INSTITUTO TECNOLOGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY

School of Engineering and Sciences

TECNOLÓGICO DE MONTERREY®

Application of differential evolution algorithm to optimization problems in optical networks

A dissertation presented by

Fernando Lezama Cruzvillasante

Submitted to the

School of Engineering and Sciences

In partial fulfillment of the requirements for the degree of

Doctor of Philosophy

In

Information Technologies and Communications

Major in Telecommunications

Monterrey, Nuevo León, December 13th, 2014

Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

School of Engineering and Sciences

Application of differential evolution algorithm to optimization problems in optical networks

A dissertation presented by

Fernando Lezama Cruzvillasante

Submitted to the
School of Engineering and Sciences
In partial fulfillment of the requirements for the degree of

Doctor of Philosophy

In

Information Technologies and Communications
Major in **Telecommunications**

Monterrey, Nuevo León, December 13th, 2014.

Dedication

To my dear family, my parents M. Cristina S. Cruz Villasante Barrita and Mariano Lezama Olguin, and my brother Alejandro and sister Mariana.

For their love, patience, understanding and support.

Acknowledgment

To my Advisor Dr. Gerardo A. Castañón, for his guidance, friendship, and support throughout this work.

To my thesis committee members, Dra. Ana Maria Sarmiento, Dr. Gabriel Campuzano, Dr. César Vargas and Dr. Walter Cerroni, for their valuable comments, readiness, and strategic vision applied to this research.

To the management of CONACYT and ITESM scholarships for sponsoring me throughout my graduate studies.

To all my friends, for their valuable help, good will and friendship.

To my lovely family, for the continuous love and affection.

To God for his endless love.

Fernando Lezama Cruzvillasante

Instituto Tecnológico y de Estudios Superiores de Monterrey
December 2014

Application of differential evolution algorithm to optimization problems in optical networks

by

Fernando Lezama Cruzvillasante

Abstract

It is well-known that telecommunications are developing almost exponentially worldwide in response to the ever-increasing bandwidth demand and transmission distances required in communication networks. Wavelength division multiplexing (WDM) optical networks have led to substantial research, which has eventually emphasized the modifications required in the optical network architectures to achieve their full potential. Optical networks are a field quite rich of optimization problems ranging from simple to multiobjective combinatorial ones. In WDM networks, the routing and wavelength assignment (RWA) and the survivable virtual topology mapping (SVTM) issues are of paramount importance in network optimization. With the evolution of optical WDM networks to a more flexible architecture such as OFDM optical networks, new problems such as routing and spectrum allocation (RSA) arises. RWA, SVTM and RSA problems in an arbitrary mesh network are known to be NP-complete.

Computational intelligence emerges as a crucial tool to deal with those complex optimization problems. In computational intelligence, nature-inspired algorithms encompass a set of heuristics that base their operation on the imitation of nature's behavior. It has been proved that those algorithms can be applied to a wide range of optimization problems in diverse areas of the engineering field obtaining near-optimal solutions in an acceptable amount of time.

In this doctoral dissertation we present the application of differential evolution (DE) algorithm to the RWA, SVTM and RSA problems in optical networks. We also propose the analysis of the control parameters of the DE algorithm on the system performance's improvement. Additionally, we propose strategies to improve the efficiency of the algorithm. We present experiments that demonstrate the effectiveness and efficiency of the algorithm.

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

Nowadays, it is well-known that telecommunications are developing almost exponentially worldwide in response to the ever-increasing bandwidth demand and transmission distances required in the communication networks. The systems fit to cope with this exponential growth are leaded by optical technologies which have superior features over other wired systems. These characteristics are, for example, a higher bandwidth (in the order of terabits), lower signal attenuation, lower distortion, lower power consumption, less space required by the material, among others. A mature technology to address the aforementioned growth and which may meet the bandwidth demand is the wavelength division multiplexing technology (WDM) [1]. WDM technology has led to substantial research which has eventually emphasized the modifications required in the optical network architectures to achieve their full potential. In this context, the routing and wavelength assignment (RWA) [2] and Survivable Routing [3] problems are of paramount importance in optical networks optimization. Moreover, flexible optical network (FON) architectures has been proposed as a new more agile network infrastructure needed to provide flexibility and efficiency in the use of resources [4]. In FONs, the optical spectrum is divided into frequency slots of finer size than the established ITU-T WDM grid (50 Ghz). The connections may occupy multiples of these slots according to transmission rate, modulation format and distance required [5].

The RWA problem can be formally stated as follows [6]: given a set of traffic demands between any given pair of nodes in a network, establish paths and assign wavelengths to each of those paths, so that all demand is met and the Network Wavelength Requirement (NWR) is minimized, subject to the wavelength capacity and continuity constraints. The RWA problem can generally be categorized into two cases: RWA with static off-line traffic and RWA with incremental dynamic on-line traffic.

On the other hand, survivability is the ability of a network to withstand and recover from failures, and is one of the most important requirements of networks. Its importance is magnified in fiber optic networks with throughputs in the order of gigabits and terabits per second. In an IP-over-WDM network, a virtual topology can manage the reconfiguration of the traffic to recovery from a failure. A virtual topology is defined by a set of virtual nodes and virtual links (ligthpaths) connecting the nodes while the physical topology is composed by the physical nodes and physical links. The problem of routing virtual links into a physical topology in such a way that the virtual topology (lightpaths set up on the physical network) remains connected in the presence of a physical link failure is known as the Survivable Virtual Topology Mapping (SVTM) problem [7].

With the evolution from rigid to flexible, the well-known routing and wavelength assignment (RWA) problem in WDM networks becomes the RSA in FONs. However, new challenges arise on the networking level since the previous WDM algorithms can no longer be applied directly.

In a simple definition, optimization consists on finding the best solution to a specific problem. However, the most noticeable characteristic of combinatorial or NP-complete problems is that no fast solution is known because the number of feasible solutions increases rapidly as the size of the input data increases. Here is when computational intelligence becomes fundamental.

The RWA, SVTM and RSA problems are well known to be NP-complete problems [7–9], and their importance can be assessed by the number of approaches proposed in the literature to solve them. So far, these problems have been analyzed by integer linear programming formulations (ILP), heuristic strategies and the application of optimization algorithms such as genetic algorithms, particle swarm optimization (PSO), ant colony optimization (ACO), etc.

Even when there have been applied many algorithm to these problems, due to their importance it is a necessity to developed new optimization tools to find efficient solutions in less amount of time. Differential evolution (DE) is arguably one of the most powerful stochastic real-parameter optimization algorithms in current use, [10]. So a study on the application of DE to these optimization problems seems promising.

1.1 Problem Statement

Technological developments in the area of telecommunications, especially the internet, have generated that users demand higher quality services, with better features such as greater transmission capacities, higher speed, among other advantages. Optical networks seem to be the technology that can provide all these characteristics. However, in order to do this, they require an efficient management of the bandwidth with a careful use of optical components and network planning, which makes optical networks a field quite rich of optimization issues ranging from simple distinct problems, to multiobjective combinatorial ones, such as implementation of physical and logical topologies, optimal placement of components, routing and wavelength assignment, survivability in networks, etc. [11].

Providing quality solutions to these problems is not a trivial thing. Many of these problems belong to the class of combinatorial problems; whose main feature is that is not known a fast solution to them. That is, the time required to obtain an acceptable solution, using known algorithms, increases rapidly with respect to the dimension of the problem. So far, these problems have been solved by heuristics strategies or by the application of optimization tools such as genetic algorithms among others [12, 13].

For this reason, the problem of this doctoral dissertation tackle is that there are no tools that guarantee an optimal solution in an adequate time to such problems. Even when in the literature there are different approaches that provide near optimal solutions for these optimization problems, it is necessary to provide new optimization tools or techniques with the ability to find quality solutions (optimal) in acceptable computation time.

We propose to efficiently solve, using an evolutionary algorithm called Differential Evolution (DE), optimization problems in optical networks. We will solve specifically not only the problems of routing and wavelength assignment (RWA) and the survivable virtual topology mapping problem (VTM) in WDM networks, but also the routing and spectrum allocation problem (RSA) in elastic optical networks.

The DE algorithm was developed by K. Price and R. Storn in 1995 [14] for global optimization. The performance of this algorithm depends on the control of a few parameters that can be applied to different problems.

Particularly, it is really interesting to see how these two trends, optical networks and evolutionary computation, can be merged to achieve a common goal. On one hand, we have the growth in demand of communication networks which will be covered by optical networks, and on the other hand we have the design and develop of tools for solving optimally the problems related to optical networks which could be solved by evolutionary computation. So, there is a direct dependency between evolutionary computing and optical networks.

The problem is really wide, because solve all the optimization issues that arise in optical networks through just this dissertation seems impossible. For this reason, we chose to tackle the problems that we consider of paramount importance (RWA, SVTM and RSA), and limiting the research to the use of DE. We hope that this dissertation can be scale-up to the application of the algorithm to other problems, but a delimitation was necessary in order to be more consistent with the scope established.

1.2 Hypothesis and Research Questions

Following, the hypotheses that this doctoral dissertation holds are presented:

- Hypothesis 1: DE improves results compared to those reported using other approaches.
- Hypothesis 2: Through the use of evolutionary algorithms, it is possible to optimize different parameters within an optical network, such as the network wavelength requirement (NWR), the average path length (APL), the blocking probability, among others.
- Hypothesis 3: DE can attain the lower bounds regarding the network wavelength requirement (NWR).
- Hypothesis 4: An efficient virtual topology mapping can be done using DE.
- Hypothesis 5: DE can be applied in a more flexible scenario, solving the RSA problem.
- Hypothesis 6: The convergence rate of the DE algorithm can be improved, thus improving the efficiency and quality of solutions.
- Hypothesis 7: The pre-processing strategies developed will result in better efficiency of the proposed DE algorithm.

In order to have a more detailed scope of what we are solving with the work presented in this doctoral dissertation, we present the research questions, which will be answered:

- It is possible to apply an evolutionary algorithm to solve combinatorial problems in optical networks obtaining better results than those proposed in the literature?
- The routing and wavelength assignment (RWA) problem can be optimized using DE?
- Is it possible to do an optimal virtual topology mapping (VTM) using DE?
- Through the use of evolutionary algorithms, is it possible to optimize different parameters within an optical network, such as the network wavelength requirement (NWR), the average path length (APL), the blocking probability, among others?
- Can the VTM be solved optimally avoiding not only a single link failure but also multiple link failures or node failures?

- Can we applied the algorithm in an elastic scenario, solving the routing and spectrum allocation (RSA) problem?
- How can stagnation in local optimums be avoided?
- DE has three crucial control parameters: the mutation constant (M), the recombination constant (RC) and the population size (NP). Can we get the optimal values of these parameters through parameter tuning?
- Can we improve the response of DE using pre-processing strategies, such a better routing or the introduction of modifications on the original DE algorithms?
- Will DE improve or obtain at least equal results than those reported using other approaches for the RWA, SVTM and RSA problems?

