *p*-cycles in the current state of the network and selects the best scored *p*-cycle amongst them. Where, score is a ratio of number of protection provided by a *p*-cycle and cost of spare capacity required for constructing a *p*-cycle. However, outcome of the DCPC iteration totally depends on the state of the network. State of the network means unit of protection required and unit of spare capacity availability at each and every span of the network. In each iteration, after discovery of a global best *p*-cycle it will construct the *p*-cycle on the network. Construction of *p*-cycles means to make a cross connection at the nodes of a *p*-cycle, update the number of uncovered working links and available spare capacity of the corresponding spans.

In the beginning of *p*-cycles formulations, unprotected working capacities and spare capacities are generally available over each and every span of the network and therefore DCPC discovers large *p*-cycles with good scores. Since the state of the network changes after each iteration and therefore as the work progresses DCPC may delivers medium sized *p*-cycles. The noticeable point of the algorithm observed at the final stage is that the working capacities are unprotected over scattered spans. Therefore, DCPC finds local small sized *p*-cycles with poor score. The small sized *p*-cycles increase the spare capacity requirement and due to unavailability of spare capacity around the span where some working capacities remains unprotected. Generally DCPC terminates even if some working capacities remain unprotected and spare capacity is available on different part of the network. These limitations of a DCPC also reported in paper [7-8]. The table1 shows the same in most popular test case networks often used in analysis of *p*-cycles.

Networks	**Working capacity**	**Spare capacity provisioned**	**Working capacity unprotected**	**Spare capacity unused**
Net-1	2546	1766	280	302
Net-2	984	754	102	104
Net-3	422	300	40	50
Net-4	316	194	16	24

TABLE 1: Performance of DCPC with different test case networks

In this paper, our contribution is to enhance DCPC so that it provides 100% protection of working capacity in case of single failure scenario. Consider the test case network shown in figure 2. The spans are labeled with unprotected working capacities and two *p*-cycles, referred as *p*-cycle-1 and *p*-cycle-2, are depicted on the figure.

