A Defragmentation-Ready Simulation Framework For Elastic Optical Networks

Rodrigo Stange Tessinari, Didier Colle, Anilton Salles Garcia

***Abstract*—In this paper we explore in details the framework ElasticO++, which is a simulation framework for Elastic Optical Networks using OMNeT++. ElasticO++ is the first software available capable of working with spectrum defragmentation in dynamic network scenarios. The flexibility offered by the discussed tool allows both academia and industry to develop and evaluate new algorithms and techniques for Elastic Optical Networks. In this paper, we focus on ElasticO++ architecture, algorithms, and the features which we consider makes our framework unique. Additionally, we present a simulation case study and the newer features we are currently developing.**

***Index Terms*—Elastic Optical Networks; Fragmentation; Spectrum Defragmentation; Spectrum Allocation; RMSA; Algorithms; Simulation; OMNeT++.**

I. Introduction

In the past few years, Elastic Optical Networking (EON) emerged as the "next generation" core network technology [1]–[3], intended to surpass Wavelength-Division Multiplexing (WDM) weaknesses and limitations [4]–[6]. WDM is the most successful and widely used technology in the optical networks backbone by this date. However, in recent years Internet traffic in the core network has been doubling almost every two years, and predictions indicate that it will continue to exhibit exponential growth due to emerging applications such as high-definition and real-time video communications [1].

Traditional WDM-based networks offer the possibility to establish rigid connections (wavelengths) at fixed rates, using optical channels modulated with a fixed modulation format and spaced at a 50 GHz fixed grid. As a result, the process of upgrading/modifying the network to adapt to changing traffic and network conditions becomes challenging [4]–[7].

In contrast, EON relies on Optical Orthogonal Frequency Division Multiplexing (OOFDM) and advanced modulation technologies that enhance spectral efficiency and flexibility [1], [6], [8]. OOFDM allows the aggregation of multiple subcarriers to form super-channels, thus changing the paradigm of the network from fixed-size WDM channels to variable-sized EON channels that can reduce spectrum waste up to 60% [9].

Elastic Optical Networking concepts have been firstly introduced by [4], [9] to improve flexibility. The term "elastic" refers to this ability of the network to adjust its resources dynamically, according to the requirements of each connection. With EON it is possible to use the optical spectrum with a finer granularity and in a more flexible way than WDM, by creating "elastic channels"; i.e., that can accommodate both "sub-" and "super-" wavelength channels according to each requested demand. By adjusting the spectrum assigned to a demand for its needs, EON has the potential to improve the overall network capacity, regarding resource utilization. In addition, EON presents several other benefits such as high spectral and energy efficiency and elastic over time bandwidth variation [4], [9], [10]. In consequence of all those advantages, Elastic Optical Network is being considered as the technology for upgrading WDM networks [2], [3], [11].

Although several works have pointed EON benefits [4], [9], no technology is perfect, and the added efficiency and flexibility comes at the price of increased complexity and new problems, such as spectrum fragmentation losses [12], [13] and service unfairness [12], [14].

A considerable amount of work has been done on both fragmentation and unfairness problems, introducing a broad range of solutions, which raises the following question: "how to compare existent solutions and how to identify which one is better suited for the required scenario?" Usually, this is done whenever a new algorithm or technique is proposed, by comparing the proposal to some simple, well-known solution [15]–[18], or by a survey publication that usually brings a qualitative comparison [1], [7], [19], [20]. A major problem with both approaches is the difficulty in comparing published algorithms, particularly because of different simulation scenarios (e.g., network topologies, simulation setup) and, not rarely, missing information regarding simulation/testbed setup or other particular parameter [1], [7], [15]–[19].

In this context, the simulation framework ElasticO++ was first introduced in [21] with the objective to enable testing a whole range of routing, modulation, spectrum assignment, and defragmentation algorithms, parameters, and topologies. We believe this framework has the potential to become a useful tool to help other researchers in their research projects, especially newcomers. The framework provides a set of instruments that allow rapid implementation, testing, and analysis of new algorithms; and enables a common and well-controlled environment for comparing existing algorithms. In its current version, the framework comes with twelve traditional algorithms already implemented, which can be used standalone or in combination with new ones.

Our framework was built on top of the well-established discrete event simulator OMNeT++ [22] and as a consequence inherited the ability to run and extract data from a significant

R. S. Tessinari and A. S. Garcia are with LabTel, Federal University of Espírito Santo, Brazil (e-mail: {stange, anilton}@inf.ufes.br)

D. Colle is with IMEC, Ghent University, Belgium (e-mail: didier.colle@ugent.be)

The Ad Hoc Associate Editor coordinating the review of this manuscript and approving it for publication was Prof. Divanilson Rodrigo de Sousa Campelo.

Digital Object Identifier: 10.14209/jcis.2017.5

volume of simulations with a reduced workload for the users. This flexibility allied with the framework fragmentation and defragmentation capabilities are what we consider the most valuable strengths of our tool and what makes it unique. In this paper, we explore in details the flow and algorithms of Elastico++.

This paper is divided as follows: Section II presents some EON concepts, whereas the framework architecture and simulation validation/case study are presented in Section III and Section IV respectively. Section V concludes this paper.

II. Elastic Optical Networks

This chapter presents a succinct overview about Elastic Optical Networks (EON), focusing on the key points necessary to the understanding of this paper. For other topics not included in the following sections, following literature is recommended as a more extensive information regarding other EON aspects: basics concepts [9], [23], architecture and enabling technologies [4], [6], surveys [1], [7], and routing and spectrum assignment [19], [24].

A. Routing, Modulation, and Spectrum Assignment

One of the key challenges in Elastic Optical Networks is how to optimize resource utilization within the network. This optimization has the potential to decrease network cost and energy consumption as well [4], [25]. How to allocate resources among client's requests is defined as the Routing, Modulation, and Spectrum Assignment (RMSA) problem.

Historically, the RMSA subject in EONs evolved from the problem of calculating routes and wavelengths for connections in conventional WDM networks, called the Routing and Wavelength Assignment (RWA) problem. For each arriving request, the network needs to evaluate if there are enough free resources to attend that request. The network performs an *allocation algorithm* to search for resources, and if it finds, then the requisition is accepted by the network, thus creating a new connection and occupying those resources. If the allocation algorithm fails, the requisition is rejected. The minimum quantity of resources necessary to accomplish the received request is determined based on two factors: (i) by physical conditions along the lightpaths, such as signal-to-noise ratio (SNR), and linear and nonlinear distortions and (ii) the algorithms and heuristics used to optimize resource utilization [9]. Notice that restrictions do exist and must be respected to perform the spectrum assignment correctly: *spectrum continuity* constraint, and the *spectrum contiguity* constraint [26].

The spectrum continuity constraint is inherited from the wavelength continuity constraint in WDM networks. This constraint states that the same wavelength must be maintained along the whole path between both end-points associated with the connection (i.e., lightpath). In EONs, instead of the same wavelength, the restriction obliges the use of the same frequency range in the lightpath. Spectrum allocation in EONs introduces another constraint regarding the available spectrum on each fiber link along the end-to-end route. As each optical channel may use a different amount of bandwidth, frequency-overlapping becomes a possibility and needs to be taken into account [6].

Moreover, recent technology advances allow the use of adaptive modulation schemes, enabling more flexibility and efficiency, as best suitable modulation can be utilized according to physical layer conditions (e.g., fiber length or signal-to-noise ratio). For example, an efficient and dense format (e.g., 64-QAM) could be selected for short distances, and more robust and less dense format (e.g., QPSK) for longer paths [5], [27]. Modulation formats play a crucial role in the context of the RMSA problem as they translate between the bitrate requested by clients (usually in Gbps) and the bandwidth physically utilized to fulfill the demand (in GHz).

At this point, it is important to state different categories of the resource optimization problem. Two are the more common scenarios when discussing resource optimization: *static* (or *offline*) and *dynamic* (or *online*) scenarios [28].

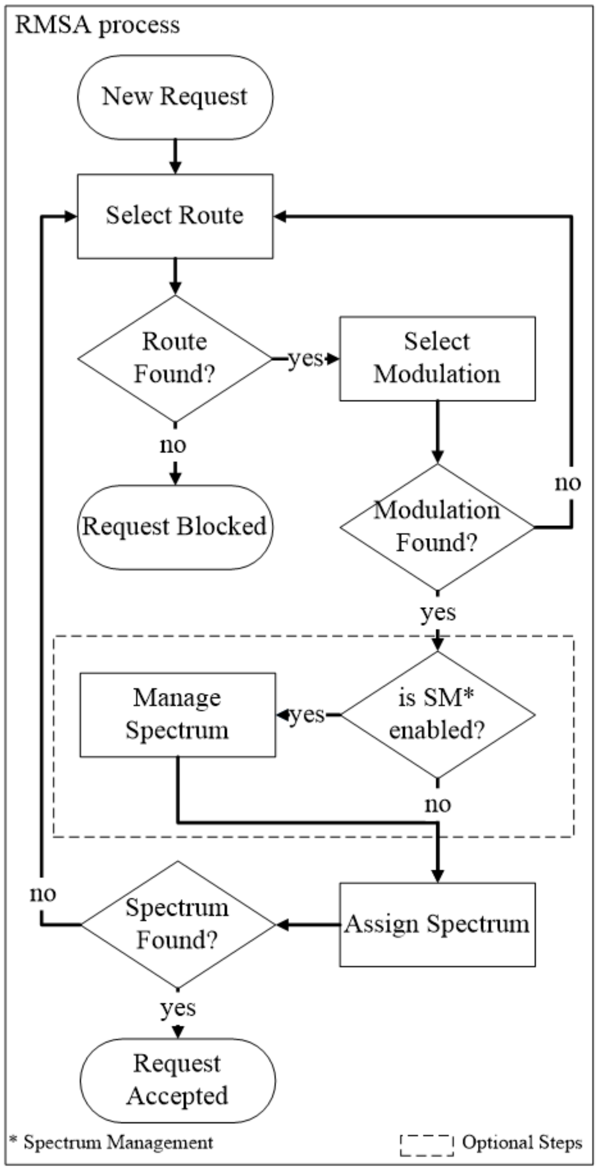


Fig. 1. An example of RMSA process.

