Efficient Heuristics for GRWA

Brigitte Jaumard
CIISE - Concordia Institute
for Information Systems Engineering
Concordia University
Email: bjaumard@ciise.concordia.ca

Francois Vanderbeck
Universite de Bordeaux 1
Email:
francois.vanderbeck@math.u-bordeaux1.fr

Benoit Vignac
Centre de Recherche
sur les transports
Universite de Montreal
Email:
benoit.vignac@crt.umontreal.ca

***Abstract*— This paper presents effective heuristics to solve GRWA problem where the objective is to minimize network cost. We present our greedy, tabu search and rounding off column generation based heuristics and we test them on some instances to make some comparison.**

I. Introduction

In optical networks with grooming capability, many streams can share the same lightpath. This lead to an efficient use of network bandwidth but can increase network cost because of the expensive equipments that have to be installed at nodes. Then the grooming, routing and wavelength assignment problem has become a big issue.

Many authors have studied the case where the objective is to maximize the number of accepted requests [14] [13] [12] [10] but the network cost is actually the main objective function to optimize [4] [11] [3] [5] [9] [1] [10] [8] [2] [13] [12] [6] [7] because of the increase of the available bandwidth with WDM equipment. In parallel, network were first build as ring [4] [11] [3] [5] [9] [2] [10] [7] but mesh topology [8] [1] [14] [13] [12] [6] [7], which is more flexible with multiple paths available, get more interest for research. Moreover the single hop routing, as in RWA, is replaced by multi-hop routing [4] [3] [5] [9] [8] [1] [14] [13] [12] [7], to allow the grooming of requests with different end points. Grooming requests with the same granularity, e.g, four requests on a wavelength [4] [11] [3] [5] [9] [2] [10] [8] [1], was replaced by traffic with different granularities belonging to standard bandwidth value, e.g., OC-1, OC-3, OC-12, OC-48, [14] [13] [12] [6] [7]. The GRWA is very difficult to solve by exact method (mathematical programming) because of its complexity, so mathematical formulation are only provided used for very little instances and heuristics are used to build good solutions.

Under general assumptions for the network topology and the traffic, we develop three heuristics to solve the GRWA. The objective is to minimize the overall number of network optical ports while satisfying the traffic demand. We restrict the number of optical hops to two in order to minimize the regeneration delay.

The greedy heuristic try to build a feasible solution by adding requests to the network iteratively. The Tabu Search is a local search where the next candidate is selected from a pool of solutions obtained by slightly modify the current solution. The rounding off uses column generation process and iteratively select a column, which correspond to the routing on a single wavelength, to build a GRWA solution.

After introducing the GRWA problem in more details we will describe our heuristics, that will be compared after some computational expirements on some instances. We finish by some conclusions and remarks.

II. Statement of the GRWA Problem

Given an optical network with grooming capability, we try to assign an optical path, i.e., a physical path and a wavelength to each request of the traffic demand, so as to minimize the network cost, counted as the overall number of optical port.

We will assume general topology network with uniform wavelength capacity U, e.g., OC-192, and uniform number of wavelength W per optical fiber. There are two directed optical fibers between each connected node. Traffic granularity are OC-1,3,12 and 48, and there can have many requests with the same granularity and end-points. In view of modeling the Internet traffic, we will consider asymmetrical traffic, meaning that the uploading and downloading traffic between two nodes can be different. We assume that there is no wavelength conversion and no bifurcated flows, i.e., each request is routed over an unique optical path on an unique wavelength.

Let us consider an optical network represented by a graph $G = (V, A)$ with node set V where each node is associated with a node of the physical network, and arc set A where each arc is associated with a directional optical fiber link of the physical network. Let $n = |V|$ and $m = |A|$. The traffic is given by a three dimensional $n \times n \times |T|$ matrix D, with $T = \{1, 3, 12, 48\}$ the set of standard granularities, where each element D_{sdt} defines the number of requests with a granularity OC-t between s and d or equivalently, the (s, d, t) demand.

