Capacity Optimization based on Traffic Grooming in Transport Networks

Madanagopal Ramachandran
Dept. of CSE
Indian Institute of Technology Madras
Chennai, India
cs14d009@smail.iitm.ac.in

Anix Anbiah
Dept. of CSE
Indian Institute of Technology Madras
Chennai, India
anix@cse.iitm.ac.in

Krishna M. Sivalingam
Dept. of CSE
Indian Institute of Technology Madras
Chennai, India
skrishnam@iitm.ac.in

***Abstract*—The demand for increasing capacity in service provider's networks has led to the development of different transport technologies such as PDH, SDH/SONET, WDM/DWDM, OTN, Ethernet, MPLS-TP etc. which support different types of traffic requirements. Transport networks get augmented due to the surge in capacity requirements. The need for shorter time to provision capacity along with the continuous growth in transport networks typically result in sub-optimal utilization of network resources. Optimization involves consideration of the existing traffic demands for services already provisioned and providing alternate paths and resources based on some global metrics. In multi-layer multi-technology transport networks, this work focuses on optimization of capacity consumed globally for the complete network instead of optimizing capacity in each layer network independently so that it could result in significant benefits in utilization of the existing network. Techniques like traffic grooming across layers and topologies between every pair of nodes by creating tunnels or bearers are used to find the optimum capacity usage. In terms of total capacity used and the number of flows or cross-connections created, the performance of the proposed approach is compared with the provisioning simulated for service requests one-by-one and it is found that the grooming based approach results in lower capacity utilization and number of flows or cross-connections created.**

I. Introduction

Service providers employ different transport technologies like PDH, SDH/SONET, OTN, DWDM, Ethernet, MPLS-TP etc. to support different types of traffic and service requirements. Service providers require faster provisioning of services to satisfy more customer requests in less time. Typically, services are provisioned manually by the service providers by logging into management systems and this leads to sub-optimal capacity utilization. Added to this, the transport network gets augmented with the addition of new network elements and links between them, which typically means that the path earlier used to satisfy service requests may not be optimal after the augmentation.

This leads to the scope for capacity optmization in transport networks. Periodic optimization of the network resources for existing services by computing shorter paths and grooming of traffic would help in getting the best out of the network capacity resources already in place. With the advent of Software Defined Networking (SDN) and centralized control of the network, automation of capacity optimization would be of real benefit to service providers so that it would result in better utilization and scope for accepting new service requests using the existing network infrastructure.

This work addresses the capacity optimization problem in multi-layer multi-technology transport networks by optimizing capacity across layers globally instead of optimizing bandwidth in each layer independently. This would result in significant benefits in utilization of the existing network due to the global level of optimization. Optimization is achieved by traffic grooming low rate signals or capacity into high rate signals or capacity by creation of tunnels or bearers.

The proposed approach uses the path computation algorithm for transport networks proposed in [6] so that the adaptation and topology related requirements are satisfied. For unprotected service requests, the grooming approach tries to create one set of tunnels or bearers and for protected service requests, two separate set of tunnels or bearers (one for the working path and the other for the protection path) are attempted to be created. The performance of the proposed approach is evaluated for the parameters the number of cross-connections created and the tunnel usage ratio. It is found that the benefit of optimization is significant where it resulted in lower number of flows or cross-connections created when compared with the path computation for simulated service provisioning of service requests one-by-one.

II. Background and Related Work

A. Transport Network

A depiction of a portion of a service provider transport network is shown in Fig. 1. A transport network of a typical service provider is deployed such that it uses different transport technologies for different bandwidth and service requirements. It comprises of a national long distance (NLD) network that transports traffic from one city to another. It is typically formed as a mesh network with Automatically Switched Optical Network (ASON) functionality implemented. To cater to the increasing bandwidth demands, optical fibers supporting WDM are deployed in the NLD network. One choice for the NLD network is an OTN mesh network supporting ASON at ODU-k layer which multiplexes multiple OTN signals over WDM. Each city in the country has one or more OTN nodes