In static scenarios all traffic demand is known beforehand, therefore it is possible to optimize resource utilization according to some metric (e.g., reduce the use of spectrum or smaller hop count average). As all demands are known, the optimization process is done offline and all calculation needed is done beforehand. As computational time is not an issue, optimization algorithms can take several minutes or even days to ensure optimized results. Usually, discussing the fragmentation problem in the static case is not very common since in general those solutions are typically used in the planning phase of the network, which is pretty much optimized. An exception is in pseudo-static scenarios, where multiple phases of allocation/release of resources are taken into account [29].

In the other hand, in dynamic scenarios, new requests arrive and leave randomly without any previous knowledge of the network. Provisioned connections end randomly as well. Because of the lack of information regarding traffic demand, all resource allocation must be done online, as quickly as possible while clients wait for an answer. Therefore, the processing time of algorithms cannot be too high, and as the RMSA problem is NP-Hard, usually heuristics are used [6], [30]. Because of this more unstable environment, it is easily visualized that fragmentation losses will happen more often than in static scenarios. Fragmentation problem is discussed in Section II-B.

It is important to notice that by resources, we are referring to a spectrum range. Currently, the minimum granularity of spectrum that can be assigned in an EON is a 12.5 GHz frequency range, and it is known as *frequency slot* or just *slot* [26]. This terminology is used in the rest of the text.

Fig. 1 illustrates an example of RMSA process in which the dashed rectangle represents optional steps. The RMSA process starts with a selection of a suitable route between desired source-destination nodes. After routing selection, it is necessary to define how much bandwidth (accounted in slots) is required to transport the requested bitrate. This definition is done by assigning a modulation format, according to link length, signal power, and other physical parameters. An optional spectrum management method can be used. Those management techniques usually consist of splitting the resources into partitions and establishing policies to use it. For example, some policies allocate only demands with the same "total path length" [31] or "number of slots" [32] in the same partition. Finally, the last step in the RMSA process consists of finding the necessary number of contiguous slots on the links belonging to the selected route.

B. Fragmentation/Defragmentation

In a dynamic network scenario, incoming requests are established and released in an entirely random fashion. This randomness induces spectral resources to be highly fragmented and consequently, "gaps" are unavoidably introduced leading to the so-called spectrum fragmentation problem, thus degrading spectrum utilization efficiency [33].

When connections are uniform regarding spectrum width, as in WDM networks, fragmentation is caused by the wavelength continuity constraint. Though there are wavelengths available on every link that constitute the path, there may not be a commonly available one on all links. This lack of continuity is called *inter-link* fragmentation and exists in EON as well.

In addition to the continuity constraint, EON also experiences the contiguity constraint. This contiguity constraint in association with a heterogeneous environment leads to the second type of fragmentation, the *intra-link* fragmentation. In general, defragmentation algorithms try to reduce the intra-link fragmentation effect [33].

Fragmentation losses lead to inefficient resource utilization and overall network performance degradation, increasing blocking probabilities as the unused slots remain scattered over the links and not enough contiguous spectrum slots may be available for new requests to be established. Several spectrum defragmentation techniques have been developed to prevent performance degradation. The primary goal is to rearrange existing connections in a way that available slots remain continuous, clearing more space for new incoming requests, thus reducing blocking probabilities. Four main spectrum defragmentation techniques are proposed in the literature: (a) reoptimization technique [34], (b) make-before-break [27], (c) push-and-pull technique [35], and (d) hitless technique [36] / hop tuning [37].

In addition to the Defragmentation Algorithm, once a defragmentation is requested, it is recommended to evaluate if it is indeed required in order to prevent unnecessary operations that, depending on which algorithm is used, may be very costly (e.g., interrupting connections for a period). This prevention can be achieved by evaluating a Fitness Function, such as "Spectrum Compactness" [38] or "Utilization Entropy" [39]. Fig. 2 is an example of this defragmentation evaluation process described.

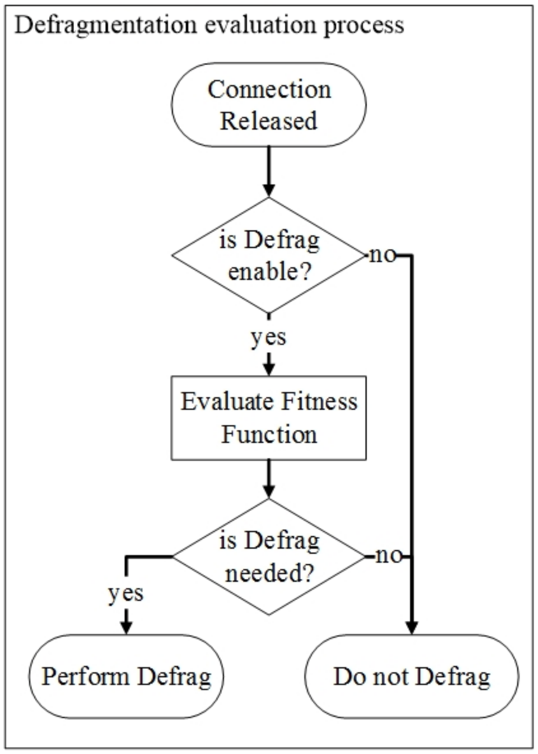


Fig. 2. An example of defragmentation evaluation process.

III. Framework Overview

The development of this work is driven to fulfill a gap found in academia. The proposed framework is aimed to simplify the creation and testing of new algorithms, by reducing the amount of time consumed with non-critical but yet necessary tasks, such as logging, data handling, and chart generation. Our intention is that new users could use the framework as a "black box," thus focusing on the current problem and its solution. We developed the framework based on the following requisites:

R1. Perform Elastic Optical Networks simulations;
R2. Support fragmentation and defragmentation;
R3. Allow easy test of algorithms;
R4. Be flexible and easily upgraded if required;
R5. Capable of running batch simulations;
R6. Support of charts and statistics generation;
R7. Free software publicly available[1].

ElasticO++ is heavily based on OMNeT++[2] [22]. OMNeT++ is a discrete event simulator created in C++ for modeling communication networks, multiprocessors, and other distributed or parallel systems; and is one of the most popular simulators in the research area of communication networks. OMNeT++ is open-source and can be used under the Academic Public License that makes the software free for non-profit use.

A. Related Work

Before starting the development of the proposed framework, we searched for simulation tools capable of allowing implementation and testing of new algorithms for elastic optical networks, focusing on allocation and defragmentation scenarios. Since EON technology is the direct evolution of the WDM networks, and as mentioned in Section II-A, the RSMA problem has its roots in the RWA problem, we started our search by looking for WDM simulation tools. From the myriad of simulation tools found, some solutions focused on the RWA and grooming problems [40], [41], whereas others take in consideration more complex physical layer modeling [42]. Unfortunately, those simulators were not updated to support Elastic Optical Networks.

Regarding EON simulators, [43] is capable of working both with the RWA and the RMSA problems, as well the "Net2Plan: The open-source network planner" [44]. In addition to algorithm testing, Net2Plan also has a network planner feature. In total it provides four different tools: *offline network design*; *traffic matrix generation*; *online simulation*; and *reporting*, which permits generation of user-defined logs. Although Net2Plan was initially developed to simulate WDM Networks, it is being updated to work with EON as well. At last, the Complex Elastic Optical Network Simulator (CEONS) [45] is a test environment aimed to address three EON problems: Routing and Spectrum Assignment (RSA) problem; Routing, Modulation, and Spectrum Assignment (RMSA) problem, and the regenerator placement problem. Unfortunately, besides the short paper [45], there is not much information available online.