This study further assumes that signal regeneration occurs on the same wavelength whenever there will be an O/E/O conversion and the optical path must be composed by less than H optical hops.

In summary, the constraints of the GRWA are :

- *Demand covering.* Each traffic requested must be granted.
- *Fiber capacity.* No more than W wavelengths are available on each directional fiber link.
- *Wavelength capacity.* The overall bandwidth of the requests groomed on a given fiber link and a given wavelength cannot exceed the capacity U.
- *Optical-Electrical-Optical (OEO) conversions.* The signal must be converted to electric whenever the grooming

needs to be modified (adding or dropping a traffic stream, merging or splitting several streams). Otherwise, the signal remains in the optical domain. This is called an “optical bypass” (or simply bypass).
- *Wavelength continuity.* The signal must be regenerated on the same wavelength after each OEO conversion at an intermediate node.
- *Port installation.* In order to perform the OEO conversions, a port must be installed at each endpoint of an optical hop.
- *Optical hops.* The number of optical hops in a optical path is bounded by H.

III. Grooming preprocessing, potential paths and sorting functions

Grooming preprocessing: as the number of requests can be very large, increasing the number of routing and grooming patterns, the set of feasible solutions should be reduce to focus our search on good potential solutions. We fix some grooming decisions by aggregate requests with same (s, d), they will be route on the same optical path.
We note $D_{sd} = \sum_{t \in T} D_{sdt}$ the total demand for the (s, d) pair, then:

- Maximal traffic aggregation: requests for the (s, d) pair are replaced by $\frac{D_{sd}}{U}$ requests that required one OC_U, call *tunnel* and one request that required $\text{OC}_{D_{sd} \% U}$. For instance, if $U = 192$ and $D_{sd} = 211$, new requests for (s, d) pair will be an OC_{192} and an OC_{29}.
- Partial traffic aggregation: we only try to build tunnels, the remainder, OC_{29} in the previous example, is not aggregated.

The goal of such preprocessing is to groom requests like they would be in an optimal solution. So we create as much as possible tunnels because they always use only two ports as they are routed on single hop optical path. Note that maximal aggregation gives a lower number of requests but the partial aggregation keep more flexibility in the routing decisions. Granularity multiplicity permits to perform both aggregations, i.e., the disaggregate solution will always satisfy GRWA constraints.

Potential paths: for each (s, d) pairs, we create $P_{\max}$ potential paths in P_{sd}. $P_{\max}$ is a parameter that can be adjusted depending on the number of nodes, of arcs and the density of the network. Potential paths are the first $P_{\max}$ shortest paths, in number of arcs. As traffic aggregation, fixing a limited number of path for each requests reduce the solution space and the use of shortest paths allow us to focus on good routing decision as more longer paths are not expected to be in an optimal solution.

Request sorting function: in greedy and tabu search heuristics we will need to add a list of requests to a partial solution to build a solution. The order of the requests can influence the quality of the new solution so we have to use sorting function that depend on:

- requests bandwidth requirement,
- length of the shortest path.

We have use four sorting functions:

- $\text{sort}_{l+d}()$ sorts requests by increasing length of the shortest shortest path and then by decreasing bandwidth requirement.
- $\text{sort}_{l-d}()$ sorts requests by decreasing length of the shortest shortest path and then by decreasing bandwidth requirement.
- $\text{sort}_{dl-}()$ sorts requests by decreasing bandwidth requirement and then by increasing length of the shortest shortest path.
- $\text{sort}_{dl+}()$ sorts requests by decreasing bandwidth requirement and then by increasing length of the shortest shortest path.

Those functions aim to use the lower network capacity possible. When we sort the list of requests, tunnels are always at the beginning of the list and they are sorted by increasing or decreasing length of the shortest shortest path; for tunnels we only have to choose their optical path as there are not groomed with other requests.

