Unrestored Flow Optimization in Survivable Networks Based on *p*-Cycles

Adam Smutnicki

Chair of Systems and Computer Networks,
Wrocław University of Technology,
Wybrzeze Wyspianskiego 27, 50-370 Wrocław, Poland
adam.smutnicki@pwr.wroc.pl

Abstract. This paper deals with Unrestorable Flow Optimisation (UFO) problem in networks protected by p-cycles. This novel protection technique is used as the efficient tool for ensuring survivability of computer networks. In this paper there have been formulated mathematical model of UFO problem, discussed its theoretical properties, and proposed the original solution algorithm based chiefly on metaheuristics. The algorithm combines k-shortest paths method, multi knapsack problem, p-cycles generator, linear programming and some local search procedures.

Keywords: computer network, survivability, optimisation, p-cycles, UFO problem.

1 Introduction

Survivability of computer networks and systems is located among the most important subjects in modern computer engineering and science. This research topic embraces the wide spectrum of particular technological and theoretical problems derived from computer architecture area, network topology, communication protocols, transmission, coding, cryptography, etc. The topology of the computer network has crucial meaning for its survivability, since physical creation of the net links is much more time-consuming and troublesome than producing a new (or spare) device, furthermore faults of network links and nodes are still the common problem. The idea of usage *p*-cycles is quite new, but have been widely developed among recent years. *p*-Cycles are very favorable in comparison with traditional ring or mesh topologies. In this paper, we present and discussed the new optimization problem, generated by the concept of network protection by using *p*-cycles, where the criteria is not a cost, but the level of restorability ensured by using currently available net resources.

This paper is organized as follows. Section 2 provides brief introduction into *p*-cycles idea. Section 3 describes, in detail, new problem of unrestorable flow optimization, called hereinafter the UFO problem. Section 4 discusses some its properties and solution methods. Conclusions one can find in Section 5. Because of the strong NP-hardness of UFO problem, this paper introduces basically the

G. Allen et al. (Eds.): ICCS 2009, Part I, LNCS 5544, pp. 325–334, 2009.

mathematical model, then discusses its essential properties including layer decomposition and computational complexity and, at the end, provides a wide spectrum of solution methods. Particular algorithms, as well as experimental results based on standard benchmarks are shown in separate our paper [1].

2 Background of *p*-Cycles

Traditionally, the most popular solution for providing restorability in computer networks was to use either *ring* or *mesh* topology. Ring offers short restoration time as well as simple restoration scheme, however its design and operation is rather complex and the usage of total transport bandwidth is inefficient. Mesh is easy to design, optimize and operate, but have greater than ring restoration time. Mesh networks don't require as much spare capacity as rings, because in the restoration process capacity demand can be split between different links. On the other hand, rings are so efficient in the restoration process because there are no need to search for restoration path. Obviously, there is a great need to find topology, which aggregates all advantageous properties of mesh and ring networks. This idea was fully realized in the concept of p-cycles, that means "fast as ring", "efficient as mesh", preconfigurable and protected.

In normal non-failure state, routing for flow demands between pairs of nodes, is done using one of many routing techniques. Set of p-cycles is formed in advance while configuring network, to be ready to use in case of any failure and perform real-time recovery. p-Cycles are not an ordinary cycles. Let us consider a mesh network and choose some cycle (Fig. 1 a). In classical cycle protection approach, this cycle protects all spans being "on-cycle". In the paper [2] it is shown that cycle established on mesh network protects also "straddling spans", i.e. spans between cycle nodes, but not belonging to the cycle (Fig. 1 b). Observe, that in case of failure of "straddling span" the arc of cycle can be used to transfer whole flow from this failed span. This property allows one to extend protection provided by p-cycles on straddling spans as well. Fig. 1 c shows which spans cannot be protected using this properties.

In case of failure of "on cycle" span, there is one path which can be used to transfer flow (Fig. 2 a). But for failure of "straddling span" there are two different paths, which can be used for recovery process. Arc of the cycle can be used as a path, or both arcs to achieve lower load on links (Fig. 2 b and c). Because without using any additional links and spare capacity we achieve much higher level of protection, protected are not only cycle spans but also "straddling spans". "Straddling spans" have twice the leverage of an on-cycle span in terms of efficiency because when they fail, the cycle itself remains intact and can thereby offer two protection paths for each of unit of protection capacity. This spans are not limited to be inside a cycle, each span between two nodes of the cycle is protected by this idea.

