# Routing in All-Optical Label Switched-based Networks with Small Label Spaces

Fernando Solano Donado, Joanna Moulierac

# Routing in All-Optical Label Switched-based Networks with Small Label Spaces

Fernando SOLANO DONADO
Warsaw University of Technology, Poland
Email: fs@tele.pw.edu.pl

Joanna MOULIERAC
University of Nice, CNRS I3S INRIA, France
Email: joanna.moulierac@sophia.inria.fr

***Abstract*—With the development of All-Optical Label Switching (AOLS) network, nodes are capable of forwarding labeled packets without performing Optical-Electrical-Optical (OEO) conversions, speeding up the forwarding. However, this new technology also brings new constraints and, consequently, new problems have to be adressed. We study in this paper the problem of routing a set of demands in such a network, considering that routers have limited label space, preventing from the usage of label swapping techniques. Label stripping is a solution that ensures forwarding, concerning these constraints, of all the paths at expenses of increasing the stack size and wasting bandwith. We propose an intermediate feasible solution that keeps the GMPLS stack size smaller than label stripping, in order to gain bandwidth resources. After proposing an heuristic for this problem, we present simulations that show the performance of our solution.**

## I. INTRODUCTION

With the success of the Generic MultiProtocol Label Switching (GMPLS) protocol [1] in packet and circuit switched networks, GMPLS has been foreseen as the standard mechanism to handle the control plane of emergent technologies. However, the adaptation of GMPLS to these new technologies is not always straightforward and yields to new challenges.

One of these new challenges is a reduced label space in the routers. As an example, we consider All-Optical Label Switching (AOLS) technologies [2]. AOLS implements forwarding functions of GMPLS directly at the optical domain, enabling extremely fast packet forwarding. By using optical labels, the packets are processed by the optical switch without any Optical-Electrical-Optical (OEO) conversions aiding the forwarding decision taking. This implies that the label is extracted and processed optically and the forwarding is achieved via all-optical sub-systems.

The main disadvantage of AOLS-based networks is that each label requires its own hardware (a correlator and an incoming address generator [2]). Therefore, it is of major importance to reduce the number of employed correlators in every node (referred as the number of labels that are maintained, or label space, throughout the paper).

In this paper we consider the problem of finding, for each demand, a route and an efficient label assignation, so the label space is efficiently used in every node and the amount of bandwidth wasted due to large label stacks is reduced. This problem is subject to constraints specific to the usage of AOLS technology, that imply a limited number of optical correlators in each node and a fixed maximum stack size in the header.

This paper is organized as follows. In §II the basic concepts of GMPLS label forwarding mechanism are explained. In addition, the techniques label stacking and label stripping are briefly discussed. In §III we formally state the problem addressed in this paper. In §V, we propose a new heuristic aiming at solving the presented problem. Simulation results can be seen in §VI. Finally, conclusions are given in §VII.

## II. LABEL SWITCHING MECHANISM IN GMPLS

The connections established in GMPLS are called Label Switched Paths (LSP). Packets are associated to LSPs by means of a label, or tag, placed in the header of the packet. In this way, routers - called Label Switched Routers (LSR) in GMPLS - can distinguish and forward packets.

In addition, in GMPLS, it is allowed to carry a set of labels in the header of a packet; conforming a stack of labels. Even though a packet may contain more than one label, LSRs must only read the first (or top) label in the stack in order to take forwarding decisions. Stacking labels and label processing, in general, is standardized by the following set of operations that an LSR can perform over a given stack of labels:

- SWAP: replace the label at the top by a new one,
- PUSH: replace the label at the top by a new one and then push one or more onto the stack, and
- POP: remove the label at top in the label stack.

The labels stored in the table have a local meaning to the node and they are swapped all along the LSP. In this paper, we assume that nodes use a *per-platform label space*, i.e., the label has the same meaning regardless of the packet's incoming interface.

### A. Label stacking

When two or more LSPs follow the same set of links, they can be routed together 'inside' a higher-level LSP, henceforth a *tunnel*. In order to setup a tunnel, multiple labels are placed in the packet's header: a method known in the literature as *label stacking*.