978-3-903176-15-7

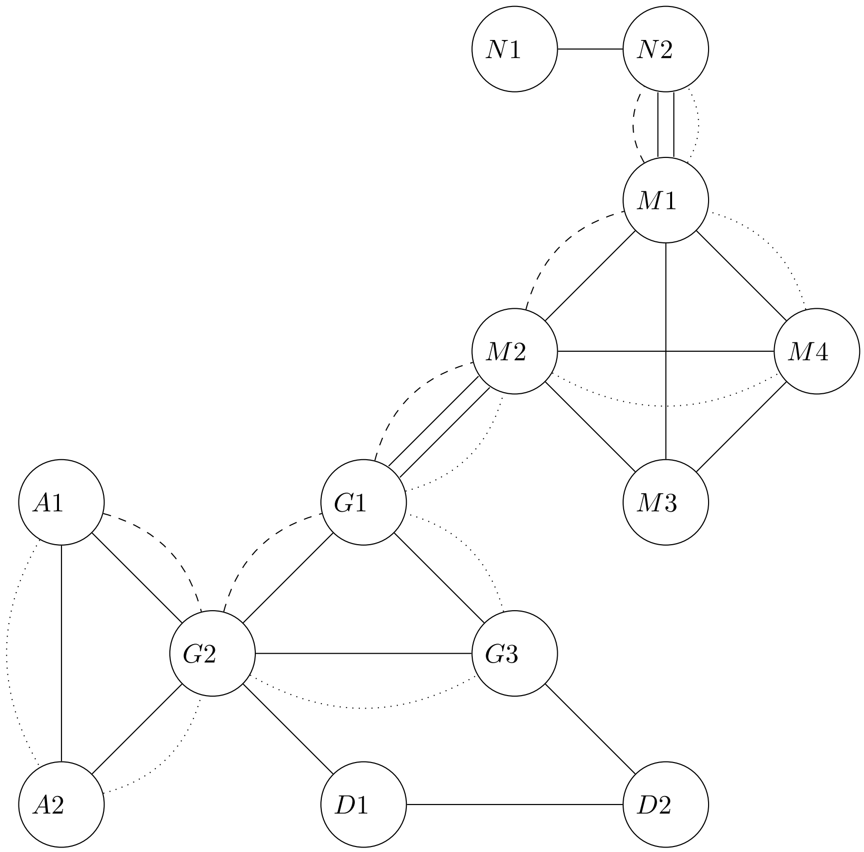

Fig. 1: Transport Network

which aggregates traffic from the city to transport over the NLD network to other cities in the country. Other choices for the NLD network are OTN with OCH ASON (Wavelength Switched Optical Network - WSON) or OTN with Flexgrid which supports ASON at OCH spectrum layer (Spectrum Switched Optical Network - SSON). In the Fig. 1, the nodes *N1*, *N2* and many such nodes in each city or important location (not shown due to lack of space) and edges (typically WDM links) connecting them form the NLD network.

The city or metro network is again a WDM network which transports SDH and Ethernet traffic within the city. The traffic to other cities is handed over to the NLD aggregation node which is part of the NLD network. SDH/SONET is used to carry voice and other fixed rate traffic and Ethernet is used to carry variable rate traffic like IP, IP-MPLS and MPLS-TP etc. In Fig. 1, nodes *M1* to *M4* and edges (typically WDM links) connecting them form the metro core WDM network in one city where the metro core node *M1* is connected to the NLD aggregation node *N2* in that city.

The SDH metro network is typically organized as multiple aggregate rings at STM-64 or STM-16 rate and these rings are connected to the metro core nodes at some common locations. Each aggregate ring has multiple access rings typically at STM-4/STM-1 rates. The Ethernet metro network is also typically organized as multiple core rings at 10 Gbps rate. Each core ring has multiple aggregate or access rings typically at 1 Gbps rate. In both SDH/SONET and Ethernet metro networks, the aggregate and access rings could be subtended at one node or at two nodes (dual homing rings).

In the Fig. 1, the nodes *G1* to *G3* and edges connecting them form one of the aggregate ring in a location where the node *G1* is connected to the metro core node *M2* in that location. The nodes *A1* and *A2* and edges connecting them form an access ring subtended at *G2* and the nodes *D1* and *D2* and edges connecting them form an access ring subtended at *G2* and *G3*.

B. Path Computation in Transport Networks

Path computation in transport networks has to consider the different technologies and the layers supported in each of those technologies. Also, for path computation to be practically used in service provider's transport network, topologies like ring, dual homing, linear etc. have to be considered so that the provisioned path is easy to troubleshoot and maintain. This is achieved by constructing an auxiliary graph structure with support for physical, logical, adaptation edges and special edges to consider topologies [6]. Using this graph structure with initial and dynamic weight assignment for edges, an algorithm for unprotected and protected path computation is proposed in [6].

