This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

Approximation Algorithms for Traffic Grooming in WDM Rings

Kevin Corcoran, Seth Flaxman, Mark Neyer, Peter Scherpelz, Craig Weidert, Ran Libeskind-Hadas

***Abstract*— This paper addresses the problem of traffic grooming in WDM rings in which all traffic emanates from a single node and all other nodes are destination nodes. This "one-to-many" scenario arises in metropolitan access networks in which one node serves as a "hub" connecting the ring to a larger network as well as in video-on-demand and other multimedia services where a single source node serves a collection of subscriber nodes. The ring comprises a given number of wavelengths of uniform capacity and a variable number of tunable Add-Drop Multiplexers (ADMs) at each node. Given a set of requests at the destination nodes, where each request comprises a bandwidth demand and a profit for fulfilling the request, our objective is to select a subset of the requests and pack ("groom") them onto the wavelengths such that no wavelength's capacity is exceeded and the total profit of the selected requests is maximized. Although this problem is NP-complete, we give polynomial time approximation algorithms with excellent theoretical performance validated with experimental results.**

***Index Terms*— Optical networks, WDM rings, traffic grooming, approximation algorithms.**

I. Introduction

In this paper we propose new approximation algorithms and heuristics for the problem of traffic grooming in optical ring networks employing wavelength division multiplexing (WDM). Wavelength division multiplexing permits a single fiber to carry multiple connections by placing them on different non-conflicting wavelengths. However, a single wavelength may have substantially more capacity than is required by individual connection requests and thus multiple requests may be packed onto the same wavelength. Traffic grooming is the process of packing multiple "low rate" streams onto the available wavelengths.

We consider WDM networks in which a single node is the source node and all other nodes may be destination nodes. This scenario arises in "single-hub" networks where a local ring network is connected to a backbone network through a unique hub node [1], [2] as well as in applications such as video-on-demand where a single node provides data to subscriber nodes [3].

This work was supported by the National Science Foundation under award 0451293 to Harvey Mudd College.

Kevin Corcoran is with the Department of Computer Science at the University of Oregon, Seth Flaxman is with the Media and Design Laboratory at Ecole Polytechnique Fédérale de Lausanne, Mark Neyer is with the Department of Computer Science at the University of North Carolina, Peter Scherpelz is with the Department of Physics at the University of Chicago and is supported by the Fannie and John Hertz Foundation, Craig Weidert is with the School of Computing Science at Simon Fraser University, and Ran Libeskind-Hadas is with the Department of Computer Science at Harvey Mudd College.

We assume that the ring network has been provisioned with a given number of wavelengths of identical capacity. The source node is assumed to be able to transmit on all wavelengths. In addition, each destination node has some number of tunable ADMs each of which may be tuned to any wavelength in order to deliver or "drop" a request groomed on that wavelength to its node. We further assume that a request may be split (or "bifurcated") into integral amounts (e.g. multiples of OC-3) and groomed over multiple different wavelengths [4], [5].

In the model considered here, each destination node can make an arbitrary number of connection requests of the source node. We assume that the connection requests are given off-line (i.e. are known a priori). Each request has some demand and associated "profit". The profit may represent a priority level or some other measure of the value of satisfying that request. We receive the profit associated with a request if and only if its full demand is groomed onto the available wavelengths. In contrast, the on-line version would receive requests one-by-one and would determine whether or not to satisfy each request and how to groom it. The on-line version is an interesting direction for future research.

The ring may be unidirectional, such as a UPSR SONET ring [6], or it may be bidirectional. In the case that the ring is bidirectional, we assume that the traffic has been partitioned, by some algorithm or heuristic, into clockwise and counter-clockwise traffic. Grooming is then performed independently in each direction. In some cases, partitioning is not necessary. For example, if the bidirectional ring employs dedicated path protection, each connection is made in both the clockwise and counter-clockwise directions and thus the single-source grooming problem comprises a pair of unidirectional problems [7]. The algorithms presented here consider a collection of connection requests that are to be routed in the same direction.