As mentioned before, the LSRs in the core of the network route data solely on the basis of the topmost label in the stack. This helps to reduce both the size of the forwarding tables that need to be maintained on the core LSRs and the complexity of managing data forwarding across the backbone.

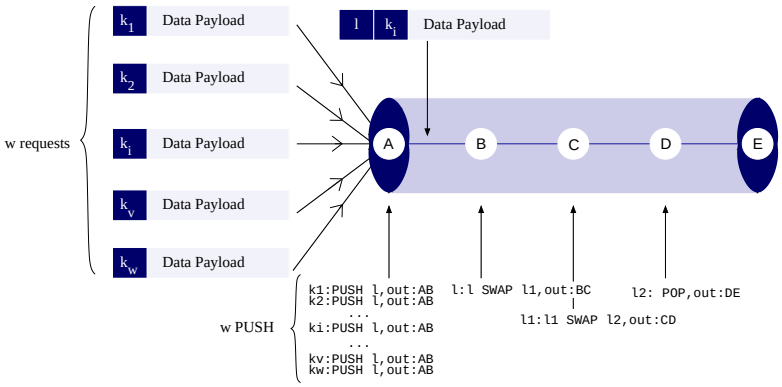


Fig. 1. GMPLS Operations performed at the entrance and at the exit of a tunnel.

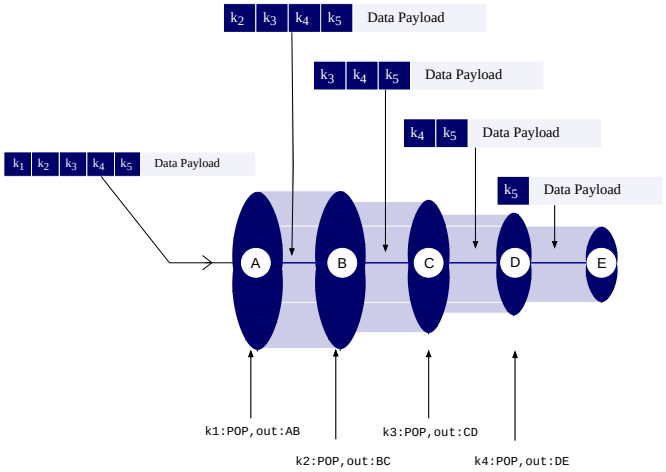


Fig. 2. Illustration of label stripping

Fig. 1 shows the general operations needed to configure a tunnel with the use of label stacking. At the entrance of the tunnel, $w$ PUSH are performed in order to route the $w$ requests through the tunnel. Then, only one operation (either a SWAP or a POP at the last node but one of the tunnel) is performed in all the nodes along the tunnel, regardless of $w$. In this figure, a stack of size two is used to route the $w$ LSPs in one tunnel from node $A$ to node $E$. The top label $l$ is swapped and replaced at each hop: by $l_1$ at node $B$, by $l_2$ at node $C$, and is finally popped at node $D$. The $w$ requests, at the exit of the tunnel at node $E$ can end or follow different paths according to their bottom label $k_i, \forall i \in \{1, 2, ..., w\}$ in the stack. Therefore, the labels in node $E$ are relevant for the next tunnel or LSP leaving from $E$.

### B. Label Stripping

The best way for employing the minimum number of correlators at any node is by performing *label stripping* [3], [4]. Label stripping refers to a technique that encodes the route of an LSP in the stack, so at every hop the pertinent LSR strips off (pops) the top label and determines the next hop based on its content. In label stripping, labels are never swapped nor pushed at core nodes. Therefore, since all the paths use one label for every hop, every node $v_i$ must store at most $\Delta(v_i)$ labels, where $\Delta(v_i)$ is the degree of the node $v_i \in V$.

Clearly, the number of labels that must be encoded in the stack (henceforth the stack size) is equal to the number of hops of the route, if label stripping is used. This fact leads to the employment of larger amounts of bandwidth.

