A Study on Cross-Layer Multi-Constraint Path Computation for IP-over-Optical Networks

Xi Yang[1], Tom Lehman[1], Kenichi Ogaki[2] and Tomohiro Otani[2]
[1]USC/ISI East, 3811 N Faifax Drive, Suite 200, Arlington, VA 22203, U.S.A.; Email: {xyang, tlehman}@isi.edu
[2]KDDI R&D Laboratories, Inc., 2-1-15 Ohara Fujimino Saitama, 356-8502, Japan; Email: {ogaki, otani}@kddilabs.jp

Abstract — **A powerful path computation element is a must for the next-generation IP-over-Optical networks to support on-demand service provisioning crossing layers. The main purpose of path computation in multi-layer networks (MLN) is to enable traffic engineering (TE) under a multitude of constraints. A common objective of such traffic engineering is to minimize resource utilization while satisfying all the applicable constraints. In this study, we investigate a variety of MLN path computation methods based on real-world control plane implementations and network settings. We present some techniques for handling the specific TE constraints that concern the IP-over-Optical networks. Experimental results are presented for performance evaluation and comparison of the investigated multi-constraint path computation methods and techniques.**

I. INTRODUCTION

Today's IP-over-Optical networking technologies are clearly showing their potential to drive down the price and provide more efficiency and flexibility in traffic aggregation, routing and end-to-end service delivery. From the users' perspective, there is a requirement for network services beyond what are typically available on best-effort infrastructures. This includes truly dynamic provisioning with fine control on bandwidth granularities and service times. This is particularly important for those high-performance applications and high-end users who can now have the high-capacity, deterministic and dynamic network services at affordable prices.

A typical IP-over-Optical network could be an IP-over-WDM network that transports aggregated IP packets over lambdas. Traditionally, the IP routers are connected with static routes bound into optical channels. Such a static two-layer infrastructure is only suitable to deliver best-effort traffic. Even if Multi-Protocol Label Switching (MPLS) tunnels can be formed to accommodate certain Quality of Service (QoS) requirements at the IP layer, the optical layer remains static. Such an infrastructure is inflexible when having to adapt to fast traffic changes, because dynamic allocation of resources across both IP and optical layers could not be efficiently done without a control plane that unifies both layers. Further complicating the scenario, a heterogeneous network environment, with fiber switching, SONET and/or carrier-class Ethernet technologies at optical layer, will make efficient on-demand service provisioning impossible.

A unified control plane is the most promising solution capable of efficient dynamic provisioning across both IP and optical layers. Such a control plane, typically using Generalized MPLS (GMPLS), is aware of the comprehensive resource information from a Multi-Layer Network (MLN) that utilizes multiple technologies and/or multiple types of vendor equipment. To accommodate a service request with the rich information, the very important but complex function for the control plane to perform is to compute a path. The entity for this functionality is called Path Computation Element (PCE).

The IETF PCE working group has standardized the PCE architecture, message formatting, client-server and inter-PCE communication protocols [1]. However, the question of how to compute the actual paths in MLN networks remains open. A number of classic works have been done in finding paths in single-layer networks [2][3]. There has been significant work addressing multi-constraint QoS path computation in the IP layer [4][5]. Some recent work has also covered multi-layer multi-constraint path computation [6][7]. However, there have been few evaluations of the multi-constraint path computation within the context of realistic multi-layer networks using unified control plane technologies. This is due in part to there being no control plane software available for open research. In lack of real-world context, none of the previous work addressed the full set of TE constraints for a typical IP-over-Optical network, nor did they develop the specific techniques to tackle those complex constraints from a pragmatic perspective.

In this study, we investigate a variety of MLN path computation methods that are integrated into an open source GMPLS control plane software developed by the DRAGON project [8]. The same control plane software has been deployed in the Internet2 Dynamic Circuit Network (DCN) [9] and other real-world networks. This study addresses the two-layer IP-over-Optical networks, which are exemplified by IP-over-WDM networks in experiments. Due to limited scope of the paper, this study only considers single domain networks and is focused on path computation algorithms and techniques. The path control schemes that dictate interactions between path computation, routing and signaling are not considered. Also not considered is optimization of a sequence of dynamic demands. In other words, we only optimize single request over the network topology at a given time instant even when the network carries dynamic traffic.