For example, to compute a protected path from *A1* to *N2* and beyond, the algorithm for path computation proposed in

[6] would result in a working path involving the nodes *A1*, *G2*, *G1*, *M2*, *M1*, *N2* and the protection path involving the nodes *A1*, *A2*, *G2*, *G3*, *G1*, *M2*, *M4*, *M1*, *N2*. This is highlighted using the dashed line for the working path and the dotted line for the protection path in Fig. 1. Even though this looks straightforward for the given sample network, the real transport network would have many topologies and links from a node and the path computation algorithm should consider the correct adaptation between layers and ensure that the same set of topologies are used in the working and protection paths.

C. Traffic Grooming

Traffic grooming is a commonly used method to route multiple low bandwidth traffic into high bandwidth streams which are then routed over a network [5]. In this problem, given a network topology and a set of traffic requests with different bandwidth requirements, the optimal number of high bandwidth traffic streams has to be determined such that the low bandwidth traffic flows can be groomed into them. This problem is well studied in the context of WDM networks under different scenarios and is shown to be NP-Hard [8].

A comprehensive survey of traffic grooming in WDM networks is carried out in [2]. Different techniques for static traffic grooming are studied and they are categorized into non wavelength traffic exchange and wavelength traffic exchange related works. Different techniques for dynamic traffic grooming are also studied.

A method of creating an auxiliary graph model for traffic grooming in WDM mesh networks is proposed in [7]. It proposes creation of multiple nodes and edges to represent access layer, lightpath layer and wavelength layer and a heuristic algorithm which computes paths considering these layers in the auxiliary graph to satisfy a set of connection requests. Different ways of ordering the connection requests are proposed and the different ways of assigning weights to the edges are proposed to optimize various performance parameters. The performance of the proposed strategies are evaluated and compared for representative small and large networks.

The problem of grooming IP/MPLS traffic over a meshed optical network is studied in [1]. It proposes a two stage approach where all the IP/MPLS traffic is routed over the cheapest path called the Forward Synthesis stage and then the underused links are removed and traffic through them is redirected to other links called the Design Tightening stage. The same two stage approach is then repeated in the optical layer. It also incorporates a feedback logic to improve the grooming output.

Although traffic grooming has been applied to IP/MPLS and WDM networks in past research to optimize capacity, the usage of traffic grooming along with path computation for multi-technology multi-layer transport networks using a single auxiliary graph structure is something which has not been attempted before. This work aims to use traffic grooming by creation of tunnels or bearers to groom multiple low rate traffic signals so that the number of cross-connections or flow table entries is reduced and capacity optimization is achieved.

III. Problem Statement

Due to constant increase in the bandwidth demands, the amount of traffic flows from service requests that need to be handled in a transport network is continuously increasing. This results in corresponding increase in the number of cross-connections or flow table entries in a transport network element. Also, the cross-connections or flow table entries may change if the service requests are short-lived. Further to this, the continuous change in the network topology due to re-configuration or growth of the network would result in constant updates in the path of the service requests. This may not lead to the optimal utilization of capacity in the network since the updates happen manually without the global view of the network.

A typical solution to overcome this problem is traffic grooming [5] where multiple low bandwidth traffic streams (fine-granular flows) are groomed into high bandwidth traffic streams (bearers or tunnels) which are then routed over the transport network so that the number of cross-connections or flow table entries are minimized. The bearers or tunnels are pre-established or dynamically created when the bandwidth demands increase and the low bandwidth streams use the bearers or tunnels to reach from one part of the network to the other part of the network. But the traffic grooming problem, which tries to establish the high bandwidth traffic streams to satisfy a set of low bandwidth flow requests, is shown to be NP-Hard [8]. In multi-technology multi-layer transport networks, where different layers of adaptation are used to carry traffic in different parts of the network, capacity optimization is even more difficult.