Although the problem under consideration here is shown to be NP-complete in general, we show that some special cases can be solved using existing polynomial time approximation schemes and *for the general case we describe a novel and practical polynomial time approximation algorithm.* We first analyze this approximation algorithm theoretically and give experimental results for a heuristic that refines the algorithm. These results confirm that our approach finds solutions that are generally very close to optimal.

Traffic grooming in rings has received significant attention in recent years. A substantial body of literature has considered the problem of designing a network or "logical topology" of minimum cost for a given traffic pattern [1], [6], [8], [9], [10]. The assumption here is that all traffic requests should

978-1-4244-3435-0/09/$25.00 ©2009 IEEE

be satisfied and the network should be configured to satisfy these requests at minimum cost.

In contrast, this paper addresses the problem of finding optimal traffic grooming solutions in a network in which the number of wavelengths has been specified and the ADMs have been deployed at nodes. The objective is to tune the ADMs at each node and select and groom a subset of the given requests in order to maximize the sum of the profits of the groomed requests. The assumption here is that while the network may have been configured for a particular idealized traffic pattern, the actual traffic patterns may in fact be different. Moreover, the network may not be able to accommodate all of the requests and thus an optimal subset of the requests must be selected.

II. Definitions and Problem Statement

We formally define the Tunable Ring Grooming Problem (TRGP). An instance of TRGP comprises[1]:

1) a set N of nodes that may make requests from the source node,
2) a function $\alpha : N \rightarrow \mathbb{Z}^*$ where $\alpha(n)$ represents the number of tunable ADMs at receiver node n.
3) a set Λ of wavelengths,
4) a constant C representing the capacity of each wavelength,
5) a set R of requests where each request $r \in R$ is an ordered pair (n_r, d_r) where $n_r \in N$ is the node making the request and d_r is a positive integer (e.g. multiple of OC-3) indicating the demand of the request, and
6) a function *profit* : $R \rightarrow \mathbb{Z}^+$ where *profit*(r) is the profit gained upon satisfaction of request r.

For simplicity, we assume that no request demands more than the capacity C of a wavelength which is a reasonable assumption for most types of network traffic. Given an instance of TRGP, a feasible solution comprises a selection $S \subseteq R$ of requests that will be satisfied, a tuning of the ADMs at each node, and a grooming of the requests onto wavelengths such that the capacity of each wavelength is not exceeded and each request in $(n, d) \in S$ is satisfied by merit of the demand d of the request being partitioned into integer parts over a set of wavelengths such that node n is tuned to those wavelengths. The objective is to find a feasible solution that maximizes the profit.

III. The Tunable Ring Grooming Problem

Because the requests in a TRGP instance can be fulfilled on any wavelength, the problem is similar to the Multiple Knapsack Problem (MKP) [11], [12], [13]. The difference between these two problems is that TRGP allows splitting of requests (objects) across multiple wavelengths (knapsacks), whereas MKP does not. MKP is known to be NP-Complete in the strong sense even in highly restricted cases [12]. When each node in the ring has a single ADM, TRGP does not permit splitting and thus the problem becomes MKP. Thus, since TRGP is a generalization of MKP it is also NP-Complete in the strong sense.

In this section, we present approximation algorithms for TRGP. We begin by showing that for certain special cases, existing results for MKP and Knapsack can be exploited to find good approximation algorithms for TRGP. Then, we give our main result: an approximation algorithm for the general case.

A. Applying Known Results for Special Cases

In this section, we briefly mention how known approximation techniques for knapsack problems can be applied to three special cases of TRGP.

1) Exactly one wavelength: If the number of wavelengths, $|\Lambda|$, is exactly 1 then we use an existing Knapsack fully polynomial time approximation scheme (FPTAS)[2] [14] on all requests with a knapsack of capacity C. Because all requests must go on only one wavelength, the Knapsack FPTAS will choose a set of requests whose profit is within $(1 - \varepsilon)$ of optimal.

