Performance Evaluation of the Grade-of-Service-based Routing Strategies for Optical Networks

Andrzej Szymanski, Artur Lason, Jacek Rzasa, and Andrzej Jajszczyk
AGH University of Science and Technology, Department of Telecommunications, Krakow, Poland
Email: szymans@agh.edu.pl, {lason, rzasa, jajszczyk}@kt.agh.edu.pl

Abstract—The paper is focused on an optical network in which the network operator needs to achieve differentiated connection blocking probability between different classes of lightpath requests. We propose and investigate several routing strategies, which may be used for that purpose, i.e., to maintain the blocking probability of the high priority requests below an assumed value in a broad range of network loads. Simulation results show that all proposed strategies achieve the stated goal, but their performance varies considerably. The paper presents differences in performance of individual strategies.

I. INTRODUCTION

Optical networks play an important role in today’s networking. An increase in the available capacity promotes the introduction of new services, each having different network requirements. We believe that diversified services, tailored to customers’ needs are the key for both retention of old and attraction of new customers. On one hand this requires high capacity, on-demand provisioned connections, on the other, providing connections with the quality of service (QoS) and grade of service (GoS) adequate to customers’ requirements. Quality of Service (QoS) is recognized in this paper as every parameter that affects data flow after the connection (lightpath) is established, Grade of Service (GoS) includes all parameters applicable to connection setup.

Inherently, GoS is directly related to the Routing and Wavelength Assignment (RWA) algorithm used in the network. Several Routing and Wavelength Assignment algorithms for optical networks appeared in the past, for a review see [1], [2]. These papers concentrate on an algorithm, or a set of algorithms, that would provide the best results either in terms of the blocking probability or carried load. However, the papers focus on a network that serves a single class of requests, and little work is available on networks with multiservice traffic.

In [3] we have provided a tutorial overview of the problems associated with providing controlled GoS in optical networks, and we have proposed several mechanisms that are able to differentiate requests that belong to different classes. The purpose of this paper is to investigate routing strategies that combine the presented mechanisms, compare their performance and assess whether such combinations are worth considering. The goal of the investigated strategies is to achieve the blocking probability of the high priority requests below an assumed value in a wide range of network loads. Simultaneously, the strategies should have a low impact on the low priority requests and require as little additional information as possible, to keep the control plane complexity at a reasonable level.

The rest of the paper is structured as follows. The next section contains descriptions of strategies that will be investigated and mechanisms that are the building blocks for those strategies; the section that follows provides some details on the simulation environment used for investigation. We then discuss the obtained results and, finally, we conclude the paper.

II. MECHANISMS AND STRATEGIES

A Grade-of-Service routing strategy is built on one or more mechanisms that may be used to achieve differentiated services in the optical network. As it was discussed in [3], there are at least three groups of mechanisms that may be used to provide GoS differentiation:

(a) mechanisms which preserve network resources for high priority requests,
(b) mechanisms in which a different routing algorithm or a set of candidate routes is considered for each class of requests,
(c) mechanisms in which an existing lower class lightpath may be preempted or rerouted if the resources it uses are needed for a higher class lightpath.

Here, we will consider only the strategies that are built upon mechanisms that either preserve network resources or use a different routing algorithm or a set of candidate routes. We will first present the mechanisms in a brief form (for details see [4], [3]), and then, we will follow with a description of strategies using those mechanisms.

A. Mechanisms which preserve network resources for high priority requests

The general idea behind this group of mechanisms is to provide a notion of resources that are protected from being used by low priority lightpaths. Effectively, this increases
the future high priority requests will find sufficient resources in the network at the time they arrive. The preservation of resources is a kind of “educated guess,” since the future high priority requests are not known at the time resources are preserved.

In the first mechanism, called First Link Capacity Threshold (flcap), a given number of wavelengths, $T$, must remain free on the first link of a path when a low priority request is being served. This mechanism does not require any changes to the control plane, since the number of free wavelengths on the first link of the path should be readily available in the source node of that path. The numerical complexity of the presented algorithm may be estimated as $O(w)$, since we only have to check the set of $w$ wavelengths on a single link, and this information is available in the local node.

In the second mechanism, named Link Capacity Threshold (lcap), a given number of wavelengths, $T$, must remain free on each link for use by the high priority lightpath requests. Low priority requests are accepted on a given link as long as there are more than $T$ free wavelengths on the link. This mechanism requires two changes to the control plane. First, lightpath reservation messages must carry the priority of the request. Second, each node must know the threshold value. The numerical complexity of the algorithm may be estimated as $O(lw)$ since we need to check all links on a candidate path, the number of which is upper bounded by $l$, and on each link we need to check $w$ wavelengths.