To answer the research questions is not a simple task. To do that, we must conduct a well structured research. The best way to achieve this goal is through the establishment of objectives. The objectives will have a direct relation with the research questions. The main and particular objectives are presented in the following section.

1.3 Objectives

The main objective of this Doctoral Dissertation is to propose and validate a tool (based on an evolutionary algorithm) which provides optimal solutions to optimization problems in optical networks.

The specific objectives of this dissertation are intended to cover the following points:

- To apply an evolutionary algorithm, namely Differential Evolution (DE), to the RWA, SVTM and RSA problems in optical networks
- Optimally solve the routing and wavelength assignment problem.
- Optimally solve the survivable virtual topology mapping problem in all optical networks.
- Optimally solve the routing and spectrum allocation problem in elastic optical networks.
- To find the optimum set of DE parameters to achieve the best results.
- To design pre-processing strategies, such as a better routing or the introduction of new parameters in the original DE scheme to improve the efficiency of the DE algorithm.
- In addition, to compare and show how our results are better than other approaches proposed in the literature when applied to real sized networks.

1.4 Justification

The RWA, SVTM and RSA problems are well known to be NP-complete problems [7–9]. These are classical, yet important, combinatorial problems and their importance can be assessed by the number of approaches that have been proposed in the literature to solve them. On the other hand, the RSA problem has appeared in a new paradigm called elastic optical networks, and has attracted a lot attention in the research community.

To provide a quality solution to the RWA or the RSA problems can bring us different benefits, such as the reduction of resources to establish connections, the increase on the number of users in the network, the reduction of the number of blocked calls, among others [4,12,15]. All these benefits not only will increase the satisfaction of the end user, but also will reduce the network cost for the provider of the service.

Furthermore, as the capacity of optical networks increases with throughputs on the order of gigabits and terabits per second, survivability is a critical concern. Due to this, survivability, the ability of a network to withstand and recover from failures, is one of the most important requirements of optical networks. There is, naturally, much work on network protection or survivability, for instance, [3,7,13] present different approaches to protect a mesh-based WDM optical network from element failures, such as node and link failures. Solving efficiently the SVTM is another option to provide survivability in IP-over-WDM networks.

1.5 Scope and limitations

The problematic is kind of general, because solve all the optimization issues that arise in optical networks through just this doctoral dissertation seems impossible. So, we need to specify which optimization tool will be use, and which problems will be solved.

In this doctoral dissertation the main objective is to efficiently solve, using an evolutionary algorithm called differential evolution (DE), optimization problems in optical networks, specifically the problems of routing and wavelength assignment (RWA) and the virtual mapping problem survivor (SVTM) in WDM networks. Then, we will pass from WDM networks to elastic networks solving the routing and spectrum allocation (RSA) problem.

The analysis will be done through simulations implemented in MATLAB©and using well-known network topologies.

1.6 Contributions

Provide optimal solutions to problems in optical networks give different benefits to the users and operators. For that reason, in this dissertation we deal with problems as the routing and wavelength assignment which is a classical, yet important, problem in WDM networks. Also, guaranteeing survivability in optical networks is a critical requirement in todays optical networks that we can achieve solving the survivable virtual topology mapping problem. More recently, in the new elastic optical network paradigm, researchers are very interested in provide a solution to the routing and spectrum allocation problem.

As evidence of the validity of our research, different contributions to the scientific field throughout publications in international conferences and index journals have being made in this research.

In the telecommunication field, we have contributed with nine international papers published so far. From these papers, five are conference papers [16–20] and four are Journal published papers.

Among the Journal published papers, one was published in the Computer Networks journal from Elsevier [21], two in the Photonic Network Communication journal from Springerlink [22, 23] and one more in the Communications Letters journal [24] from IEEE.

Moreover, we published one chapter book [25] and also we are waiting for the approval of the publication of two more journal papers already submitted and another chapter book (accepted for publication).

At the end of this doctoral dissertation it is included a list with all the related published work. The validity of this research is well documented through these papers.

1.7 Methodology

We are encouraged to show how DE algorithm produces excellent results finding optimal solutions to the RWA and SVTM problems in arbitrary optical networks. We also want to scale-up our research and show that the DE algorithm can be also applied in an elastic scenario solving the RSA problem.

The RWA, SVTM and RSA problems are well known to be NP-complete problems [7,8]. Given the complexity of these three problems, their solutions have been analyzed so far by heuristic strategies and the application of optimization algorithms such as genetic algorithms, particle swarm optimization (PSO), ant colony optimization (ACO), etc.

In the category of evolutionary algorithm, DE is arguably one of the most powerful stochastic real-parameter optimization algorithms in current use. [10] presents a comprehensive survey of the DE algorithm, and shows how this algorithm has many applications on the engineering field.

For these reasons, we are optimistic to propose DE as an option to solve the RWA, SVTM and RSA more efficiently than other approaches proposed in the literature.

Speaking of efficiency refers to providing better solutions in more acceptable times for each problem. An optimal solution is different depending on the problem we are addressing. For the RWA problem, it is intended to decrease the number of wavelengths required (NWR) to meet traffic demand through the application of DE. The NWR has an economic implication: the lower the number of wavelengths required, the lower the cost of the network. At the same time, it is also expected that the paths established will have the shortest path length in order to minimize the average path length (APL) of the network. The reduction of the APL has a primary impact on the delays and transmission impairments of the signal; it also helps to reduce network resource wastage.

In a similar way, for the RSA problem, it is intended to decrease the spectrum utilization (SU). The SU has an impact in the well distribution of the resources, providing quality of the signal, and also more space for future connections. As with the RWA problem, minimize the APL is desired as well.

Regarding the SVTM, Considering the IP level restoration scenario in IP-over-WDM networks [26], an important challenge is to make the routing of the virtual topology (VT) on to the physical topology (PT) survivable. To achieve the IP restoration level the virtual topology needs to remain connected after a failure occurs. The failure can be of many types: node failure, link failure or multiple link failure. Single link failure is the most common failure in optical networks. At the same time, it is expected to minimize the number of wavelength links because this measurement gives an indication of the distribution of network resources.

DE can be modeled in such a way that optimize these objectives. By establishing an objective function and doing an appropriate encoding for individuals within the algorithm, we are able to select the metric to optimize. DE is a multiobjective optimizer, so not only can optimize a single metric, but also can be set to optimize multiple metrics at once, in order to achieve global optimal solutions.

Chapter 2

BACKGROUND

Optical networks are an extensive field of optimization issues ranging from simple distinct problems, to multiobjective combinatorial ones. Computational intelligence emerges as a crucial tool to deal with these complex optimization problems. In computational intelligence, nature-inspired algorithms encompass a set of heuristics whose methodology is based on the emulation of nature's behavior. Some of the most popular and modern algorithms in this category include population-based algorithms, such as genetic algorithms (GA) and differential evolution (DE) optimization. Swarm intelligence includes algorithms such as ant colony optimization (ACO), particle swarm optimization (PSO) or artificial bee colony (ABC). Research has demonstrated that those algorithms can be applied to a wide range of optimization problems in diverse areas of engineering, obtaining acceptable near-optimal solutions in adequate computational time. This chapter reviews the application of nature-inspired algorithms in the area of optical networks. Networking design and optimization problems are categorized, identifying opportunity research areas where nature-inspired algorithms could be applied. Also, as the development of optical networks and computational intelligence is highly dynamic, new trends and directions are identified and discussed. The Chapter aims to be a starting point for those interested in learning the basis of some important nature-inspired algorithms and their applications in optical network technologies.

2.1 Introduction

In recent years the field of telecommunications has experienced an impressive importance growth, mainly due to the popularity of the internet. In the context of the continued growth for broadband access systems including Asymmetric Digital Subscriber Line (ADSL) and

Fiber To The X (FTTX), the strong market for video streaming and other cloud services are driving a continual increase in Internet traffic volume [27].

Wavelength routed optical networks (WRON) promise to meet the high transmission quality and large bandwidth desired by end users for transmitting multimedia traffic. This capacity is obtained through the use of optical technologies with components that provide routing and restoration at the wavelength level. The origin of the optical networking technology is linked to WDM which provides additional capacity on existing optical fibers.

Figure 2.1 shows a WDM network, which consists of routing nodes interconnected by point-to-point fiber-optic links. A routing node or an optical cross-connect (OXC) can route an optical signal from an input fiber to an output fiber without performing optoelectronic conversion.

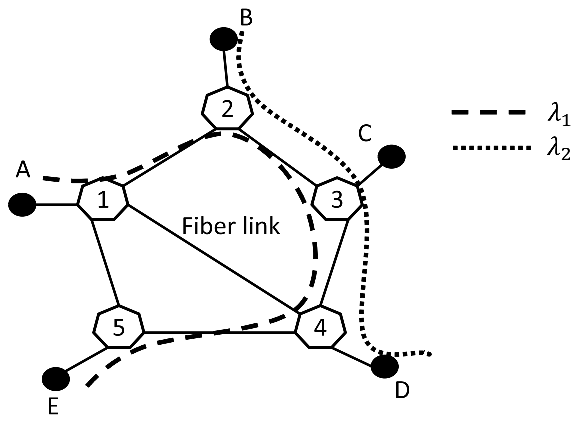


Figure 2.1: A WDM network consisting of routing nodes interconnected by point-to-point fiber-optic links.

Wavelength division multiplexing (WDM) [28] technology divides the bandwidth of a typical optical fiber into some non-overlapping channels operating at different wavelengths providing the opportunity to explore the tremendous bandwidth of fibers in optical networks. WDM networks are considered as connection-oriented networks and have led to substantial research, which has eventually emphasized the modifications required in the optical network architectures to achieve their full potential. However, Optical networks are a field, quite rich, of optimization issues ranging from simple distinct problems, to multiobjective combinatorial ones, such as the implementation of physical and logical topologies, optimal placement of components, routing and wavelength assignment, survivability in networks, etc. [11].

Providing quality solutions to these problems is not a trivial matter, since many of them are combinatorial problems whose distinctive characteristic is that they do not have fast solving methods. Meaning that the computational time required to obtain acceptable solutions, using known algorithms, increases rapidly with the dimension of the problem. So far, these problems have been solved by heuristic strategies or by the application of optimization tools such as genetic algorithms, evolutionary algorithms, among others [12].

In the category of computational intelligence, nature-inspired algorithms are some of the most powerful stochastic real-parameter optimization algorithms in current use [29]. Nature-inspired algorithms encompass a set of heuristics that base their operation on the imitation of nature's behavior. Some of the most popular and modern algorithms in this category include population-based algorithms, such as genetic algorithms (GA) and differential evolution (DE) optimization. Another classification, swarm intelligent, includes algorithms such as ant colony optimization (ACO), particle swarm optimization (PSO) or artificial bee colony (ABC). These metaheuristics have a wide range of applications in different engineering domains, due to their efficiency and effectiveness in the solution of complex optimization problems [30].