Fig. 2 depicts an example of label stripping. Node $A$ receives a packet that is addressed to node $E$. In the packet, the whole path from $A$ to $E$ is encoded in a stack of labels $(k_1, ..., k_5)$. Node $A$ looks at the top label in the stack, *i.e.* $k_1$, looks at its GMPLS table, pops the top label and forwards the packet to $B$ according to the label $k_1$. Therefore from $A$ to $B$, the stack of labels is only $(k_2, ..., k_5)$. When the packet reaches node $E$, the stack of labels has been consequently reduced as the final destination has been reached.

It is our aim to propose in this paper a routing and label assignment heuristic in which the bound in the number of labels per node is not violated while aiming at reducing the size of the stack as much as possible.

It should be highlighted that, in AOLS, *the process of generating labels (or label stacks) in a source node does not require any optical correlator.*.

## III. Problem Definition

In AOLS networks, an optical device is required for each maintained label in the node. Therefore, it is of major importance to keep a low number of optical devices that are used in each node. Moreover, there is a strong restriction on this number when the network is already dimensionned.

The problem we adressed in this paper is to find a set of GMPLS LSPs or tunnels for a given set of requests, considering the restrictions on the number of correlators[1] on the node. A trivial solution is to use label stripping. However, its major inconvenience is that the label stack size is increased, together with the consumed bandwidth. Therefore, our aim is to find a solution using a smaller stack size.

The problem addressed here can be described as follows.

Inputs:

- the network $G = (V, E)$,

[1]The terms correlator and label are used indistinctly since, under this architecture, one label needs one correlator.

- $h(v_i)$ represents the number of optical correlators available $\forall v_i \in V$,
- a set of demands $\{(v_i, v_j)\} \subseteq V \times V$

OUTPUT: Find a set of paths and label bindings satisfying all the demands while respecting the constraints on the number of optical devices $h(v_i)$ per node.
OBJECTIVE: Minimize the maximum stack size for all the demands.

The nodes on the graph have a limited number of optical correlators of at most $h(v_i)$. In this paper, without loose of generalization, we assume that the number of available correlators is proportional to the degree of the node, therefore $h(v_i) = C \cdot \Delta(v_i)$, for some given constant $C$. The problem consists of finding a set of paths and a set of label forwarding entries for each router satisfying the constraints on the number of correlators in the node. The main objective of our proposal is to minimize the stack size, which will eventually lead to a reduction in the bandwidth utilization (as our experiments corroborate).

We conjecture that the problem belongs to the $NP$-hard class. However, a proper study of the complexity of the problem is left for further analysis.

*Extreme cases:* The two cases $C = 1$ and $C \to \infty$ correspond to extreme cases with trivial solutions: label stripping and label swapping, respectively. Between these two extremes, the problem will be to provide the routing for all the demands while respecting the limited number of labels to be stored per node. Therefore, we focus in the following on the case $C > 1$.

## IV. Related Work

To the best of our knowledge, this problem has not been treated before. However, solutions to similar problems related with the optimal usage of the label space can be found in the literature.

The first works related to the problem are not based on the usage of stack, but on a technique called label merging (not discussed here). Saito *et. al.* proposed a integer linear program in [5] for reducing the label space using label merging. Later, Bhatnagar *et. al.* proposed an heuristic in [6]. An analysis of the label merging technique was further extended in [7].

In [8] the authors deal with the problem of minimizing the number of used labels, when routes are given and the stack depth is limited to two. In [9], the authors extend this problem by assuming that routes should be found as well, considering that links have capacities. In these two contributions, the authors have as objective the minimization of the usage of the label space while keeping the stack depth to a maximum of two. Contrarily, in this paper, we want to reduce the stack depth while using the resources that already available at the nodes.

The study of the optimal usage of the stack was developed mainly by Gupta *et. al.* in [10] (and similarly in [11]). They performed a mathematical analysis for studying the trade-off between label space sizes and stack depth in some special network configurations. They focus in network configurations in which all nodes are interconnected either: $a$) along a path, or $b$) along a tree. Comparing it with our contributions in this paper, our problem considers more generic networks and, therefore, finding the routes is part of the problem.