2) Exactly one ADM per node: In the case that each node making a request has exactly one ADM, the problem is equivalent to MKP. As stated above, the difference between TRGP and MKP is the possibility of splitting. When each ADM in a TRGP instance has only one ADM per node, no splitting can occur. Thus, a polynomial time approximation scheme (PTAS) for MKP due to Chekuri and Khanna [12] can be applied in this case.

3) At least two ADMs per node: Next, consider the case that each node making a request has at least two ADMs. In this case we can apply an FPTAS for Knapsack [14] as follows: Given $|\Lambda|$ wavelengths of capacity C, we construct an instance of Knapsack with the same set of requests and a single knapsack of capacity $|\Lambda|C$. Next, we apply the Knapsack FPTAS on this Knapsack instance. The solution returned by the FPTAS is now "sliced" into blocks of size C and placed into the original $|\Lambda|$ wavelengths. This slicing may divide some requests over at most two wavelengths since each request has demand at most C. Since each node has at least two ADMs, the ADMs may be tuned appropriately to accommodate this slicing. Due to the correspondence between solutions to the TRGP instance and the constructed instance of Knapsack, the $1-\epsilon$ approximation for the Knapsack is a $1-\epsilon$ approximation for TRGP with at least two ADMs per node.

Evidently, the TRGP problem is most challenging to approximate in the more general case when some nodes have a single ADM and others have multiple ADMs. We now address the general case.

B. An Approximation Algorithm for TRGP

In this section we describe a $\frac{q}{q+1}$-approximation algorithm for TRGP for the case in which the demand d_r of each request

[1]Throughout the paper, we denote the positive integers as $\mathbb{Z}^+$, the non-negative integers as $\mathbb{Z}^*$, and the power set of set S as $\mathcal{P}(S)$.

[2]Recall that a polynomial time approximation scheme (PTAS) is an infinite family of approximation algorithms such that for any $\epsilon > 0$ there exists a $(1-\epsilon)$-approximation algorithm that runs in time polynomial in the problem size, where the degree of the polynomial may increase as a function of ϵ. In contrast, a fully polynomial time approximation scheme (FPTAS) is a PTAS in which the degree of the polynomial is constant and does not depend on ϵ.

r is at most C/q for some $q \in \mathbb{Z}^+$. Thus, if a request can demand as much as the maximum capacity of a wavelength then $q = 1$ then the algorithm is guaranteed to give at least $\frac{1}{2}$ of the optimal profit. However, if each request demands at most $\frac{1}{3}$ of the maximum capacity of a wavelength, then the algorithm is guaranteed to give solutions that are with $\frac{3}{4}$ of the optimal profit.

Throughout the following discussion we use the terms "wavelength", "bin", and "knapsack" interchangeably. We use the *First Fit (FF)* and *First Fit Decreasing (FFD)* algorithms which are defined as follows: The First Fit algorithm numbers the wavelengths (bins) in some arbitrary order and places each request in the lowest numbered wavelength which can accommodate it. The First Fit Decreasing algorithm first sorts the requests in non-increasing order of demand and then applies the First Fit algorithm to the requests in this order.

We begin by considering the special case that $q = 1$. The following lemma is used for this case:

Lemma 1: Consider an instance of MKP in which $K \geq 2$ bins of capacity C are available. Let $s(i)$ denote the size of item i where $s(i) \leq C$. If $\sum_i s(i) \leq KC/2$ then at least one bin will remain empty when the items are packed using First Fit.