In this work, the focus is on usage of traffic grooming to achieve capacity optimization globally in multi-technology multi-layer transport networks. The problem can be formally stated as follows:

Given:

1) A graph of the network $G = (V, E, C)$ where V is the set of nodes, E is the set of edges and C is the capacity values for the edges,
2) The capacity of a tunnel c_{tunnel}, and,
3) The set of demands $D = \{(a, z, c_f), a, z \in V\}$ where c_f is the size of the demand

Find:

1) Set of tunnels $T = \{(a, z, i), a, z \in V\}$ where i is the id of the tunnel and,
2) A mapping of the flows to their corresponding tunnels $g() : D \rightarrow T$

Such that:

1) $\forall t \in T, \sum_{f} c_f^t \leq c_{tunnel}$ where c_f^t is capacity of a flow using tunnel t
2) $\forall e \in E, n_t^e \times c_{tunnel} \leq c_e$, where
    a) n_t^e is the number of tunnels using e
    b) c_e is the capacity of edge e

The objectives are the following:

1) Minimize total number of cross-connections or flow entries
2) Maximize the bandwidth usage within the tunnels

The path computation problem for single service request in multi-layer networks is shown to be NP-Complete in [3] and [4]. Hence, heuristic path computation algorithm is proposed using a auxiliary graph structure in [6]. The optimization using traffic grooming in transport networks is shown to be NP-Hard in [8]. These factors mean that the capacity optimization problem in transport networks have to be addressed using heuristic approaches.

A. *Traffic Grooming Approach*

A heuristic approach is proposed for grooming of traffic flows from service requests to achieve capacity optimization in transport networks. The following design principles were used while coming up with the approach:

1) Aggregate end-to-end flows into tunnels first. That is, longer tunnels (in terms of number of hops taken through the network) are decided upon first.
2) As much as possible, do not leave bandwidth unused in the longer tunnels. The longer tunnels take up more bandwidth in the network and should be filled to capacity as much as possible. Thus, a threshold of minimum usage is set based on the cost of the path between the ends of the tunnel. The threshold is determined as specified as given in the following equation:

$$\mathcal{T} = 1 - 1/Cost_{path} \tag{1}$$

   where $\mathcal{T}$ is the threshold and $Cost_{path})$ is the cost of the path between the tunnel ends.
3) Shorter tunnels can be created more liberally, with the threshold for minimum usage being a lower value than for longer tunnels.
4) More tunnels (that is, more aggregation) are required at the core of the network. This is due to the fact that most end-to-end flows tend to flow through the core of the network. Also, flows that spillover from the longer tunnels congregate with other similarly spilled over flows and are accommodated in shorter tunnels.

The above approach is applied to every pair of nodes between which service requests are configured in the transport network. For each pair of such nodes, an unprotected or protected path is computed based on the protection requirement in the service request. The protected path is broken into working and protection paths. Each path is evaluated for traffic grooming by checking for tunnel or bearer creation between every combination of nodes in the path. This is done by first considering pair of nodes in the path which are farthest followed by less farther apart until adjacent nodes are reached. While considering a pair of nodes for tunnel creation as part of traffic grooming, a check whether the nodes are supporting same layer and technology, is done so that the tunnels are properly configured.

For the working path from *A1* to *N2* in Fig. 1, possibility of creating a direct tunnel from *A1* to *N2* is checked by considering the path cost using which a threshold is set. The higher the path cost, higher the threshold so that longer tunnels are created only when the request capacity across many service requests is significantly more. When the request capacity from *A1* and *N2* across many service requests exceeds tunnel capacity, then some number of full capacity tunnels are created computed by dividing request capacity by tunnel capacity. For the remaining capacity, if the ratio of request capacity and tunnel capacity exceeds the threshold, then an extra tunnel is created. If a tunnel is created, then all the service requests which contributed to the tunnel creation are removed from the intermediate nodes of the tunnel so that they are not considered for tunnel creation between the intermediate nodes. The above process is repeated for the next pair of nodes *A1* and *M1*, *G2* and *N2* whose path cost is lesser than the cost from *A1* and *N2*. Similar process is repeated for the protection path for the same service request. After a pair of nodes in a service request is processed, then the next service request is processed. This is explained as a algorithm in the next section.

IV. Capacity Optimization Algorithm

The algorithm for capacity optimization is presented below.