## V. Proposed method for the general case

We describe in this section the method and heuristic that provide a solution to the problem in question.

Initially, we consider the solution of *label stripping* using shortest path over the input network. The idea of the proposed method is that at each iteration we augment the network by adding new virtual links to the input network. The added virtual links in each iteration correspond to tunnels traversing physical links. Obviously, to setup a tunnel, new labels are needed. We add as many virtual links as possible.

When no more virtual links (or tunnels) can be added, we route demands using shortest path over the virtual network, which consists of both physical and virtual links. In each hop of the computed shortest path, one label is striped. However, when a virtual link is traversed, labels are swapped in order to route the packets through the physical hops of the tunnel.

Since the augmented network in each iteration has more links, the diameter of the (virtual) network is potentially reduced. By considering the shortest path algorithm and the label stripping scheme over the virtual network, we use smaller stack headers as new virtual links are added.

Clearly, it is desirable to add tunnels (virtual links) that will pay off for the reduction in the stack sizes. Selecting the new tunnels that are going to be constructed is the core of the problem. The selection is made by maximizing the number of free optical correlators in the nodes in order to distribute fairly the labels among all the nodes of the network. We now focus in supporting procedures of the main heuristic that deals with the set of requests.

### A. Notation

Let $\Delta(v_i)$ be the degree of vertex $v_i$.

Let $h(v_i)$ be the size of the free label space for $v_i \in V$. At the beginning of the algorithm, the free label space $h(v_i)$ equals to $C \cdot \Delta(v_i)$.

Let $SP_G(v_i, v_j)$, with $v_i, v_j \in V$, be the shortest path in graph $G$ from $v_i$ to $v_j$ assuming hop count as a metric.

Let $MP_G(v_i, v_j)$ be the path from $v_i$ to $v_j$ such that $\min h(v_k), \forall v_k \in MP_G(v_i, v_j)$ is maximum.

### B. Computing paths that maximizes the free label space

The path $MP_G(v_i, v_j)$ is the one that maximizes the free label space for all the nodes traversed by the path. This algorithm ensures to use the routes that traverses nodes that are not overloaded and that have still enough free label space. This is specially useful when the value of $C$ is small and when few optical correlators are available in the nodes. The algorithm can be seen as Algorithm 1.

**Algorithm 1:** Path with Maximal label space

**Input**: $G = (V, E)$, source $s \in V$, destination $d \in V$
**Output**: Path from $s$ to $d$ with maximal label space
**begin**
  The set $U$ contains only the source: $U \leftarrow \{s\}$
  The set $N$ contains the neighbours of nodes in $U$
  **while** $d \notin N$ *and* $\exists v_i \in (V - U) \cap N, h(v_i) > 0$ **do**
    $U \leftarrow U \cup \{v_i\}$, where:

$$v_i = \arg \max_{v_k \in N - U} h(v_k)$$

    Update set $N$ with the neighbours of nodes in $U$
  **if** $d \in N$ **then**
    $U \leftarrow U \cup \{d\}$
  **else**
    **return** $MP(s, d) = \emptyset$
  Construct $G' = (U, F)$, where $G'$ is the graph induced by the nodes $U \subset V$ computed by the first steps.
  **return** $MP(s, d) =$ shortest path in the graph $G'$ between $s$ and $d$
**end**

### C. Main Heuristic

In this subsection, we propose a heuristic for the problem of finding a set of paths and tunnels for a set of demands.

At the beginning of the heuristic, the graph $G'$ is a copy of $G$. The remaining free label space is reduced by the respective degree of each node: $\forall v_i$, $h(v_i) \leftarrow h(v_i) - \Delta(v_i)$. This is to ensure that at least a solution using label stripping will be found for any demand. Then, the demands are considered in the decreasing order of the length of their shortest paths. A $MP_G(v_i, v_j)$ path is computed (always in $G$) for each $(v_i, v_j)$. Entries are added in all the nodes along the path, creating a tunnel. When it is not possible to find such a new tunnel (because nodes lack of labels), routing the demand is postponed for the second step of the heuristic. For every hop of the new tunnel, a new link in the graph $G'$ is added.