In this Chapter, we present a survey and a summary of problems related to optical networks in which nature-inspired algorithms have been applied. To the best of our knowledge, this is the first survey of nature-inspired applications in the field of optical networks. We aim for this work to be used as a starting point for those interested in learning the basis of some important nature-inspired algorithms and their applications in optical network technologies. Also we identify open research problems where these algorithms could be applied.

The Chapter layout is as follows, the proposed search algorithms are discussed in Sect. 2.2. Section 2.3 presents a brief description of different areas of optical networks where the different nature-inspired algorithms have been applied. Section 2.4 presents the discussion and research opportunities. Concluding remarks are addressed in Sect. 2.5.

2.2 Nature-inspired Algorithms

Since the beginning of human history, the first approach to solve problems has been by trial and error. This could be called a heuristic or metaheuristic approach, and indeed our day to day learning experience is based on it.

Methauristics as a scientific method to solve optimization problems is a modern phenomenon. Recently, algorithms with randomization and local search are being called metaheuristics. Due to the complexity of the problems of interest, metaheuristics aim to find good feasible solutions in an acceptable computational time.

Metaheuristics can be classified in many ways. For example population-based algorithms could include GA, PSO, DE, and any other algorithm that considers a population of solutions. Another example could be swarm intelligence algorithms, which use the observed self-organized behavior of colonies of ants or bees.

As is evident, more and more metaheuristics are under development and new appear in the literature periodically. In this survey paper, we present intelligence computational algorithms under the classification of nature-inspired algorithms, which emulate nature's behavior. In the next sections we present a brief description of some of the modern and most important nature-inspired metaheuristics, including GA, PSO, DE, ACO and ABC.

2.2.1 Genetic Algorithms

The genetic algorithm (GA) was introduced by John Holland between 1960s and 1970s [31]. This is one of the most popular evolutionary algorithms in terms of the diversity of its applications.

The GA is a stochastic search technique inspired in some of the processes observed in natural evolution. As many of the evolutionary algorithms, GA stars with an initial set of random solutions called a population. The number of solutions (NC) is the first parameter to be defined. This characteristic of diversity reduces the probability of being stuck in local optimums. In GA, each solution in the population is called a chromosome. A chromosome is a fixed-length binary string of 0s and 1s which represents an encoded solution to the problem. The chromosomes evolve trough an iterative process, in which every iteration is called a generation. To create the next generation with new solutions, two chromosomes are combined using a crossover operator, or a single chromosome is modified using a mutation operator. The new solutions are called offspring and are evaluated by a fitness measure. Finally, some of the offspring and parents are selected according to their fitness value. The algorithm hopefully will converge, after a termination criterion is reached (for example a predefined number of generations), to an optimal or sub-optimal solution to the problem.

Algorithm 1 shows the pseudocode of a simple GA, which has the following components:

a population of binary strings, control parameters, a fitness function, a crossover and mutation operator and a selection mechanism.

Algorithm 1 Pseudocode of the simple GA

```
Set the control parameters $p_c$, $p_m$ and $NC$.
Create an initial Pop.
Evaluate the fitness of Pop.
repeat
    Apply selection operator.
    Apply crossover operator.
    Apply mutation operator.
until a satisfactory solution is obtained or a termination criterion is reached.
```

Encoding

In GA, this is a fundamental mechanism, in which the variables of the optimization problem are encoded as a fixed-length binary strings. The encoding depends on the nature of the problem under study. For example, in determining the optimal flows in a transportation problem, the variables assume continuous values, while in the traveling salesman problem the variables are binary. Nevertheless, the encoding mechanism should map each solution to a unique binary string.

For instance, we present a common method of encoding when the problem has real-value continuous variables. Each variable is first mapped to an integer value defined in a specific range, and then it is encoded using a fixed number of binary strings. For example, if the domain of a continuous variable x is $[-2.5, 2.5]$, we can encode this variable with a precision of one decimal point, by multiplying this range by 10. Thus the continuous variable is then mapped to an integer range of $[-25, 25]$. Finally, if we divide this range in 50 equal units, the length of the string has to be 8 because $49 = 7^2 < 50 < 8^2 = 64$. The binary code corresponding to each integer can, in this way, be easily computed.

Genetic Operators

To form a new individual in GA three operators are applied: selection, crossover and mutation. First, a selection of parent solutions is performed. The selection mechanism is based on the level fitness of the parents. There are many different selection strategies such as the proportionate distribution, the ranking selection, the tournament selection or the genitor selection [32]. The solutions with the best fitness have better probability of being selected. In the simple GA, solutions with best fitness have a higher chance of survive to the next generations.

Once we have selected the parents, the crossover operation is performed. Two strings are picked up from the population and combined in a crossover point which is in the range of $[1, L]$, where L is the length of the string. The two strings exchange a portion beyond the crossover point to form two new strings. The crossover operation is controlled by a parameter called crossover rate (p_c). After the selection of the parents, we apply the crossover operation only if a number between $[0, 1]$ is greater than p_c.

After the crossover operation, strings are subjected to mutation. A new parameter called probability of mutation (p_m) is introduced. Mutation is very simple: bits of the string will change from 0 to 1 and vice versa with a probability of p_m. Every bit is independent, therefore the probability of changing one bit does not affect the probability of changing others.

2.2.2 Particle Swarm Optimization

The particle swarm optimization (PSO) algorithm was first introduced by Kennedy and Eberhart [33]. The original objective of their research was to mathematically simulate the social behavior of bird flocks. PSO can handle many complex engineering and science optimization problems. A complete theoretical analysis of the algorithm by Clerc and Kennedy is presented in [34].

The PSO is a population-base evolutionary algorithm. Similar to other evolutionary algorithms, PSO is initialized with a population of random candidate solutions, called particles. These particles move synchronously together as a swarm and have a fitness value that represents the quality of each as a solution. A particle swarm is a collection of particles. A neighborhood is defined as a sub-collection of particles that are within a certain distance from each other. A certain velocity and position are assigned to each particle, iteratively moving through the search space. Particles will update their position trying to improve with respect to their own performance, best swarm experience and their previous velocity vector.

The PSO algorithm has many variants in the literature, but the standard is the global model (Gbest model) [35] in which the whole population is considered as a single neighborhood. The PSO approach only involves two model equations, making it attractive for its simplicity. As mentioned before, each particle represents a possible solution associated with two vectors, position (x_i) and velocity (v_i). A swarm consists of a number of particles that move through the search space looking for the optimal solution. Each particle updates its position and its previous velocity according to the followings equations:

$$v_i^{k+1} = w * v_i^k + c_1 * r_1(pbest * t_i^k - x_i^k) + c_2 * r_2(gbest^k - x_i^k) \tag{2.1}$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{2.2}$$

where c_1 and c_2 are two positive constants, r_1 and r_2 are two randomly generated numbers within the range $[0, 1]$, w is a constant weight, $pbest_i^k$ is the best position of a particle i based on its own experience ($pbest_i^k = [x_{i1}^{pbest}, x_{i2}^{pbest}, ..., x_{iN}^{pbest}]$), $gbest_i^k$ is the best overall particle position of the swarm ($gbest^k = [x_1^{gbest}, x_2^{gbest}, ..., x_N^{gbest}]$), and k is the generation index.

As we can see in the equations, each particle updates its position according to its own best position, best particle position in the swarm and its previous velocity. A pseudocode summarizing the PSO is presented in algorithm 2.

Algorithm 2 Pseudocode of the PSO

```
Randomly create an initial swarm of particles.
For each particle: randomly initialized the position and velocity vectors.
repeat
  Evaluate the fitness of each particle and store the particle with the best fitness value
    (gbest).
  Update position and velocity according eq. 2.1 and eq. 2.2.
until a satisfactory solution is obtained or a termination criterion is reached.
```

Many other optimization algorithms, such as GA, ACO, among others, compete against PSO. Nevertheless, some of the advantages of PSO over other techniques are that it has few parameters to adjust, it is easy to implement and program, it does not require a good initialization process, it has the ability to avoid stagnation in a local optimal and it is flexible enough to work with other algorithms forming hybrid metaheuristics.

Despite its simplicity, it has been proved that PSO overcomes other optimization techniques in different combinatorial problems [36–38] which makes the PSO a solid and good option to solve optimization problems.

2.2.3 Ant Colony Optimization

The Ant Colony Optimization (ACO) algorithm was first introduced by Colorni et al. [39] and Dorigo et al. [40]. ACO is a swarm intelligent technique inspired in the behavior of real ants and targets combinatorial discrete optimization problems. Nevertheless, the ACO algorithm is a technique that encompasses different kind of models derived from the observation of real ants behavior, which is also a source of inspiration of the design of novel algorithms for the solution of optimization and distributed control problems [41]. A survey of the application of ACO in different engineering domains can be found in [42].

The main idea behind the ACO model is that the self-organized principles of real ant colonies can be exploited to coordinate a population of artificial agents that collaborate to solve optimization problems. So, to understand ACO, an explanation of the behavior of real ants has to be introduced. Examples of different aspects of the behavior of ant colonies that can inspire ant algorithms are division of labor, brood sorting, cooperative transport and foraging. To achieve these tasks, ants communicate with each other through stigmergy. The term "stigmergy" was introduced by French biologist Pierre-Paul Grass in 1959, and describe a non-symbolic form of communication mediated by the environment [43]. To achieve stygmergy, ants leave a trace of pheromones when they go from the nest to the food source and back again. Other ants can perceive the traces of pheromones and tend to follow the routes where the concentration is higher. This mechanism allows ants to transport food to their nest in an effective and easy way.

To explain in more detail the effect of the pheromone, an experiment called the double bridge experiment was designed by Pastel et al. [44]. In this experiment, the nest was connected to a food source by two bridges of equal size. A scout ant starts to explore the routes and eventually reaches the food. In its way out to reach the food, the scout ant leaves a concentration of pheromones. Initially, each ant randomly chooses one of the bridges. Nevertheless, after a while one of the two bridges presents a higher concentration of pheromones and therefore the colony of ants converges to follow this route.

A variant of the double bridge experiment was studied in [45]. In this experiment, one of the bridges was significantly longer than the other. In this case, the ants that chose the short bridge will reach the food and will go back to the nest before the ones that initially chose the longer bridge. Therefore, the short bridge will receive a higher amount of pheromone earlier than the longer bridge, increasing the probability of being selected by the ants. Finally, in [46] a mathematical model that describes this behavior was developed. Assuming that at a given time, m_1 ants have used the shorter bridge and m_2 ants have used the longer one, the probability for an ant to choose the shorter bridge is:

$$p_1 = \frac{(m_1 + k)^h}{(m_1 + k)^h + (m_2 + k)^h} \tag{2.3}$$

where k and h are constant parameters to fit the experimental data. By changing these values one can achieve the impact of shorter path and less congestion path. In the double bridge experiment the probability to choose the other bridge is $p_2 = 1 - p_1$.