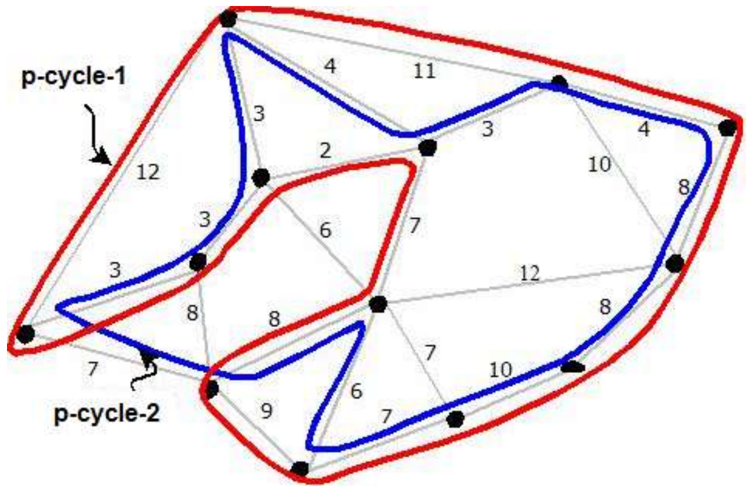


FIGURE 2: Test case network with two Hamiltonian p-cycles

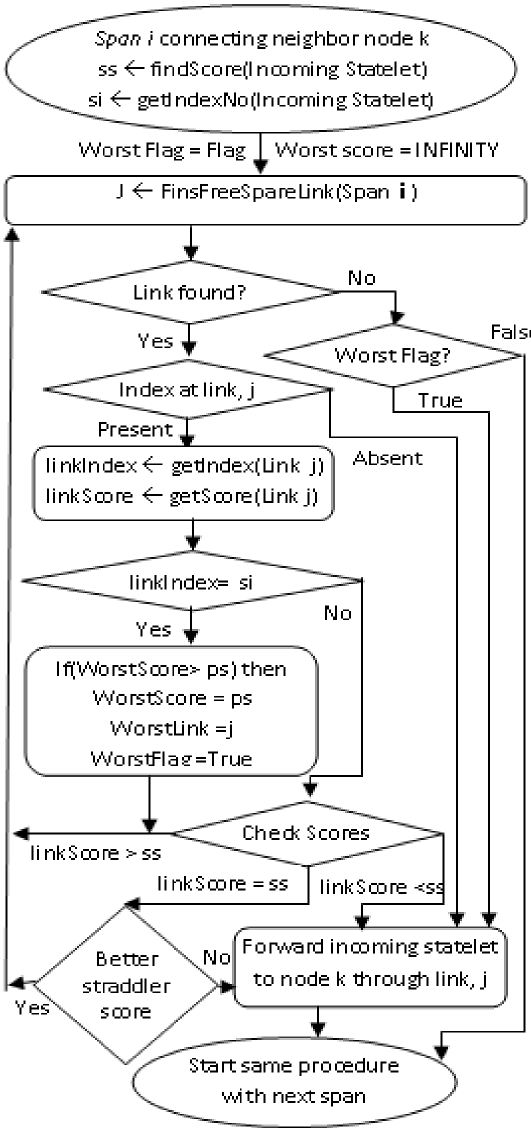
Here *p*-cycle-1 and *p*-cycle-2 are Hamiltonian and provides 33 unit of protection (13 unit on-span and 20 units on straddle spans) and consumes only 13 unit of spare capacity. The efficiency score of both *p*-cycles is 2.57 and therefore DCPC selects any one of the *p*-cycle. However, it will be better to select the *p*-cycle which has straddle spans at locations where unprotected capacities are more. This will balance the unprotected working capacities over the network and therefore more likely be able to find large sized *p*-cycles in further iterations of the DCPC. For example, sum of the unprotected working capacity of straddle span at *p*-cycle-1 is 65 (3 + 4 + 3 + 10 + 12 + 7 + 6 + 6 + 8 +7) where as 84 (11 + 10 + 12 + 7 + 9 + 7 + 6 + 2 + 8 + 12) at *p*-cycle-2. Therefore selecting *p*-cycle-2 may provide better solution at the end of the algorithm as compared to selecting the *p*-cycle-1.

4. PROBLEM FORMATION

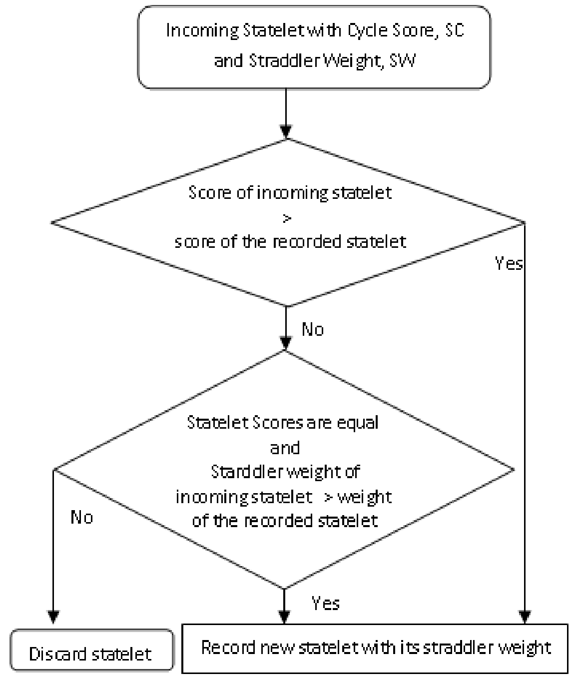
The idea behind proposed work is to incorporate contribution of unprotected working capacity during selection of efficient *p*-cycles amongst available *p*-cycles. Since a *p*-cycle provides two protection paths for failure of straddle span and only one protection path for on-cycle span failure, therefore it is better to make spans as straddle spans where unprotected working capacity left is more and also on-cycle span where unprotected working capacities are less.

The DCPC statelet has number of fields to carry the important information required for selection of good scoring *p*-cycle. The statelet fields numPaths and hopCount contains number of useful protection paths candidate *p*-cycle can provide and size of the candidate *p*-cycle. These two fields are used to calculate efficiency score of the *p*-cycle. DCPC does not relay any information regarding the number of straddle spans and on-cycle spans of a candidate *p*-cycle. For the same we have added new field with the structure of the statelet named as straddle *score* – sum of unprotected working capacity at straddle spans of the *p*-cycle. Tandem node calculates and updates the straddle score during statelet broadcast. Here, we have presented two different approaches to incorporate the significance of straddle score in *p*-cycle selection criteria: Incremental and cumulative.