The remaining of the paper is organized as follows. Section II describes the network model. Section III presents the investigated cross-layer multi-constraint path computation solutions in details. Section IV presents our experimental results for evaluation and comparison of these path computation methods. Section V concludes this study.

II. NETWORK MODEL

A. *IP-over-Optical Network*

In IP-over-Optical networks, the IP layer is packet switching capable (PSC). The optical layer is either lambda switching capable (LSC) or SONET/SDH TDM. In this study, we consider all the IP layer services as of circuit/tunnel fashion, which means a guaranteed amount of bandwidth has to be reserved. The IP packets are aggregated into optical channels in the form of IP/MPLS tunnels. Each optical channel has to be allocated in the form of lambdas or TDM time slots as a whole. The lambdas in WDM networks are either fixed or convertible while the timeslots in TDM are exchangeable.

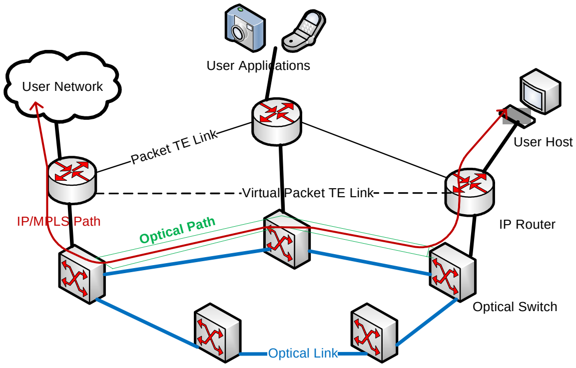

Figure 1: Illustration of IP-over-Optical network.

An IP-over-Optical network is illustrated in Figure 1. The network is physically connected by optical links. A PSC TE link is established between two IP routers on top of existing static optical connections. When a dynamic optical path is created, a corresponding PSC TE link is also established (as depicted in fine dashed line). A user end-to-end IP/MPLS path is carried in such optical path(s) that connects the source and destination IP routers.

B. *Traffic Engineering Constraints*

The various TE constraints in the IP-over-Optical networks generally fall into the following categories:

- **Prunable constraints**: including bandwidth, switching type, encoding type, service times and policy-induced exclusion etc.
- **Additive constraints**: including path length, latency and optical-layer linear impairments (e.g. dispersion) etc.
- **Non-additive constraints**: including optical wavelength continuity, Ethernet VLAN continuity and optical-layer non-linear impairments (e.g. cross-talk) etc.
- **Adaptation constraints**: The adaptation constraints give conditions for traffic to cross layers. In addition, they can modify some of the above constraints into relaxed or more stringent forms. For example, a constraint may relax the wavelength continuity constraints by allowing for wavelength conversion. In a contrasted example, when IP traffic is mapped into TDM layer, the adaptation constraints may narrow down the choices for SONET signals. For example, 120 Mbps IP/MPLS tunnel has to be mapped into STS-3c, which is around 150 Mbps.

The main challenges in tackling these constraints are two-fold. They firstly lie in the computational complexity in handling a multitude of constraints, especially those non-additive and adaptation constraints. Secondly each layer has its characteristic constraints. For example, bandwidth in IP layer can be allocated with arbitrary continuous values as long as they do not exceed the maximum available capacity while bandwidth in optical layer has to be allocated in fixed values. Putting these layer-specific constraints together, design of path computation algorithms has to be further complicated.

C. *Path Computation Model*

The main purpose of path computation in an MLN is to enable traffic engineering under the various constraints. The objective of traffic engineering is to optimize resource utilization while satisfying all the applicable constraints. The typical optimization criteria for resource utilization include:

- path length in number of hops or actual mileage.
- administrative route cost assigned by operators.
- cost of physical resources such as optical transponders.

