Efficient and Scalable Design of Protected Working Capacity Envelope

S. Sebbah(1) B. Jaumard(2)

(1) ECE, Concordia University, Canada
(2) CIISE, Concordia University, Canada

Networks Conference - 2008
Budapest, Hungary

Outline

1. Problem definition and motivation
    - Protected Working Capacity Envelope
    - Problem statement
    - Motivation
2. Design of PWCE using p-cycles
    - Literature review
    - Proposed solution method
3. Performance results
    - Evaluation metrics
    - Performance evaluation

Outline

1. Problem definition and motivation
    - Protected Working Capacity Envelope
    - Problem statement
    - Motivation
2. Design of PWCE using p-cycles
    - Literature review
    - Proposed solution method
3. Performance results
    - Evaluation metrics
    - Performance evaluation

Outline

1. Problem definition and motivation
   - Protected Working Capacity Envelope
   - Problem statement
   - Motivation
2. Design of PWCE using p-cycles
   - Literature review
   - Proposed solution method
3. Performance results
   - Evaluation metrics
   - Performance evaluation

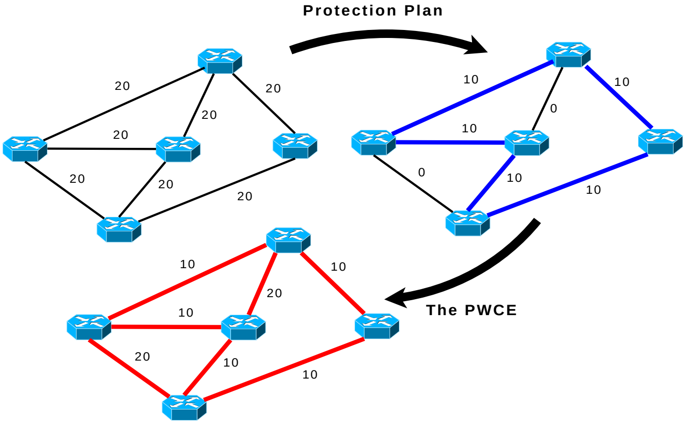
Protected Working Capacity Envelope(PWCE)

The PWCE is a new design approach of survivable WDM networks (Grover 2004), it is characterized by:

- The protection plan is designed ahead of the working plan, i.e., the protection capacity is deployed before any working capacity or failure in the network.
- The deployed protection capacity creates an implicitly protected capacity called the The Protected Working Capacity Envelope.

p-cycle-based PWCE

- An example of a design of resilient WDM network based on p-cycle-PWCE



Advantages

- Network operation is largely simplified: Provisioned working capacity within the PWCE is implicitly protected.
- Provisioning is less depend on network state information (routing databases ...), thus results in a more consistent and coherent scheme.
- The integrity of backup paths is guaranteed as an active monitoring and tuning of signal parameters can be actively performed.

Advantages

- Network operation is largely simplified: Provisioned working capacity within the PWCE is implicitly protected.
- Provisioning is less depend on network state information (routing databases ...), thus results in a more consistent and coherent scheme.
- The integrity of backup paths is guaranteed as an active monitoring and tuning of signal parameters can be actively performed.

Advantages

- Network operation is largely simplified: Provisioned working capacity within the PWCE is implicitly protected.
- Provisioning is less depend on network state information (routing databases ...), thus results in a more consistent and coherent scheme.
- The integrity of backup paths is guaranteed as an active monitoring and tuning of signal parameters can be actively performed.

The design problem

The associated PWCE design problem can be defined as follows:

- **Input**
  - A network/link spare capacity budget.
  - A set of link protection *p*-cycles.

The design problem

The associated PWCE design problem can be defined as follows:

- **Input**
  - A network/link spare capacity budget.
  - A set of link protection *p*-cycles.
- **Objective**
  - PWCE of maximum size.
  - Shaped PWCE to suit a certain potential traffic forecast.

Motivation

- The design of a PWCE is a hard combinatorial problem (equivalent to the Travelling Salesman problem TSP).
- Integer Linear Programming based solutions have been proposed for it, but unfortunately they are not scalable and/or not efficient.

Motivation

- The design of a PWCE is a hard combinatorial problem (equivalent to the Travelling Salesman problem TSP).
- Integer Linear Programming based solutions have been proposed for it, but unfortunately they are not scalable and/or not efficient.

Motivation

- The design of a PWCE is a hard combinatorial problem (equivalent to the Travelling Salesman problem TSP).
- Integer Linear Programming based solutions have been proposed for it, but unfortunately they are not scalable and/or not efficient.