[1]https://bitbucket.org/Stange/elastico/
[2]https://omnetpp.org/

All those works have one characteristic in common: they are all simulators by themselves. Those tools were created from scratch, forcing the developers to focus on the technology specific models (e.g., algorithms, and equipment) but also in the inner works of the simulator itself (e.g., event queuing and scheduling, random number generation, and object addressing). A different option is using a discrete event simulator as a base and then build extensions to it. This approach has the advantage of free the developers from all labor associated with the simulator itself, allowing them to focus on the problem. This method was taken by "An elastic networks OMNeT++-based simulator" [46], which consists of a framework for OMNeT++ software. The framework is focused on simulating scenarios for testing algorithms and network architectures. According to the authors, the tool is capable of simulating an active Path Computation Element (PCE) controller for Elastic Optical Networks, and is configurable, allowing implementing and testing new algorithms and architectures [46]. However, the software is not publicly available for download. Our solution follows the same approach, using OMNeT++ as starting point.

Since the gap still exists, we decided it was worthy to invest time and effort to create a new framework, specialized enough to handle specific details related to Elastic Optical Networks, but sufficiently flexible to be modified and used by other fellow researchers. Section III-B describes the architecture and main features, which we believe makes our framework unique.

B. Architecture Description

The proposed framework is divided into three main parts: *Extra Tools*, *Equipment*, and *Controller* (Fig. 3). Extra Tools is a set of scripts created to automate time demanding tasks, such as data handling and chart generation. Two main features are worth a highlight: *i) CSV (Comma Separated Values) Handler script* is responsible for aggregating each simulation result and preparing it to be used by the *ii) Chart Generator Script*. Chart Generator Script reads organized data stored in .csv files and creates ready-to-plot files, with statistics such as standard deviation, variance, and 95% confidence interval. Currently, the framework supports three "chart outputs formats": Microsoft Excel, OriginLab Origin, and Matplotlib.

Equipment represents a simple implementation of some conventional equipment found in an optical network, such as transponders, Reconfigurable Optical Add-Drop Multiplexer (ROADM), and optical fiber. We created a C++ class to

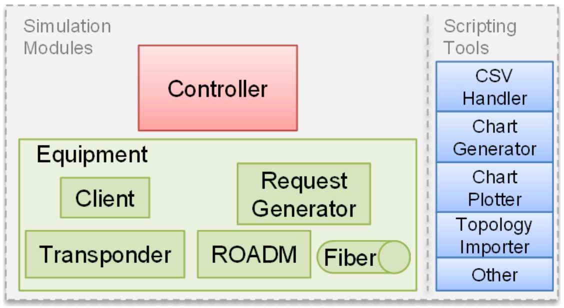


Fig. 3. Framework architecture description.

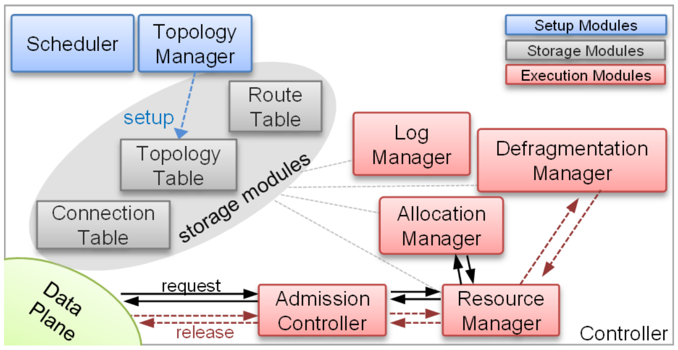


Fig. 4. Controller modules.

represent an optical fiber. In this fiber class are included parameters such as length and attenuation, which together influence the "Input/Output power" ratio. those parameters can be used by the *ModulationScheme* class (Section III-C) to take decisions regarding modulation formats. Besides those two characteristics, in the current version, no other physical layer characteristic is used. Also, it is worth mentioning the *Request Generator* module, in which is possible to configure different traffic patterns to be utilized in the simulation.

Finally, the most important part of ElasticO++: the *Controller*. It is composed of several modules as shown in Fig. 4. Those modules can be categorized into three groups: *setup*, *storage*, and *execution*. Setup modules are activated only at the beginning of the simulation to coordinate other modules (e.g., *Scheduler module*), and to detect network topology and resources (e.g., *Topology Manager*). Topology Manager stores all information collected in *Topology Table* module for later use by execution modules. In addition to the adjacency matrix, the Topology Table also stores a matrix of Links. A Link is a C++ class created to represent network resources. Each Link has attributes that characterize the optical fibers, including a spectrum state representation. Since version v0.1.70 we utilize strings of 0s and 1s, with each character meaning a frequency slot and its occupation state. This method reduced code complexity and allowed algorithms to use standard C++ string operations, such as find() when looking for resources. *Route Table* and *Connection Table* complete the storage group, keeping, respectively, routes found during Setup Phase and information regarding active connections during the Execution Phase.

Modules involved in the Execution Phase of the simulation forms the third and last group. We explain these modules through an example. When a new request (solid black line in Fig. 4) arrives at the controller, it is received by the *Admission Controller*. The Admission Controller is the interface between equipment and other controller modules, and its primary job is to convert the format of received information to a "common data structure" understood by all other modules of this group. This data structure is particularly interesting as it greatly improves the flexibility of the framework, simplifying the process of adding new parameters to the framework, thus removing the need to modify function call between Controller Modules. New

```
struct AllocationRequest {
public:
  int source;
  int destination;
  int bitrate;
};

struct AllocationRestriction {
public:
  int lower_slot_index;
  int higher_slot_index;
  int num_slots;
  int route_index;
  int route_id;
  double route_longest_link_length;
};

struct AllocationResult {
public:
  string allocation_log;
  long int connection_id;
  int lower_slot_index;
  string modulation;
  int number_of_slots;
  bool rejection_after_defragmentation;
  int rejection_cause;
  int route_id;
  bool success;
  bool success_after_defragmentation;
};

struct AllocationConfiguration {
  int route_k;
  ConnectionTable* connection_table;
  RouteTable* route_table;
  TopologyTable* topology_table;
};
```

Fig. 5. Controller common data structures associated with allocation.

parameters could be added to create new and more complex algorithms. Those data structures also guarantee backward compatibility as long as previously implemented parameters are not removed. As Fig. 5 shows, there are at the moment four different data structures related to the allocation process used by controller modules: AllocationRequest, AllocationRestriction, AllocationResult, and AllocationConfiguration.

Still, the Admission Controller forwards the request to the *Resource Manager* module, which is responsible for coordinating the Allocation Manager and the Defragmentation Manager, create/remove connections from the Connection Table, and call the Log Manager to create simulation logs. The *Log Manager* module is responsible for maintaining statistics and log reports, and in the current version of the framework is capable of generating a simulation result (in a human understandable format) and generating a .csv version to be read by module *CSV Handler* previously mentioned. Next, the *Allocation Manager* is responsible for assembling, managing, and executing *allocation algorithms* (Section III-C), runs an algorithm to try to find resources to attend the request. Once the allocation algorithm runs, it updates the AllocationResult object and returns it to Resource Manager. At this point the AllocationResult can carry a success or a fail, in both cases, the Resource Manager will activate the Log Manager to update statistics with this particular request and result, and then, only in the event of success, it will update the Connection Table,

```
class RoutingClass {
public:
  assignRoute( ) {
    /* calculate route(s) */
  };
}

class SpectrumAssignmentClass {
public:
  assignSpectrum( ) {
    /* select a suitable frequency slot */
  };
}

class RsaClass {
private:
  RoutingClass* route_algo;
  SpectrumAssignmentClass* spectrum_algo;

public:
  assignResources( ) {
    route = route_algo->assignRoute( );
    slot = spectrum_algo->assignSpectrum( );
    /* return result */
  };
};
```

Fig. 6. An example of algorithm classes.

thus creating a new connection. At last, an answer is returned to the requesting client informing either a connection ID or rejection.

The release process (dashed red line in Fig. 4) work in a very similar way to the admission process. The exception is when the release request reaches the Resource Manager module it is not forwarded to the Allocation Manager. Instead, the connection is terminated and optionally the *Defragmentation Manager* module is called if defragmentation is enabled in the .ini file. After this process, the same way as Admission Process, Log Manager is called, statistics are recorded, and an answer is returned to the requesting client informing the termination of the requested connection.

C. Algorithms

We developed an architecture that focuses heavily in algorithm re-utilization. The idea is to create, in a distinct way, smaller algorithms to handle specific problems and then put them together using C++ pointers and inheritance. Both allocation and defragmentation algorithms are formed by two different types of classes: specific classes, used to achieve the desired behavior (e.g., routing) and base classes used for connecting those specific classes altogether.

As an example, in the context of EONs, a basic Allocation Algorithm could be an RSA (Routing and Spectrum Assignment) algorithm. Such algorithm would be responsible for determining a suitable route (R) and a frequency slot (SA) for each new request in the network. This RSA algorithm could be implemented in ElasticO++ by three classes in total: two specific classes, one routing class (e.g., Dijkstra) and one spectrum assignment class (e.g., Fist Fit), and one base class responsible for connecting those two previous classes. Fig. 6 illustrates the concept.