Algorithm 1 capOpt(graph, flowList)

1: Route all the flows in $flowList$ using the physically shortest path through the network
2: Calculate the pair-wise count of flows for all pairs of nodes in the network i.e., add the flows that pass through each pair of nodes
3: Process the pairs of nodes in the decreasing order of path cost
4: Create as many tunnels as are required between the two nodes to carry the flow while making sure that tunnel usage threshold is exceeded for each tunnel. Therefore, flows that do not fit into the created tunnels are allowed to spill over
5: For each flow that traverses a tunnel, remove the flow (or its corresponding cross-connection or forwarding table entry) from the nodes that are intermediate between the source and destination of the tunnel
6: Do not process node pairs that are adjacent to each other ($hopcount < 2$).
7: Once all node pairs have been processed, compute the overall tunnel usage ratio as a weighted average of the usage ratio of all tunnels (the weight being the hop count).
8: Similarly, calculate the number of cross-connections or forwarding table entries required for both flows and tunnels to carry all the flows end-to-end.

The algorithm for capacity optimization (Algorithm 1) takes as input the *flowList* which needs to be optimized. For each flow in that list, the shortest path between the source and destination of the flow is computed using the *findPath* algorithm proposed in [6]. In case the flow is not protected, the *findUnprotectedPath* procedure described in [6] would be used, else, if the flow is protected, then link-disjoint path pair (LPP) has to be found for which *findLPP* procedure described in [6] would be used internally. Since the shortest path is used for processing, the value N is set as 1 while calling *findPath* algorithm proposed in [6]. For each pair of nodes in the graph, the count of flows between them is then found. Each pair of nodes is processed starting with the pair having the greatest path cost. If path costs are same, then the pair with the greater number of flows between them is processed first. For the current pair of nodes, many tunnels are created between the nodes such that either the tunnel capacity is full or the tunnel usage exceeds the threshold as specified in equation 1. Since the tunnels would now carry the flows, flows are assigned to the created tunnels and the flow related forwarding table entry or cross-connection is removed from all the nodes part of the tunnel. Tunnels are not created for pair of nodes that are one hop away. By using this method, one flow could get carried over multiple tunnels between different pair of nodes along the route of the flow. After all the node pairs are processed, the total tunnel usage ratio is computed for all tunnels. For each tunnel, the tunnel usage ratio is computed as the ratio of the capacity of the tunnel used and the total capacity of the tunnel. Also, the total number of cross-connections or forwarding tables entries required to carry all the flows in the *flowList* is computed.

Computational Complexity: The computational complexity of Algorithm 1 in the average case when all the flows are unprotected is $\Theta(F((E+V)\log V) + V^2(\log(V^2)+V))$ and when all the flows are protected is $\Theta(FV((E+V\log V)+((E+V)\log V)) + V^2(\log(V^2)+V))$ where F is the number of flows considered for optimization, V and E are the number of nodes and edges in the auxiliary graph. The complexity of Step 1, when all the flows are unprotected uses $findUnprotectedPath$ procedure described in [6], is $\Theta(F((E+V)\log V))$ and when all the flows are protected uses $findLPP$ procedure described in [6], is $\Theta(FV((E+V\log V)+((E+V)\log V)))$. After all the flows are routed, the number of node pairs could be V^2 in case full mesh of tunnels are required. The complexity of sorting the node pairs based on path cost in Step 3 is $\Theta(V^2\log(V^2))$. The complexity of Step 4 and Step 5 is V since the path could contain atmost V nodes. Since Step 4 and Step 5 are executed for each node pair, the complexity of Steps 3 to 5 is $\Theta(V^2(\log(V^2)+V))$ and therefore the complexity of the algorithm which is the sum of the complexities of Step 1 to 5 is $\Theta(F((E+V)\log V)+V^2(\log(V^2)+V))$ for unprotected flows and $\Theta(FV((E+V\log V)+((E+V)\log V))+V^2(\log(V^2)+V))$ for protected flows.

V. Performance Results

The performance of the described algorithm using the auxiliary graph construction for the network described in [6] is evaluated and the results obtained are provided in this section. The same network described in the performance evaluation section in [6] is used for evaluation, which resulted in 2955 network elements, which in turn resulted in 10455 nodes (due to creation of nodes for each technology and layer combination within every network element), 480 hub nodes for dual homing topologies and 10380 adaptation edges, 5393 physical links and 3540 special edges for a total of 19313 edges in the complete auxiliary graph.