Proof: Suppose First Fit is applied and that in the resulting solution no bin is empty. We wish to show that $\sum_i s(i) > KC/2$. By the definition of First Fit, no item that was packed in bin i could have been packed in bins 1 through $i-1$. Let b_i denote the total amount of capacity of bin i used by First Fit. In general, for $i, j \in \{1, \ldots, K\}$ and $i < j$, we must have $b_i + b_j > C$. If no bin is empty, we have this inequality for all pairs of bins, so by generating all $\binom{K}{2}$ inequalities and summing them, we find

$$(K-1)b_1 + (K-1)b_2 + \cdots + (K-1)b_K > \frac{(K-1)KC}{2}$$

which implies that

$$b_1 + b_2 + \cdots + b_K > \frac{KC}{2}.$$

Thus, we have shown that if no bin is empty,

$$\sum_i s(i) > \frac{KC}{2}.$$

Equivalently, if

$$\sum_i s(i) \leq \frac{KC}{2},$$

then some bin must be empty. ■

For the special case of $q = 1$, we sort the requests by non-increasing *density* which is defined as the ratio of the profit of the request divided by its size (demand). Next, we pack the requests into the wavelengths by First Fit. By Lemma 1, this will fill at least half of the total capacity of the bins before one request fails to fit. Thus, over half of the capacity will be filled with the most dense requests possible, so this is a $1/2 = q/(q+1)$-approximation when $q = 1$.

Assuming $q \geq 2$ and $|\Lambda| \geq 2$, we first present two lemmata about approximating BIN PACKING or KNAPSACK using the First Fit Decreasing algorithm (FFD) Algorithm.

Lemma 2: Given an instance of a packing problem with bin capacity C and $s(i) \leq C/q, \forall i$, if an item does not fit into a bin using FFD, then that bin must be filled to strictly more than $\frac{q}{q+1}$ times its capacity.

Proof: If an item does not fit into a bin, then we consider the first such item, i. There are two cases. First, we consider the case in which $s(i) > \frac{C}{q+1}$. Note that since any q items must always fit in a bin and items are inserted in non-decreasing order of size, there are already at least q items of size at least $s(i) > \frac{C}{q+1}$ in the bin and the total filled space is greater than $C\frac{q}{q+1}$. If $s(i) \leq \frac{C}{q+1}$, there must be less than $\frac{C}{q+1}$ free space in the bin so the filled space is greater than $C\frac{q}{q+1}$. ■

Lemma 3: Given K bins of capacity C and items with $s(i) \leq \frac{C}{q}, \forall i$, any set of items with total size at most $C\left(\frac{Kq+1}{q+1}\right)$ will fit in the bins when packed using FFD.

Proof: Consider an arbitrary set of items with total size at most $C\left(\frac{Kq+1}{q+1}\right)$. When packing these items using FFD, either all items will fit in the first $K-1$ bins, or the first $K-1$ bins must all be filled to greater than $C\frac{q}{q+1}$ capacity by Lemma 2. This means that the subset of items which are not in the first $K-1$ bins will have size at most $C\left(\frac{Kq+1}{q+1}\right) - C(K-1)\frac{q}{q+1} = C$. These items will fit in bin K, so the entire set of items will fit as well. ■

Now we present Algorithm 1 where q is the maximum value such that $d_r \leq C/q, \forall r \in R$.

Input: An instance of TRGP with $d_r \leq C/q, \forall r \in R$
1 Sort elements by non-increasing density into list S
2 Let $A = S$ if total demand $\leq \frac{C|\Lambda|q}{q+1}$ and otherwise let A be the minimal prefix of S with total demand $> \frac{C|\Lambda|q}{q+1}$
3 Pack A onto wavelengths with FFD
4 **if** *some request in A was not packed* **then**
5 Let r_b denote the first request considered by FFD that was not packed
6 Let B be the set comprising r_b and all packed requests with size $\geq d_b$
7 Discard from B the request with least profit
8 **if** r_b *was not discarded* **then**
9 Pack r_b in place of the discarded request
10 **return** the assignment

Algorithm 1: $q/(q+1)$-Approximation for TRGP

In line 1, the algorithm sorts the requests by non-decreasing density, resulting in a list S. In line 2, the algorithm finds the shortest prefix of S such that the sum of the demands of these requests in the prefix exceeds $\frac{C|\Lambda|q}{q+1}$. This prefix is denoted by A. Next, in line 3, the list of requests A is packed onto wavelengths using FFD, which involves re-sorting A by size and placing the requests in the re-sorted list using FF. If some request fails to be packed, the algorithm continues on to determine if subsequent requests can still be packed. If request r is packed into wavelength λ, this means r will tune one of its ADMs to λ, and will be carried entirely by λ. Next, if some request was not packed in this process, we determine the first such request, r_b, in line 5. In line 6, the

set comprising r_b and all packed requests with demand greater than r_b are collected into a set B. In line 7, the least profitable request in B is discarded. If this request was r_b, the packing is complete. Otherwise, the removed request is replaced by request r_b and the packing is complete.