The behavior of real ants described before is emulated in the ACO technique by artificial ants. Now, while ACO is inspired by real ants, there are few but important differences, as it is intended as an optimization tool, not as a simulation of ants in nature [47]. For example, the pheromone of real ants is reduced over time as the pheromone is a chemical substance that evaporates. On the other hand, in ACO the evaporation of the pheromone can be established as a constant rate according to the necessities. Also, artificial ants can have memory to store information about the places they have been or the actions they have performed; and they can have a global vision based on pheromone level, traffic flow, congestion, etc.

Dorigo et al. [39, 48] have defined several variants of ACO based on the aforementioned behavior of real ants. A general pseudocode of the ACO algorithm is presented in algorithm 3. ACO can be used in different versions of the same problem, which makes it versatile. It can be applied to different combinatorial problems with minimum changes, which makes it robust. It is a population-based algorithm that allows positive feedback to be used as the primary search mechanism [47].

Algorithm 3 Pseudocode of the ACO

```
Create a construction graph.
Initialize pheromone values.
repeat
   Create all ant solutions considering the pheromone values.
   Perform local search.
   Update pheromone values.
until a satisfactory solution is obtained or a termination criterion is reached.
```

2.2.4 Artificial Bee Colony

The artificial Bee Colony (ABC) algorithm was first introduced by Karaboga [49]. ABC is a swarm-based algorithm inspired on the behavior of honey bees. It is a recent algorithm that exploits fundamental characteristics as self-organization and division of labor. The ABC algorithm can tackle continuous, combinatorial, constrained, multi-objective and large-scale optimization problems. ABC is one of the most used swarm-based algorithms and the number of its applications increases day by day. A comprehensive survey of the algorithm and its applications can be found in [50].

As another swarm-based algorithm, ABC explodes different characteristics of an intelligent swarm. To be called swarm intelligence, an algorithm must satisfy some principles of self-organization and division of labor. Bonabeau et al. in [51] described four mechanisms of self-organization, called positive feedback, negative feedback, fluctuations and finally multiple interactions. Additional to these principles, the division of labor is another important feature of an intelligent swarm. The use of specialized laborers for specific tasks is supposed to increase the performance of the entire swarm [49].

In a Bee colony, there are three components needed to emerge as a collectively intelligent swarm: food source, employed foragers and unemployed foragers. The model defines two leading modes of the behavior of the swarm: the recruitment to a rich nectar source and the abandonment of a poor source.

Food source: the value of a food source depends on many factors such as its proximity to the nest, the concentration of energy or the ease of extracting this energy.

Employed bees: they are associated with a particular food source. They bring in the information about this particular food source to the nest with a certain probability.

Unemployed bees: there are two types of unemployed bees, scouts and onlookers. Scouts bees are always randomly searching the environment looking for new food sources. Onlookers are waiting in the nest and establishing the position of a food source using the information shared by employed bees.

The most important part in the formation of collective knowledge is the exchange of information. The exchange of information among bees takes place in a dancing area. The dance is called a waggle dance. All the information about a rich food source is available to an onlooker bee in the dance area. Because of this, there is a greater probability of an onlooker bee to employ herself at the most profitable food source. Hence, the recruitment is proportional to the profitability of the food source [52].

In the ABC algorithm, the position of a food source represents a solution, and the nectar amount of a food source corresponds to its fitness. In its basic form, the number of employed bees is equal to the number of solutions since each employed bee is associated with one food source. The general idea of the ABC algorithm is given in algorithm 4.

Algorithm 4 Pseudocode of the ABC

```
Initialization step: Using scouts to find initial food sources.
repeat
    Employed bee phase: Send employed bees to food sources.
    Onlooker bee phase: Send onlooker bees to food sources based in probabilities given by
    the employed bees.
    Scout bee phase: Send scouts to discover new food sources.
    Memorize the best food source so far achieved.
until a satisfactory solution is obtained or a termination criterion is reached.
```

In an initialization step, a population of food sources is discovered by the scout bees, and control parameters are set.

In the employed bee phase, employed bees search for new food sources having more nectar within the neighborhood of their associated food source. The fitness of new solution is evaluated and a selection process is applied for the employed bee. The information is then shared with onlooker bees in the hive by dancing in the dancing area.

In the onlooker phase, a probabilistic decision is taken by the onlooker bees based on the information provided by the employed bees. The decision can be made through roulette wheel selection for example, based on the fitness. After a selection has been made by an onlooker bee, a neighborhood source is determined, and through the fitness, a greedy selection is applied as in the employed phase.

In the scout bee phase, if an employed bee finds that a source food does not improve after a limited number of trials, the food source is abandoned and the employed bee becomes a scout bee searching for a new food source randomly.

These three phases (employed bee, onlooker bee and scout bee) are repeated until a satisfactory solution is obtained or a termination criterion is reached.

2.2.5 Differential Evolution Optimization

Differential Evolution (DE) is a very simple mathematical model, which represents a very complex process of evolution. Intelligently using the differences in the population generated, a simple but fast linear operator called differentiation makes the DE unique.

The genetic annealing developed by K. Price [53] was the beginning of the DE algorithm. Then, DE was introduced by K.Price and R. Storn in a series of papers presented in different conferences and journals [14, 54–56]. Since then, the algorithm has been applied to different optimization problems in many engineering fields.

Nowadays, DE is very popular because of its effectiveness in solving multiobjective problems. An internet search reveals that the number of DE research articles indexed in science citation index (SCI) database over the span of 2007-July 2009 is 3964 and out of these, there are more than thousands of application papers in diverse areas [10].

DE algorithm uses a population of individuals and iterates by creating new populations until an optimal solution is obtained. An individual in the algorithm is a vector of dimension D, where D is the problem's dimension, and represents a specific solution to the problem.

At the beginning of the algorithm, assuming that there is not information about the optimum, the initial population is created randomly. DE employs repeated cycles of recombination and selection to guide the population towards the vicinity of a global optimum. The probability operators which are crossing and mutation are applied to each individual in a population to obtain new individuals (children). These new individuals have some properties of their ancestors; these ancestors are kept or deleted by selection. The term generation is used to designate the conversion of all individuals into new ones, i.e., to move from one population to another.

The algorithm is run for a limited number of generations. DE has three crucial control parameters: the mutation constant (M), which controls the mutation strength, the recombination constant (RC) and the population size (NP). Throughout the execution process, the user defines the population size NP. At each generation, all individuals in the population are evaluated in turn. The individual being evaluated is called the target vector. Three other individuals are randomly chosen from the population and are mixed with each other; this operation is referred to as mutation, and results in a mutant individual (which is also a vector).

The mutant individual is then mixed with the current target vector by an operator called recombination, the result of this recombination process is a vector called the trial vector.

Finally, the selection operator is applied. If the trial vector improves the objective function, it is accepted and replaces the current target vector in the new population that is being created. Otherwise, it is rejected and the current target vector passes on to the next generation, in this case the trial vector is not retained.

A pseudocode of the DE algorithm is presented in algorithm 5. Following is a description of the operators used in the DE algorithm that find the most promising region in the search space.

For each target vector $x^i, i = 1, \ldots, NP$, the mutant individual m^i is generated according to the next equation:

$$m^i = x^{r1} + M(x^{r2} - x^{r3}) \tag{2.4}$$

where $x^{r1}, x^{r2}, x^{r3} \in \{1, \ldots, NP\}$; $x^{r1} \neq x^{r2} \neq x^{r3} \neq x^i$. x^{r1}, x^{r2} and x^{r3} are three random individuals from the population, mutually different and also different from the current target vector x^i, and M is a scaling factor called the mutation constant which must be $M > 0$.

The mutation operator is used to control the magnitude of the difference between two individuals, this operator manages the trade-off between exploitation and exploration on the search process. This operator is the one guiding the convergence of the algorithm.

The recombination operator RC is applied to increase the diversity in the mutation process. As mentioned before, this operator is the last step in the creation of the trial vector. To create the trial vector, the mutant individual, m^i, is combined with the current target vector. Particularly, for each component j, where $j = \{1, 2, \ldots, D\}$, of the mutant individual m^i, a random number $rand$ is chosen in the $[0, 1]$ interval. Next, this number $rand$ is compared to the parameter RC, which is called the recombination constant. If $rand \leq RC$, the j^{th} element of the mutant individual is selected as the j^{th} element of the trial vector t^i, otherwise, the j^{th} element of the target vector is selected as the j^{th} element of the trial vector. It is important to note that a small value in RC yields to the cancelation of the mutation operator, since the target vector will become the new trial vector. This is because $rand \leq RC$ will probably not be true in most cases if RC is small.

Finally, the selection operator is applied. This operator is a simple rule of elitist selection of the vectors that improve the objective function. This is done by comparing the fitness between the trial vector and the target vector in the objective function using:

$$pop_i = \begin{cases} t^i & \text{if } f(t^i) < f(x^i) \\ x^i & \text{otherwise} \end{cases} \tag{2.5}$$

where pop_i is the population of the next generation, that changes by accepting or rejecting new individuals.

The best individual in the population and the global best individual are kept at the end of each generation, to keep track of the best solution found so far.

Algorithm 5 Pseudocode of the DE algorithm

```
Set the control parameters $M$, $RC$ and $NP$.
Create an initial Pop.
Evaluate the fitness of every individual.
repeat
  for each individual $x \in Pop$ do
    Select three individuals from Pop.
    Apply mutation.
    Apply recombination.
    Verify boundary constraints.
    if Boundary constraints are violated then
      modify the infeasible elements.
    end if
    Apply selection operator.
    Update Pop.
  end for
until a satisfactory solution is obtained or a computational limit is exceeded.
```

2.2.6 Complexity of Nature-inspired Algorithms

Substantial discussion arises when trying to decide for the best algorithm for a combinatorial NP-hard problem. It is clear that the nature-inspired algorithms presented in this survey have similar characteristics. In general terms of computational theory, the complexity of an algorithm can be defined by the number of elementary computations required for its execution. However, there is not a standard measurement for nature-inspired algorithms that would allow us to unify and compare their complexity. It is common, in the literature, to make performance comparisons based on the quality of the solution and on the computational time using benchmark problems [57] or based on the number of evaluations of the objective function [22]. However, as mentioned before, a unified criterion for classification based on complexity is still missing. Among other reasons, this is due to the many aspects to be considered when analyzing the complexity of an algorithm of this nature. For instance, if we base the complexity analysis on the number of evaluations of the objective function, we also need to consider that the number of evaluations may change from problem to problem. Also, the objective function of a particular problem has its own complexity; therefore even the specification of the complexity of a single algorithm seems difficult. Furthermore, the encoding of a nature-inspired algorithm has also an impact on its performance in a particular problem. Nevertheless, we believe that the ability to make complexity comparisons of nature-inspired algorithms is very important and relevant to the research community.