IV. Greedy heuristic

Greedy heuristic is an iterative process where a requests is added to the partial solution at each iteration without changing previous iterations decisions. We can perform the traffic aggregation prior to the greedy and also sort the list of requests. Requests are added to the network in the order of the sorted list of requests and for each request r, we test available optical paths, i.e., each (path, wavelength) pairs:
for each available path p from 1 to P_{sd}:
  for each available wavelengths λ from 1 to W
    test the attractibility of (p, λ) for r.

The attractibility is the inverse of the evaluation function value:
$f^{eval}(p, \lambda)$ = number of ports added + $\beta \times$ capacity violation.
We select the optical path with no more than H optical hops that gives the minimum value of the evaluation function but priority is given to optical paths that satisfy capacity constraints, i.e., whenever there is an optical path that satisfy capacity constraints, we select it so as to always try to find a feasible solution.
When two solutions have the same attractibility, we select the ones with the lower number of optical hops because the number of optical hops is expected to grow up when we will add other requests, making the number of feasible optical path lower.
The solution obtained can be infeasible, i.e., it can happen that the capacity constraints are violated.

V. Tabu Search

The tabu search is a local search based heuristic where the next candidate is selected in the neighborhood of the current solution and gives the best value of an evaluation function, eventually different from the objective function. It allows to deteriorate the evaluation function to escape from a

local optimal solution and keep in memory previous solution attributes to avoid cycling on the same solutions.
To build the neighborhood of the current solution, we use moves, that are slightly modification of the current solution. We will use three different moves to allow our tabu search to not only intensificate the search in a potentially good solution space area but also to diversicate the search to visit solution with different routing pattern.
The first solution of the tabu search is the solution of the greedy heuristic that can be infeasible. This is another feature of our tabu search that allow the current solution to be infeasible (capacity violation) so that we can cross an infeasible solution space instead of to go round it and increase the distance to travel.
Our tabu search is composed of three phases:

- Removing port (intensification)
- Moving tunnel (diversification)
- Moving requests causing capacity violation (feasibility recovering)

A. Port move

We define the port move of our tabu search as the removing of an existing port by re-routing requests that use this port. This aims to improve the objective function by removing ports that seems to be useless or not well used. This move is used in the *removing port phase*.
Selecting the port to remove: we use two criterion to select the port to be removed:

- the port that gives the greater percentage of transit requests.
- the port that gives the greater percentage of demand use for transit requests.

Those two criteria aim to evaluate the good use of ports, transit requests are not expected to pass through a port.
Re-routing requests when we have selected the port to remove, we determine requests that have to be re-routed to remove this port. We have two criteria:

- Re-route all requests that use this port.
- Re-route the minimal number of requests to create a bypass.

Those two methods differ in the number of requests to re-route, the second has the advantage to keep a bypass.
The rerouting of requests is done in the same way as in the greedy heuristic.
Tabu List: we keep in memory the last $TABU_1$ port removed and forbid to re-create them.

B. Rerouting of Single Hop Tunnels

To diversicate the explored solution space, we reroute some tunnels, i.e., we change their optical path. We use the same evaluation function as in greedy heuristic. This move is used in the *rerouting tunnels phase*.
The port removing is not expected to produce tunnels rerouting because they use well their port. So if we want to make room for others requests and thus diversicate the explored space, we have to explicitly reroute them.
Tabu List: We keep in memory the last $TABU_2$ tunnels rerouted and forbid to re-route them one the same optical path.

C. Connection move

This move consists in changing the optical path of some request, that is change its path or / and its wavelength. This move is used in the *removing port phase*. To reroute the request we act as in the greedy heuristic, i.e., we search for the optical path giving the best evaluation function. The only difference is that we use tabu status to forbid some optical path but they are removed if the solution is better than the best solution ever visited. Even are broken by selecting the (path, wavelength) that gives the minimum number of hops.
Tabu List: we keep in memory the last $TABU_3$ requests rerouted and forbid to re-route them one the same optical path.