The investigated path computation solutions in this study are based on the following context and assumptions. Firstly, we address requests for end-to-end deterministic services. A path traverses the IP layer or both IP and optical layers between a pair of source and destination IP routers. The sequence of traversed TE links and the amount of reserved bandwidth and/or optical channel on each link has to be determined as the result of path computation. Secondly, we assume that routers and switches in the network are working in peer mode, which means the PCE can obtain dynamic link states for the entire two-layer network topology. It has all the information needed to handle any of the applicable TE constraints. Thirdly, we assume that creation of an optical-layer path will trigger a PSC TE link advertisement in the IP layer.

III. CROSS-LAYER PATH COMPUTATION SOLUTIONS

A common previous solution for cross-layer path computation was to handle upper and lower layers separately, which we call a *base-line solution*. Using the path computation model described in Section II a PCE has full view of both IP and optical layer topology with all traffic engineering information and can therefore support new solutions that optimize resource allocation jointly for both layers. Such joint optimization solutions generally have two approaches to the path computation algorithm. The first is to search the path on the raw topology graph that simply consists of the actual nodes and links with traffic engineering information as link states. All the constraints have to be judged on-the-fly by a powerful search process, which we call *constrained search*. The second is to transform the raw topology graph into a logical form which is sophisticated enough to represent some or all the constraints in the graph construction. This is called a *graph transformation* technique. Thus a simplified path selection algorithm, say a modified Shortest Path First (SPF), can find an optimal path with given resource utilization objectives. Other simplified path selection algorithms based on *heuristic search* techniques can also be used to trade optimality for reduce computation complexity.

A. Constrained Search

Constrained search can be based on any existing SPF search algorithm. One good candidate is Breadth First Search (BFS). BFS has some good properties such as choosing a path with least number of hops when multiple paths have same cost and allowing for negative link metric which is useful for computing diverse paths. Our Constrained BFS (C-BFS) algorithm checks all applicable constraints for specific layers and applies cross-layer adaptation constraints when proceeding from one layer to another. To address these constraints, the following extensions have to be added. Their implications on computation complexity are also discussed.

- *Handling prunable constraints and additive constraints*: These constraints add constant computation time to each search hop but do not introduce extra search branches.
- *Handling cross-layer adaptation constraints*: Combinations of inter-layer adaptations introduce O(X^2) additional search branches to each layer-crossing border node, where X is the number of layers.
- *Handling VLAN or wavelength continuity constraints*: We use the common vector technique to handle these non-additive constraints. A computation time of $O(W)$ complexity is added to each search hop, where W is the number of Ethernet VLAN tags or wavelengths.
- *Loop avoidance logic*: C-BFS may re-enter a node multiple times due to complicated constraints, which may cause looping situations that are not present in a basic BFS. A computation time of $O(D)$ is added to each hop for loop avoidance, where D is the network diameter.
- *Parallel link handling logic*: There could be situations that some of the parallel links can satisfy certain constraints while others cannot. All the parallel links have to be searched. Each parallel link adds one extra search branch to the BFS search.
- *Link and node reentry:* Unlike a basic BFS that only visit each node and link once, C-BFS has to reenter some nodes and links multiple times. One scenario is that C-BFS reenters a node from other unvisited inbound links to find feasible paths under certain constraints. C-BFS also re-enters visited links to search for feasible paths or minimize optimization objectives. The addition in reentry search branches varies by topology and TE parameters.
- *Preserving and restoring scenes for active search process*: With multiple interim parameters are generated in the search, preserving interim search states upon branching and restoring them upon reentry is important for generating a globally optimal path. Each search hop needs a constant number of stack operations for restoring and preserving the search scene at the head node.
- *Other technology-specific constraints*: Some technology/layer-specific constraints also add to the complexity. For example, traffic adaptation into the TDM and LSC layers needs to round the bandwidth to multiple of minimum reservable bandwidth at these layers. In another example, a special constraint may be applied to the LSC layer to guarantee that a wavelength is exclusively allocated to one of the outgoing links on a WDM switch so that two paths will not collide on the same wavelength.