At the end of the heuristic, a shortest path in the graph $G'$ is used for the demands whose route computation was postponed. Label stripping is employed over $G'$ considering that the virtual links use one position in the stack along all virtual link's physical path. The complete heuristic can be read as Algorithm 2.

We illustrate an example of our heuristic with Fig. 3. Let us assume that we want to route a demand between every pair of nodes in the grid network. Let us assume $C = 1.5$, dimensioning nodes with degree two with three labels, nodes with degree three with five labels and node $E$ with six labels in total. Initially, we seize the label stripping entries.

Obviously, the diameter of the network is four, given by the shortest paths between $A$ and $I$, or $C$ and $G$. Arbitrarily, we first consider the demand $(A, I)$ among the four demands of length four in the graph. We consider the path with maximum free label space: $A \rightarrow B \rightarrow E \rightarrow H \rightarrow I$. We setup a tunnel over this path, seize the corresponding labels in the network and add the links $A-I$, $B-I$ and $E-I$ to $G'$. The process is repeated this time with demand $(C, G)$. The path found for it is $C \rightarrow F \rightarrow E \rightarrow D \rightarrow G$, filling up node $E$ label space and adding three more links to $G'$. Similarly, in the third iteration, a tunnel for demand $(G, C)$ is routed using $G \rightarrow D \rightarrow A \rightarrow B \rightarrow C$, filling up the label space for nodes $A$, $B$ and $D$. In the next iteration, a path for the demand $(I, A)$ cannot be found, due the lack of labels. After few iterations in which no demand can be routed, we are able to route demands $(G, F)$ and $(I, C)$, creating two more tunnels, as shown.

**Algorithm 2:** Main Heuristic

**Input**: $G = (V, E)$, set of demands $\{(v_i, v_j)\} \subseteq V \times V$
**Output**: A set of tunnels satisfying all the demands
**begin**
  Compute $SP_G(v_i, v_j), \forall v_i, v_j$ in the graph $G$
  Sort the demands in the order of decreasing length of their shortest paths and include them in stack $S$
  Let $H \leftarrow \emptyset$
  **for** *pop a demand* $(v_i, v_j)$ *from the stack* $S$ **do**
    Compute $MP_G(v_i, v_j)$ in $G$
    **if** $MP_G(v_i, v_j) = \emptyset$ **then**
      $H \leftarrow H \cup (v_i, v_j)$
    **else**
      Create a tunnel from $v_i$ to $v_j$ following the path $MP_G(v_i, v_j)$
      Let $h(v_i) \leftarrow h(v_i) - 1, \forall v_i \in MP_G(v_i, v_j)$
      **for** $\forall v_k \in MP_G(v_i, v_j) - v_j$ **do**
        Add in $G'$ an edge between $v_k$ and $v_j$
        **if** $\exists (v_k, v_j) \in S$ **then**
          $S \leftarrow S - (v_k, v_j)$
  **for** $(v_i, v_j) \in H$ **do**
    Compute a shortest path $SP_{G'}(v_i, v_j)$ in $G'$
    **if** *Several paths have the same length* **then**
      Choose the path with the smaller stack and if several with the shortest length in $G$.
    Label stripping is used for the part of the path that corresponds to original edges in $G$ and label swaping for the part of the path that corresponds to tunnel (virtual links in $G'$)
**end**