D. Feasible vs infeasible solution

During the tabu search we may select solutions that do not satisfy capacity constraints. As we want to find the best feasible solution possible, we increase the capacity violation penalty β if we select an non-admissible solution. However it's not obvious that increasing indefinitely the penalty will insure that the search will select an admissible solution. This is why we have to restore admissibility using a special phase, that, starting from an infeasible solution, search for an admissible solution.
Restore phase: We select the (arc, wavelength) with the greater capacity violation and re-route requests that use this (arc, wavelength). Requests are sorted and we add them on (path, wavelength) that have enough capacity and leads to a number of hops lower than H. We use the objective function to select the new optical path. Even are broken by selecting the (path, wavelength) that gives the minimum number of hops and tabu status is always remove.

E. Multiphase Tabu

There are three phases in our Multiphase Tabu:

- The first phase, is composed of port moves. This phase is the intensification phase that search for good solution by trying to decrease the number of blades.
- The second phase is the diversification phase that reroute tunnels that would not be reroute with the first phase.
- The third phase is the feasibility recovering phase that move requests that are on (arc, wavelength) where there is violation of the capacity constraints.

Stopping criteria

1) number of iterations in the i phase greater than $ITER_i$,
2) number of iterations with an infeasible solution greater than $ITER^I_{\max}$,
3) number of iterations without decreasing the number of blades greater than $ITER_{obj}$.

The third stopping criteria is not used for the feasibility recovering phase because we only focus to decrease the capacity constraint violation. The multiphase tabu stop if we have perform more than $ITER_{\max}$ iterations.

β **updating** The penalty of the capacity constraint violation is update during the process to take in account the feasibility of explored solutions:

- If the solution choose in the phase 1 and 2 is infeasible: $\beta := \beta \times 2$, to make infeasible solutions less attractive,
- If the current solution is feasible during four iterations: $\beta := \frac{\beta}{2}$.

We begin the Multiphase Tabu by the port phase to decrease the number of blades. If this phase stop because of criteria 1 or 3, then we perform the tunnel rerouting phase, else we perform feasibility recovering phase.
When the tunnel rerouting phase is stopped because of criteria 1 or 3 and the solution is feasible then we perform phase 1, if the solution is infeasible we perform phase 3 to try to recover feasibility. If the the tunnel rerouting phase is stopped because of criteria 2, we perform phase 3.
When the infeasibility recovering phase stop, we always perform phase 1.

VI. Mathematical Formulation

Our formulation of the GRWA is based on the Dantzig-Wolfe decomposition that exhibit a subsystem of equations that can be more tractable than the whole problem, and use these subsystem to define variables of the so-called *master* program. The motivation of this kind of reformulation is that it gives a better LP relaxation bound and remove symmetry (same solution represented by multiple indices permutations).
As we have made the assumption of wavelength continuity we can decompose the problem over each wavelength, the subsystem defines a potential routing pattern on a wavelength. We define the *independent routing configuration* (IRC) as a traffic pattern that can be carried out on a single wavelength over the optical network in partial fulfillment of the connection requests. It satisfies wavelength capacity, OEO conversions, port installation and optical hop constraints (defining the subsystem). Let $\mathcal{C}$ be the set of IRCs, for each IRC $c \in \mathcal{C}$, we define its traffic pattern by x^c_{sdt} that gives total traffic for any (s,d,t) demand. The cost $\textsf{cost}_c$ of IRC c represents the number of port installations in the configuration.
We also implicitly allows exchanges of larger granularity traffic into smaller granularity one, for the same (s,d), to stabilize the column generation process. It leads to scale the reward for (s,d,t) demand so as to give easier column generation process.
The definition of IRCs can be further restricted to *maximal proper* IRCs. An IRC is *maximal* if it cannot carry anymore traffic than it does without increasing its cost. It is *proper* if the traffic it carries does not exceed cumulative demand, i.e.,

$$\sum_{t' \in T : t' \le t} t' \, x^c_{sdt'} \;\le\; \sum_{t' \in T : t' \le t} t' \, D_{sdt'} \quad t \in T. \tag{1}$$