Notice, we do not need any additional spare capacity to protect "straddling spans", because the spare capacity from ring spans is used to protect those spans. This means that we can protect much more spans and link capacity using the same amount of spare capacity as in the ring model. Thus, under the some costs, we can achieve higher level of network survivability.

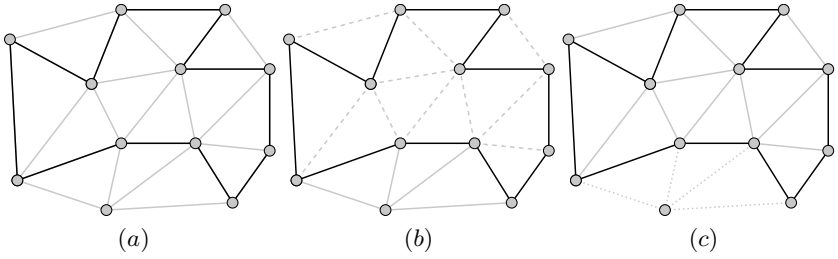

Fig. 1. p-Cycles in the mesh network

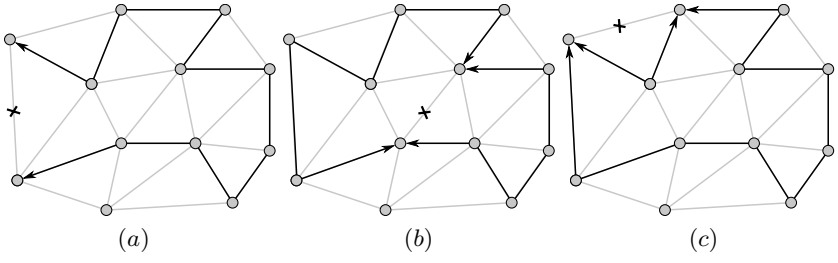

Fig. 2. p-Cycles protection schemes for various types of failure

3 The UFO Problem

In the literature, most papers dealing with survivable networks concentrate on ensuring 100% survivability. But only few authors have considered (not only mentioned) problem, where 100% of restorability may not be achievable. For networks protected by p-cycles, in [3] there has been proposed problem in which *the level of restorability* is maximized, assuming fixed amount of spare capacity.

Another paper [4] mentions about different idea — minimization of unrestorable flow in the network with fixed capacities, where no additional capacity is necessary, since restoration is done within available spare capacity, left after optimization of all flow demands. This problem, called by us Unrestorable Flow Optimisation (UFO), will be studied extensively in our paper. Below, we define it formally as follows. (Notation based on [5] and [6] will be used.)

We have given: network topology, link capacities, traffic demand matrix, candidate paths for demands, p-cycles configuration. Optimization over working flows in normal non-failure state of the network is done, for protection in the case of single link failure. The objective is to minimize the unrestored flow, i.e. flow that due to limited link capacity cannot be restored using p-cycles.

Indices

$e, l = 1, \ldots, E$ network links (spans)
$d = 1, \ldots, D$ demands

$p = 1, \ldots, P_d$ candidate paths for flows realizing demand d
$q = 1, \ldots, Q$ p-cycles
$s = 1, \ldots, S$ failure states

Constants

$\mathcal{D}$ set of all demands
$\mathcal{E}$ set of all spans
$\mathcal{Q}$ set of all p-cycles
$\mathcal{Q}_e \in \mathcal{Q}$ set of p-cycles which can be used for restoration span e
$\mathcal{D}_e \in \mathcal{D}$ set of demands using span e
δ_{edp} $= 1$, if link e belongs to path p realizing demand d; 0 otherwise
h_d — volume of demand d
c_e — capacity of link e
β_{eq} $= 1$, if link e belongs to p-cycle q; 0 otherwise
ϵ_{eq} $= 1$, if p-cycle q can be used for restoration of link e; 0 otherwise i.e. link e either belongs to p-cycle q or is a straddling span of q
γ_{eq} — coefficient of restoration paths provided for failed link e by an instance of p-cycle q ($= 1$ for an on-cycle link; $= 0.5$ for a straddling span; $= 0$ otherwise)