Fig. 7 presents in more detail how the allocation algorithms are implemented in the current version of the framework. We implemented five abstract classes, one base class (*Allocation-Algorithm*) that is used internally in the AllocationManager module as an interface to any algorithm implemented, allowing the use of different logic for the allocation process. The abstract class AllocationAlgorithm contains two methods: assignResources() and setup(). The method assignResources() is called when a new request for resources arrives in the network, whereas the setup() method is used to provide the information needed to other specific classes (e.g., pointers to storage tables or even parameter values, such as the K value from the K-Shortest Paths algorithm). The other four abstract classes implemented are *RouteAlgorithm*, *ModulationScheme*, *SpectrumAlgorithm*, and *SpectrumManagement*. Each class has two basic methods that are needed to be implemented: *setup()* and a class-specific assign method (e.g., *assignRoute()* for the RouteAlgorithm class).

The names of the classes are related to its functionality, in that manner, a RouteAlgorithm class is responsible for obtaining routes. It can be done *online*, i.e., calculating the route for request by evaluating the network current status, or *offline*, during a setup phase of the network, when routes are calculated and stored in the RouteTable module. In the latter case, during a new request, the algorithm simply select a route previously calculated from the RouteTable. The ModulationScheme class applies the modulation format to the request, given the characteristics of the chosen route, providing a “translation” between the demanded bitrate (usually in Gbps) and the bandwidth physically utilized to fulfill the request (in GHz or frequency slots). The SpectrumManagement class is used to divide the spectrum into partitions. At least, the SpectrumAlgorithm class is responsible for searching a range of free slots large enough to accommodate the arriving request.

When creating a functional algorithm, it is necessary to inherit from one of the abstract classes and implement the virtual methods mentioned. Implementing the virtual methods is a necessary step since the AllocationManager (Fig. 4) relies on C++ template concept to work, and uses pointers to call the AllocationAlgorithm setup() and assignResources() methods when needed. In this sense, when a user of the framework wants to create a new routing algorithm, he must first inherit it from the RouteAlgorithm class, and so forth.

At this point, it should be noted the importance of the common data structures mentioned in Section III-B and its synergy with this algorithm design. The methods of all classes related to the allocation process receive the same structures as parameters, allowing significant flexibility, as new parameters can be implemented to be used by newer algorithms. For example, a new parameter related to the quality of the service could be added in the AllocationRequest structure, and without any other coding, be visible to all algorithms and controller modules. Finally, Fig. 8 presents the RMSA allocation algorithm class implemented and illustrates the described concepts.

We opted for this design as it improves code re-utilization and provides flexibility since it is easy to match different combinations of algorithms without writing extra code. Once

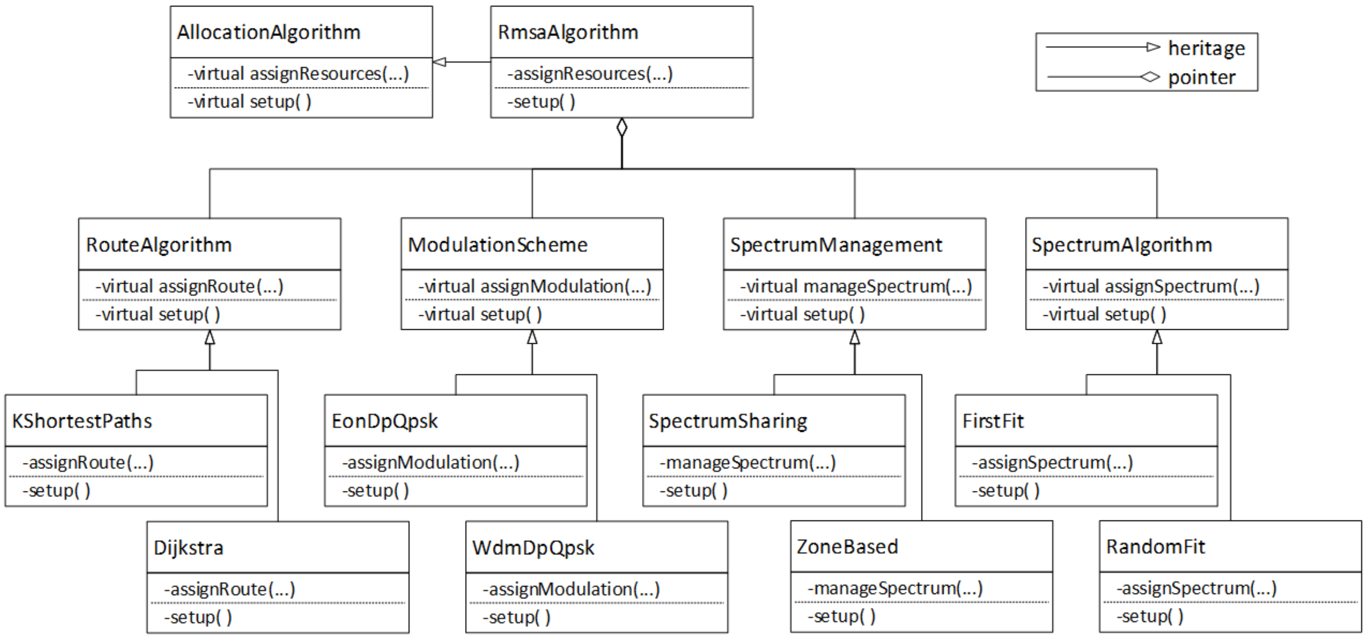


Fig. 7. Relationship of classes related with the Allocation Algorithms.

```
class RsmaAlgorithm : public AllocationAlgorithm
{
 protected:
   ModulationScheme* modulation_scheme_;
   RouteAlgorithm* route_algorithm_;
   SpectrumAlgorithm* spectrum_algorithm_;
   SpectrumManagement* spectrum_management_;:

 public:
   AllocationResult& assignResources(
      AllocationResult& result,
      const AllocationRequest& request,
      AllocationRestriction& restriction);

   void setup (AllocationConfig& config);
 };
```

Fig. 8. The class RmsaAlgorithm, used in all tests performed in Section IV.

an algorithm is implemented, it can be used with any other algorithms, e.g., a Spectrum Partition based algorithm can be used in combination with a First Fit or Random Fit algorithm, with any routing algorithm, and any modulation scheme. In the current version of the framework, eight algorithms related to resource allocation are implemented, as Table I presents.

Defragmentation Algorithms follow the same logic, but instead of five base classes, it uses three: an algorithm class, a fitness function class, and an interface class. Similarly to allocation algorithms, all defragmentation algorithms also share common data structures to transmit information and results between classes. There are in the current version two structures: DefragmentationRequest and DefragmentationResult, both represented in Fig. 9. Finally, Table II provides a summary of the defragmentation algorithms implemented in the latest version of the framework. Those algorithms are in-house implementations of the algorithms found in the literature.

```
struct DefragmentationRequest {
  long connection_id;
  int lower_slot_index;
  int num_slots;
  long route_id;
  int situation;
};

struct DefragmentationResult {
  string defragmentation_log;
  int moved_connections;
  bool success;
};
```

Fig. 9. Controller common data structures associated with defragmentation.

More information regarding the ElasticO++ framework, including installation tutorial, screens, and documentation, can be found at the project website[3].

IV. Simulation

The objective of this section is to present some of the capabilities and flexibility of the developed framework. Notice that the results of previous works [21], [32], [47] were obtained through the use of ElasticO++. Also, in [47] is presented comparisons of simulations performed using ElasticO++ and the original publications, in addition to some reflections about simulation and result comparison in our research field in general.

For this case study, we established a "baseline" algorithm without defragmentation. We then chose four other algorithms each one being a variation of the baseline. For the last comparison, we enabled the defragmentation algorithm to work

[3]https://bitbucket.org/Stange/elastico/.

TABLE I
ALLOCATION ALGORITHMS IMPLEMENTED.

Acronym	Name	Type
DIJK	Dijkstra [48]	Routing
KSP	K-Shortest Paths [49]	Routing
EDQPSK	EON DP-QPSK [9]	Modulation
WDQPSK	WDM DP-QPSK [9]	Modulation
SS	Spectrum Sharing [12]	Spectrum Management
ZB	Zone Based [32]	Spectrum Management
FF	First Fit [5]	Spectrum Assignment
RF	Random Fit	Spectrum Assignment

TABLE II
DEFRAGMENTATION ALGORITHMS AND VARIATIONS IMPLEMENTED.

Acronym	Name	Type
MBB	Make-Before-Break [27]	Defragmentation
PP	Push-Pull [50]	Defragmentation
AO	Always ON	Fitness Function
SC	Spectrum Compactness [38]	Fitness Function
ACR	After Connection Release	Policy
ARR	After Request Rejection	Policy
AN	All Network	Scope
RO	Route Only	Scope

with the baseline algorithm. More about the algorithms is covered in Section IV-A.

A. Setup and Algorithms

Simulations were performed using American topology NSFNET[4], composed of 14 nodes and 42 links. In this test, we configured 360 slots[4] ($360{\cdot}12.5$ GHz $= 4.5$ THz) as maximum bandwidth available per fiber (C band).

A dynamic network operation scenario is simulated with new requests arriving at λ Poisson rate and holding time exponentially distributed (with a normalized mean of $1/\mu = 1$). Network load is given by $\rho = \lambda/\mu = \lambda$ (Erlang). In each simulation run, 10^6 requests[4] are generated, and for each chart point, 30 simulations[4] with different random seeds. Each new request is composed of a source, destination, and bitrate requirement; following uniform distribution[4]. In this particular scenario, four services[4] are provided by the network with bitrates of 40 Gbps, 100 Gbps, 400 Gbps, and 1 Tbps[4]. For each arriving request the controller evaluates if the network has enough resources available, as shown in Fig. 1.