2.3 Areas of Applications in Optical Networks

Technological developments in the area of telecommunications, especially the Internet, have generated that users demand higher quality services, with better features such as greater transmission capacities, higher speed, among others. Optical networks seem to be the technology that can provide these features. However, due to its complexity, optical network design encompasses a high diversity of optimization issues ranging from simple distinct problems, to multiobjective combinatorial ones, such as implementation of physical and logical topologies, optimal placement of components, routing and wavelength assignment, survivability in networks, etc. [11].

Different design and optimization tools have been considered for those combinatorial problems. However, as the complexity of such problems increases with the network's size, the use of computational intelligence (CI) to solve them is becoming a very important tool. Nature-inspired algorithms have already been applied with success to a wide range of engineering fields, such as signal processing, artificial neural networks, pattern recognition and image processing, electromagnetism, propagation, and microwave engineering, among others. Also, there are a great number of problems in the optical communication field in which nature-inspired algorithms have already been applied. Being a rapidly growing area, a survey and evaluation of these, helps to identify and discuss new trends and open research opportunities.

The following are the main areas in optical network design in which different algorithms included nature-inspired algorithms have been applied. This survey gives a general view of combinatorial problems and possible open research issues in which nature-inspired algorithms can be applied.

2.3.1 Routing and Wavelength Assignment

The routing and wavelength assignment is arguably one of the most studied, yet important, problem in optical networks. In a WDM network a connection in the optical layer is done when data needs to be sent from a source to a destination node. This is accomplished by establishing a path between two nodes, which is referred to as a lightpath. However, in the absence of wavelength conversion, these lightpaths must be chosen without violating any of the following two constraints [6]:

i) *Wavelength capacity constraint:* states that a wavelength may be used only once per fiber at any given point in time; and

ii) *Wavelength continuity constraint:* states that the lightpath uses the same wavelength on all links it traverses from source to destination.

In this context, the RWA [12, 15] problem is of paramount importance in designing and planning optical networks. The RWA problem seeks to optimally establish routes and adequate wavelengths for the requested connections according to an objective function. Physical and operational constraints can also be included [58]. It can generally be categorized into two cases: RWA with static off-line traffic and RWA with incremental dynamic on-line traffic [6].

The RWA problem is known to be an NP-complete problem [8], and its importance can be assessed by the number of approaches proposed in the literature to obtain near optimal solutions [59]. Integer Linear Programming (ILP) models [12, 60] have been successfully used to solve the static RWA problem in small optical networks. But, as the network's size increases so does the dimension of the ILP model, whose solution typically requires an extensive computation effort (and execution time) which renders them impractical for medium to large-scale networks. Therefore, different heuristic-based algorithms have been proposed to solve the problem. Nature-inspired algorithms such as genetic algorithms (GA), particle swarm optimization (PSO), ant colony optimization (ACO), etc. are among them.

In [61] ACO is used to analyze the RWA problem, considering wavelength conversion. The application of ACO algorithm to the problem has been evolving through the years. In [62], an ant-based algorithm for dynamic RWA in WDM optical networks under the wavelength continuity constraint is proposed. Then, in [63], the authors propose the use of an ACO algorithm to solve the intrinsic problem of the RWA under consideration of the wavelength continuity constraint. Its approach also provides survivability to network failures or traffic congestion.

In a more recent paper [64], ACO algorithm is used to solve the dynamic anycast RWA problem in wavelength-routed WDM networks. Finally, in [65] a novel idea to enhance the ACO method for solving the problem of RWA is proposed. In this paper, not only the static and dynamic cases are analyzed, but also no conversion, partial conversion and full conversion cases are studied. Also for the dynamic case, in [66] a novel genetic algorithm is proposed. The algorithm has different attributes, since it not only reduces the call blocking probability, but it also provides fairness among connections, fault tolerance capability and employs a very short computation time. Using a different strategy, in [67] the static RWA problem in optical networks is formulated as a single objective optimization problem and it is solved in a novel way using a GA. Similar to it, [68] presents the use of a PSO algorithm to obtain near-optimal solutions to the NP-complete RWA problem in optical networks, without a wavelength conversion capability. In [69] a heuristic approach inspired by PSO is proposed for solving the static RWA problem and a new encoding scheme for members of the swarm population is proposed as well. The results from [69] are compared to those from [68] showing a significant improvement both in terms of the number of iterations required and in the average path length (APL). Then in [70], a novel chaotic particle swarm optimization (CPSO) based scheme is proposed for solving the dynamic RWA problem in all-optical WDM networks without any wavelength conversion. An evolutionary framework is presented in [71], where a genetic algorithm with random-keys is developed to minimize the number of wavelength in the assignment. Another application of GA for solving the RWA problem is presented in [72].

Regarding the applications of DE and ABC algorithms, in [73] a population-based evolutionary algorithm using DE is introduced. The DE algorithm is modeled for a multiobjective context to solve the static-RWA problem, under the consideration of wavelength conversion capability. In a different approach, a new idea based on the ABC algorithm is introduced in [74]. The proposed RWA-ABC approach is evaluated for both path length (propagation delay) and hops count optimization schemes and compared against GA algorithm. More recently, in [75] the ABC algorithm is used to solve the RWA problem in a multiobjective context. A similar approach is presented in [22], in which DE is modeled to solve the static RWA problem without a wavelength conversion capability. In their study, the DE-RWA approach is compared against GA and PSO. Not just nature-inspired algorithms have been applied to the RWA problem. For instance, in [60] a large RWA problem is partitioned into several smaller sub-problems, each of which may be solved independently and efficiently using well-known approximation techniques.

More recently, in [76] the authors evaluate the average-case performance of eight off-line heuristic algorithms to solve the routing and wavelength assignment problem and the related throughput maximization problem in WDM optical networks.

As we can see, the RWA problem can be considered as a classical well-known problem in optical networks. Nevertheless, its NP-complete nature and importance makes the problem very interesting to the research community, which always seeks to improve the efficiency and effectiveness of the applied algorithms.

2.3.2 Protection and Restoration

Optical networking is the most effective technology to meet the high bandwidth network demand. However, survivability, which is the ability of a network to withstand and recover from failures, is one of the most important requirements in today networks. Its importance is magnified in fiber optic networks with throughput in the order of gigabits and terabits per second. It is because of their huge throughput that a failure in just one of their components, such as a link failure or a node failure, can lead to a huge loss of information.

Survivable networks are based either on dedicated resources or on dynamic restoration. Each of these architectures has its own benefits. On one hand, dedicated resources offer a faster restoration time and guarantee the restoration ability, with the drawback of less efficient usage of the available resources compared to dynamic restoration. On the other hand, dynamic restoration offers a more efficient usage of the resources due to multiplexing spare-capacity, but it cannot guaranty to a 100% the survivability of the network in case of a component failure.

In WDM networks the high bandwidth capacity can be divided into different transmission channels which can be associated to a different optical connection; through which the upper layers (IP, Ethernet, etc.) can transmit data [77]. Then, the survivable virtual topology mapping is introduced, as a mean of providing protection and restoration in WDM networks.

Survivable Virtual Topology Mapping

In IP-over-WDM networks, a virtual IP network has to be routed on top of a physical optical fiber network. An important challenge hereby is to make the routing survivable. A virtual topology can be set up, which is defined by a set of virtual nodes and virtual links (lightpaths) connecting the nodes while the physical topology is composed by the physical nodes and physical links. The problem of routing virtual links into a physical topology in such a way that the virtual topology (lightpaths set up on the physical network) remains connected in the presence of a physical link failure is known as the survivable virtual topology mapping (SVTM) problem [7].

To illustrate the SVTM, consider the physical topology (WDM network) presented in Fig. 4.1(a). Figure 4.1(b) presents a virtual topology with virtual links $e^v_{1,3}$, $e^v_{3,4}$, $e^v_{4,5}$, $e^v_{2,5}$ and $e^v_{1,2}$, which in fact are lightpaths (IP links) that need to be mapped on the WDM network. Figure 4.1(c) presents a SVTM against a single link failure. Observe that a single link failure disconnects at most one virtual link of the virtual topology, so the virtual topology remains connected achieving the IP restoration level. To show a not survivable mapping, in Fig. 4.1(d) we have routed the lightpath $e^v_{2,5}$ through the physical links $e^p_{2,4}$ and $e^p_{4,5}$. It can be seen that a failure in the physical link $e^p_{4,5}$ disconnects the virtual links $e^v_{2,5}$ and $e^v_{4,5}$ of the virtual topology (dashed lines in Fig. 4.1(b)), leaving the virtual node 5 isolated in the virtual topology, which clearly indicates that the mapping is not survivable.

The SVTM problem is known to be NP-complete [7]. Because of its complexity, for real-life size networks, it is not possible to solve the problem optimally in an acceptable amount of time using classical optimization techniques. Therefore, heuristic approaches should be used. In [78] a fast and efficient algorithm that finds a survivable (i.e., robust to single fiber failures) mapping of IP topology on the mesh of fibers in IP-over-WDM networks has been developed. [79] presents a local search algorithm, called FastSurv, which can provide survivable routing in the presence of physical link failure. In [80] the spare capacity allocation (SCA) problem is analyzed using a matrix-based model, and a fast and efficient approximation algorithm termed Successive Survivable Routing (SSR) is developed. [81] studies the survivable VTM problem under single node/SRLG (Shared Risk Link Group) failure model, and proves

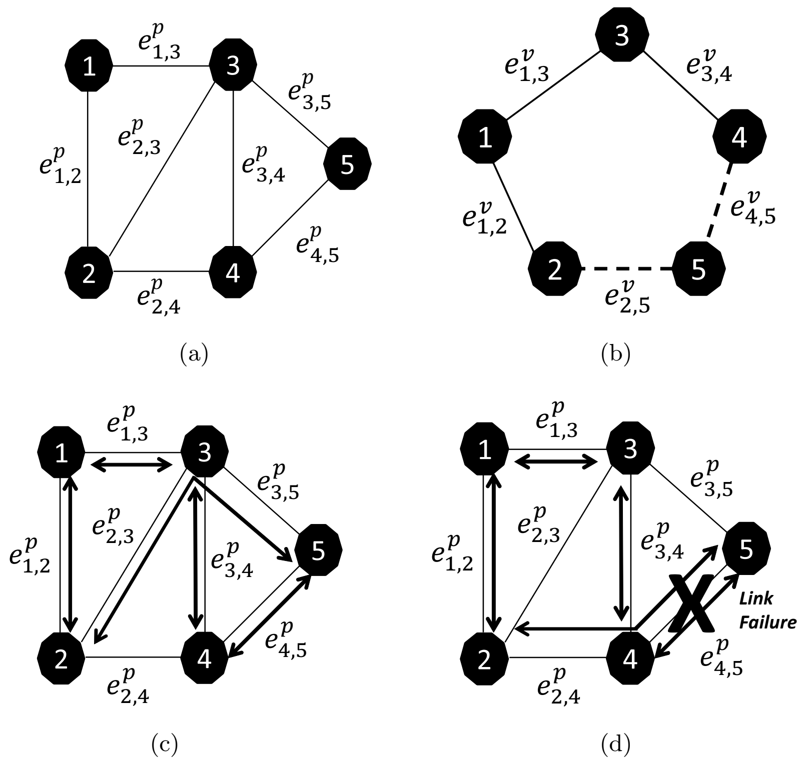


Figure 2.2: Illustrative survivable and unsurvivable virtual topology mapping for a simple 5-nodes network. (a) Physical topology. (b) Virtual topology. (c) Survivable mapping. (d) Unsurvivable mapping.

that the survivable VTM problem under single node/SRLG failures is also NP-complete. As for the use of nature inspired algorithms, [77] and [82] study the survivable VMT problem applying a GA and an ACO algorithm respectively. Also, in [18] an approach to solve the VTM using DE is introduced.