Variables

x_{dp} $= 1$ if demand d uses path p; 0 otherwise (binary)
f_e — load of link e associated with working demands
y_{deq} $= 1$ if demand d uses path p-cycle q for restoration in the case of failure of link e; 0 otherwise
z_{de} $= 1$ if demand d is not restored in the case of failure of link e; 0 otherwise
g_{el} — load of link e associated with p-cycle in the case of failure of link l

Overall optimization criteria function is:

$$\min_{y,z} U(y, z; x, \mathcal{Q}) = \min_{y,z} \sum_{e} \sum_{d} z_{de} h_d, \tag{1}$$

with constraints:

$$z_{de} + \sum_{q} \epsilon_{eq} y_{deq} = A_{de},\ e = 1, \ldots, E,\ d = 1, \ldots, D \tag{2}$$

$$g_{el} = \sum_{d} \sum_{q} B_{eldq} y_{dlq},\ e = 1, \ldots, E,\ l = 1, \ldots, E \tag{3}$$

$$g_{el} \le s_e, \quad e = 1, \ldots, E, \quad l = 1, \ldots, E, \tag{4}$$

where A_{de}, B_{eldq}, are constants used to make equations more clear for fixed x:

$$f_e = \sum_{d} \sum_{p} \delta_{edp} x_{dp} h_d, \quad e = 1, \ldots, E, \tag{5}$$

$$A_{de} = \sum_{p} \delta_{edp} x_{dp}, \quad e = 1, \ldots, E, \quad d = 1, \ldots, D, \tag{6}$$

$$B_{eldq} = \sum_{p} \delta_{ldp} x_{dp} \beta_{eq} \gamma_{lq} h_d, \quad e = 1, \ldots, E,$$
$$l = 1, \ldots, E, \quad d = 1, \ldots, D, \quad q = 1, \ldots, Q, \tag{7}$$

and:

$$s_e = c_e - f_e, \quad e = 1, \ldots, E. \tag{8}$$

Auxiliary variable s_e, defined in (8) and used in constraint (4), will be called *residual spare capacity*, left after fulfilling all traffic demands from set $\mathcal{D}$ transfered over paths determined by x. Auxiliary constant A_{de} takes only binary values, $A_{de} \in \{0, 1\}$, which means that either there is no possibility to restore demand d in case of failure of span e (then $z_{de} = 1$ and all $y_{deq} = 0$, $q = 1, \ldots, Q$) or there exists possibility of restoring d in case of failure of span e using one chosen p-cycle q (then $y_{deq} = 1$ and $z_{de} = 0$). The formulated optimization problem is a Mixed Integer Linear Programming (MILP) task. However, because of the problem size, and well-known weakness of the general MILP solution methods, we give up MILP techniques and transform the problem into new one.

Using equation (2) in (1) and substituting equation (3) into inequality (4), the problem receives the following form:

$$\min_{y,z} U(y, z; x, \mathcal{Q}) = \min_{y} \sum_{e} \sum_{d} (A_{de} - \sum_{q} \epsilon_{eq} y_{deq}) h_d$$
$$= \sum_{e} \sum_{d} A_{de} h_d - \max_{y} \sum_{e} \sum_{d} \sum_{q} \epsilon_{eq} h_d y_{deq}, \tag{9}$$
$$\sum_{d} \sum_{q} B_{ledq} y_{deq} \leq s_l, \; e = 1, \ldots, E, \; l = 1, \ldots, E. \tag{10}$$

In these equations index e stands for span being damaged; when another index l is used also for spans, it refers to other span which has been influenced by damaged span e.

First element in equation (9) is constant and does not have any influence on the form of optimal solution, but only on its value. Problem (9) described by second element with constraint (10), can be solved by a sequence of problems known in literature as Knapsack Problem (KP) or Multiple Knapsack Problem (MKP). Constraint (10) determines whether there is a need to consider KP either or MKP case. Transformation of (9) – (10) into MKP is shown below.