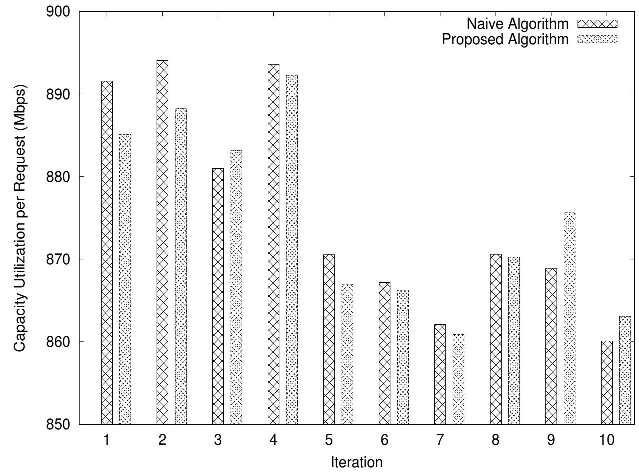

Fig. 2: Capacity Utilization per Service Request

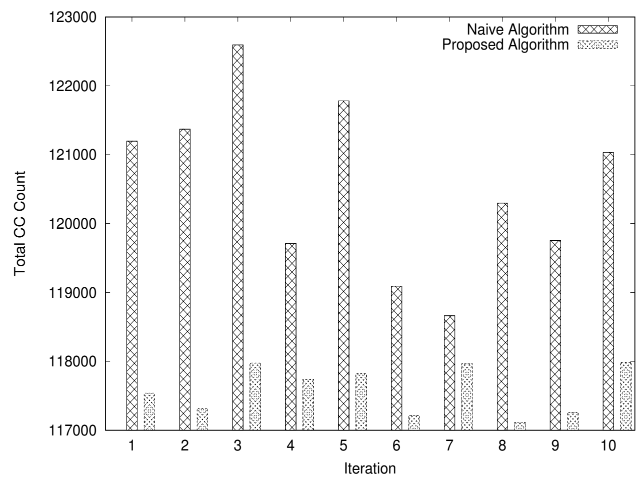

Fig. 3: Total No. of CCs or Flow Table Entries Created

To evaluate the performance for finding LPP paths, 5000 service requests between randomly selected source and destination nodes are generated. The service requests are generated at SDH layer with rate VC12, VC3, VC4 and Ethernet layer with random capacity between 1 to 200 Mbps. The process is repeated for 10 iterations where the dynamic weight approach Piece-wise Linear Function based on Utilization (PLF) is used and the values α, β, γ and η are all set as 0.5 since this combination gave better results as part of the evaluation in [6]. The tests were performed on a machine with 4-core Intel i5 processor (3 GHz) and 64 GB RAM.

The first performance evaluation that is carried out is to check whether there is reduction in capacity utilization and

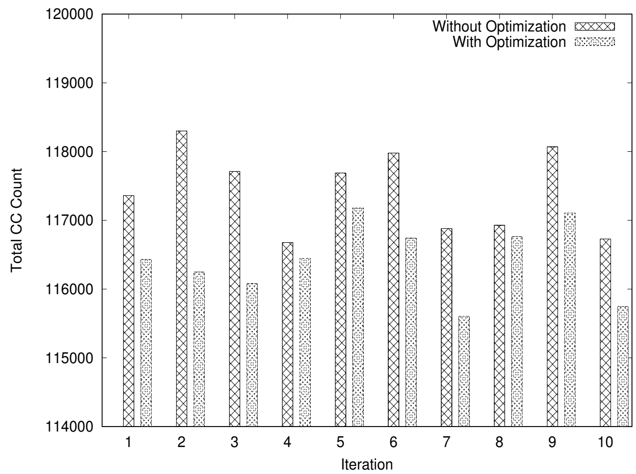

Fig. 4: Total No. of CCs or Flow Table Entries Created

number of cross connections or flow table entries as a result of using the proposed algorithm. To check the reduction in capacity utilization, a naive path computation algorithm is designed to satisfy LPP requests and compared with the proposed algorithm. The naive path computation algorithm uses the Dijkstra's shortest path algorithm to find the working path between the given source and destination in the auxiliary graph structure. While finding the protection path, all the physical and logical links part of the working path are excluded (adaptation edges are not excluded) and the Dijkstra's shortest path algorithm is again used to find a protection path.