From the above description, we can understand that the cross-layer C-BFS has much higher complexity than the basic BFS algorithm. This is due to the nature of cross-layer adaptation and non-additive constraints. Compared to exhaustive search, however, complexity of constrained search has been considerably reduced by trimming out infeasible branches by apply constraints during the search process.

B. Graph Tranformation Techniques

The constrained search solution is complex because TE constraints have to be checked in every search hop. In addition, all the constraints are double checked with every link reentry and some constraints are also rechecked for node re-entry. A topology transformation solution can reduce part of the complexity due to search reentry. A topology transformation differs from the constrained search solution in that it does not conduct path computation on the network graph of the original topology. Instead, it first transforms the network graph into a new form that can take some constraints into the graph construction. Here we discuss two specific techniques that concern the IP-over-Optical networks.

1) Label-Layer Graph

Both IP and optical layers may need to handle non-additive constraints such as wavelength continuity. Such constraints can be captured by label-layer graph transformation, a technique we developed to handle generic data channel continuity constraints. Each label (VLAN tag or wavelength) is represented as a graph-layer. The network graph is split into a number of label layers. Searching on the label-layer graph does not need to consider the VLAN tag or wavelength constraints, because the resulting path from a simple SPF has already included a chosen VLAN tag or wavelength.

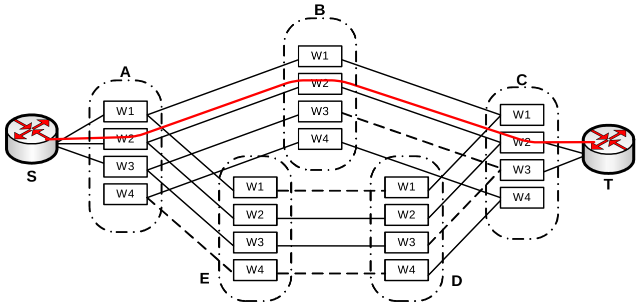

Figure 2: Illustration of label-layer graph transformation.

Figure 2 illustrates label-layer graph transformation for wavelength continuity constraints in an IP-over-WDM network. The original network topology is the same as that depicted in Figure 1. Each bidirectional optical link carries four wavelengths, labeled w_1, w_2, w_3, w_4. Solid and dashes lines represent the available and allocated wavelengths respectively. In the transformed label-layer graph, an IP router is connected to some or all the label layers depending on transporters available on that node. A path can traverse any label layer as long as it is available on the links. For example an optical path from node A to C can use wavelength w_2.

2) Channel Graph

Channel graph is another graph transformation technique that captures the adaptation constraints. This technique was first proposed in [7]. A channel graph is the dual of the network graph. It translates each link triplet *<head, tail, switching_capability>* into a node and add an edge between two constructed nodes *<v1, v2, swcap1>* and *<v2, v3, swcap2>* if the switching capability swcap1 on link *<v1, v2>* can be adapted to switching capability swcap2 on link *<v2, v3>*. Performing a simple SPF on the channel graph can result in a path which itself represents the needed cross-layer adaptations. VLAN translation and wavelength conversion types of adaptation can be similarly handled.

Figure 3 illustrates the channel graph that describes cross-layer adaptation between IP (PSC) and WDM (LSC) links whose original topology is shown in Figure 2. To simplify the drawing, we only show two wavelengths on each link. Each vertex in the graph represents one or a portion of a unidirectional link. For example (V_S, V_A, PSC) represents the PSC link from IP router *S* to WDM switch *A*. $(V_A, V_S, LSC\text{-}W_1)$ represents wavelength W_1 on the LSC link from WDM switch A to IP router S. Two types of adaptation constraints are captured in this graph transformation. Shown on the upper left is cross-layer adaptation between IP (PSC) and WDM (LSC) layers. Shown in the bottom is wavelength conversion between wavelengths W_1 and W_2 at nodes *D* and *E*.