Consider right side of equation (9) for fixed span e in a form:

$$\max_{y} \sum_{d} \sum_{q} C_{edq} y_{deq}, \tag{11}$$

where:

$$C_{edq} = \epsilon_{eq} h_d, \tag{12}$$

with constraints:

$$\sum_{d} \sum_{q} B_{ledq} y_{deq} \leq s_l, \quad l = 1, \ldots, E. \tag{13}$$

For fixed e, exists $d \times q$ binary decision variables y_{deq}. So the criteria function (11) corresponds to optimal packing Q knapsacks with elements $1, 2, \ldots, D$. In order to match constraint (13) with constraints from MKP [7], the new notion r_q will be used, called hereafter *residual capacity of p-cycle* q. Value r_q stands for maximum flow which can be added to all spans in p-cycle q, without exceeding residual capacity of the spans. Calculation of p-cycle residual capacities is a complex problem – we will discussed it, in detail, in Section 4.3. For further considerations, we assume that capacity of p-cycle meet constraint $r_q > 0$ and:

$$s_l = \sum_q \beta_{lq} r_q. \tag{14}$$

Substituting (14) in constraint (13) one can obtain:

$$\sum_d \sum_q B_{ledq} y_{deq} \le \sum_q \beta_{lq} r_q, \quad l = 1, \ldots, E. \tag{15}$$

Condition (15) is then transformed into sequence of $Q \times E$ conditions in the following form:

$$\sum_d B_{ledq} y_{deq} \le \beta_{lq} r_q,\ q = 1, \ldots, Q,\ l = 1, \ldots, E, \tag{16}$$

what makes stronger constraint that original (15). Summing inequalities (16) by sides for each l (this is relaxation) inequality (13) will be received. Observe in definition (7) that value B_{ledq} does not depend on l, so:

$$B_{ledq} = B^*_{edq} = \sum_p \delta_{edp} x_{dp} \gamma_{eq} h_d, \tag{17}$$

because (16) is fulfilled obviously for $\beta_{lq} = 0$, while this condition have to be fulfilled additionally for $\beta_{lq} = 1$. Using (17), constraint (15) can be transformed into:

$$\sum_d B^*_{edq} y_{deq} \le r_q, \quad q = 1, \ldots, Q. \tag{18}$$

for those l for which $\beta_{lq} = 1$. It corresponds to missing constraint for knapsacks capacities $1, \ldots, Q$ in MKP (constraints (18) are identical for all l). If $Q = 1$ (special case) MKP is simplified and takes form of KP.

Finally, value:

$$\max_y \sum_e \sum_d \sum_q C_{edq} y_{deq}, \tag{19}$$

can be calculated step by step, analyzing possibility of restoring flow f_e using p-cycles after the failure of span e, for all $e \in \mathcal{E}$ such $f_e > 0$. If only $f_e = 0$ nothing have to be restored — $y_{deq} = 1$, $z_{de} = 0$, $e \in \mathcal{E}$ because $h_d = 0$ for $d \in \mathcal{D}$. So (19) can be finally written in a form:

$$\max_y \sum_{e \in \mathcal{E}_+} \sum_{d \in \mathcal{D}_e} \sum_{q \in \mathcal{Q}_e} C_{edq} y_{deq}, \tag{20}$$

where: $\mathcal{E}_+ = \{e \in \mathcal{E} :\ f_e > 0\}$.

4 Solution Proposal

Solution proposal tends to decomposition of the problem (1) – (8) leading to the series of known, simpler cases.

4.1 Decomposition

In the context of Section 3 one can propose quite natural layer decomposition of optimization problem into several sub-problems. At the begin we have the network with known topology, span capacities and costs and a set of flow demands. Each demand determines flow transfer between pair of nodes (from source to destination). In order to satisfy these demands routing paths have to be found. This problem is known in the literature as multicomodity flow (MCF). MCF problem requires the set of paths (for example k shortest paths k-SP) between specified pair of nodes, among which MCF selects the set of the best ones ([5]). Observe, the configuration of paths satisfying demands need be advantageous for p-cycles configuration. Our overall aim is to find the optimal set of p-cycles; this can be done by metaheuristic algorithm, which goes through the solution space by certain search trajectory verifying candidates on p-cycles. Successive p-cycles in this space are pre-generated by using a reasonable generator. One among many described in literature can be used.