For this case study, we established a "baseline" allocation algorithm (Algo-0). This baseline algorithm is a combination of the following base algorithms: Yen K-Shortest Paths [49] as the routing algorithm, EON DP-QPSK [9] as the modulation scheme, Spectrum Sharing [51] as the management technique, and First Fit [5] as spectrum assignment algorithm. The K-Shortest Paths algorithm has no particularities, besides the use of K=5 in all tests. Regarding the modulation scheme, as mentioned in Section III-B, the physical layer is implemented by the optical fiber class. Although some parameters (e.g., fiber length) are available to this EON DP-QPSK class, this particular implementation does not make use of it, thus following Table III

[4]These values are configurable and can be changed as desired.

TABLE III
RELATIONSHIP BETWEEN THE REQUESTED BITRATE AND THE NUMBER OF SLOTS NEEDED TO ALLOCATED THE CONNECTION IN THE NETWORK [9].

	EON-DP-QPSK		WDM-DP-QPSK	
Requested Bitrate (Gbps)	Bandwidth (GHz)	# slots	Bandwidth (GHz)	# slots
40	25 + 10	3	1 x 50	4
100	37.5 + 10	4	1 x 50	4
400	75 + 10	7	4 x 50	16
1000	190 + 10	16	10 x 50	40

TABLE IV
CASE STUDY ALLOCATION ALGORITHMS

Algo-#	Description
Algo-0	K-Shortest Paths EON DP-QPSK Spectrum Sharing First Fit
Algo-1	*Dijkstra* EON DP-QPSK Spectrum Sharing First Fit
Algo-2	K-Shortest Paths *WDM DP-QPSK* Spectrum Sharing First Fit
Algo-3	K-Shortest Paths EON DP-QPSK *Zone Based* First Fit
Algo-4	K-Shortest Paths EON DP-QPSK Spectrum Sharing *Random Fit*
Algo-5	K-SP EON DP-QPSK Spectrum Sharing First Fit + *Spectrum Compactness Make-Before-Break*

values. The Spectrum Sharing (SS) spectrum management algorithm is a policy that does not restrict resource access between services, i.e., any division nor spectrum partitioning is done. At last, the First Fit (FF) is often used as a benchmark in publications of new assignment algorithms. FF is a good overall solution regarding blocking rates and with a low computational cost associated.

We compare Algo-0 against four other algorithms: each one changing one of the four base algorithms, as presented in Table IV. Algo-1 uses Dijkstra Shortest Path algorithm [48] instead of K-Shortest Paths. Algo-2 is a representation of WDM technology using fixed grid of 50 GHz according to [9]. In practice, the difference of both schemes in this implementation is the number of slots used to attend each service type. As shown in Table III, EON technology is much more efficient than WDM in some scenarios because of its capability to form super-channels [52]. The extra 10 GHz in the EON-DP-QPSK column refer to the guard band between channels.

Algo-3 uses the Zone-Based partitioning (ZB) [32] as Spectrum Management algorithm. The idea the technique is to divide the spectrum into a set of smaller and homogeneous environments, which can hold only similar services (i.e., with same bandwidth requirements), ensuring each partition has the same capacity (i.e., can accommodate the same maximum number of connections at a given time). The Zone-Based algorithm differs from the Spectrum Sharing technique, which makes no division whatsoever. Finally, Algo-4 uses Random Fit spectrum assignment algorithm as a variation of the First Fit algorithm.

Three metrics are used to evaluate algorithms performance in following tests, requests blocked rate (RBR), bitrate blocked rate (BBR), and service fairness. RBR is defined as

$$RBR = R_b/R_t$$

where R_b represents the number of requests blocked at the end of the simulation and R_t the total number of requisitions

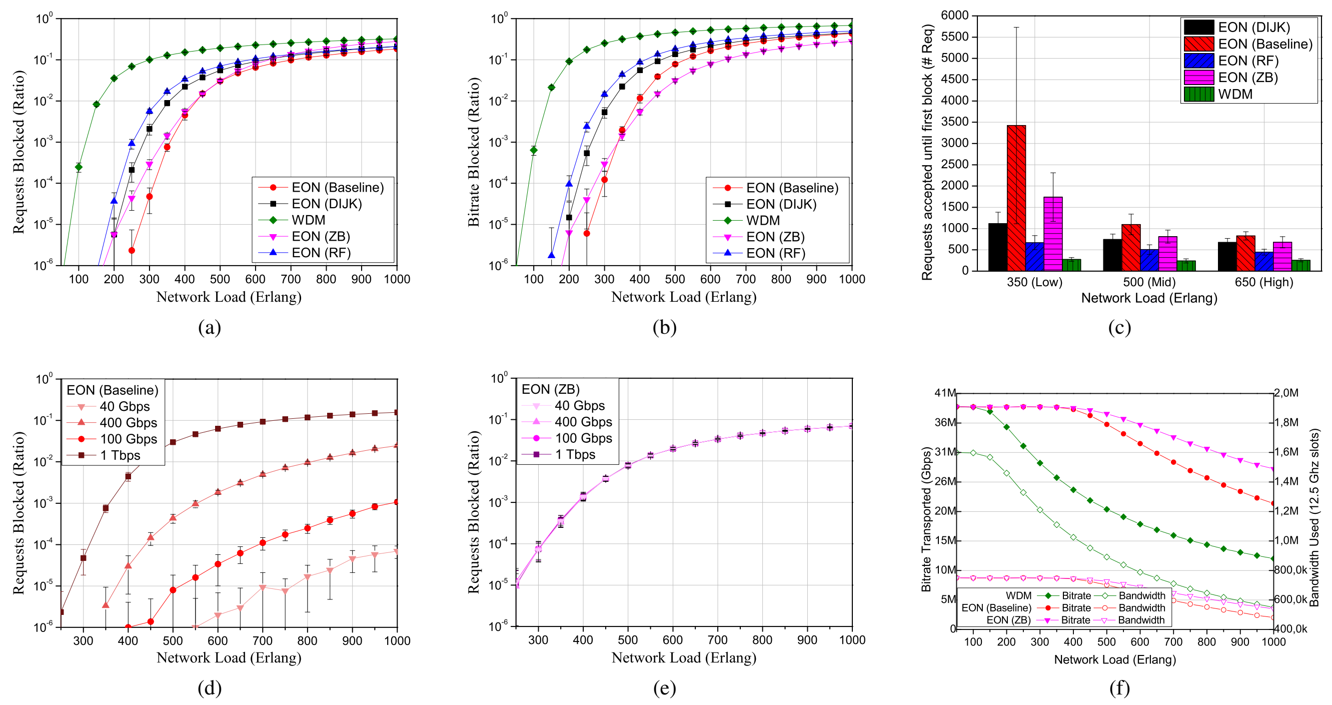


Fig. 10. Allocation results: (a) Requests blocked ratio (RBR); (b) Bitrate blocked ratio (BBR); (c) Number of requests accepted before the first block; Fairness between services: Baseline Algo-0 (d), and Zone-Based Algo-3 (e); (f) Relationship between bitrate transported and bandwidth used in the process.

generated. Likewise, the BBR is given by

$$BBR = B_b/B_t$$

where B_b is the total bitrate blocked, and B_t is the total bitrate requested. Finally, the service fairness is evaluated by comparing the service requests blocked rates (RBR_{Sti}) between all service types. RBR_{Sti} in defined as

$$RBR_{Sti} = R_{bSti}/R_t$$

where R_{bSti} stands for the number of requests blocked of service type St_i. The greater is the difference between the blocked rates the more unfair is the algorithm in the scenario analyzed.

We also run simulations combining Algo-0 with enabled defragmentation, using Make-Before-Break [27] as Defragmentation Algorithm and Spectrum Compactness as Fitness Function [38]. Simulation results are presented in Section IV-B.

B. Result Analysis

As our intention with this paper is to present the Framework capabilities, the results presented here are already known by the community. As we do not commit to presenting any new result we ask the reader to focus instead on the tools available to analyze the results. Fig. 10 and Fig. 11 present all results obtained from the case study.

Fig. 10 (a) and Fig. 10 (b) present requests and bitrate blocked ratio, respectively. Both charts can be used to evaluate the general behavior of tested algorithms. It can be seen that all EON-based algorithms outperform the WDM-based. Also, K-Shortest Paths (K = 5) outperforms Dijkstra Shortest Path algorithm; and First Fit outperforms Random Fit. Comparing BBR curves it is noticiable that when increasing the network load, the Zone-Based becomes more efficient than Spectrum Sharing, reducing the total bitrate blocked. Fig. 10 (c) shows the number of accepted requests before first rejection in the network. Same pattern from (a) and (b) is observed.

Fig. 10 (d) and (e) show fairness for baseline algorithm and Zone-Based variant, respectively. Fairness is represented by the difference between blocking ratios for all services in the network. In charts (d) and (e) each curve accounts for a different service, and the closer the curves are, the fairer is the Allocation Algorithm. As expected, Zone-Based outperforms Spectrum Sharing as it is designed to increase fairness in the network.