In the third mechanism, named Wavelength Pools (pool), the set of wavelengths is partitioned into two pools. The first (common) pool may be used by all lightpath requests. The second (preserved) pool may be used only by high priority lightpath requests. This pool is used if it is not possible to set up a request using the common pool. With this mechanism, control plane modifications are even smaller as it is only needed to know the division of wavelengths into pools in each node. No further changes are necessary, since for the low priority requests it is possible to report reserved wavelengths as busy in the source node of a lightpath. This mechanism was first investigated in [5], where the authors analyzed this concept using both a continuous-time Markov chain and computer simulation. The complexity of the mechanism may be estimated as $O(lw)$.

The fourth mechanism, called the Path Capacity Threshold (pcap), is similar to the second one. However, threshold $T$ now defines how many continuous wavelengths must remain free on a path a low priority lightpath request is going to take. If the number of free continuous wavelengths is above the threshold, then the low priority request is admitted. Otherwise, the low priority request is blocked. High priority requests are unaffected by this threshold. Similarly to the second mechanism, the necessary changes to the control plane include dissemination of the threshold value and adding the request priority to the reservation messages. The numerical complexity of the algorithm may be estimated as $O(lw)$ since we need to check all links on a candidate path, which is upper-bounded by $l$, and on each link we need to check $w$ wavelengths to remove busy wavelengths from the set of available wavelengths, finally we need to count the number of elements in the resulting set of available wavelengths which may contain, at maximum, $w$ elements.

The Global Capacity Threshold (gcap) mechanism also operates by preserving network resources. This mechanism falls at the opposite end with regard to complexity and requirements enforced on the control plane. Path evaluation in this mechanism is quite complex and requires the global network state to be known in the source node of the lightpath. The idea behind this mechanism is to preserve wavelength continuous paths for future high priority requests, so that after a low priority lightpath is set up, there should be enough resources in the network to set up at least $T$ high priority lightpaths between any pair of nodes, where $T$ is a given threshold.

When a low priority request is served, its route is determined first and a wavelength is assigned by using one of the standard RWA algorithms. This is step 1. In step 2, the free resources in the network must be evaluated to determine whether the low priority request can be accepted or not. Thus, for each source - destination pair it is necessary to evaluate the potential number of high priority lightpaths originating in the source and terminating in the destination that could be served: a) before the considered low priority lightpath is set up, b) after the considered low priority lightpath is set up.

If the setup of the low priority lightpath reduces the number of potential lightpaths (for any pair) below a given threshold $T$, then the low priority lightpath should not be accepted on a given route and wavelength. The algorithm may revert to step 1 to find and evaluate another path-wavelength pair.

The numerical complexity of this algorithm may be estimated in the following way. To evaluate a candidate path we need to evaluate at most $w$ wavelengths. For each wavelength we need to evaluate $n(n-1)$ communicating pairs. Assuming that the potential paths for each potential pair are precomputed and assuming the complexity of evaluating each s-d pair as $O(lw)$ (the complexity of the Path Capacity Threshold mechanism), the resulting complexity is $O(n^2lw^2)$, which is considerably worse than the complexity of the other algorithms.

The detailed algorithms for pcap and gcap mechanisms may be found in Appendix.

B. Mechanisms in which a different routing algorithm or a set of candidate routes is considered for each class of requests

In this class of mechanisms the low and high priority requests are differentiated by using a different set of potential routes or a different routing algorithm for each class of requests. Low priority requests are served with a simple algorithm, which provides an inferior blocking probability, while high priority requests are served with a more advanced algorithm that provides a lower blocking probability.

One of the possible implementations of this mechanism is the alternate routing with a different number of paths for each class of requests. High priority requests are offered more
paths while low priority requests are offered fewer paths to consider. This implementation has been used in the subsequent investigations.

We used the following algorithm to compute the set of alternate paths:

1: Set the metric of all links in the topology to 1;
2: The first path between two nodes is obtained by using the shortest path algorithm;
3: Set the metric to $n$ for all links that belong to the first path; $n$ is the number of nodes in the network;
4: Compute the second path using the shortest path algorithm;
5: Set the metric to $n$ for all links that belong to the second path; $n$ is the number of nodes in the network;
6: Compute the third path using the shortest path algorithm.