For fixed set of p-cycles and fixed routing paths realizing demands we have to solve the UFO problem. Its optimal solution is created by independent checking of restorability for each span with nonzero flow. If this span transfers single commodity flow, its restorability case can be simply evaluated. Otherwise, if the span transfers multicommodity flow, the Multiple Knapsack Problem (MKP) is used to find optimal restoration scheme (described briefly in Section 4.2). Both cases require the evaluation of so called *spare capacity of the cycle*. For disjoin cycles, inside current p-cycles configuration, such evaluation can be made independently and the problem is not troublesome. If cycles inside current p-cycles configuration are not disjoint, the problem of finding real *spare capacity of cycles* can be written as special linear programming (LP) problem — described in Section 4.3.

4.2 Minimisation of Unrestorable Flow

Let us consider criteria function (1) in the mathematical model from Section 3. In order to calculate its value we need to known matrix z_{de}, where h_d is known in advance as the input data for this problem. According to the description in Section 3, all routing paths have to be chosen for this calculations (realizing all flow demands) and dedicated configuration of p-cycles also have to be known, with its residual capacities calculated. Algorithm for calculation of z_{de} values is presented in Lis. 1.1. All symbols are consistent with those from Section 3. This algorithm refers to equations (9) – (10). It examines all spans $e \in \mathcal{E}_+$ in the network, whether whole flow from D_e can be restored using p-cycles $\mathcal{Q}_e$ in case of failure of each particular span.

Listing 1.1. Pseudocode of algorithm for calculation z_{de} values.

```
for e ∈ E do
  begin
   (R_e, U_e) = checkRestoriationPosibility(e, D_e, Q_e);
   for d ∈ R_e do z_de = 0;
   for d ∈ U_e do z_de = 1;
  end;
```

The key role plays function *checkRestoriationPosibility*(); it determines whether all flow demands can be restored or not in case of failure of span e. If all flow demands cannot be restored, the function returns two sets of: R_e for restorable, and U_e for unrestorable demands, minimising the total amount of unrestored flow.

One can distinguish four following scenarios in case of failure of span e (We say that particular span belongs to p-cycle, if it is either on-cycle span or a straddling-span on this p-cycle.):

1. span e does not belong to any p-cycle, so $\mathcal{Q}_e = \emptyset$; thus definitely flow from span e cannot be restored, so $y_{deq} = 0$, $q = 1, \ldots, Q$ and in consequence $z_{de} = 1$, $d \in D_e$;
2. flow on span e consists of only one commodity $|\mathcal{D}_e| = 1$ and span e belongs to one p-cycle, so $|\mathcal{Q}_e| = 1$; flow can be either restored or not, depending on residual capacity of this p-cycle;
3. flow on span e is a multicommodity flow, $|\mathcal{D}_e| > 1$ and span e belongs to one p-cycle, so $|\mathcal{Q}_e| = 1$ — flow can be restored or not depending on residual capacity of this p-cycle;
4. flow on span e is a multicommodity flow and span e belongs to more than one p-cycle, so flow can be restored or not depending on residual capacity of those p-cycles.

Case (1) is obvious. Case (2) is simple – flow can be restored if $h_d \leq r_q$ (h_d is a value of flow for demand d, r_q is a residual capacity of p-cycle q). In case (3), the total flow for span e can be restored if $\sum_{d \in D_e} h_d > r_q$. In this case, the selection of demands to be restored is significant — it is the optimization problem, modeled by using Knapsack Problem. The most complex case (4) is modeled by MKP, as already mentioned.