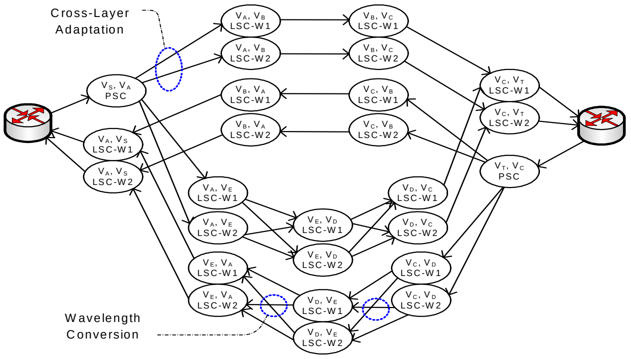

Figure 3: Illustration of channel graph transformation.

On the channel graph, cost on an edge is assigned to be the average of the two original links it bridges. For example, cost of the edge $<(V_A, V_B, LSC\text{-}W_1), (V_B, V_C, LSC\text{-}W_1)>$ will be ½ (cost$< V_A, V_B >$ + cost$< V_B, V_C >$). A sink edge at source or destination have cost equal to half of the link it connects to. By so doing a path selected in the channel graph has the exactly same cost of its dual path in the original graph. Then an optimal path in the channel graph can be translated back to an optimal path in the original graph.

As illustrated in Figure 3, the complex layer-specific wavelength (or VLAN) continuity constraints and cross-layer adaptation constraints can be embedded in graph construction and thus provide an alternative to the constrained search solution for IP-over-Optical networks.

C. Heuristic Search

Compared to the constrained search solution like C-BFS, complexity of a solution using the above graph transformation techniques is not necessarily lower. With some constraints embedded in the topology construction, a path computation process does not need to recheck these constraints during search reentry situations, which does reduce the complexity. On the other hand, topology transformation itself consumes time and the resultant topology graph is often bigger than the original network graph, which in turn increase complexity. To cope with the complexity, we can use heuristic search that has reduced search space.

Enumerating the numerous heuristic path computation algorithms is beyond the scope of this study. A reasonable implementation choice for us is the commonly used K-Shortest Path (KSP) search. KSP search is a typical technique to reduce the path computation complexity. The basic idea is to run a KSP algorithm on a topology by ignoring complex TE constraints. With K paths are returned, we can verify if there is a feasible path among them by applying the TE constraints along the paths. Since K is a constant, time complexity of KSP is polynomial and constraints verification on a fixed set of paths is also polynomial. Therefore this solution can scale to very large networks. There are "bad" cases in which no feasible path is among the K shortest paths or none of paths is feasible even if such a path exists. To reduce the chance of hitting bad cases, K needs to be reasonably large although this is no guarantee for success or optimization. A heuristic search technique such as KSP can be used as an addition to both the constrained search and graph transformation solutions when we are seeking faster and predictable running times with tolerance of some sub-optimal results.

We have implemented some enhancements to the KSP heuristic for IP-over-Optical networks. Firstly our heuristic uses the IP layer whenever possible because when an IP layer path can be found, using unnecessary optical channels poses higher cost and increases contention. The trick is to assign smaller TE metric to the IP links as compared to that of optical links. This will force path computation to try the IP layer first. However, manipulating link cost metrics like that introduces another problem. Even with a fairly large K, the KSP heuristic may only return K-shortest IP-layer paths. In some scenarios, none of the K IP-layer paths would be feasible while those feasible cross-layer IP-Optical paths are not even tried. To solve this problem, our second trick is to prune the IP layer links without sufficient bandwidth before running the KSP heuristic. This will eliminate most of unqualified IP-layer paths and give more tries to cross-layer IP-Optical paths.