Originally, we wanted to compute alternate paths as link disjoint paths. However, due to the fact that the underlying topologies have the connectivity degree between 2 and 3, it is often impossible. Thus, we switched to the version of the algorithm presented above, which allows re-using links from previously computed paths, but with a great penalty, greater than any path length in the network. For the same reason we restricted the number of alternate paths to three.

C. Strategies

Based on the presented mechanisms we have defined and tested 30 strategies. Half of them employ just one mechanism, the rest are a mix of two mechanisms. Single mechanism strategies employ one of the mechanisms that preserve resources and alternate routing with the same number of paths (1, 2 or 3) for each class of requests. The alternate paths are searched for using the algorithm presented earlier.

The mixed strategies differentiate requests using two mechanisms simultaneously: a mechanism that preserves resources and alternate routing with a different number of paths for each class of requests. We have used the following combinations of the number of alternate paths for low and high priority requests, respectively: (1,2), (1,3) and (2,3).

We did not include in our investigations the single mechanism strategies that used just the different number of paths for different classes of requests (without resource preservation) since the preliminary studies have shown that they are unable to reach the assumed target, which is to maintain the blocking probability of the high priority requests below 0.005 for a specified range of the offered traffic.

III. Simulation Environment

The simulation environment was built using the OMNeT++ simulator [6]. The default pseudo-random number generator in version 2.3 of OMNeT++ has been replaced with Mersenne-Twister generator [7] with a period length of $2^{19937} - 1$ from the Boost Random library [8]. The results were obtained using the batch means method and evaluated at the 0.95 confidence level.

To assess the behavior of the presented strategies we have used two networks: the 28 nodes Pan-European reference network shown in Fig. 1 and the 14 nodes NSF network (c.f. [9]) shown in Fig. 2. Each link consists of two fibers, one in each direction and each fiber was assumed to carry 80 wavelengths. There are no wavelength converters in the network and the wavelength assignment is done using the first-fit scheme.

The offered traffic is generated based on a uniform traffic matrix. In the base case each node pair generates bidirectional lightpath requests, with mean intensity of 0.033 lightpath per unit time for the Pan-EU network and 0.165 lightpath per unit time for the NSF network. The interarrival time was exponentially distributed. A given fraction (10 – 50%) of the offered traffic belongs to the high priority class, while the remaining traffic belongs to the low priority class. For each simulated node four separate generators were used: two to generate lightpath interarrival times, one for each traffic
priority class, and, similarly two generators for generating the lightpath holding times. The lightpath holding time is also exponentially distributed with mean of 10 units time, independent of the class. As a result, the total network load in the base case is \(28 \times 27 \times 0.033 \times 10 = 249.48\) Erl for the Pan-EU network and \(14 \times 13 \times 0.165 \times 10 = 300.3\) Erl for the NSF network. To observe network behavior under different loads, this base case traffic is scaled by the factor ranging from 1.0 to 1.8. The scaling factor is used in the subsequent text and figures as the offered load intensity measure. Traffic intensity is varied by changing the mean lightpath interarrival time while the mean lightpath holding time remains constant. The range of the offered traffic has been chosen so that in low load region the network without GoS mechanisms is able to provide less than 0.005 blocking probability for all requests and in the high load region the blocking probability is greater than 0.05.

The simulation studies have been performed in the networks with the centralized control, where the central controller is responsible for all computations and all resource allocation, thus possesses the perfect knowledge about the network state. Investigations of the networks with distributed control are planned for further studies.

### IV. RESULTS

The comparison of strategies is quite difficult. We have tested the strategies under 9 values of offered traffic and for each value we have tested 5 different mixes of traffic: 10/90, 20/80, 30/70, 40/60, and 50/50, where the left and right value shows the percentage of, appropriately, high and low priority traffic. This gives in total 45 combinations of the offered traffic. Second, for each strategy it is possible to choose a different value of parameter \(T\), which regulates the amount of resources preserved for high priority traffic. As a result, this value affects the balance between the blocking probability of the low and high priority requests. To be able to compare the strategies, we needed to have some reference criterion, to which the strategies would be normalized. We have assumed that: 1) the blocking probability of the high priority requests should be below 0.005 for each of the 45 points of the offered traffic considered, and 2) the value of \(T\) should remain constant with changing traffic conditions, i.e., in all 45 points of the offered traffic investigated. This is a potential scenario of an ASON network with variable load and fixed network parameters.