Fig. 10 (f) is a doubled Y-axis chart utilized for analyzing spectrum efficiency of tested algorithms. Left y-axis represent the total bandwidth used (curves with solid shapes) to transport the total of bitrate requested (curves with blank shapes). At lower loads, WDM transporting the same amount of bitrate at a cost of much higher bandwidth, while comparing with EON-based technologies. EON (SS) and EON (ZB) performs equally for lower loads. When the network load increases, WDM starts to block much more than the other two algorithms, and both WDM curves decrease proportionally. It is interesting to notice that this difference between WDM and EON is expected since EON is indeed more efficient because the use of OOFDM. However, Fig. 10 (f) also shows that a spectrum management technique can increase the efficiency as well. Indeed, the enhanced efficiency of EON (ZB) can be seen on the right side of the chart, where EON (ZB) and EON (SS) consume almost the same bandwidth while EON (ZB) transport more bitrate than EON (SS).

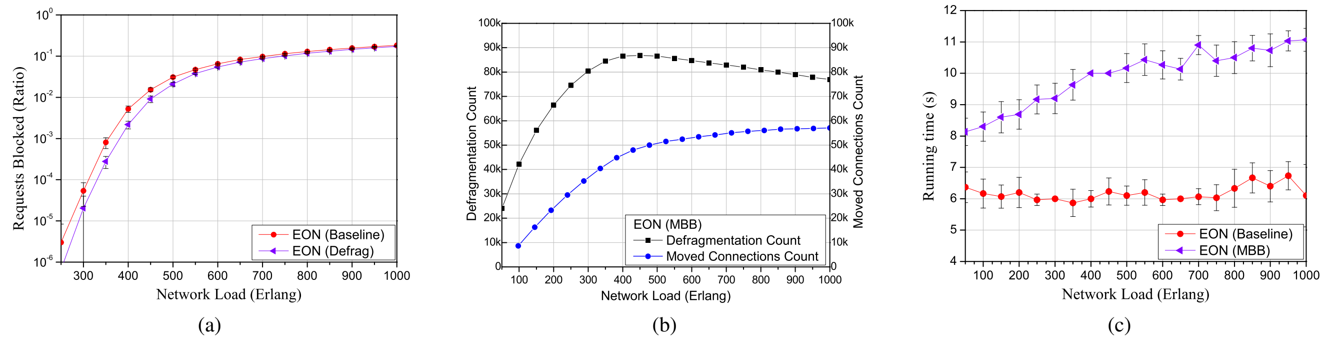


Fig. 11. Defragmentation results: (a) Comparison between baseline algorithm with and without defragmentation; (b) Relationship between Defragmentation count and the total number of connections moved in the process; (c) Average simulation time.

In Fig. 11, charts (g), (h), and (i) show results related to the use of a Defragmentation Algorithm. Chart (g) is the same chart as (a) but showing that some gain can indeed be achieved by using a Defragmentation Algorithm. As expected, defragmentation increases the performance of the network, by allowing more requests to be accepted. Chart (h) shows how many times the Defragmentation Algorithm was activated (left y-axis, the black curve with squares), and the number of moved connections (right axis, the blue curve with circles). Chart (h) can be used to measure how well the fitness function and the defragmentation activation threshold perform, by observing if unneeded defragmentation activation occurs (e.g., small amount of moved connections). It is important to notice that the "moved connections count" metric can be used by RMSA algorithms as well, representing the number of connections disrupted during allocation of new requests. The number of disrupted connections can be an important benchmark for those algorithms as it may impact overall network quality of service. We can observe that when the load increases, there is not more free space to allocate new connections, thus saturating the blue curve. At last, chart (i) presents some metric about simulation performance. Although it is not a very precise measurement method, it serves well to observe that simulations without defragmentation have simulations time on the same plateau, taking around six seconds to complete; while simulations using Defragmentation Algorithms tend to need more time as the network load is increased. That happens because the network is more used and by consequence gets more fragmented. As defragmentation algorithms are very demanding, simulation time increases (up to 60% in this tested scenario).

C. Future Work

During the dozens of tests performed during the development of the presented framework and algorithms, we identified a potential problem concerning dynamic network simulation scenarios. Consider the scenario where the simulation provides same traffic pattern throughout the time. In this case, the values of blocking rates measured at the end usually give a good idea of what happened during the simulation since those rates tend to a stable value. Fig. 12 illustrates this idea. However, when the traffic pattern changes over the simulation, the observed final result may not be significant to characterize what happened during the simulation. This can potentially induce misleading conclusions. Notice that if the simulation in Fig. 13 stopped before the last two-thirds of events, both curves SZB and DZB would get very similar results.

In both Fig. 12 (a) and (b), the X-axis represents the simulation evolution using the number of requests as the unit. The Y-axis in (a) accounts for the request blocked ratio experienced during this "step" (i.e., R requests in the simulation). After R requests, the monitored data is reset, and the accounting for the next step starts. The Y-axis in (b) shows the "average" values accumulated up to that point in the simulation. Fig. 13 follows the same logic of Fig. 12.

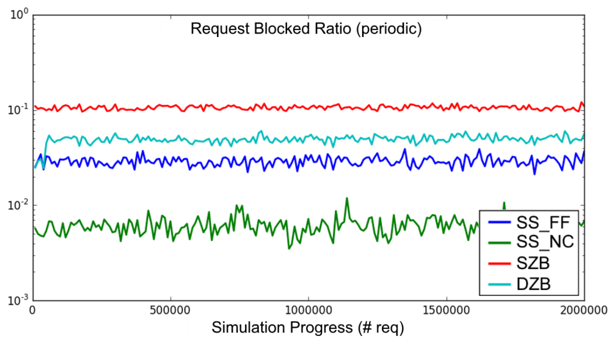
All charts in presented Fig. 12 and Fig. 13 were created by the under-development *Simulation Progression Monitoring* feature of ElasticO++ and plotted automatically by the framework, using Matplotlib [53]. This feature allows monitoring the behavior of the chosen metrics throughout the simulation. At the current version of the framework, the resulting chart represents only one simulation run, and as a consequence can not be used to infer any quantitative conclusion. We intend to keep working on this feature and exploring more results in this area.

V. Conclusion

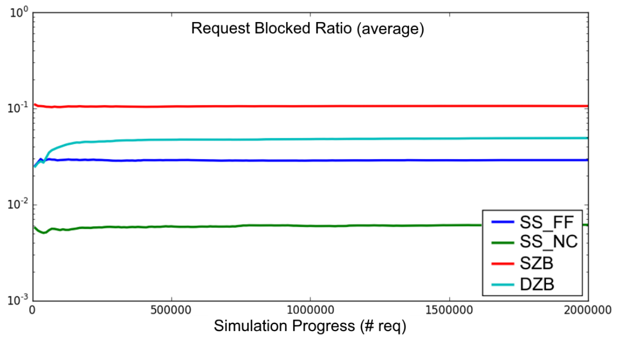
In this paper we explore in details the framework ElasticO++, a simulation framework that enables rapidly testing of Allocation and Defragmentation Algorithms, using a different set of parameters and topologies. To the best of our knowledge, our framework is the first non-commercial software available capable of handling fragmentation and defragmentation scenarios. We believe this framework has the potential to grow and become a useful tool to help other fellow researchers in their research projects, providing a set of instruments to implement rapidly, test, and analyze results for new algorithms, as presented in Section IV.

As future works, we intend to implement a more robust physical layer model, which will improve the quality of the simulation results. We also plan the implementation of new algorithms, giving especial attention to Defragmentation Algorithms. We also plan to keep developing the *Simulation Progression Monitoring* feature mentioned in Section IV-C.

More information regarding the ElasticO++ framework, including installation tutorial, screens, and documentation, can

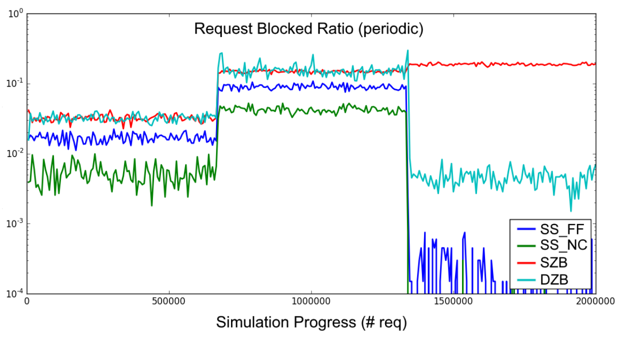


(a)

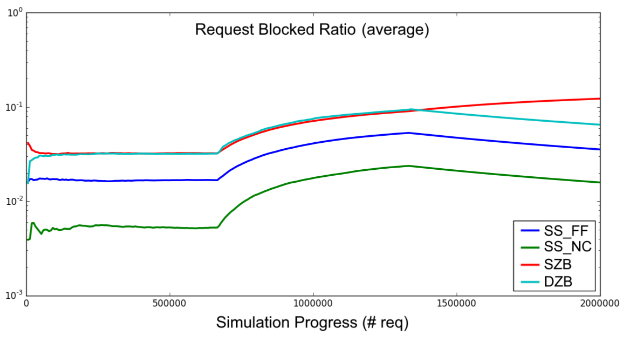


(b)

Fig. 12. Results of a simulation with a static traffic pattern over its duration. (a) Periodic values of request blocked ratio (b) Average values of request blocked ratio.