IV. PERFORMANCE EVALUATION

A. Evaluation and Experiment Design

In this section we evaluate performance of the following cross-layer multi-constraint path computation solutions for IP-over-Optical networks: i) *base line solution*; ii) *C-BFS constrained search solution*; iii) *KSP heuristic search solution*; and iv) *graph transformation based KSP heuristic search solution*. The base line solution is described as follows. It firstly tries to select a shortest path in the IP layer. When failed, an optical path is requested between the pair of optical switches that are nearest to the source and destination IP routers, and an IP/MPLS path is tunneled across the justcreated optical path. This is a typical design in networks that use lower-layer User Network Interface (UNI) to provision upper-layer services. While the base line is a separate-layer solution, all other investigated solutions compute paths with both layers jointly. In graph transformation, we have only implemented the channel graph transformation technique which was described in Section III. The reason we evaluate the graph transformation technique in combined with the KSP heuristic search technique is that a transformed topology is usually intended for use of a simplified/heuristic search procedure in practice.

We implemented our investigated path computation solutions in DRGAON Network Aware Resource Broker (NARB). DRGON NARB is our open source software module that functions partly as PCE in the DRAGON control plane. Path computation solutions such as C-BFS have been used by NARB for real network deployments for years. We patched the code with the graph transformation techniques to run the complete experiments. Our experiment network is a simulate environment whose optical layer consists of 22 WDM optical switches with 4-wavelength bidirectional links interconnected in a topology that mimics the Internet2 DCN. We assume none of switches is capable of wavelength conversion. Each wavelength has a 10 Gb/s capacity. One of the four wavelengths is used for static IP layer connections while the other three can be dynamic allocated. In the IP layer there are 22 core IP routers each connected to an optical switch with two to four 10 Gb/s interfaces. As reasoned in Section I, we are only concerned with optimizing every path computation request for a given topology whose TE link states depend on network traffic load at a given time instant. We do not use a dynamic provisioning heuristic to offset optimality of a current path in order to better accommodate future paths. For a given topology, we run path computation for individual 1, 2 and 5 Gb/s IP/MPLS requests between every pair of source and destination IP routers. Although the requests are computed on an individual basis, we can still simulate a dynamic environment by repeating the same path computation when the network is averagely 0%, 20%, 40% and 60% loaded. To fill the network with the loaded traffic, we simulate the provisioning activities for a random set of requests until the desired network load is achieved. From this point, we start running the actual path computation experiments on the partially loaded network topology.

In the experiments, we evaluate the path computation solutions using the following performance metrics. The first is path computation complexity measured by running times. We compare the average and maximum running times of the path computation solutions. The second is path computation effectiveness measured by blocking rate, which equals to the number of failed paths divided by the total number of requests. The third is path computation efficiency measured by the bandwidth-normalized average path hop-length, which is calculated by the following equation:

$$L_{Avg-Norm} = \left(\sum_{p \in \{paths\}} L(p) \times B(p) \right) / \left(\sum_{p \in \{paths\}} B(p) \right)$$

where *{paths}* is the set of all computed paths, *L(p)* is the length of path *p*, *B(p)* is the bandwidth of path *p*.

B. Numerical Results

We present the numerical results from our experiments in the following table and figures. We use *BASE* and *C-BFS* to denote the base line and C-BFS constrained search solutions. We ran two separate rounds of experiments, namely *KSP10* and *KSP100*, for the KSP heuristic search solution with K=10 and K=100 respectively. The graph transformation based KSP heuristic search has K=100, denoted by *GT-100*. Table I shows the average and maximum running times for these investigated path computation solutions in the network topologies with different traffic loads. The running times are measured by milliseconds. Configuration of the PCE host server is described below*.