A. Incremental Method: Original rules of DCPC are used during statelet forwarding at tandem node and recording at cycler node, except in case when score of incoming statelet and available score both are same. In this case, the incoming statelet will be forwarded if it is better in respect of straddle score. Figure 3(a) depicts the procedure to forward incoming statelet at tandem node in detail. Similarly, Cycler node also accepts same score incoming statelet which is better in respect of straddle score. The exact working of Cycler node presented in Figure 3(b).



(a)



(b)

FIGURE 3 : Enhanced DCPC (a) Statelet forwarding procedure of tandem node (b) Recording procedure of the cycler node.

B. Cumulative Method: The idea behind the proposed work is to make span straddle of the *p*-cycle as per amount of its unprotected working capacity. In paper [9], author talked about capacity weighted (actual) efficiency of a *p*-cycle, which is dependent not just on the number of on-cycle and straddling span, but also on the working capacities of those spans. The formula used for calculating actual efficiency of a *p*-cycle is -

$$\mathbf{E_w(p)} = \left(\sum_{\forall i \in s} \mathbf{w_i x_{p,i}}\right) / \left(\sum_{\forall i \in s \,|\, x_{p,i=1}} \mathbf{c_i}\right)$$

Where w_i is the amount of unprotected working capacity on span i at the time of calculation of actual efficiency. This new quantity gives us not only a guess of a *p*-cycle's ability to protect hypothetical working capacity, but also gives us an indication of a *p*-cycle's actual suitability in a specific working capacity state. Here our idea is to use actual efficiency of a *p*-cycle in place of A priori efficiency (AE) during statelet forwarding. The tandem node forwards the incoming statelet if its Ew(p) score is larger than the exiting statelet score. Similarly cycler node accepts the incoming statelet with its *p*-cycle score which is larger than previously received best statelet score.

5. SIMULATION AND RESULTS

The performances of proposed modifications in DCPC are evaluated with the most popular test case networks shown in figure 4. In this paper we referred these test case networks by name: Net-1(28 nodes, 45 spans, 3.21 A.N.D.), Net-2 (19 nodes, 28 spans, 2.95 A.N.D.), Net-3 (14 nodes, 21 spans, 3.0 A.N.D.) and Net-4(13 nodes, 23 spans, 3.54 A.N.D.). These networks are mostly used by researchers working in the area of *p*-cycles for evaluating the performance of their proposed work. We used most popular and efficient network simulator named as NS-2. The simulation test bed is supposed to have the following properties-

- The traffic is assumed to be one unit between each node pair, i.e. unit traffic matrix.
- The routes for working paths have been identified with shortest path Dijkstra's algorithm, using hop count as metric.
- The working capacity w_i on every span is equal to the total number of working paths passing through that span.
- A spare capacity provisioned on each span is as per the solution of spare capacity optimization model.