4.3 *p*-Cycles Residual Capacities

According to description in Section 3 there is no possibility to add any spare capacity, only existing spare capacity in working spans can be used. This assumptions generates several additional constraints. The most important is a decision problem how much of spare capacity have to be assigned to each of used p-cycles, in situation when at least two cycles have common span — Fig. 3. In this situation sum of p-cycles capacities cannot exceed span spare capacity. Additionally

maximization of sum of whole p-cycles spare capacity is desired. Mathematical formulation of this problem is described below. Let us define:

r_q — capacity of p-cycle q;
s_e — available spare capacity on span e;
r_q^{max} — maximum potential capacity of p-cycle q (defined in equation (21));
$\beta_{eq} = 1$ if span e belongs to p-cycle q;

Maximum potential capacity of each p-cycle is bounded by value:

$$r_q^{max} = \min\{s_e : e = 1, \ldots, E, \quad e \in q\}. \tag{21}$$

For each span we have constraint:

$$\sum_q r_q \beta_{eq} \leq s_e, \quad e = 1, \ldots, E \tag{22}$$

Total amount of p-cycles capacity should be maximised:

$$\max \sum_q r_q, \quad q = 1, \ldots, Q \tag{23}$$

taking into account constraints:

$$0 \leq r_q \leq r_q^{max}, \quad q = 1, \ldots, Q \tag{24}$$

The problem (23) – (24) is a typical linear programming task, so simplex method can be recommended here to solve it.

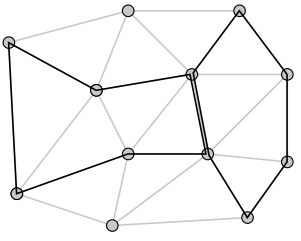

Fig. 3. Example of configuration where two p-cycles have common span

4.4 Problem Complexity

Two described previously problems, namely the finding of z_{de} values and calculation of residual capacities, are only activities performed during the evaluation of the current (fixed) p-cycles and the current routing paths configuration. In the pessimistic case, we have to solve certain MKP problem(s), which is, according to [7], strongly $\mathcal{NP}$-hard. So the process of evaluation single solution is strongly $\mathcal{NP}$-hard, too. Note, that the generation of the set of p-cycles candidates (in the local search algorithm) appears to be also quite complicated problem, see [1] for detail, because the number of possible cycles in network has non-polynomial character; the similar remarks refers also to generation of the set of paths – candidates for realizing routes for each demand.

5 Conclusions

We have formulated problem of increasing computer network survivability by using p-cycles as an combinatorial optimization problem. The problem alone has been neither discussed nor solved in the literature, yet. Although the problem can be modeled as a general MILP task, presented by us transformations show that it is quite complicated case. We have also shown that problem is strongly $\mathcal{NP}$-hard, which suggests that the most suitable solution methods should rely on metaheuristic approaches. We have also shown original decomposition method and then proposed highly dedicated algorithm for each particular subproblem. Due to limited size of the paper, we have presented here the primal part of extensive studies made for the problem stated – detailed analyzes of the algorithm as well as wide experimental research on common benchmark test are presented in the complementary paper [1], also submitted for this conference.

References

1. Smutnicki, A., Smutnicki, C.: An Algorithm for Flow Optimization in Survivable Networks Based on p-Cycles. Submitted for International Conference on Computational Science, ICCS 2009 (2009)
2. Grover, W.D., Stamatelakis, D.: Cycle-oriented Distributed Preconfiguration: Ring-like Speed with Mesh-like Capacity for Self-planning Network Restoration. In: Proceedings of ICC 1998 IEEE International Conference on Communications, Atlanta, Georgia, USA, June 1998, pp. 537–543 (1998)
3. Grover, W.D.: Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking. Prentice Hall PTR, New Jersey (2003)
4. Smutnicki, A., Walkowiak, K.: Modeling Flow Allocation Problem in MPLS Network Protected by p-Cycles. In: Peringer, P., S., J. (eds.) Proceedings of 42nd Spring International Conference on Modelling and Simulation of Systems, Hradec nad Moravici, Czech Republic, MARQ, April 2008, pp. 35–42 (April 2008)
5. Pióro, M., Medhi, D.: Routing, Flow, and Capacity Design in Communication and Computer Networks. Elsevier Inc., San Francisco (2004)
6. Chołda, P., Jajszczyk, A., Wajda, K.: The Evolution of the Recovery Based on p-Cycles. In: Proceedings of the 12th Polish Teletraffic Symposium PSRT 2005, Poznań, Poland, pp. 49–58 (September 2005)
7. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer, Heidelberg (2004)