The SVTM was chosen because there are different nature-inspired algorithms applied to this problem, so that fits very well for the survey. However, there are also other survivability approaches [13, 26, 83, 84] in optical networks different from the SVTM.

2.3.3 Optimal Component Placement

Photonic networks can use components capable of implementing wavelength conversion operations in WDM networks. A wavelength converter is a device that converts one wavelength carrying data at the input to a different wavelength at the output. With the use of wavelength converters in a WDM network it is clear that the wavelength continuity constrain is relaxed, providing more flexibility to the network. However, due to the cost of wavelength converters, it is a critical concern to make a careful decision about the number used and the location to place them. This introduces an important optimization problem called the optimal wavelength converter placement. The problem of component placement is more general and involves not only the placement of wavelength converters, but also amplifiers placement [85].

Through the next sections we will analyze different applied approaches in the area of optimal component placement, given a brief description of the problems.

Placement of Wavelength Converters

To further improve the WRON, wavelength converters are placed in network nodes to reduce the network call blocking probability [86, 87]. It is well understood that wavelength converters are presently very expensive and exhibit only a limited wavelength conversion range. That is why a network in which optimal nodes are equipped with wavelength conversion capabilities is economically more convenient. These networks are often referred to as networks with sparse wavelength conversion [88]. However, if a network needs to be equipped with a limited number of wavelength converters, the problem becomes a decision of where to place them in order to get the minimum blocking probability.

K.R. Venugopal et al. [89] used a new heuristic approach for placement of wavelength converters (PWC) to reduce blocking probabilities. Their study includes both, static and dynamic light path establishments. C. Vijayanand et al. [90] proposed new integer linear program (ILP) formulations for the static and dynamic RWA problems to reduce the number of conversions. In their study, a genetic algorithm (GA) was used for placing limited-range wavelength converters in an arbitrary mesh WRON. This was among the first applications of evolutionary algorithms to the wavelength converters placement problem. Later, in [91], a GA is proposed to determine the optimal nodes in the networks where a given number of converters must be placed. Li et al. [92] have proven that the optimal solution on a path could be achieved when all the segments (a path segment between two consecutive converters or between an end node and its nearest converter) on the path have equal blocking probabilities. Based on this theorem, some algorithms of linear complexity were proposed to obtain near-optimal solutions for converter placement on a path. Later, in [93] a network model upon an algorithm for placing a given number of wavelength converters is presented. In addition, some heuristics are proposed and evaluated over different network examples. In short, most of the proposed techniques are either complicated, need large storage capacity, or require massive computation time. Considering these factors, C. F. Teo et al. [94] have proposed a novel evolutionary PSO algorithm to find the optimal placement of wavelength converters to achieve minimum blocking probability. More recently, in [21] a similar approach inspired in DE was applied, in which the authors show that DE can overcome the results obtained with the PSO algorithm.

Placement of Optical Amplifiers

As with the wavelength converters, the optical amplifiers improve the efficiency of WRON. Nevertheless, to obtain a real benefit from these devices, a careful placement has to be done [95]. Erbium-doped fiber amplifiers (EDFA) and RAMAN amplification effect are typical examples of such devices. Also, Semiconductor optical amplifiers (SOAs) have attracted much attention for their cost effectiveness and simplicity as compared to EDFAs [96]. The approach of their placement have to take into consideration physical characteristics such as amplified spontaneous emission (ASE), four wave mixing (FWM), non-linarites, etc. [97]. However, this survey paper intends to focus on the network planning. The placement of these components in the network is an important problem where nature-inspired algorithms can be applied.

Similar to the PWC problem, the placement of optical amplifiers (POA) is referred to the placement of the minimum number of optical amplifiers as well as their position to guarantee that all the signals are adequately amplified improving the performance of the network. In [98] the POA with power equalization in a multiwavelength Optical LAN/MAN network is formulated as a mixed integer linear program (MILP) that can be solved by a linear program solver. In [99] a simulated annealing algorithm is proposed to cope with the POA in broadcast-and-select WDM networks. Later, in [100] an algorithm that takes into account transmission impairments is proposed. Then, in [101] two techniques, based on integer programing, are presented to solve the POA problem in metropolitan WDM ring networks. As with other NP-complete problems, the MILP cannot be tractable for big instances. Then, in [102] an original approach for solving the POA problem in switch-based WDM optical network using GA is presented. More recently, in [103] a heuristic solution that divides the problem into sub-problems taking the interdependency of these into consideration is presented. Finally, in [104] the placement of a SOA amplifier is analyzed in a fiber link. However, the placement of SOAs in an entire network and its optimization process is still an open research topic.

Raman amplifiers became one of the most renewed research subjects. This is due to the development in laser technologies and also to the increase in bandwidth of the optical networks. Despite the advantages presented by the Raman amplifiers, a challenge arises because of the complexity of the design parameters and its high cost (around 10 times an EDFA amplifier) [105]. In [105, 106], GA is applied to improve the characteristics of design of a Raman amplifier. The application of other nature-inspired algorithms to this problem is still a wide open reasearch area.

As we can see, the use of nature-inspired algorithms is low in the POA problem. As the complexity of the networks increases, and taking into consideration the many physical impairments as well as the cost impact, it is clear that the application of more powerful algorithms is greatly still needed.

2.3.4 QoS Routing Services

RWA is a classical problem in WDM networks that considers the continuity and capacity constrains [6]. However, new demanding applications in optical networks are emerging requiring particular specifications to provide guaranteed classes of services. In this more realistic scenario, quality of services (QoS) metrics, such as bounded end-to-end delay, aggregated delay, bandwidth, cost, jitter, among others, have to be considered. The QoS routing is a challenging problem that considers these metrics or a combination of these, attracting a lot of interest in the scientific community. The objective of QoS routing, as in the RWA, consist in identifying and selecting paths and assigning wavelengths to those paths to meet specific QoS metrics [107].

Considering these attributes, QoS routing is an attractive field for applications of intelligent computation approaches. In [108] a routing approach based in GA is proposed to solve the QoS routing problem, considering different constraints, such as delay, bandwidth, packed loss rate, and cost. Simulation results proved that the GA-based algorithm is not only robust and efficient, but also has a great convergence rate and it is very easy to implement. In [109], a DE algorithm was applied to find the multicast tree with maximum reliability degree and user's QoS satisfaction degree. Few years later, hybrid approaches have been also proposed.

In [110] authors study the application of a combination of gaming analysis and PSO algorithm to find a QoS unicast path with Pareto optimum under Nash equilibrium on both the network provider utility and the user utility. The computed results show these approaches are both feasible and effective. A combination of ACO and an artificial immune algorithm was proposed in [111]. More recently, in [112] authors present some hybrid approaches based on GA and PSO to solve the multi-constraint QoS routing. Another nature-inspired application using the ABC algorithm is presented in [113]. The authors present a considerable number of experiments and compare the performance of multicast tree, convergence time and multicast tree cost to other algorithms such as GA and ACO. They find that the ABC algorithm shows improvements in the optimization ability and convergence speed.

2.4 Discussion and Research Opportunities

Through the paper, we present some of the most important nature-inspired algorithms, and also the important role they play in the solution of optimization problems. We have focused our attention in the optical networking field, in which the applications of optimization algorithms are vast due to the high complexity of the problems that arises during the evolution of the technologies. Nevertheless, the field of optical networking optimization as well as the growing diversity of approaches in the literature opens a window for research opportunities. For this reason, we consider of paramount importance to identify these research opportunities, hoping that this paper may help as a guide for future applications of nature-inspired algorithms.

Table 2.1 summarizes the references presented in this review, categorized by application areas and by nature-inspired algorithms. As is evident from the table, there are still many open application areas for some of the optimization algorithms. From the survey, it can be stated that GA is at the base of many of the other algorithms, and it is also the first algorithm often considered for the resolution of an optimization problem. However, it can also be observed that as soon as a new problem emerges, algorithms are applied and compared to the existing ones, showing almost always an improvement in the reported results. Therefore, there are still some areas of opportunity, especially with the application of the ABC algorithm. This algorithm was developed recently compared to the others, and has demonstrated outstanding performance in different problems. Another research opportunity is in the area of components placement.

The POA problem presents a lot of opportunities, since it can take into consideration diverse physical characteristics. Also, the power consumption is an issue that has attracted attention recently. That fact makes the optimization of POA problem an interesting topic to explore. On the other hand, it can be observed that the RWA and QoS routing seem to be entirely covered by nature-inspired algorithms. Nevertheless, these problems are of paramount importance in the area of networking planning. That is the reason why the research community still gives them a lot of attention always proposing different solution approaches that work better than the previous ones.

Table 2.1: Summary on different approaches that have been proposed to solve optimization problems in optical networks.

	GA	DE	PSO	ACO	ABC
Routing and wavelength assignment	[66], [67], [71], [72]	[73], [22]	[68], [69], [70]	[61], [62], [63], [64], [65]	[74], [75]
Static RWA	[67], [71], [72]	[22]	[68], [69]	[65]	[74]
Static RWA with conversion		[73], [22]		[61], [65]	[75]
Dynamic RWA	[66]		[70]	[62], [63], [65]	
Survivable virtual topology mapping	[77], [83]	[18]		[82]	
Placement of components	[90], [91], [99], [102]	[21]	[94]		
Placement of wavelength converters	[90], [91]	[21]	[94]		
Placement of optical amplifiers	[99], [102]				
EDFA, SOA	[99], [102]				
RAMAN amplification	[105], [106]				
QoS routing	[108], [112]	[109]	[110], [112]	[111]	[113]

When we analyze the RWA problem and the QoS routing problems in a more detailed way, we may still be able to identify research opportunities. First, it is true that the classic RWA problem is the most studied. Nevertheless, the RWA problem is called that way in general terms. Strictly speaking, this problem may be categorized in several ways depending on the scenario under consideration. For this reason, one single approach could not handle all the possible scenarios are of interest.