Consequently, we get a lower bound for parameter \(T\) since with increasing its value the amount of resources devoted to high priority requests increases, and as a result the blocking probability of the high priority requests decreases. However, increasing the value of the parameter \(T\) increases the blocking probability of the low priority requests which is undesirable. Thus, it is reasonable to keep the value of \(T\) as low as possible to avoid unnecessary drop in performance of the low priority requests. As a result, for each strategy there is only one possible value of parameter \(T\) and we have obtained this value by trial-and-error method. In the rest of the article we will use only this value of parameter \(T\) for a given strategy. The obtained values are presented in Tables I and II, names \(altXY\) reference the alternate paths algorithm with \(X\) paths for low priority requests and \(Y\) paths for high priority requests.

The assumption that we made in the previous paragraph has one more implication. We can safely assume that the high priority class is served in a satisfactory manner in all assumed traffic conditions, for all considered strategies. Thus we can safely disregard it in the subsequent discussions and focus on the comparisons of the blocking probability of the low priority requests, since this reflects the price we pay for granting the high priority requests a superior grade of service.

The question that remains to be answered is, how can we compare the blocking probability of the two strategies in 45 different cases of offered load. We have decided to use the number of cases in which a given strategy is better than the other ones. Note, that in a statistical sense we have assumed that each offered traffic case is equally probable.

For the rest of the paper the following notation will be used: \(altXY-algo\), where \(X\) is the number of paths for a low priority request, \(Y\) is the number of paths for a high priority request, \(algo\) is the abbreviated name of the algorithm. For example \(alt13-pcap\) is the Path Capacity Thresholds strategy with one path for low priority requests and 3 paths for high priority requests, the value of \(T\) for this scenario is 2 (see Table I).

Table III presents the number of cases in which a given strategy is the best one according to the specified criteria. The numbers show, that the scenario \(alt33-gcap\), combining the Global Capacity Thresholds strategy and the alternate paths strategy with three paths for each traffic class achieved the best results for the PanEU network, and achieved the majority of the best results (35 out of 45 cases) for the NSF network. However, this scenario is considerably more complex one, compared to the others, and requires a great deal of information about the network state. Gathering such information may not be desirable for performance and scalability reasons. Thus,
TABLE III
THE BEST STRATEGIES

<table>
<thead>
<tr>
<th>Network</th>
<th>Scenario</th>
<th>No of cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>PanEU</td>
<td>alt33-gcap</td>
<td>45</td>
</tr>
<tr>
<td>NSF</td>
<td>alt33-gcap</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>alt33-pcap</td>
<td>10</td>
</tr>
</tbody>
</table>

TABLE IV
THE BEST STRATEGIES IF GCAP IS NOT CONSIDERED

<table>
<thead>
<tr>
<th>Network</th>
<th>Scenario</th>
<th>No of cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>PanEU</td>
<td>alt33-pcap</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>alt33-lcap</td>
<td>1</td>
</tr>
<tr>
<td>NSF</td>
<td>alt33-pcap</td>
<td>45</td>
</tr>
</tbody>
</table>

it might be interesting to know which scenario would perform the best if strategies using gcap mechanism were excluded from comparisons. The result is shown in Table IV.

The strategy that performed the best is the Path Capacity Thresholds algorithm coupled with the routing strategy that provided three alternate paths for each request, both low and high priority. It achieved the lowest blocking probability for low priority requests in all, but one case, for the PanEU network and in all cases for the NSF network.

It might be interesting to know, whether it is beneficial to choose gcap and pcap if, for some reason a network operator wants to run the network with a given alternate routing variant. The answer to that question is positive (see Table V). For all investigated alternate routing variants the best algorithm to use is gcap, and if gcap is excluded for the reasons presented earlier, pcap.

In Fig. 3 we have presented a slightly different comparison of the investigated strategies. The figure shows the blocking probability of the low priority requests vs. offered load for different mechanisms. The network is the PanEU network, with 20% of high priority traffic and alt33 routing. For all strategies the blocking probability rises gradually with the increasing load. However, in the low load region the strategies utilizing fcap and pool mechanisms introduce high blocking probability, which is unnecessary and harmful. On the other hand, gcap and pcap mechanisms behave very well in this aspect. For completeness Fig. 4 presents the blocking probability in the high priority class.