To evaluate the performance in terms of capacity utilized in the whole simulated network, all the physical links part of the metro network are considered for the utilized capacity since the service requests terminate in the metro network and the logical links created over the National Long Distance (NLD) OTN-WDM network is used to carry the service requests from metro network. To evaluate the performance for finding LPP paths, same set of 5000 requests are generated for 10 iterations and the total capacity utilized per service request is compared for the naive algorithm with the proposed algorithm.

The results obtained for the total capacity utilized per service request in 10 iterations of the simulation using the naive path computation algorithm (legend bar - Naive Algorithm) and the proposed algorithm (legend bar - Proposed Algorithm) is shown in Fig. 2. From the results in the graph, it can be observed that the proposed algorithm results in lower capacity utilization compared to the naive path computation algorithm. The improvement is expected to be signinficant in real networks since constant network changes and manual provisioning would have resulted in more capacity utilization. The higher capacity utilization in real networks would be reduced using the proposed capacity optimization algorithm.

The number of cross connections or flow table entries created when using the naive path computation algorithm (legend bar - Naive Algorithm) and the proposed algorithm (legend bar - Proposed Algorithm) is shown in Fig. 3. From the results in the graph, it can be observed that the proposed algorithm results in lower number of cross connections or flow table entries compared to the naive path computation.

The second performance evaluation that is carried out is to check whether there is reduction in the number of cross connections or flow table entries as a result of using the proposed algorithm when compared with using the $findPath$ algorithm proposed in [6] for same set of service requests one-by-one. This evaluation is done for 10 iterations.

The results obtained for the number of cross connections or flow table entries created in 10 iterations of the simulation using the approach proposed in this work (legend bar - With Optimization) when compared with the number created when using $findPath$ algorithm proposed in [6] for all service requests one-by-one (legend bar - Without Optimization) is shown in Fig. 4. From the results in the graph, it can be observed that the proposed approach resulted in lower number of cross connections or flow table entries. This improvement is expected to be significant in real networks since the manual provisioning would have resulted in larger number of cross connections or flow table entries. For the 5000 service requests simulated, it has been observed that the average tunnel usage ratio is around 25% to 30% in all the 10 iterations.

VI. Conclusions

In this paper, capacity optimization in transport network, using multiple technologies and multiple layers within them, is addressed using traffic grooming by creation of tunnels or bearers. Tunnels or bearers are created whenever the capacity provisioned between a pair of nodes is above a threshold using the path computation algorithm of multi-technology multi-layer transport networks. The main reason for using path computation across layers and technologies is that it would result in cross-layer optimization which would not be possible if each layer or technology optimization is done independently.

The performance of the proposed approach is evaluated for the capacity utilization and the number of cross-connections or flow table entries created by comparing the proposed grooming based approach with using a naive path computation algorithm where the proposed approach results in lower capacity utilization and the number of cross-connections or flow table entries created. When the proposed approach is compared with using the path computation algorithm for satisfying the service requests one-by-one, it is found that the number of cross-connections or flow table entries created is lower for the proposed grooming based approach.

It is expected that in real service provider transport networks, capacity benefits could be realized using the grooming based approach proposed in this work since the provisioning which is typically done manually would not have considered the network changes that took place over a period of time.

Acknowledgment

This work was supported by an IRDA grant from IIT Madras (2017-2020) and by a DST grant (EMR/2016/003016) from Government of India (2017-2020).

REFERENCES

[1] S. De Maesschalck, M. Pickavet, D. Colle, and P. Demeester. Multi-layer traffic grooming in networks with an IP/MPLS layer on top of a meshed optical layer. In *Proc. IEEE GLOBECOM*, volume 5, pages 2750–2754, 2003.

[2] R. Dutta and G. N. Rouskas. Traffic grooming in WDM networks: Past and future. *Network, IEEE*, 16(6):46–56, 2002.

[3] F. Kuipers and F. Dijkstra. Path selection in multi-layer networks. *Computer Communications*, 32(1):78–85, 2009.

[4] M. L. Lamali, N. Fergani, J. Coheny, and H. Pouyllau. Path computation in multi-layer networks: Complexity and algorithms. In *Proc. IEEE INFOCOM*, pages 1–9, Apr. 2016.

[5] E. Modiano. Traffic grooming in WDM networks. *Communications Magazine, IEEE*, 39(7):124–129, 2001.

[6] M. Ramachandran and K. M. Sivalingam. Path Computation for Provisioning in Multi-Technology Multi-Layer Transport Networks, 2019; arXiv:1901.01531.

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

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