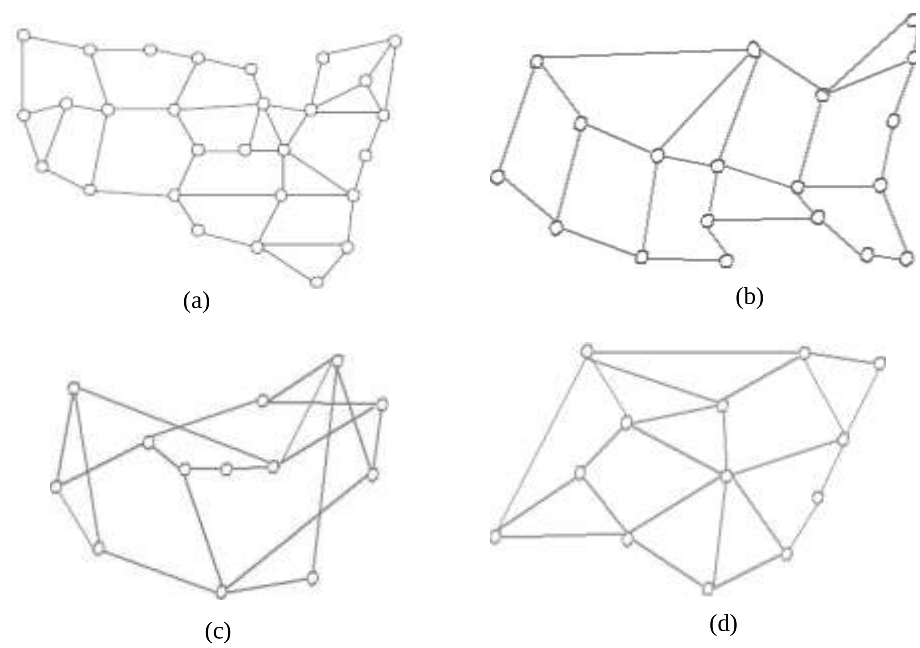


FIGURE 4: Test case networks: (a) Net-1 (b) Net-2 (c) Net-3 (d) Net-4

We have simulated the traditional DCPC which is based on A-priori Efficiency(AE) of a *p*-cycle, proposed cumulative approach based on Actual Efficiency(Ew) and Incremental approach which is based on AE with straddle score (AE with std_score). The simulation results of all three different versions of DCPC (AE, Ew and AE with std_score) are tabulated on table 2. The results of proposed algorithms are compared with the result of conventional DCPC in terms of number of *p*-cycles constructed, total protection available and utilization of spare capacity. The results clearly shows that conventional DCPC-AE) is unable to provide 100 protections in all the test case networks. In Net-1, 11% working capacity remains unprotected even through lot of spare capacity remains available in the network. Similar results are observed with Net-2, Net-3 and Net-4. This happened due to unavailability of spare capacity around the span where working capacity remains unprotected. We have suggested modifications in the statelet forwarding rules to manage the spare capacity around the unprotected working capacity.

Net-works	Total working capacity (WC)	Provisioned spare capacity (SC)	Total no. of *p*-cycles constructed			Total unprotected working capacity (%)			Total unused spare capacity (%)		
			AE	Ew	AE with std_score	AE	Ew	AE with std_score	AE	Ew	AE with std_score
Net-1	2546	1766	92	99	102	280 (11.0%)	154 (6.0%)	83 (3.3%)	302	241	272
Net-2	984	754	44	47	49	102 (10.4%)	52 (5.3%)	25 (2.5%)	104	72	99
Net-3	422	399	20	23	24	40 (9.5%)	21 (4.7%)	07 (1.6%)	149	127	139
Net-4	316	194	16	17	18	16 (5.0%)	8 (2.5%)	01 (0%)	24	20	20

TABLE 2: Simulation results of Incremental and Cumulative approach.

The results of cumulative approach (Ew) clearly shows the effectiveness of new metric straddle score which has been considered during forwarding of the statelet. It minimizes the unprotected working capacity approximately half from the original DCPC, in all the test case networks. However, it increases the used spare capacity. Further, the results of incremental approach (AE with straddle score) shows noticeable enhancement in survivability level on all the test case networks. In Net-1, approximately 100% protection available, whereas 96%, 97% and 98% protection are available in Net-1, Net-2 and Net-3 respectively. The main contribution of the proposed work is that it improves the performance of DCPC protocol without much incrementing used spare capacity.

6. CONCLUSION

We have explored complexity involved in the optimal design of *p*-cycle for the survivability of WDM networks. Numbers of approaches are reported in the literature where ILP is most efficient approach for the same. The DCPC is a distributed protocol to design optimal *p*-cycles in WDM networks. However, the DCPC enable to provide 100% protection under SCO model. The proposed modifications in DCPC, by force tried to form span as straddle of the *p*-cycle where more protection is required. The cumulative approach forwards the incoming statelet based on actual efficiency of the corresponding *p*-cycle. However, Incremental approach works exactly as DCPC except when score of incoming statelet and existing statelet both are same. In this case, it forwards the incoming statelet if their straddle score is larger than straddle score of existing statelet. Results clearly shows that proposed approaches provides more protection as compared to conventional DCPC under spare capacity optimization model only by using few more spare capacity. The simulation results clearly shows that incremental approach provides approximately 100% protection over Net-4 and Net-3 whereas 97-98 % over Net-2 and Net-1. The noticeable point is that proposed modifications increased the protection by using just more spare capacity.

7. FUTURE SCOPE

In this paper we have given an idea for selection of *p*-cycles to balance the unprotected working capacity over spans of the network. The *p*-cycle selection is based on their cumulative/global straddle score and therefore a span where more unprotected working capacity may not be guaranteed to become straddle of the *p*-cycle. So the performance of proposed algorithm may be enhanced by consideration of local straddle score.