In our opinion strategies based on gcap and pcap are strong candidates for implementing them in real networks. An important observation is that pcap follows closely the performance of gcap (see Fig. 3), which might be an indication that in the overall cost-benefit comparison pcap might be preferred, due to its much lower complexity and a similar performance to gcap. As far as other algorithms are considered, lcap and pool offer a similar complexity to pcap, but significantly worse performance, and fcap delivers unacceptable performance with a slightly less complexity. As a result strategies based on lcap, pool and fcap are questionable candidates for implementing in real networks.

V. CONCLUSIONS

The GoS differentiation allows a network operator to deliver services tailored to customers’ requirements, while simultaneously ensuring efficient resource utilization. In the paper we investigated several strategies that are able to successfully deliver GoS-differentiated services in the optical network.

All strategies managed to achieve the stated goals in assumed network conditions. However, the cost of implementing
those strategies, being the decreased performance of low priority requests and complexity of control procedures, was quite different.

The best strategies were based on either Global Capacity Threshold and Path Capacity Threshold mechanisms, the former performing slightly better than the latter at the cost of radically increased complexity and requirements. These two mechanisms were the most successful ones with each variant of alternate paths routing that we have tested. However, the best results with those two algorithms were obtained when all requests, regardless of their class, were offered the maximum allowed number of paths. For those two strategies, introducing an additional level of GoS differentiation by offering a different number of paths for each class of requests resulted in a performance drop. In an overall view, Global Capacity Threshold and Path Capacity Threshold are preferred mechanisms to be implemented in future optical networks, due to their superiority in performance.

Along with the development of new services delivered to customers and growing users’ needs, the complexity of GoS mechanisms will increase in order to deal with more traffic classes and less predictable traffic patterns. The improved GoS mechanisms will provide more precise control over such GoS parameters as the blocking probability and connection set up time and their distribution among communicating node pairs. This seems to be the focus of the future work in this field.

**APPENDIX – ALGORITHMS**

The algorithms operate on a lightpath request $R$ from a node $S$ to a node $D$. The request $R$ is either a low or a high priority request. Each link carries $W$ wavelengths in each direction and $\Omega = \{\lambda_1, \ldots, \lambda_W\}$ is the set of wavelengths in a network. $P$ denotes a path (a sequence of links), which will be used for setup of the request $R$. $A$ is the set of wavelengths considered for a lightpath setup. $|A|$ is the number of elements in $A$. $A_0$ is the set of unused (free) wavelengths on the first link of path $P$. $T$ is a configurable threshold parameter that determines how much resources are preserved for high priority requests.

The **algorithm for the Global Capacity Threshold strategy**

1: Compute the first candidate path $P$ from the node $S$ to the node $D$ according to the preferred routing algorithm.
2: Compute the set of available wavelengths $A \subset \Omega$ so that each $\lambda_i \in A$ is free on all links on the path $P$.
3: If $A = \emptyset$ then decline the candidate path $P$ and go to step 7.
4: If $R$ is a low priority request and $|A| \leq T$ then decline the candidate path $P$ and go to step 7.
5: Choose a wavelength $\lambda_j \in A$ according to the preferred wavelength assignment algorithm.
6: Set up the request $R$ on the path $P$ and the wavelength $\lambda_j$ and FINISH.
7: Compute the next candidate path $P$; if it exists go to step 2, otherwise block the request $R$ and FINISH.

The **algorithm for the Path Capacity Threshold strategy**

1: Compute the first candidate path $P$ from the node $S$ to the node $D$ according to the preferred routing algorithm.
2: Compute the set of available wavelengths $A \subset \Omega$ so that each $\lambda_i \in A$ is free on all links on the path $P$.
3: If $R$ is a low priority request and $|A| \leq T$ then decline the candidate path $P$ and go to step 8.
4: If $A = 0$ then decline the candidate path $P$ and go to step 8.
5: Choose a wavelength $\lambda_j \in A$ according to the preferred wavelength assignment algorithm.
6: If $R$ is a low priority request then assess the candidate path-wavelength pair $(P, j)$ using the assessment algorithm (presented later in the text); if the candidate path-wavelength pair is accepted then go to step 7, otherwise remove the wavelength $j$ from the set $A$ and go to step 4.
7: Set up the request $R$ on the path $P$ and the wavelength $\lambda_j$ and FINISH.
8: Compute the next candidate path $P$; if it exists go to step 8, otherwise block the request $R$ and FINISH.

**ACKNOWLEDGMENT**

This work was funded by the Polish Ministry of Science and Higher Education under Grant No. N51701332/2131 and PBZ-MNiSW-02/I1/2007 G018/T02/2007.

**REFERENCES**