However,

- Mathematical modeling is not unique for a given problem
- Optimization techniques for large scale mathematical models have not been used yet
- Column generation techniques, in which optimization is performed without having to enumerate all the variables a priori are very efficient optimization tools in large scale systems

Literature review

- The classical design methods are based on a two-step approach: an explicit enumeration of all/most promising *p*-cycle followed by an ILP optimization method.
  - A method that explicitly enumerates -(brute strength)- all the *p*-cycles is time and effort consuming, and results in a huge optimization model thus, not scalable.
  - A method that performs a selective enumeration (e.g., SLA, Expand, Grow) [Grover et all 2004] may have the advantage of scalability, but does not guarantee an optimal solution.

Literature review

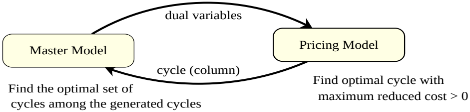
- The classical design methods are based on a two-step approach: an explicit enumeration of all/most promising *p*-cycle followed by an ILP optimization method.
  - A method that explicitly enumerates -(brute strength)- all the *p*-cycles is time and effort consuming, and results in a huge optimization model thus, not scalable.
  - A method that performs a selective enumeration (e.g., SLA, Expand, Grow) [Grover et all 2004] may have the advantage of scalability, but does not guarantee an optimal solution.

Literature review

- The classical design methods are based on a two-step approach: an explicit enumeration of all/most promising *p*-cycle followed by an ILP optimization method.
  - A method that explicitly enumerates -(brute strength)- all the *p*-cycles is time and effort consuming, and results in a huge optimization model thus, not scalable.
  - A method that performs a selective enumeration (e.g., SLA, Expand, Grow) [Grover et all 2004] may have the advantage of scalability, but does not guarantee an optimal solution.

A column generation based PWCE design approach

- The whole design problem is divided into two sub-problem: master and pricing problems which are executed sequentially.
  - The pricing problem generates dynamically relevant *p*-cycles (this is not a complete enumeration).
  - The master problem optimizes the generated p-cycles.



Master Model

Pricing Model

dual variables

cycle (column)

Find the optimal set of cycles among the generated cycles

Find optimal cycle with maximum reduced cost > 0

- The process is iterated until all the potential *p*-cycle are generated.

Master Problem

Objective

$$\max \quad \sum_{s \in S} u_s w_s,$$

where u_s is the shaping factor.

Parameters

$$a_s^c = \begin{cases} 2 & \text{if } s \text{ is a straddling-cycle span in } p\text{-cycle } c \\ 1 & \text{if } s \text{ is an on-cycle span in } p\text{-cycle } c \\ 0 & \text{otherwise,} \end{cases}$$

$$b_s^c = \begin{cases} 1 & \text{if } s \text{ is an on-cycle span in } p\text{-cycle } c \\ 0 & \text{otherwise.} \end{cases}$$

Master Problem

- Actually, this is a restricted master problem, it grows as new *p*-cycles are added by the pricing problem.
- Variables are w_s (number of protected channel on s) and z^c(number of copies of *p*-cycle c).
- The constraints of the master problem are:

$$\begin{aligned}
\sum_{c\in C} a_s^c z^c \quad - \quad w_s &\geq 0 && s \in S && (1)\\
\sum_{c\in C} b_s^c z^c \quad + \quad w_s &\leq w_s^{max} && s \in S && (2)\\
w_s &\geq w_s^{\min} && s \in S && (3)\\
w_s, z^c \in \mathbb{Z}^+ & && && (4)
\end{aligned}$$

Pricing problem

Objective

$$\max \left(\sum_{s \in S} a_s^c t_s^1 - \sum_{s \in S} b_s^c t_s^2 \right),$$

Variables

$$x_s^o = \begin{cases} 1 & \text{if } s \text{ is an on-cycle span in } p\text{-cycle } c \\ 0 & \text{otherwise,} \end{cases}$$

$$x_s^s = \begin{cases} 1 & \text{if } s \text{ is a straddling-cycle span in } p\text{-cycle } c \\ 0 & \text{otherwise,} \end{cases}$$

$$\max \left(\sum_{s \in S} \overbrace{(x_s^o + 2x_s^s)}^{a_s^c} t_s^1 - \sum_{s \in S} \overbrace{x_s^o}^{b_s^c} t_s^2 \right),$$

Pricing problem

Feed the master problem with new relevant *p*-cycle.

$$\sum_{s \in \mathcal{E}(v)} x_s^o \leq 2 \qquad v \in V \tag{5}$$

$$\sum_{s' \in \mathcal{E}(v): s' \neq s} x_{s'}^o \geq x_s^o \qquad v \in V, s \in \mathcal{E}(v) \tag{6}$$

$$4x_s^s \leq \sum_{v \in \mathcal{V}(s)} \sum_{s' \in \mathcal{E}(v)} x_{s'}^o \qquad s \in S \tag{7}$$

$$\sum_{s \in \mathcal{E}(N)} x_s^o \geq x_{s'}^s \qquad N \subset V, s' \in \mathcal{E}(N) \tag{8}$$

$$x_s^o, x_s^s \in \{0, 1\}. \tag{9}$$

Evaluation metrics

Classical evaluation metrics (redundancy, computing time) we have considered two other metrics

$$\text{Protection gap ratio} = \frac{\sum_{s \in S} \text{unprotected capacity on } s}{\text{overall protected capacity}}$$

$$\text{Protection overhead ratio} = \frac{\sum_{s \in S} \text{protection overhead on } s}{\text{overall protection capacity}}$$

$$\text{Protection overhead} =$$
$$\text{supposed protected capacity} - \text{the protected capacity.}$$