8. REFERENCES

[1] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, “Survivable WDM mesh network,” *Journal of Lightwave Technology,* vol. 21, no. 4, pp. 870–883, 2003.

[2] Grover W.D. , “Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking,” *Prentice Hall (2004)*

[3] W.D. Grover and D. Stamatelakis, “Bridging the Ring-Mesh Dichotomy with *p*-cycles,” *Proc. Of design of Reliable Communication Network(DRCN 2000), technical University Minich, germany,* April 9-12, 2000

[4] W., Somani, A., “Comparison of protection mechanisms: capacity efficiency and recovery time,” *In IEEE international Conference on Communications—ICC,* pp. 2218–2223 (2007)

[5] W. D. Grover and D. Stamatelakis, “Cycle-oriented distributed pre-configuration: Ring-like speed with mesh-like capacity for self-planning network restoration,” *Proc. IEEE International Conference on Communications (ICC)*. *Atlanta, Georgia, USA,* (Jun, 1998), pp. 537-543

[6] M. S. Kiaei, C. Assi, and B. Jaumard, “A Survey on the *p*-Cycle Protection Method,” *IEEE Commun. Surveys Tutorials, vol. 11, no. 3, pp. 53-70,* July 2009.

[7] Asthana, R., Singh, Y.N., Grover, W.D, "*p*-Cycles: An overview," *Communications Surveys & Tutorials, IEEE*,” vol.12, no.1, pp.97-111, First Quarter 2010

[8] D. Stamatelakis, W.D. Grover, "Distributed Preconfiguration of Spare Capacity in Closed Paths for Network Restoration," *U.S. Patent Pending*, July 11, 1997.

[9] C. Liu and L. Ruan, “Finding good candidate cycles for efficient *p*-cycle network design,” *Proc. 13th International Conference on Computer Communications and Networks (ICCCN 2004),* pp. 321–326, 2004.

[10] D. Schupke, C. Gruber, and A. Autenrieth, “Optimal configuration of *p*-cycles in WDM networks,” *IEEE*, pp. 2761–2765, 2002.

[11] H. Zhang, O. Yang, “Finding protection cycles in DWDM networks,” *Proc of IEEE ICC,* Pages 2756-2760, April/May 2002.

[12] Kungmang Lo, Daryoush Habibi ant atl., “Efficient *p*-Cycle Design by Heuristic *p*-Cycle Selection and Refinement for Survivable WDM Mesh Networks,” *Proc. of IEEE Global Telecommunications Conference(GLOBECOM), San Francisco*, USA, *IEEE Communication Society.*

[13] Zhenrong Zhang and et al., "A heuristic method for design of survivable WDM networks with *p*-cycles," *Communications Letters, IEEE ,* vol.8, no.7, pp. 467- 469, July 2004

[14] Doucette, J.; He, D.; Grover, W.D.; Yang, O.,"Algorithmic approaches for efficient enumeration of candidate *p*-cycles and capacitated *p*-cycle network design," *Design of Reliable Communication Networks, 2003. (DRCN 2003), Fourth International Workshop*, vol., no., pp. 212- 220, 19-22 Oct. 2003

[15] R.Yadav, R. S. Yadav and H. M. Singh, “Enhanced Intercycle Switching in *p*-cycle Survivability for WDM Networks” *Journal of Optical Communications and Networking*, vol. 2, issue 11, November 2010, pp. 961-966

[16] B. Wu, K. L. Yeung and S. Z. Xu, “ILP formulation for *p*-cycle construction based on flow conservation,” *IEEE GLOBECOM '07*, Nov.2007.

[17] B. Wu and P.-H. Ho, "ILP formulations for p-cycle design without candidate cycle enumeration," *IEEE/ACM Trans. Netw*., vol. 18, no. 1, pp. 284-295, Feb. 2010

[18] B. J. Donald, "Finding all the elementary circuits of a directed graph," *SIAM J. Comput*ation., vol. 4, no. 1, pp. 77-84, Mar. 1975.

[19] W. D. Grover, "Method and apparatus for self-healing and self provisioning networks," *U.S. Patent* No. 4,956,835, 1990.

[20] S. Ramamurthy and B. Mukherjee, "Survivable WDM mesh networks, part 1- protection*," in Proceedings of Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. INFOCOM'99, vol. 2, (New York, NY, USA), pp. 744–751, 1999.*