(c)



(d)

Fig. 13. Results of a simulation with a dynamic traffic pattern over time. (a) Periodic values of request blocked ratio (b) Average values of request blocked ratio.

be found at the project website: https://bitbucket.org/Stange/elastico/.

VI. Acknowledgement

This work was funded in part by CAPES Foundation, Ministry of Education of Brazil.

References

[1] G. Zhang, M. De Leenheer, A. Morea, and B. Mukherjee, "A survey on ofdm-based elastic core optical networking," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 1, pp. 65–87, 2013, doi: 10.1109/SURV.2012.010912.00123.

[2] A. Mayoral, V. Lopez, G. de Dios, and J. P. Fernandez-Palacios, "Migration steps toward flexi-grid networks," *Optical Communications and Networking, IEEE/OSA Journal of*, vol. 6, no. 11, pp. 988–996, 2014, doi: 10.1364/JOCN.6.000988.

[3] I. Stiakogiannakis, E. Palkopoulou, D. Klonidis, O. Gerstel, and I. Tomkos, "Dynamic cooperative spectrum sharing and defragmentation for elastic optical networks," *Optical Communications and Networking, IEEE/OSA Journal of*, vol. 6, no. 3, pp. 259–269, 2014, doi: 10.1364/JOCN.6.000259.

[4] M. Jinno, H. Takara, B. Kozicki, Y. Tsukishima, Y. Sone, and S. Matsuoka, "Spectrum-efficient and scalable elastic optical path network: architecture, benefits, and enabling technologies," *Communications Magazine, IEEE*, vol. 47, no. 11, pp. 66–73, 2009, doi: 10.1109/MCOM.2009.5307468.

[5] M. Jinno, B. Kozicki, H. Takara, A. Watanabe, Y. Sone, T. Tanaka, and A. Hirano, "Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network [topics in optical communications]," *Communications Magazine, IEEE*, vol. 48, no. 8, pp. 138–145, 2010, doi: 10.1109/MCOM.2010.5534599.

[6] K. Christodoulopoulos, I. Tomkos, and E. Varvarigos, "Elastic bandwidth allocation in flexible ofdm-based optical networks," *Journal of Lightwave Technology*, vol. 29, no. 9, pp. 1354–1366, 2011, doi: 10.1109/JLT.2011.2125777.

[7] I. Tomkos, S. Azodolmolky, J. Sole-Pareta, D. Careglio, and E. Palkopoulou, "A tutorial on the flexible optical networking paradigm: State of the art, trends, and research challenges," *Proceedings of the IEEE*, vol. 102, no. 9, pp. 1317–1337, 2014, doi: 10.1109/JPROC.2014.2324652.

[8] J. Armstrong, "Ofdm for optical communications," *Lightwave Technology, Journal of*, vol. 27, no. 3, pp. 189–204, 2009, doi: 10.1109/JLT.2008.2010061.

[9] O. Gerstel, M. Jinno, A. Lord, and S. B. Yoo, "Elastic optical networking: A new dawn for the optical layer?" *Communications Magazine, IEEE*, vol. 50, no. 2, pp. s12–s20, 2012, doi: 10.1109/MCOM.2012.6146481.

[10] B. Kozicki, H. Takara, Y. Tsukishima, T. Yoshimatsu, T. Kobayashi, K. Yonenaga, and M. Jinno, "Optical path aggregation for 1-tb/s transmission in spectrum-sliced elastic optical path network," *Photonics Technology Letters, IEEE*, vol. 22, no. 17, pp. 1315–1317, 2010, doi: 10.1109/LPT.2010.2055046.

[11] X. Yu, M. Tornatore, Y. Zhao, J. Zhang, X. Wang, S. Zhang, R. Wang, J. Wang, J. Zhang, and B. Mukherjee, "When and how should the optical network be upgraded to flex grid?" in *Optical Communication (ECOC), 2014 European Conference on*. IEEE, 2014, pp. 1–3.

[12] R. Wang and B. Mukherjee, "Spectrum management in heterogeneous bandwidth optical networks," *Optical Switching and Networking*, vol. 11, pp. 83–91, 2014, doi: 10.1016/j.osn.2013.09.003.

[13] M. Klinkowski and K. Walkowiak, "Routing and spectrum assignment in spectrum sliced elastic optical path network," *IEEE Communications Letters*, vol. 15, no. 8, pp. 884–886, 2011, doi: 10.1109/LCOMM.2011.060811.110281.

[14] F. Callegati, L. H. Bonani, and W. Cerroni, "Service fairness in flexible optical networks," in *Optical Fiber Communication Conference*. Optical Society of America, 2013, pp. OTh1H–4.

[15] Y. Yin, H. Zhang, M. Zhang, M. Xia, Z. Zhu, S. Dahlfort, and S. B.

Yoo, "Spectral and spatial 2d fragmentation-aware routing and spectrum assignment algorithms in elastic optical networks [invited]," *Journal of Optical Communications and Networking*, vol. 5, no. 10, pp. A100–A106, 2013, doi: 10.1364/JOCN.5.00A100.

[16] X. Wang, Y. Zhao, J. Zhang, X. Yu, and J. Zhang, "Consecutiveness loss-aware routing and spectrum assignment algorithm in flexible bandwidth optical networks," in *Communications and Networking in China (CHINACOM), 2012 7th International ICST Conference on*. IEEE, 2012, pp. 262–266.

[17] A. Rosa, C. Cavdar, S. Carvalho, J. Costa, and L. Wosinska, "Spectrum allocation policy modeling for elastic optical networks," in *High Capacity Optical Networks and Enabling Technologies (HONET), 2012 9th International Conference on*. IEEE, 2012, pp. 242–246.

[18] X. Wan, L. Wang, N. Hua, H. Zhang, and X. Zheng, "Dynamic routing and spectrum assignment in flexible optical path networks," in *National Fiber Optic Engineers Conference*. Optical Society of America, 2011, p. JWA055.

[19] B. Chatterjee, N. Sarma, and E. Oki, "Routing and spectrum allocation in elastic optical networks: A tutorial," *Communications Surveys Tutorials, IEEE*, vol. 17, no. 3, pp. 1776–1800, 2015, doi: 10.1109/COMST.2015.2431731.

[20] N. Hua, Y. Liu, X. Wan, X. Zheng, and Z. Liu, "Dynamic routing and spectrum assignment algorithms in flexible optical networks: An overview," in *Communications and Networking in China (CHINACOM), 2012 7th International ICST Conference on*. IEEE, 2012, pp. 251–255.

[21] R. S. Tessinari, B. Puype, D. Colle, and A. S. Garcia, "Elastico++: An elastic optical network simulation framework for OMNeT++," *Optical Switching and Networking*, vol. 22, pp. 95–104, 2016, doi: 10.1016/j.osn.2016.07.001.

[22] Varga, A., "OMNeT++ web page," https://omnetpp.org/.

[23] E. Agrell, M. Karlsson, A. Chraplyvy, D. J. Richardson, P. M. Krummrich, P. Winzer, K. Roberts, J. K. Fischer, S. J. Savory, B. J. Eggleton *et al.*, "Roadmap of optical communications," *Journal of Optics*, vol. 18, no. 6, pp. 63 002–63 041, 2016, doi: 10.1088/2040-8978/18/6/063002.

[24] S. Talebi, F. Alam, I. Katib, M. Khamis, R. Salama, and G. N. Rouskas, "Spectrum management techniques for elastic optical networks: A survey," *Optical Switching and Networking*, vol. 13, pp. 34–48, 2014, doi: 10.1016/j.osn.2014.02.003.

[25] B. Wang and P.-H. Ho, "Energy-efficient routing and bandwidth allocation in ofdm-based optical networks," *Journal of Optical Communications and Networking*, vol. 8, no. 2, pp. 71–84, 2016, doi: 10.1364/JOCN.8.000071.

[26] A. N. Patel, P. N. Ji, J. P. Jue, and T. Wang, "Routing, wavelength assignment, and spectrum allocation algorithms in transparent flexible optical wdm networks," *Optical Switching and Networking*, vol. 9, no. 3, pp. 191–204, 2012, doi: 10.1016/j.osn.2012.02.001.

[27] T. Takagi, H. Hasegawa, K.-i. Sato, Y. Sone, A. Hirano, and M. Jinno, "Disruption minimized spectrum defragmentation in elastic optical path networks that adopt distance adaptive modulation," in *European Conference and Exposition on Optical Communications*. Optical Society of America, 2011, pp. Mo–2.

[28] X. Yu, Y. Zhao, J. Zhang, X. Wang, J. Wang, and J. Zhang, "Spectrum engineering in flexible grid data center optical networks," *Optical Switching and Networking*, vol. 14, pp. 282–288, 2014, doi: 10.1016/j.osn.2014.05.016.