Theorem 1: Algorithm 1 is a $q/(q+1)$-approximation for TRGP with running time $O(|R|\log|R|+|R||\Lambda|)$ for all $q>1$.

Proof: First, if the total demand of all of the requests is less than or equal to $\frac{C|\Lambda|q}{q+1}$ then, by Lemma 3, all of the requests will be packed by FFD and the algorithm therefore finds an optimal solution. Therefore, we need only consider the case that the total demand exceeds $\frac{C|\Lambda|q}{q+1}$ and thus A is the minimal prefix of S with total demand exceeding $\frac{C|\Lambda|q}{q+1}$.

If the algorithm successfully packs all requests in A at line 3, then it packs at least $\frac{q}{q+1}$ of the total capacity of the wavelengths using the densest requests, and the solution is therefore within $\frac{q}{q+1}$ of optimal.

The only remaining case is that the algorithm does not pack all of the requests in A at line 3. We begin by defining the profit and demand of a set of requests. Given a set S of requests, let $p_S=\sum_{s\in S}p_s$ and $d_S=\sum_{s\in S}d_s$. Similarly, we extend the definition of profit density to sets: The profit density of S is p_S/d_S.

We begin by identifying a set of requests, Q, whose profit p_Q is an upper bound on the optimal profit. To this end, we first sort the requests by non-increasing density. Next, we pack these requests in order into a single knapsack of size $C|\Lambda|$ until we reach a request that cannot be packed. We pack as much of this last request as possible to fill the knapsack, potentially taking a fractional part of this request and thus only a fractional part of its profit. The total profit of this packing, denoted p_Q, is an upper bound on the optimal possible profit because it contains the most dense requests and a total of $C|\Lambda|$ demand.

Note that d_A must be greater than $C\frac{|\Lambda|q+1}{q+1}$, as otherwise, by Lemma 3, all requests would have fit. Furthermore, the density of A is at least as great as the density of Q, that is,

$$\frac{p_A}{d_A}\geq\frac{p_Q}{d_Q}.$$

As $d_Q=|\Lambda|C$, we have

$$|\Lambda|Cp_A\geq C\frac{|\Lambda|q+1}{q+1}p_Q$$

which implies that

$$\frac{|\Lambda|q}{|\Lambda|q+1}p_A\geq\frac{q}{q+1}p_Q.$$

This means that if our algorithm discards at most $1/(|\Lambda|q+1)$ profit from A and returns a valid wavelength assignment, it will be a $q/(q+1)$-approximation.

We now show that r_b must be the only request in A that was not successfully packed by FFD. Consider the configuration of the wavelengths at the moment that r_b was considered and rejected. Note that since r_b did not fit in any wavelength, in particular it did not fit in the last wavelength. Therefore, the total demand placed in the last wavelength plus d_b must be greater than C. Since A is the minimal prefix with total size greater than $C\frac{|\Lambda|q}{q+1}$, we have:

$$d_A\leq C\left(\frac{|\Lambda|q}{q+1}+\frac{1}{q}\right) \tag{1}$$

Thus, the first $|\Lambda|-1$ wavelengths must contain total demand less than

$$C\left(\frac{|\Lambda|q}{q+1}+\frac{1}{q}-1\right)=C\left(\frac{(|\Lambda|-1)q}{q+1}+\frac{1}{q(q+1)}\right) \tag{2}$$