NSF Network

NSF Network (12 nodes, 21 links), S-ILP(SLA), E-ILP(Extend), G-ILP(Grow), B-ILP(Brute strength), CG(Column generation)

Table: all the spans have the same spare capacity budget - NSF Network

	NSF				
	S-ILP	E-ILP	G-ILP	B-ILP	CG
# of potential *p*-cycles	18	31	88	138	3
Computing time (sec.)	0.02	0.03	0.04	0.7	0.03
# of selected *p*-cycles	7	5	12	1	1
Average *p*-cycle size	7.9	10	9.2	14	14
Working capacity	840	880	994	1120	1120
Spare capacity	740	668	658	560	560
Redundancy	88%	76%	67%	50%	50%
Protection gap ratio	12%	15%	2.9%	0%	0%
Protection overhead ratio	12%	0.3%	5.5%	0%	0%

COST239 Network

COST239 Network (11 nodes, 26 links)

Table: all the spans have the same spare capacity budget - COST239 Network

	COST239				
	S-ILP	E-ILP	G-ILP	B-ILP	CG
# of potential *p*-cycles	26	97	763	3517	3
Computing time (sec.)	0.02	0.08	0.42	21.5	0.03
# of selected *p*-cycles	19	25	19	1	1
Average *p*-cycles size	4.5	6.1	6.95	11	11
Working capacity	1120	1292	1472	1640	1640
Spare capacity	858	788	608	440	440
Redundancy	76%	61%	41.3%	26.8%	26.8%
Protection gap ratio	9.2%	0%	0%	0%	0%
Protection overhead ratio	15.4%	1.3%	5.3%	0%	0%

USA network

USA Network (28 nodes, 45 links)

Table: all the spans have the same spare capacity budget - USA Network

	USA				
	S-ILP	E-ILP	G-ILP	B-ILP	CG
# of potential *p*-cycles	35	96	1017	7314	3
Computing time (sec.)	0.03	0.05	0.64	190	0.2
# of selected *p*-cycles	19	17	18	1	1
Average *p*-cycles size	6.7	8.36	12.38	28	28
Working capacity	1916	2045	2308	2480	2480
Spare capacity	1604	1501	1264	1120	1120
Redundancy	83.7%	73.4%	54.8%	45%	45%
Protection gap ratio	4.2%	2.7%	1.3%	0%	0%
Protection overhead ratio	11.3%	8.8%	9.5%	0%	0%

Capacity efficiency

Table: optimality gap of S-ILP, E-ILP and G-ILP

	S-ILP	E-ILP	G-ILP
NSF	62%	41.3%	25.4%
COST239	73.8%	35.5%	19.8%
USA	54.2%	39.9%	20.6%

Larger networks

France network (43 nodes, 71 links) Brazil (27 nodes, 68 links)

Table: all the spans have the same spare capacity budget - larger networks

	France			Brazil		
	G-ILP	B-ILP	CG	G-ILP	B-ILP	CG
# of potential *p*-cycles	2263	-	2	4613	-	2
Computing time (sec)	4.15	-	0.6	45.5	-	6.6
# of selected *p*-cycles	44	-	1	51	-	1
Average *p*-cycles size	16	-	43	11	-	27
Working capacity	3625	-	3960	3961	-	4520
Spare capacity	2011	-	1720	1627	-	1080
Redundancy	55.5%	-	43%	41%	-	23%
Protection gap ratio	1.21%	-	0%	0.3%	-	0%
Protection overhead ratio	4.3%	-	0%	10.1%	-	0%

Restricted spare capacity budget

Table: spans with different spare capacity budgets

	COST239					
	[40-80]			[60-80]		
	G-ILP	B-ILP	CG	G-ILP	B-ILP	CG
# of potential *p*-cycles	763	3517	7	763	3517	14
Computing time (sec)	1256	1846	8.4	1722	128	21.55
# of selected *p*-cycles	17	7	6	19	4	6
Average *p*-cycles size	6.5	7.43	8.67	6.84	8	8.33
Working capacity	1010	1056	1056	1303	1429	1429
Spare capacity	442	390	384	550	426	426
Redundancy	44%	37%	36.3%	43%	30%	30%
Protection gap ratio	0%	0.5%	1.2%	0.3%	0.1%	0.1%
Protection overhead ratio	18%	64%	55.7%	13%	28%	22%

Questions

Questions??