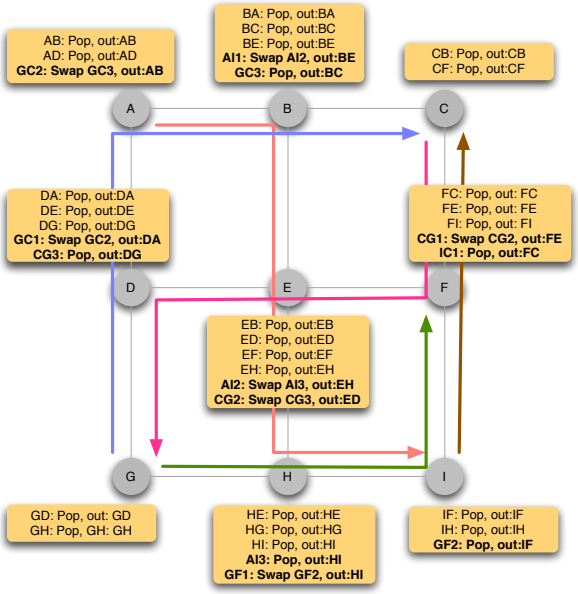


Fig. 3. Example of the proposed heuristic.

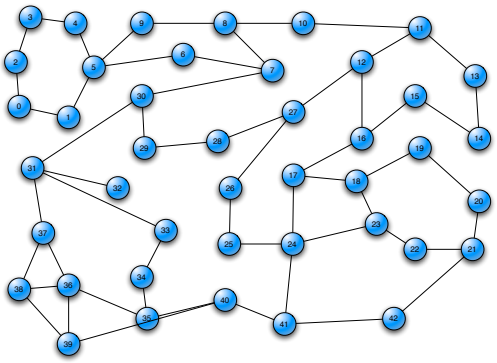

Fig. 4. Eurorings network used in the simulations.

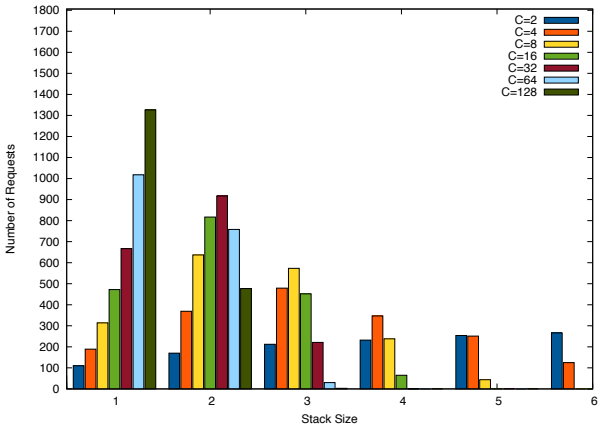


Fig. 5. Number of demands (Y) using a given stack size (X), when the number of labels is varied (series).

At this point, $G'$ has a diameter of three and no more tunnels can be created in the network. In the second step, the rest of the routes are computed using shortest-path over $G'$ and the label assignment is done using label stripping considering the placed tunnels as links.

In the example, for instance, demand $(I, A)$ will follow the path $I \rightarrow F \rightarrow C \rightarrow B \rightarrow A$ using a stack of three, even though it goes through four physical links. Concretely, node $I$ labels the packets for demand $(I, A)$ as $IC1/CB/BA$ and forwards them to node $F$. Node $F$ reads label $IC1$, pops the stack and forwards packets to node $C$. Node $C$ receives packets marked only with label $CB$, it pops label $CB$ and forwards them to node $B$. Node $B$, receives packets with label $BA$ and forwards them to $A$, after popping the last label on the stack. Node $A$ receives packets with no label, indicating that they are addressed to itself.

## VI. Simulations Results

The simulations have been run on the Eurorings network, which has 43 nodes and 55 links (see Fig. 4). We generate one demand between every pair of nodes, in total 1806 demands. Each demand has a bandwidth requirement of one unit.

Since the label assignation scheme is novel, to the best of our knowledge, no other heuristic that deals with the constraint on the number of optical devices per node exists in the literature. Therefore, for comparison purposes, we take into account label stripping and label swapping techniques.

### A. Stack size distribution

In the experiments we vary the parameter $C$, which is a multiplying factor related with the number of labels per node. For the $k$-th experiment, we set $C^{(k)} = 2^k$. Note that label stripping corresponds to $C = 1$.

In each experiment, we run our heuristic and we classify the demands according to the size of the stack needed to route them in our solution. In Fig. 5, we plot the distribution of demands according to the maximum stack size. For instance, when $C = 8$, there are a few demands that need a stack of size five (small yellow bar at category five in the plot). As $C$ increases, less demands need larger stack sizes.