Next we show that all requests after r_b will be successfully packed by FFD. As r_b did not fit in any wavelength, by Lemma 3 we have that the total size of request r_b and all requests fit in before it must have size greater than

$$C\left(\frac{|\Lambda|q+1}{q+1}\right) \tag{3}$$

Thus, from (1) and (3), the requests to be fit after r_b will have total size at most

$$C\left(\frac{|\Lambda|q}{q+1}+\frac{1}{q}-\frac{|\Lambda|q+1}{q+1}\right)=\frac{C}{q(q+1)} \tag{4}$$

Every wavelength was packed with demand at least $\frac{Cq}{q+1}$ by Lemma 2. Thus, the first $|\Lambda|-1$ wavelengths are packed with demand at least $C\frac{(|\Lambda|-1)q}{q+1}$ but not more than the demand in (2). Therefore, none of these first $|\Lambda|-1$ wavelengths hold more than $\frac{1}{q(q+1)}$ above $\frac{Cq}{q+1}$ and thus each of these wavelengths has residual capacity at least

$$C-\left(\frac{Cq}{q+1}+\frac{1}{q(q+1)}\right)\geq\frac{C}{q(q+1)} \tag{5}$$

where the rightmost inequality holds because $q>1$. Thus, by (4) and (5), all requests considered after r_b will fit in any one of the first $|\Lambda|-1$ wavelengths and thus r_b must be the only discarded request.

Finally, since each request has demand at most $\frac{C}{q}$, at least $|\Lambda|q$ requests were successfully packed before FFD attempted to pack r_b. Thus, there are at least $|\Lambda|q+1$ requests in set B in line 6. The algorithm then has $|\Lambda|q+1$ requests to choose from in line 7. Since it discards the least profitable one, it discards at most $1/(|\Lambda|q+1)$ of the profit in A. This results in a valid wavelength assignment, so we must have a $q/(q+1)$-approximation.

Algorithm 1 has running time $O(|R|\log|R|+|R||\Lambda|)$ because the dominant steps are sorting the list of requests, and packing the objects in A with the FFD algorithm, which will do no worse than scanning $|\Lambda|$ possible wavelengths for each request. ■

IV. Experimental Results

A. Algorithms and Heuristics

We refined Algorithm 1 into a heuristic presented in Algorithm 2. The heuristic invokes Algorithm 1 and then refines the solution, never performing any worse than Algorithm 1 and thus shares the worst-case analysis of Algorithm 1 derived in the previous section. However, the additional splitting performed by the heuristic may improve the quality of the solutions in practice.

Input: An instance of T-RGP
1 Run Algorithm 1
2 $p_{\text{App}} =$ profit from Algorithm 1
3 **for** $i = 1$ **to** $|R|$ **do**
4 $B =$ the set of the i most dense requests
5 Sort B by non-increasing size
6 $b_j =$ the jth element of B
7 $n_j =$ the node making request b_j
8 $d_j =$ the demand of request b_j
9 **for** $j = 1$ **to** i **do**
10 **if** $\alpha(n_j) \geq 2$ **then**
11 Place one piece of b_j in the first partially (or completely) empty bin
12 Repeat until $\alpha(n_j) - 1$ pieces have been fit in, or all of b_j's demand, d_j, is in bins
13 **if** *Part of* b_j *has not been fit in* **then**
14 Place this part into the first bin with adequate free space. If no bin has adequate space, discard b_j
15 $p_i =$ the total profit of requests fit into bins
16 $S =$ the solution corresponding to $\max\{p_{\text{App}}, p_i\}$
17 **return** S

Algorithm 2: Heuristic for T-RGP

This heuristic algorithm performs a packing similar to FFD in lines 9–14, except that it allows for splitting with objects that have 2 or more ADMs. Furthermore, in line 3 it performs this packing on ever-increasing sets of the most dense objects of the instance, and takes the packing that gives it the most profit in line 16. The running time is $O(|R|^2 \log |R| + |R|^2|\Lambda|)$, because it performs an $O(|R| \log |R| + |R||\Lambda|)$ packing $|R|+1$ times. This heuristic has several desirable features: It tries to fit more requests than does the approximation algorithm, it splits requests, and it tries both packings in which all objects are used as well as packings in which the least dense objects are not used.