The motivation for these restrictions is to accelerate the convergence of the column generation procedure.
A given IRC can be reused for different wavelengths on the network as long as the number of available wavelengths is not exceeded. Thus, a solution to GRWA can be viewed as stack of up to W IRCs each of which using its own wavelength.

$$[\text{GCSTAB}] \;\min\; z_{\text{GCSTAB}} = \sum_{c \in \mathcal{C}} \textsf{cost}_c \, \gamma_c \tag{2}$$

$$\sum_{c \in \mathcal{C}} \sum_{t' \in T : t' \ge t} \frac{t'}{t} \, x^c_{sdt'} \, \gamma_c \ge \sum_{t' \in T, t' \ge t} \frac{t'}{t} \, D_{sdt'} \quad (s,d,t) \in K_d \tag{3}$$

$$\sum_{c \in \mathcal{C}} \gamma_c \le W \tag{4}$$

$$\gamma_c \in N \qquad c \in \mathcal{C} \tag{5}$$

The cumulative demand for granularities $t' \ge t$ (3), expressed in OC-t units must be covered, we can take multiple copies of the same IRC (5), but no more than W IRC (4) because of fiber capacity.

VII. Column Generation

The formulation [GCSTAB] has typically an exponential number of columns (variables), one for each IRC. The column generation process, that solve the LP relaxation of [GCSTAB] (replacing constraint (5) by $\gamma_c \ge 0$), begins with a restricted set of columns (IRCs) and gives optimal dual prices to the pricing problem to return a column with negative reduced cost (candidate to be in the optimal solution). If the pricing returns a column with negative reduced cost, it is added to the restricted master and the process reiterate, otherwise the process stop.
The formulation of the pricing is:

$$\rho = \min_{c \in \mathcal{C}} \textsf{cost}_c - \sum_{(s,d,t) \in K_d} \left(\sum_{t' \in T : t' \le t} \frac{t'}{t} \, \nu_{sdt'} \right) x^c_{sdt} + \sigma \tag{6}$$

Where $\nu_{sdt'}$ (resp. σ) are dual prices of constraints (3) (resp. 4).

VIII. Pricing problem

The efficiency of the method depend on our capacity to modelize the subsystem defining an IRC. We first define a *Minimal Independent Routing Configuration* as an IRC that cannot be split into two sets of optical hop with no flow from the first set using optical hop of the other. A MIRC satisfies wavelength capacity constraints and port installation constraints. It will define our routing pattern. As there are a lot of different MIRCs we will use only basic ones, which are expected to be in a good solution: optical hop between s and d will define *single hop* MIRC and will satisfy only (s,d) traffic, an optical path with a stop at node i will define *two hops* MIRCs and will satisfy traffic (s,d), (s,i) and (i,d) traffic. A MIRC $m \in \mathcal{M}$ will be define by its cost $cost_m$, its traffic indicator x^m_{sdt}, its arc indicator δ^m_a.
To build an IRC with have to select a set of MIRC that are

arc disjoint:

$$[priceMIRC] \min \sum_{m\in\mathcal{M}} (cost_m\gamma_m - \sum_{(s,d,t)\in K_d} \pi_{sdt}x^m_{sdt}) - \sigma \quad (7)$$

$$\sum_{m\in\mathcal{M}} \delta^m_a \gamma_m \le 1 \qquad a \in A \quad (8)$$

$$\sum_{t'\in T: t'\le t} t'\, x^m_{sdt'} \le \sum_{t'\in T: t'\le t} t'\, D_{sdt'} \qquad (s,d,t)\in K_d \quad (9)$$

$$\sum_{t\in T} t\, x^m_{sdt} \le U\,\gamma_m \qquad m \in \mathcal{M}(sd) \quad (10)$$

$$\sum_{t\in T} t(x^m_{sdt} + x^m sit) \le U\,\gamma_m \qquad m \in \mathcal{M}(sid) \quad (11)$$