Table I: Average and maximum running times for the investigated path computation solutions in milliseconds. (**CPU: Intel Xeon Duo-Core 3.0GHz; Memory 2.0GB; OS: Fedora Linux - kernel 2.6.9*)

	0% Load		20% Load		40% Load		60% Load	
	Avg	*Max*	*Avg*	*Max*	*Avg*	*Max*	*Avg*	*Max*
BASE	4	7	4	7	4	5	4	4
C-BFS	8	97	25	62	19	51	9	24
KSP10	9	11	8	9	8	10	8	9
KSP100	37	46	35	37	37	43	32	37
GT-100	214	509	148	420	112	128	95	142

Figure 4 shows the path blocking or failure rates for the investigated path computation solutions under different traffic loads. With zero traffic, all solutions are 100% successful in finding paths for all requests. When the network is loaded with traffic, the *BASE* solution has significantly higher blocking rates than other solutions. When *BASE* cannot find an IP layer path, it requests for an optical-layer path via the UNI points nearest to the source and destination IP routers. This excludes many feasible paths that have source and/or destination segments traversed in IP layer and go down to optical layer at intermediate nodes when needed. Also a UNI style separate-layer solution excludes some other feasible paths with multiple optical segments of different wavelengths connected by intermediate IP segments that virtually make O-E-O wavelength conversion. In contrast to *BASE*, other investigated path computation solutions can effectively find such paths via cross-layer multi-constraint joint optimization.

KSP10 is the next underperformer which is incapable of generating enough candidate paths due to a small K value. The remaining solutions *C-BFS*, *KSP100* and *GT-100* have very close performance. Under a 20% traffic load, *KSP100* beats *C-BFS* and *C-BFS* beats *GT-100*. Under a 40% traffic load, *KSP100* and *C-BFS* tie while both beat *GT-100*. Under a 60% traffic load *GT-100* beats *KSP100* and *KSP100* beats *C-BFS*. All these margins are small. None of the solutions constantly beats all other solutions. Nevertheless, *KSP100* demonstrates overall better performance than *C-BFS*. Note that *C-BFS* is not an exhaustive search. It cuts search branches based on TE constraints and the criteria described in Sec III.A and its average running times are shorter than *KSP100*, which slightly increases its blocking rates. *GT-100* surprisingly underperforms under light or medium traffic loads. This can be explained by the fact that graph transformation increases the topology size, which in turn increases the search space.

This effect becomes more obvious when the network load is light and a greater number of links are available for graph transformation which generates a larger transformed network, which needs a larger K. However, further increasing K may not be an option because *GT-100* has already been running much slower than *KSP100* with the same K value.

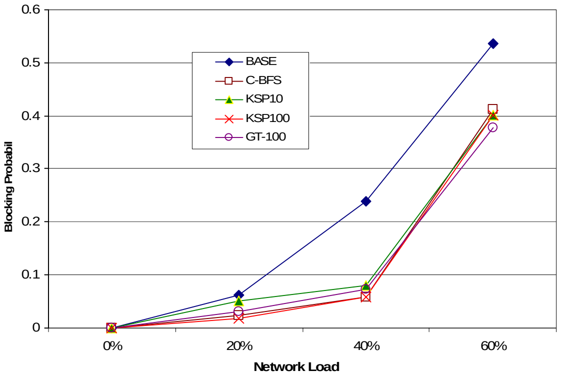

Figure 4: Blocking rates for the four path computation solutions.

Figure 5 compares the bandwidth-normalized average path length computed by the investigated path computation solutions. In each network load group, the five bars indicate the average path lengths in number of hops for the five experiment solutions. *BASE* generates the shortest paths after much higher blocking rates. *C-BFS* generates the longest paths except for under load 40%. Without counting in *BASE*, *KSP10* and *KSP100* generate the shortest paths. In particular, *KSP100* achieves this with blocking rates same or even smaller than others. Combined with modest running times, its overall performance slightly stands out among our investigated path computation solutions.