B. Methods

To test our heuristic based on the approximation algorithm, we generated random instances of TRGP and then compared the solutions found by our heuristic with the optimal solutions. In each case, we computed the ratio of the profit found by our heuristic to the profit of the optimal solution. The optimal solutions were found by formulating the problem as an integer linear program (ILP) and solving the ILP for each instance with the Lingo LP/ILP solver from Lindo Systems, Inc. Since ILP is NP-complete itself, this permitted us to solve only relatively small problem instances. The problems were generated and solved on a Dual 2.4 GHz Xeon with 4 GB of RAM running FreeBSD 6.1.

The parameters are listed in Table I along with the values we tested. Note that the density of items was either constant (same for all items) or chosen from a uniform random distribution with values ranging from $\frac{1}{2}$ to 2. Problems were generated subject to these parameters, with choices made from a uniform distribution when needed. For example, when $q = 2$ and $C = 8$, demands were chosen uniformly in the range $[1, 4]$. Note that all requests have either 1 ADM or 2 ADMs. Each request has a certain probability α of having 2 ADMs. We generated 1,000 random instances for each of the 120 combinations of parameters.

TABLE I
PARAMETERS USED IN GENERATING RANDOM INSTANCES

Parameter	Possible values
Wavelength capacity C	4, 8, 16
Number of wavelengths	5
Number of requests	16, 32
Probability α that a request has two ADMs (one ADM otherwise)	$\alpha = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1$
Demand limited to fraction $1/q$ of capacity	$q = 1,\ 2$
Density	Constant or variable ($\in U[1/2, 2)$)

For a small fraction of the generated instances of TRGP with low α values, the ILP spent over 10 hours without terminating. In these cases, an upper bound on the optimal solution was found by relaxing the integrality constraints of the ILP.

C. Results

For all runs, we calculated the profit given by the heuristic divided by optimal profit. This ratio was then averaged over the 1000 runs for each combination of variables. The mean ranged from 0.940 for $C = 16, |\Lambda| = 5, |R| = 16, q = 1, \alpha = 0$, non-uniform profit density to 1.0 for 31 different choices of the parameters, all of which had uniform profit density, and 21 of which had $q = 2$. Note, however, that as the minimum mean was found using a relaxed ILP to find an upper bound for the optimal solution, the actual mean will be slightly higher. The overall mean was 0.994, and the median of the means 0.998. We see here that the actual ratios found in practice are generally significantly better than the theoretical worst-case guarantee.

The histogram in Figure 1 shows the worst mean for our heuristic with $\alpha = \frac{1}{2}$ or more. We do not present a histogram for $\alpha = 0$ or $\frac{1}{4}$ because the data for these runs use a relaxed upper bound, which does not accurately reflect the number of heuristic runs that actually achieved the optimal solution. This histogram demonstrates that our heuristic frequently found an optimal solution and obtained a ratio of less than 0.94 of optimal less than 10% of the time.

V. CONCLUSION

We have proposed a new approximation algorithm and heuristic for the problem of traffic grooming in single-source WDM rings with bifurcating traffic. Our approximation algorithm guarantees solutions that are within $\frac{q}{q+1}$ of optimal where the largest demand of a request is for $\frac{1}{q}$ of the maximum wavelength capacity. Experimental results of a heuristic based on this algorithm suggest that its worst-case performance is even better than the theoretical lower bound.