$$\sum_{t\in T} t(x^m_{sdt} + x^m idt) \le U\,\gamma_m \qquad m \in \mathcal{M}(sid) \quad (12)$$

$$\sum_{t\in T} x^m_{sdt} \ge \gamma_m \qquad m \in \mathcal{M}(sid) \quad (13)$$

$$\sum_{t\in T} (x^m_{sit} + x^m_{idt}) \ge \gamma_m \qquad m \in \mathcal{M}(sid) \quad (14)$$

$$\gamma_m \in \{0,1\} \qquad m \in \mathcal{M} \quad (15)$$

$$x^m_{sdt} \in N \qquad m \in \mathcal{M} \quad (16)$$

where $\mathcal{M}(sd)$ (resp. $\mathcal{M}(sid)$) is the set of single hop (resp. two hop) MIRCs.

Constraints (8) impose that MIRC are arc disjoint, constraints (9) are proper column definition constraints, constraints (10 -12) are wavelength capacity constraints for single and two hop MIRCs and constraints (13 -14) are minimality constraints for two hop MIRCs.

To solve the pricing problem we use a greedy heuristic based on the previous pricing formulation: we generate all potential single and two hop MIRCs and compute the optimal traffic for each of them, i.e., we want to maximize $\sum_{(s,d,t)\in K_d} \pi_{sdt}\, x^m_{sdt}$ over MIRC definition constraints. Then we sort MIRCs in increasing order of $cost_m\gamma_m - \sum_{(s,d,t)\in K_d} \pi_{sdt}\, x^m_{sdt}$ and add them to IRC in construction if there are arc disjoint with previous added MIRCs. We also have to recompute optimal traffic for each MIRC that contains traffic previously added, because flow bounds have changed.

IX. Rounding off Heuristic

The rounding heuristic use the column generation solver to generate a set of columns corresponding to the traffic demand vector $\overline{D}$ ($\overline{D} = D$ at the first iteration) and then select a column with fractional value in the master solution to rounding it to a positive integer value.

Starting from the current solution of the linear relaxation of the master, we choose the variable γ_c with some criterion and we set it to $[\gamma_c]$, i.e., to its integer floor value $\lfloor\gamma_c\rfloor$ if $\gamma_c > 1$ and to its integer ceiling value $\lceil\gamma_c\rceil$ otherwise, i.e., we choose an IRC c and set all the routing paths on it for $[\gamma_c]$ wavelengths. If there are columns with integer value in the master LP solution we do set them to their integer value and we iterate without searching for a fractionnal column to round off.

We restart the *heuristic column generation* using residual demand values $\overline{D}$, and round off additional variables till either the master LP becomes infeasible or the master LP solution is integer. Before generating new columns we delete columns that are no longer proper according to the updated demand.

Rounding off a column as a significant impact on the integer solution we are building. It may happen that the first decision leads to a dead-end in the rounding off procedure, i.e., the master LP becomes infeasible. To overcome this drawback, we have to carefully select the columns to round off. We propose several strategies that not only take in account the γ_c value but also their "quality", i.e., some criterion that define a column that would be a part of a good master IP solution:

- Select the γ_c that is the closest to a positive integer value.
- Select the γ_c that is the farthest to an integer value.
- From the 5 best columns (i.e. with the largest γ_c) select the column that has the greatest ratio of used capacity over available capacity. This strategy aims at selecting a column with the largest network capacity usage among the 5 best columns.
- From the 5 best columns (i.e. with the largest γ_c) select the column that has the greatest satisfaction of demand constraints, i.e., the one with the largest $\sum_{(s,d,t)\in K_d} t\, x^c_{sdt}$ value. This strategy aims at selecting the column that satisfies the largest demand.
- From the 5 best columns (i.e. with the largest γ_c) select the column that has the greater ratio of satisfied demand over used capacity. This strategy selects column that makes a good use of the used capacity. This should result in selecting columns with short ligthpaths.
- Select randomly a column in the set of columns with positive master LP value.