[29] A. Eira, J. Pedro, D. Fonseca, F. Jimenez Arribas, J. Fernandez-Palacios, I. Lobato Polo, D. Schmuhl, S. Spaelter, D. Marzo, and M. Bohn, "Defragmentation of fixed/flexible grid optical networks," in *Future Network and Mobile Summit (FutureNetworkSummit), 2013*. IEEE, 2013, pp. 1–10.

[30] Y. Wang, X. Cao, Q. Hu, and Y. Pan, "Towards elastic and fine-granular bandwidth allocation in spectrum-sliced optical networks," *Optical Communications and Networking, IEEE/OSA Journal of*, vol. 4, no. 11, pp. 906–917, 2012, doi: 10.1364/JOCN.4.000906.

[31] N. Hara and T. Takahashi, "A study on dynamic spectrum assignment for fairness in elastic optical path networks," in *Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on*. IEEE, 2014, pp. 1–7.

[32] R. S. Tessinari, B. Puype, D. Colle, and A. S. Garcia, "Zone based spectrum assignment in elastic optical networks: A fairness approach," in *Opto-Electronics and Communications Conference (OECC), 2015*. IEEE, 2015, pp. 1–3, doi: 10.1109/OECC.2015.7340247.

[33] J.-L. Izquierdo-Zaragoza, P. Pavon-Marino, and M.-V. Bueno-Delgado, "Distance-adaptive online rsa algorithms for heterogeneous flex-grid networks," in *Optical Network Design and Modeling, 2014 International Conference on*. IEEE, 2014, pp. 204–209.

[34] A. N. Patel, P. N. Ji, J. P. Jue, and T. Wang, "Defragmentation of transparent flexible optical wdm (fwdm) networks," in *Optical Fiber Communication Conference*. Optical Society of America, 2011, p. OTuI8.

[35] F. Cugini, F. Paolucci, G. Meloni, G. Berrettini, M. Secondini, F. Fresi, N. Sambo, L. Poti, and P. Castoldi, "Push-pull defragmentation without traffic disruption in flexible grid optical networks," *Lightwave Technology, Journal of*, vol. 31, no. 1, pp. 125–133, 2013, doi: 10.1109/JLT.2012.2225600.

[36] X. Wang, I. Kim, Q. Zhang, P. Palacharla, and M. Sekiya, "A hitless defragmentation method for self optimizing flexible grid optical networks," in *European Conference and Exhibition on Optical Communication*. Optical Society of America, 2012, pp. P5–04.

[37] R. Wang and B. Mukherjee, "Provisioning in elastic optical networks with non-disruptive defragmentation," *Journal of Lightwave Technology*, vol. 31, no. 15, pp. 2491–2500, 2013, doi: 10.1109/JLT.2013.2268535.

[38] X. Yu, J. Zhang, Y. Zhao, T. Peng, Y. Bai, D. Wang, and X. Lin, "Spectrum compactness based defragmentation in flexible bandwidth optical networks," in *Optical Fiber Communication Conference*. Optical Society of America, 2012, pp. JTh2A–35.

[39] X. Wang, Q. Zhang, I. Kim, P. Palacharla, and M. Sekiya, "Utilization entropy for assessing resource fragmentation in optical networks," in *Optical Fiber Communication Conference*. Optical Society of America, 2012, pp. OTh1A–2.

[40] A. Soares, G. Durães, W. Giozza, and P. Cunha, "Tonets: Ferramenta para avaliaçao de desempenho de redes opticas transparentes," *VII Salão de Ferramentas do Simpósio Brasileiro de Redes de Computadores-SBRC*, 2008.

[41] F. Palmieri, U. Fiore, and S. Ricciardi, "Simulnet: a wavelength-routed optical network simulation framework," in *Computers and Communications, 2009. ISCC 2009. IEEE Symposium on*. IEEE, 2009, pp. 281–286.

[42] D. A. Chaves, H. Pereira, C. Bastos-Filho, and J. Martins-Filho, "Simton: A simulator for transparent optical networks," *Journal of Communication and Information Systems*, vol. 25, no. 1, pp. 1–10, 2010, doi: 10.14209/jcis.2010.1.

[43] L. R. Costa, L. S. de Sousa, F. R. de Oliveira, K. A. da Silva, P. J. Júnior, and A. C. Drummond, "Ons: Simulador de eventos discretos para redes ópticas wdm/eon," *Salão de Ferramentas do Simpósio Brasileiro de Redes de Computadores-SBRC*, 2016.

[44] J.-L. Izquierdo-Zaragoza and P. Pavon-Marino, "Educational and research tools for network optimization," in *Transparent Optical Networks (ICTON), 2013 15th International Conference on*. IEEE, 2013, pp. 1–4, doi: 10.1109/ICTON.2013.6602690.

[45] M. Aibin and M. Blazejewski, "Complex elastic optical network simulator (ceons)," in *Transparent Optical Networks (ICTON), 2015 17th International Conference on*. IEEE, 2015, pp. 1–4, doi: 10.1109/ICTON.2015.7193519.

[46] A. Asensio, A. Castro, L. Velasco, and J. Comellas, "An elastic networks OMNeT++-based simulator," in *Transparent Optical Networks (ICTON), 2013 15th International Conference on*. IEEE, 2013, pp. 1–4, doi: 10.1109/ICTON.2013.6602814.

[47] R. S. Tessinari, "A fairness-focused spectrum assignment algorithm for elastic optical networks," Tese de Doutorado, Universidade Federal do Espírito Santo, Vitória - ES, 2016.

[48] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959, doi: 10.1007/bf01386390.

[49] J. Y. Yen, "Finding the lengths of all shortest paths in n-node nonnegative-distance complete networks using 1 2 n 3 additions and n 3 comparisons," *Journal of the ACM (JACM)*, vol. 19, no. 3, pp. 423–424, 1972, doi: 10.1145/321707.321712.

[50] F. Cugini, M. Secondini, N. Sambo, G. Bottari, G. Bruno, P. Iovanna, and P. Castoldi, "Push-pull technique for defragmentation in flexible optical networks," in *Optical Fiber Communication Conference*. Optical Society of America, 2012, pp. JTh2A–40.

[51] R. Wang and B. Mukherjee, "Spectrum management in heterogeneous bandwidth optical networks," *Optical Switching and Networking*, vol. 11, pp. 83–91, 2014, doi: 10.1016/j.osn.2013.09.003.

[52] G. Zhang, M. De Leenheer, and B. Mukherjee, "Optical traffic grooming in ofdm-based elastic optical networks [invited]," *Journal of Optical Communications and Networking*, vol. 4, no. 11, pp. B17–B25, 2012, doi: 10.1364/JOCN.4.000B17.

[53] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing In Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007, doi: 10.1109/MCSE.2007.55.

Rodrigo Stange Tessinari is currently a post-doctoral researcher at LabTel (Laboratório de Telecomunicações) at the Federal University of Espírito Santo. He received his Ph.D. (2016) and M.Sc. (2011) degrees in Electrical Engineering from the Federal University of Espírito Santo, and a bachelor's degree in Computer Engineering from the same university (2009). During his Ph.D. he was part of the iMinds group at Ghent University. He has experience in Electrical and Computer Engineering, focusing on Telecommunications and Computational Simulations. Moreover, has participated in several R&D projects in partnership with the industry, acting mainly on the following subjects: Elastic Optical Networks (EON), Optical Transport Networks (OTN), Networking Simulation, Resource Assignment algorithms, and Control Plane technologies. He is the creator of the ElasticO++ simulation framework, which earned him runner-up in the 2016 Fabio Neri Best Paper of The Year Award.

Didier Colle is full professor at Ghent University since 2014. He was associated professor since 2011 at the same university and received a Ph.D. degree in 2002 and a M.Sc. degree in electrotechnical engineering in 1997 from the same university. He is group leader in the IMEC Smart Applications and Innovation Services Flanders (SAISF) business unit. He is co-responsible for the research cluster on network modelling, design and evaluation (NetMoDeL) inside the IDlab research group. This research cluster deals with fixed internet architectures and optical networks, green-ict, design of network algorithms and techno-economic studies. His research is mainly conducted inside international (mainly European), national and bilateral research projects together with the industry. This research has been published in more than 450 international journal and conference articles and has resulted in more than 15 PhD degrees.

Anilton Salles Garcia has a degree in Mechanical Engineering from the Federal University of Espírito Santo (1976), an M.Sc. degree in Applied Mathematics (1978), and a Ph.D. degree in Electrical Engineering from the State University of Campinas (1987). He is currently Pro-Rector of Planning and Institutional Development, and a Voluntary Professor of the Federal University of Espírito Santo of the same university, Collaborating Professor at the State University of Ceará and a retired Professor of the Federal University of Espírito Santo. He has experience in the area of Electrical Engineering, with emphasis on Telecommunications, working mainly on the following topics: Network and Service Management, Performance Evaluation and Capacity Planning, Broadband Wireless Networks, Information Modeling supported by Ontologies, Semantic Interoperability of Information Systems, Stochastic Simulation, Elastic Optical Networks and Dynamic Optical Networks.