A number of interesting related problems remain for future work. For example, we have restricted all requests to have

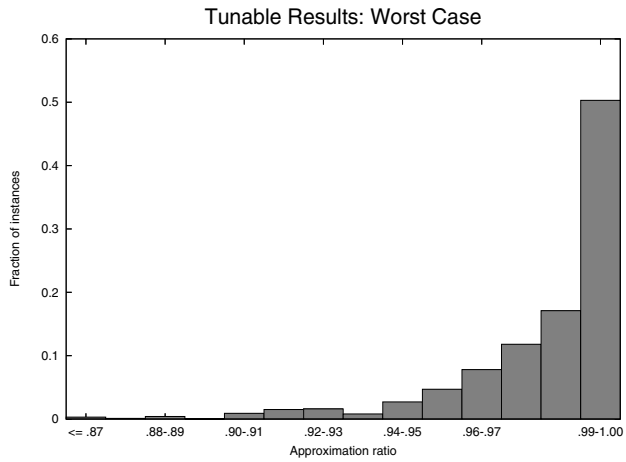

Fig. 1. Histogram of approximation ratios for $n = 16$, $\alpha = \frac{1}{2}$ with constant density, $C = 16$, $q = 1$.

demands less than or equal to the capacity of a wavelength. While this is likely to be applicable in most practical scenarios, extending the model to permit arbitrary demands is an interesting one. In addition, better approximation algorithms or tighter analyses of our algorithms may be possible. Finally, the design and analysis of on-line algorithms for this problem are currently under study.

REFERENCES

[1] O. Gerstel, R. Ramaswami, and G. H. Sasaki, "Cost-effective traffic grooming in WDM rings," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 618–630, 2000.

[2] C. Law and K.-Y. Siu, "Online routing and wavelength assignment in single-hub WDM rings," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, October 2000.

[3] K. Zhu and B. Mukherhee, "A review of traffic grooming in WDM optical networks: Architectures and challenges," *Optical Networks Magazine*, vol. 4, no. 2, March/April 2003.

[4] B. Chen, G. N. Rouskas, and R. Dutta, "Traffic grooming in WDM ring networks to minimize the electronic port cost," *Optical Switching and Networking*, vol. 2, no. 1, pp. 1–18, May 2005.

[5] B. Ramamurthy and A. Ramakrishnan, "Virtual topology reconfiguration of wavelength-routed optical wdmnetworks," in *Proceedings of GLOBECOM 2000*, vol. 2, 2000, pp. 1269–1275.

[6] H. Liu and F. A. Tobagi, "Traffic grooming in WDM SONET UPSR rings with multiple line speeds," in *Proceedings IEEE Infocom 2005*, 2005.

[7] G. Maier, A. Pattavina, S. D. Patre, and M. Martinelli, "Optical network survivability: Protection techniques in the WDM layer," *Photonic Network Communications*, vol. 4, pp. 251–269, 2002.

[8] J. Bermond, D. Coudert, and X. Munoz, "Traffic grooming in unidirectional WDM ring networks: the all-to-all u nitary case," in *7th IFIP Working Conference on Optical Network Design & Modelling*, 2003.

[9] T. Y. Chow and P. J. Lin, "The ring grooming problem," *Networks*, vol. 44, no. 3, 2004.

[10] R. Dutta and G. N. Rouskas, "On optimal traffic grooming in WDM rings," *IEEE Journal on Selected Areas in Communication*, vol. 20, no. 1, January 2002.

[11] Caprara, Kellerer, and Pferschy, "The multiple subset sum problem," *SIJOP: SIAM Journal on Optimization*, vol. 11, 2000. [Online]. Available: http://citeseer.ist.psu.edu/caprara98multiple.html

[12] C. Chekuri and S. Khanna, "A polynomial time approximation scheme for the multiple knapsack problem," *SICOMP*, vol. 35, no. 3, pp. 713–728, 2006.

[13] H. Kellerer, "A polynomial time approximation scheme for the multiple knapsack problem." in *RANDOM-APPROX*, ser. Lecture Notes in Computer Science, D. S. Hochbaum, K. Jansen, J. D. P. Rolim, and A. Sinclair, Eds., vol. 1671. Springer, 1999, pp. 51–62.

[14] O. H. Ibarra and C. E. Kim, "Fast approximation algorithms for the knapsack and sum of subset problems," *J. ACM*, vol. 22, no. 4, pp. 463–468, 1975.