X. Computational Experiments

A. Traffic and Network Instances

We tests the three heuristics on NSF and EON networks. For each network we build five instances, one with the total traffic for each (s-d) pair lower (resp. larger) than OC-192, one with a low (resp. large) percentage of small granularities and one randomly generated with total traffic for each (s-d) pair lower than two OC-192.

The number of wavelength is adapted to the traffic to make sure that a feasible solution exists.

	Greedy		Tabu Search		Rounding Off		Upper Bound	
	Value	CPU	Value	CPU	Value	CPU	Value	CPU
NSF1	364		362		370		322	
NSF2	666		660		620		418	
NSF3	1356		1356		1324		1192	
NSF4								
NSF5	514		504		478		392	
EON1	716		716		664		436	
EON2	1436		1436		1308			
EON3	850		850		774		480	
EON4	1184		1184					
EON5								

B. Comparison

XI. CONCLUSIONS

The conclusion goes here.

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] R. Battiti and M. Brunato. A multistart randomized greedy algorithm for traffic grooming on mesh logical topologies. In *Proceedings of IWDM*, 2001.

[2] R. Battiti and M. Brunato. Reactive search for traffic grooming in WDM networks. In S. Palazzo, editor, *Evolutionnary trends of the internet, IWDC2001*, volume 2170 of *Lecture Notes in Computer Science*, pages 56–66. Springer-Verlag, September 2001.

[3] R. Berry and E. Modiano. Reducing electronic multiplexing costs in Sonet/WDM rings with dynamically changing traffic. *IEEE Journal on Selected Areas in Communications*, 18(10):1961–1971, October 2000.

[4] A. Chui and E. Modiano. Traffic grooming algorithms for reducing electronic multiplexing costs in WDM ring networks. *Journal of Lightwave Technology*, 18(1):2–12, January 2000.

[5] O. Gerstel, R. Ramaswami, and G.H. Sasaki. Cost effective traffic grooming in WDM rings. *IEEE/ACM Transactions on Networking*, 8(5):618–630, October 2000.

[6] J.Q. Hu and B. Leida. Traffic grooming, routing and wavelenght assigment in optical WDM mesh networks. In *IEEE Infocom*, March 2004.

[7] B. Jaumard, Y. Solari, and A. Houle. Dimensioning WDM optical networks with minimum MSPP configuration. A publier, 2004.

[8] V.R. Konda and T.Y. Chow. Algorithm for traffic grooming in optical networks to minimize the number of trabsceivers. *Proceeding IEEE Workshop Hight Performance Switching and Rooting*, pages 218–221, May 2001.

[9] J. Wang, W. Cho, V.R Vemuru, and B. Mukherjee. Improved approches for cost-effective traffic grooming in wdm ring networks : ILP formulations and single-hop and multihop connections. *Journal of Lightwave Technology*, 19(11):1645–1653, November 2001.

[10] S. Zhang and B. Ramamurthy. Dynamic traffic grooming algorithms for reconfigurable SONET over WDM networks. *IEEE Journal on Selected Areas in Communications*, 21(7):1165–1172, September 2003.

[11] X. Zhang and C. Qiao. An effective and comprehensive approach for traffic grooming and wavelenght assigment in SONET/WDM rings. *IEEE/ACM Transactions on Networking*, 8(5):608–617, October 2000.

[12] H. Zhu, H. Zang, K. Zhu, and B. Mukherjee. Dynamic traffic grooming in WDM mesh networks using a novel graph model. *Optical Network Journal*, page ?, May/June 2003.

[13] H. Zhu, H. Zang, K. Zhu, and B. Mukherjee. A novel generic graph model for traffic grooming in heterogeneous WDM mesh networks. *IEEE/ACM Transactions on Networking*, 11(2):285–299, April 2003.

[14] K. Zhu and B. Mukherjee. Traffic grooming in an optical WDM mesh network. *IEEE Journal on Selected Areas in Communications*, 20(1):122–133, January 2002.