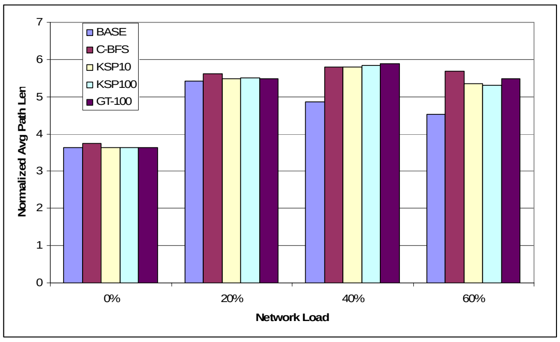

Figure 5: Bandwidth-normalized average path length for the four investigated path computation solutions.

V. CONCLUSION

In this study we addressed the issue of cross-layer multi-constraint path computation in IP-over-Optical networks. We investigated several path computation solutions in a pragmatic approach. We argued that a unified control plane technology should be used to allow for multi-constraint path computation across multiple layers. The algorithmic methods and techniques that compute paths in joint with multi-layer TE information include constrained search, graph transformation and heuristic search. We implemented these methods and techniques in the DRAGON open source control plane code that is being deployed in real-world networks. Performance of these methods and techniques are evaluated within four specific cross-layer multi-constraint path computation solutions on a 44 nodes simulated IP-over-WDM network.

From the implementation and experiments we gained some useful experience for path computation solutions in next-generation IP-over-Optical networks. They are (1) UNI-style separate-layer path computation is extremely inefficient as compared to cross-layer multi-constraint path computation; (2) constrained search, graph transformation and heuristic search are all effective cross-layer multi-constraint path computation methods and can be used in joint to improve performance; (3) dynamics in large-sized networks have complex impact on performance of any single path computation solution so that none of the investigated solutions constantly beats others in all situations; and (4) from a pragmatic perspective, a straightforward heuristic search solution such as the cross-layer multi-constraint version of KSP with a large K would provide reasonably good performance in many aspects, including computation speed, effectiveness and efficiency.

Through this study we come to the conclusion that cross-layer multi-constraint path computation solutions are both feasible and effective in achieving efficient resource allocation in the next-generation IP-over-Optical networks. A PCE making use of pragmatic path computation solutions will become an indispensable part of unified control planes that enable such networks.

REFERENCES

[1]. A. Farrel, J.-P. Vasseur, and J. Ash, “A Path Computation Element (PCE)-Based Architecture,” *IETF RFC 4655*, August 2006.

[2]. A. S. Tanenbaum, *Computer Networks*, Prentice Hall, Upper Saddle River, NJ, 3rd Edition (*Chapter 5.2*: *Routing Algorithms*), March 1996.

[3]. R. Ramaswami, *Optical Networks: A Practical Perspective*, Morgan Kaufmann Publishers, 1st Edition (*Chapter 8.2: LTD and RWA Problems*), February 1998.

[4]. S. Chen and K. Nahrstedt, “An Overview of Quality-of-Service Routing for the Next Generation High-speed Networks: Problems and Solutions,” *IEEE Network Magazine*, Vol. 12, No. 6, pp. 64-79, November/December 1998.

[5]. Y. Li, J. Harms, and R. Holte, “Fast Exact MultiConstraint Shortest Path Algorithms,” *in Proceedings of IEEE International Conference on Communications (ICC)*, pp. 123-130, Glasgow, Scotland, June 2007.

[6]. E. Oki, K. Shiomoto, et al., “Dynamic Multilayer Routing Schemes in GMPLS-Based IP+Optical Networks,” *IEEE Communications Magazine*, Vol. 43, No. 1, pp. 108-114, January 2005.

[7]. B. Jabbari, S. Gong, E. Oki, “On Constraints for Path Computation in Multi-Layer Switched Networks.” *IEICE Transactions*, Vol. 90-B, No. 8, pp 1922-1927, August 2007.

[8]. T. Lehman, J. Sobieski, and B. Jabbari, “DRAGON: A Framework for Service Provisioning in Heterogeneous Grid Networks,” *IEEE Communications Magazine*, Vol. 44, No. 3, pp. 84-90, March 2006.

[9]. Internet2, “Dynamic Circuit Network,” *Available on WWW at http://www.internet2.edu/network/dc/index.html*.