For label stripping, the maximum stack size is 14, which is given by the diameter of the network. Our heuristic shows that, if we simply have $C = 2$, for instance, we can reduce the maximum stack size to eight labels (not shown in the plot).

### B. Bandwidth usage

We now neglect the size of the header and in Fig. 6, we plot the average and maximum allocated bandwidth on links throughout our experiments (varying $C$): *only the payload*. Without doubts, our heuristic seems to use more bandwidth than the shortest path, which is explained by the fact that the traffic routes may cross more physical links. For the shortest path, no restriction has been made for the label space. Therefore, the number of optical correlators needed in that case is at maximum 559 and in average 265 which is a lot more than the number needed for our heuristic.

We can see while varying the value of $C$, the trade-off between bandwidth utilization and stack size. Indeed, the larger the stack size, the lower the bandwidth utilization.

However, we continue our analysis by taking into account the size of the header in each scenario, since label stripping needs a larger stack size. We compute the minimum ratio between the size of a label and the size of a payload for which our heuristic incurs in bandwidth savings. Let $P$ be the size of the payload and $r \cdot P$ the size of a label. Then, our heuristic performs better in bandwidth if and only if:

$$B_k \cdot (H_k \cdot r \cdot P + P) \leq B_s \cdot (H_s \cdot r \cdot P + P),$$

where $B_k$ and $B_s$ represents the average link utilization by our heuristic at the $k$-th experiment and by the shortest path, respectively; and $H_k$ and $H_s$ represents the maximum number of labels in the $k$-th experiment[2] and by the shortest path (i.e.

[2]We consider that in AOLS all the packets have the same header length, and therefore equals to the maximum stack size.

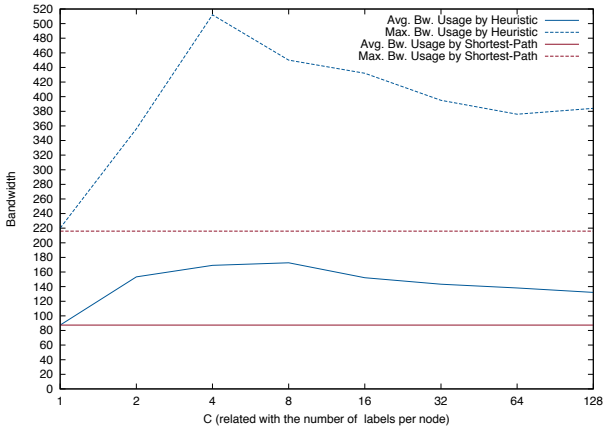


Fig. 6. Maximum (dashed lines) and average (continuous lines) allocated bandwidth on links for shortest path (blue lines) and our proposed heuristic (red lines).

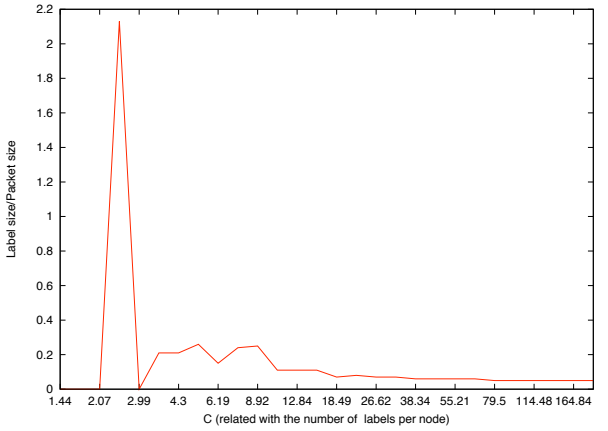


Fig. 7. Minimum ratio between the size of a label and the payload for which our solution incurs in bandwidth savings.