Table 2.1 in its rows two to four presents a simple categorization of the RWA problem in its most basic versions: static and dynamic cases. From this, it can be observed that the DE algorithm and the ABC algorithm, even when they are already applied to the general RWA problem, have not been applied for dynamic RWA, which is an opportunity for researchers. Also, the study of the static case using GA and PSO and considering wavelength conversion capability looks as an attractive field. On the other hand, the QoS routing is even more general, since it can take into account a great number of different metrics. These characteristics make the problem hard to categorize, but provides a whole field for application of algorithms and their variants.

In addition to the application of the algorithms individually, there are also hybrid applications emerging. These applications are able to solve problems in a more efficient way than the simple algorithms analyzed. However, these hybrid approaches are more complex to implement.

This survey has reviewed areas in optical networks related only to the network's planning and design. However, due to different physical devices that comprise an optical network, the algorithms can be also successfully applied to the design and optimization of such devices on the networks [114, 115]. Even when are out the scope of this survey, those applications demonstrate the relevance of the analyzed algorithms.

Finally, it can be anticipated that due to the evolution of optical networks the importance of intelligent computational tools will be of great relevance to deal with the emerging problems. The mature WDM technology, using the traditional ITU grid, is evolving towards a scenario called flexible optical networks. Technologies such as orthogonal frequency division multiplexing (OFDM) [116] and also the development of more flexible devices are characteristics that motivate interest in this new concept [117]. Moreover, long-term solution technologies, such as optical packet switching [118] or optical burst switching [119] at some point will reach maturity. Therefore, it is expected that new design and optimization problems related to these technologies will emerge, in which the use of nature-inspired algorithms may be applied taking into consideration the different methodologies used in previous communications technologies.

The evolution of systems and technology is happening not only in the field of optical networks. As the complexity of communication systems increases, more powerful algorithms will be developed. Some studies examine novel methods for adaptive routing or IT infrastructures in support of the future internet of things and its new emerging applications [120]. The methods include approaches such as evolutionary game theory (EGT) [121] among other heuristics [122]. This new applications could help us to understand the future of all-optical networks.

2.5 Concluding Remarks

In this survey, we discussed different areas of optical network optimization, focusing on network planning, where the application of computational intelligence has relevant importance in solving optimization problems. Computational intelligence algorithms can be classified in many ways, and more and more algorithms are under development. To focus the scope of the survey, we have presented some computational intelligence algorithms under the classification of nature-inspired algorithms. Nature-inspired algorithms encompass a set of heuristics whose methodology is based on the emulation of nature's behavior. We briefly described some of the most important and modern algorithms such as GA, ACO, PSO, DE and ABC. These algorithms have many applications on important optimization issues over the most varied fields of engineering. Within these fields of engineering, optical networking is a very rich field with many combinatorial optimization problems. The increase of network's complexity and development is due to the continuous growth of broadband communications, multimedia services and Internet. The entire field of optical networks is huge. For that reason, we focused on design and network planning issues, letting many other areas out the scope of this paper. Through the Chapter, we analyzed different optimization problems where nature-inspired algorithms have been applied. Also, this survey suggests that the growth in complexity of network designing problems makes the implementation of nature-inspired algorithms a key tool to solve complex problems with multiple constraints. Finally, a classification of the analyzed areas was presented. This classification shows that despite of the nature-inspired algorithms have been vastly used to solve important problems, there is still research opportunities that may be interesting to explore. Also, some directions and trends on the evolution of optical systems are given. The Chapter aims to be a starting point for those interested in computational intelligence applications in the optical networking field.

Chapter 3

DE APPLIED TO THE RWA PROBLEM

The routing and wavelength assignment (RWA) problem, known to be an NP-complete problem, seeks to optimally establish routes and adequate wavelengths for the requested connections according to an objective function. This paper presents the use of a novel approach based on a differential evolution (DE) algorithm to the RWA problem in wavelength-routed Dense-Division-Multiplexing (DWDM) optical networks. The proposed DE-RWA algorithm is modeled to optimize not only the network wavelength requirement (NWR, which is the minimum number of wavelengths needed to fulfill traffic demand) but also the average path length (APL). We present the impact of the control parameters of the DE algorithm on the system performance's improvement. Additionally, we present two strategies to improve the efficiency of the algorithm, knowing as the disjoint cut-set paths (DCS-P) algorithm and the use of a random mutation ($random - M$) parameter for DE. The proposed approach is evaluated for test bench optical networks with up to 40 nodes. Experiments show that the DE-RWA algorithm obtains results that equal the NWR lower bound for networks with and without wavelength conversion capability, whereas reduce the APL. The performance of the DE based approach is compared against results obtained with the Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) models, showing than the DE-RWA outperform those algorithms in speed of convergence and quality of solutions. The presented DE-RWA model is simple to implement and could also be extended by adding other features such as impairment-aware, energy efficient, survivability, among others in optical networks.

3.1 Introduction

Nowadays it is well known that telecommunications are developing almost exponentially worldwide in response to the ever-increasing bandwidth demand and transmission distances required in communication networks. The systems fit to cope with this exponential growth are led by optical systems which have superior features over other wired systems, these characteristics are, for example, a higher bandwidth (in the order of Tbps), lower signal attenuation, lower distortion, lower power consumption, among others. Worldwide networking and communication systems and applications use high-speed optical transport networks as appropriate backbones for connecting buildings, cities and countries [74, 123].

Dense wavelength division multiplexing (DWDM) [28] technology divides the bandwidth of a typical optical fiber into some non-overlapping channels operating at a different wavelength providing the opportunity to explore the tremendous bandwidth of fibers in optical networks. DWDM networks are considered as connection-oriented networks and have led to substantial research, which has eventually emphasized the modifications required in the optical network architectures to achieve their full potential. The connections established in DWDM networks involve two main basic operations: routing and wavelength assignment (RWA) [2].

In a DWDM network a connection in the optical layer is done when data needs to be sent from a source to a destination node. This is accomplished by establishing a path between two nodes, which is referred to as a lightpath. However, in the absence of wavelength conversion, these lightpaths must be chosen without violating any of the following two constraints [6]: i) *Wavelength Capacity constraint:* states that a wavelength may be used only once per fiber at any given point in time; and ii) *Wavelength Continuity constraint:* states that the lightpath uses the same wavelength on all links it traverses from source to destination.

In this context, the RWA [12, 15] problem is of paramount importance in designing and planning optical networks. The RWA problem seeks to optimally establish routes and adequate wavelengths for the requested connections according to an objective function. Physical and operational constraints can also be included [58]. It can generally be categorized into two cases: RWA with static off-line traffic and RWA with incremental dynamic on-line traffic [6]. In this paper we used static uniform traffic in order to compare to Nagatsu's lower bound and other heuristic approaches. This uniform traffic implies that a connection is established from any node to all the other nodes of the network.

The RWA problem is known to be an NP-complete problem [8], and its importance can be assessed by the number of approaches proposed in the literature to obtain near optimal solutions [59]. Integer Linear Programming (ILP) models [12, 60] have been successfully used to solve the static RWA problem in small optical networks. But, as the network's size increases so does the dimension of the ILP model, whose solution typically requires an extensive computation effort (and execution time) which renders them impractical for medium to large-scale networks. Therefore, different heuristic-based algorithms have been proposed to solve the problem. Among them are the evolutionary algorithms such as genetic algorithms (GA), particle swarm optimization (PSO), ant colony optimization (ACO), etc.

Given that our proposed model uses an evolutionary algorithm, the literature review we present is exclusively on models that approach the RWA problem and belong to this class. In [60] a large RWA problem is partitioned into several smaller sub-problems, each of which may be solved independently and efficiently using well-known approximation techniques. In [61] ACO is used to analyze the RWA problem, considering wavelength conversion. [124] addresses the wavelength assignment issues in interconnecting optical Local Area Networks (LANs) in which a wavelength cannot be reused for local connections. In [67] a formulation of the static RWA problem in optical networks as a single objective optimization problem is presented using a GA. Similar to it, [68] presents the use of a PSO algorithm to obtain near-optimal solutions to the problem in optical networks without a wavelength conversion capability. In [125] the RWA problem is addressed using a classical bin packing based algorithm. [69] presents a heuristic approach inspired by PSO for the static RWA problem and a new encoding scheme for members of the swarm population is proposed. In [76] the authors evaluate the average-case performance of eight off-line heuristic algorithms to address the routing and wavelength assignment problem and the related throughput maximization problem in WDM optical networks. [73] presents a population-based evolutionary algorithm for the static-RWA problem, that is restricted to networks with a wavelength conversion capability. Recently [74] presents an approach based on artificial Bee Colony (ABC).

DE is a very simple but very powerful stochastic global optimizer for a continuous search domain. It was proposed by Storn and Price [14] and represents a very complex process of evolution. Cleverly using the differences between the populations, a simple but fast linear operator called differentiation is created, which makes DE unique. Comparative studies show that DE can outperform other evolutionary algorithms in different optimization problems [10, 21, 126]. More specifically DE exploits a population of potential solutions to effectively probe the search space. The algorithm is initialized with a population of random candidate solutions, conceptualized as individuals. For each individual in the population, a descendant is created from three parents. One parent, which is called the main parent, is disturbed by the vector of the difference of the other two parents. If the descendant has a better performance resulting in the objective function, then it replaces the individual, otherwise, the original individual is retained and is passed on to the next generation of the algorithm and the descendant is discarded. This process is repeated until it reaches the termination condition. For a complete theoretical analysis of the DE algorithm the reader is referred to [14].

In this chapter we study the use of a DE algorithm to solve the RWA problem in networks with static traffic and with or without a wavelength conversion capability. Our objective function optimizes the number of wavelengths needed (Net Wavelength Requirement, NWR) and the average path length (APL). To the best of the authors knowledge we provide for the first time a practical DE model to solve the RWA problem that is simple to implement and has the versatility to analyze not only cases of networks with and without wavelength conversion capability. Our model could also be extended in further work to include other features, such as impairment-aware, energy efficient, survivability, among others in optical networks. We also investigate two techniques to improve the algorithm's performance: a new routing algorithm which computes shortest disjoint paths through the cut-set links and the use of a $random - Mutation$ parameter that improves the convergence rate and the quality of the solutions.

We apply our algorithm to real sized networks with up to 40 nodes as the USA and Japan networks. Due to the complexity of the RWA problem, we present a decomposition method that breaks the formulation into two sub-problems: a routing sub-problem and a wavelength assignment problem. Our approach considers a set of pre-computed k-shortest paths from which our algorithm chooses routes and then we use a first-fit algorithm to do the wavelength assignment. Since both sub-problems are solved sequentially, global optimality cannot be guaranteed. However, results show that the DE-RWA algorithm reaches the theoretical lower bounds presented in subsection 3.2.3, which considers uniform traffic 3.6.