14), respectively. Therefore, we have:

$$r \leq \frac{B_s - B_k}{B_k \cdot H_k - B_s \cdot H_s},$$

Which is plotted in Fig. 7. This time, in order to detail better the ratio gain factor curve (i.e. the values of $r$ shown over the Y-axis of the figure), we fine the experiments by letting grow $C$ as $C^{(k)} = 1.2^k$. The experiments show that when $C \leq 12$, it is worth using our heuristic if the size of a label is more than $20\%$ the size of the packets payload. In general, this ratio decreases as $C$ increases. For instance, when $C \geq 30$, a ratio of $5\%$ is already enough to incur in bandwidth savings.

The large peak in the plot signifies that the heuristic is not good enough to compute a good solution when the values of $C$ are small. Indeed, when optical correlators are scarce, the problem is harder to solve.

## VII. Conclusions

In this paper we propose a new method for label assignation that can be used when the label spaces of the nodes are considerably small. The proposed heuristic aims at offering an intermediate solution between label stripping and label swapping. In other words, it routes traffic using smaller header sizes than label stripping while preserving the bounds on the label spaces.

Concerning the size of the stack, our results show that if nodes have a label space dimensioned twice as large as its degree, for instance, we can reduce the stack size by $42\%$. Depending on the ratio between the size of a label and the packets payload, our heuristic may reduce the overall usage of bandwidth. For instance, when a node counts with 20 labels per neighbor and the size of a label is no less than $5\%$ the size of the payload.

It is our purpose to propose in a future mathematical models that can efficiently solve the aforementioned problem. In addition, heuristic for the specific cases in which the label spaces in a node is close to its degree are left as future work.

## Acknowledgements

The research presented in this paper has been partly funded by the project "Optimization Models for NGI Core Network" (Polish Ministry of Science and Higher Education, grant N517 397334), the CELTIC Mango project, the European project FET AEOLUS and the COLOR INRIA LARECO.

## References

[1] A. Banerjee, J. Drake, J.P. Lang, B. Turner, K. Kompella, and Y. Rekhter. Generalized multiprotocol label switching: An overview of routing and management enhancements. *IEEE Commun. Mag.*, 39(1):144–150, January 2001.

[2] Francisco Ramos et al. IST-LASAGNE: Towards all-optical label swapping employing optical logic gates and optical flip-flops. *IEEE J. Select. Areas Commun.*, 23(10):2993–3011, October 2005.

[3] Ruth Van Caenegem et al. Benefits of label stripping compared to label swapping from the point of node dimensioning. *Photonic Network Communications Journal*, 12(3):227–244, December 2006.

[4] Ruth Van Caenegem et al. From IP over WDM to all-optical packet switching: Economical overview. *J. Lightwave Technol.*, 24(4):1638–1645, April 2006.

[5] Hiroyuki Saito, Yasuhiro Miyao, and Makiko Yoshida. Traffic engineering using multiple MultiPoint-to-Point LSPs. In *Proc. IEEE INFOCOM 2000*, pages 894–901, 2000.

[6] Sudeept Bhatnagar, Samrat Ganguly, and Badri Nath. Creating Multipoint-to-Point LSPs for traffic engineering. *IEEE Commun. Mag.*, 43(1):95–100, January 2005.

[7] Fernando Solano, Ramon Fabregat, and Jose Marzo. On optimal computation of MPLS label binding for MultiPoint-to-Point connections. *IEEE Trans. Commun.*, 56(7):1056–1059, July 2007.

[8] Fernando Solano et al. Label space reduction in MPLS networks: How much can one label do? *IEEE/ACM Trans. Networking*, February 2009.

[9] Fernando Solano et al. All-optical label stacking: Easing the trade-offs between routing and architecture cost in all-optical packet switching. In *IEEE Conference on Computer Communications (Infocom 08)*, pages 655–663, Phoenix, AZ, USA, April 2008.

[10] Anupam Gupta, Amit Kumar, and Rajeev Rastogi. Traveling with a pez dispenser (or, routing issues in MPLS). *SIAM Journal on Computing*, 34(2), 2005.

[11] Anupam Gupta, Amit Kumar, and Rajeev Rastogi. Exploring the trade-off between label size and stack depth in MPLS routing. In *Proc. IEEE INFOCOM 2003*, 2003.