The Chapter layout is as follows, the RWA problem formulation and its lower bound are introduced in section 3.2. The proposed search algorithm is discussed in section 3.3. Section 3.4 presents and explains an illustrative example of the DE-RWA. Section 3.5 presents strategies to improve the efficiency of the algorithm. Results are then presented in section 3.6 and conclusions are addressed in section 3.7.

3.2 Problem Formulation

The physical network is modeled as an undirected graph $G = (V, E)$, where V is the set of physical nodes numbered $\{1, 2, ..., |V|\}$, and E is the set of physical links $\{e_1, e_2, ..., |E|\}$. The RWA problem can be formally stated as follows: given a set of traffic demands between any given pair of nodes, $R = \{(s, d, T)\}$, where s is the source node, d is the destination node, and T is the traffic demand between (s, d), establish paths and assign wavelengths to each of those paths, so that all demand is met and the number of utilized wavelengths (NWR) is minimized, subject to the wavelength capacity and continuity constraints.

The objective function comprises the minimization of the NWR as well as the APL. The NWR has an economic implication: the lower the number of wavelengths required, the lower the cost of the network. At the same time it is also expected that the paths established will have the shortest path length in order to minimize the APL of the network, which is defined as the average number of links used by all selected routes. The reduction of the APL has a primary impact on the delays and transmission impairments of the signal; it also helps to reduce network resource wastage.

Therefore, we must establish a criterion for measuring the performance of the algorithm. Our objective function or fitness function is to minimize the weighted sum of the NWR and the APL, as follows [68]:

$$minf(X) = a_1 * (NWR/n_1) + a_2 * (APL/n_2) \tag{3.1}$$

where a_1 and a_2 are the weights used to vary the relative importance of either term in the objective function, where $a_1 = 1 - a_2$, a_1 and a_2 take values from 0 to 1. n_1 and n_2 are normalizing constants that are used to maintain the NWR and APL values in the range of [0,1], since we do not wish to have one term significantly dominating the other. The way in which these normalizing constants are calculated is described in section 5.2.2.

The APL, for a given set of demands R, can be defined as:

$$APL = \frac{\sum lp_{s,d}}{Nc} \quad \forall s, d \in R \tag{3.2}$$

where $lp_{s,d}$ is the length of the path that connects nodes (s, d) and Nc is the number of requested connections in R.

Furthermore, the NWR can be analyzed in two ways: 1) considering the wavelength continuity constraint, which means the network has not a wavelength conversion capability, which in turn reduces its cost significantly since converters need not to be considered and 2) considering it does not have the wavelength continuity and capacity constraints, which means the network has a wavelength conversion capability in some nodes, this a less restrictive approach which however, assumes a most costly network. In this paper we present both cases, the calculation of the NWR is explained in the following two subsections.

3.2.1 NWR Meeting the Wavelength Continuity and Capacity Constraints

Without a wavelength conversion capability, the wavelength-continuity constraint applies to the problem and a wavelength assignment has to be done. In our approach, this assignment is done through a first-fit maximum hops $(FF-MH)$ algorithm. This allocation algorithm is easy to implement and provides good results. The study of other assignment algorithms is out of the scope of this paper. In the $FF-MH$ algorithm wavelengths are assigned by ordering the paths in decreasing number of hops, and assigning the first available wavelength to each in turn. Once the $FF-MH$ has finished, a wavelength allocation matrix $WA = \{s, d, \lambda_x\}$ will be available, showing the wavelength λ_x assigned to each requested connection between nodes s and d. Then, the NWR value will be the greatest wavelength used to satisfy all connection demands in the assignment process:

$$NWR = max \;\; WA_{sd} \quad \forall s, d \in R \tag{3.3}$$

Note that the $FF-MH$ algorithm is performed under the consideration of the wavelength continuity and capacity constraints; therefore the violation of these constraints is avoided.

3.2.2 NWR Without the Wavelength Continuity and Capacity Constraints

In the presence of a wavelength conversion capability it is possible to ignore the wavelength continuity constraint. The procedure to obtain the NWR minimal value, in this case is to minimize the flows on each link, which is equivalent to minimize the number of lightpaths passing through a particular link [60]. Then, the NWR can be defined as:

$$NWR = max \;\; L^e \quad \forall e \in E \tag{3.4}$$

where L^e is the load or number of paths that traverse a particular link e due to traffic demand and is defined as:

$$L^e = \sum_p \alpha_e^p \tag{3.5}$$

where α_e^p is equal to 1 if link e is used for path p and 0 otherwise.

In many cases, full wavelength conversion capability is not recommended or not even necessary, due to its high cost or due to the possibility of an acceptable performance of the network without it.

3.2.3 Lower Bound

In order to assess the degree of optimality, we use the well-known lower bounds for the NWR and the APL. The NWR lower bound, which is equal to or lower than the optimal minimum, can be obtained without using optimization. Theoretically, in a network with N nodes and with uniform traffic of one lightpath between nodes the NWR lower bound can be calculated as [127]:

$$NWR_{lb} = max \left\lceil \frac{K * (N-K)}{|l_{cs}|} \right\rceil \forall l_{cs} \tag{3.6}$$

where l_{cs} is any cut-set of links whose removal generates two disjoint and auto-connect subnets with K and $N - K$ nodes respectively, $\lceil\rceil$ is the ceil function, and $|l_{cs}|$ is the number of links that form the cut-set. The main cut-set links l_{cs} which maximize the Eq. (3.6) can be found by [128].

To better understand the calculation of the NWR lower bound, Figure 3.1 shows the main cut-set for a simple network of five nodes. The cut-set ($|l_{cs}| = 3$), which is formed for the three links between node pairs $1 \rightarrow 3$, $2 \rightarrow 3$ and $2 \rightarrow 4$, separates the network into two subnets, with $K = 3$ and $N - K = 2$ nodes respectively. Using Eq. (3.6), the NWR_{lb} is established as:

$$NWR_{lb} = max\left\lceil \frac{K * (N - K)}{|l_{cs}|} \right\rceil = \left\lceil \frac{3 * 2}{3} \right\rceil = 2 \tag{3.7}$$

It is important to note that this theoretical NWR_{lb} is only valid for this special case (uniform traffic i.e. one connection between all pair of nodes), but still provides a theoretical value as a benchmark. We use this method to obtain the lower bounds for all networks in this paper.

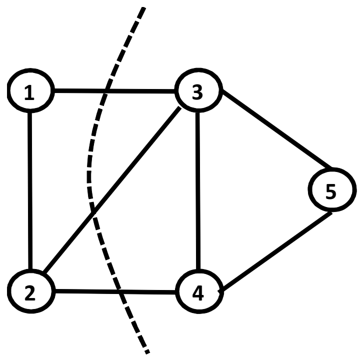


Figure 3.1: The main cut-set for a simple 5 nodes network.

Note that the minimum APL is obtained if all shortest-paths in the network are selected, which in practice can be obtained using a shortest path algorithm. However, although this would guarantee the minimum APL, it would not guarantee the minimum NWR, which, as previously mentioned, has a crucial impact on the network's cost.

Then, for a network with N nodes and with uniform traffic, the lower bound for the APL can be calculated as:

$$APL_{lb} = \frac{\sum sp_{i,j}}{N * (N - 1)} \forall i, j \tag{3.8}$$

where $\sum sp_{i,j}$ is the sum of the length, in number of hops or distance, of all shortest-paths between all nodes in the network, and N is the number of nodes.

Our approach could consider the pre-calculation of paths based on hops or distance. For simplicity, the paths are pre-calculated based on the least number of hops. However, in cases of having two or more paths with the same number of hops the algorithm selects the path with the shortest distance. In order to do this, we artificially consider that the length between nodes is $1 + dist(s, d)/Large_{number}$. Where $dist(s, d)$ is the actual distance in km of the link divided by a $Large_{number}$. Based on the previous rule the algorithm that finds the shortes path selects the path with the least number of hops and shortest distance. Also in cases where there are paths with different number of hops, the routing algorithm gives priority to paths with fewer hops.

3.2.4 The Normalizing Constants

Now that we have defined the NWR and the APL, we can explain how to calculate the normalizing constants from Eq. (3.1) n_1 and n_2. To calculate n_1, that is an upper bound for NWR, one should consider the worst case scenario for the NWR in a network, which occurs when all nodes in a subnet need to be connected through just one link to all nodes in the other subnet (recall that subnets are obtained when the cut-set links in a network are removed). Then:

$$n_1 = K * (N - K) \tag{3.9}$$

where K and $N - K$ are the number of nodes in the respective subnets.

Similarly, n_2 is an upper bound for APL. In order to compute n_2, first we find k-shortes paths that can be calculated using a k-shortes path algorithm or the disjoint cut-set shortes paths algorith that we will explain in subsection 3.5.1. Then, the worst case scenario for the APL is when all longest routes are considered for the connections between node pairs that have been already calculated. Therefore, n_2 can be calculated by:

$$n_2 = \frac{\sum sp_{max(i,j)}}{N * (N - 1)} \forall i, j \tag{3.10}$$

where $sp_{max(i,j)}$ is the longest path from the set of calculated paths for the connection between nodes (i, j).

Note that since n_1 is the upper bound for the NWR and n_2 is the upper bound for the APL, then, dividing both terms (NWR and APL) by their respective upper bounds, we guarantee that the number obtained by this division is in the range of $[0, 1]$.

3.3 Differential Evolution (DE) Algorithm

As mentioned before, DE algorithms use a population (Pop) of individuals and iterate by creating new populations until an optimal solution is obtained. DE algorithms are run for a limited number of iterations (generations).

DE has three crucial control parameters: the mutation constant (M), which controls the mutation strength, the recombination constant (RC), which increases the diversity in the mutation process, and the population size (NP). Throughout the execution process, the user defines M and RC values in the range of $[0, 1]$ and the Pop size NP which is an integer that depends on the dimension of the problem. These parameters are maintained fixed throughout the execution of the algorithm.

For a more detailed explanation of our approach consider the simple network of five nodes in Figure 3.1. Our approach, inspired in [67], assumes that a set of paths to meet demand is available. Therefore, in an initialization step we calculate k paths between all pairs of nodes, using the k-shortest paths algorithm [129].

These paths are stored in an array and represent our solution space. Table 5.1 shows $k = 3$ shortest paths for our example.

Table 3.1: The selected paths for the individual test are in bold.

Node	1	2	3	4	5
1	[]	**[1, 2]** [1, 3, 2] [1, 3, 4, 2]	[1, 3] **[1, 2, 3]** [1, 2, 4, 3]	[1, 2, 4] **[1, 3, 4]** [1, 2, 3, 4]	**[1, 3, 5]** [1, 2, 3, 5] [1, 3, 4, 5]
2		[]	[2, 3] [2, 1, 3] **[2, 4, 3]**	**[2, 4]** [2, 3, 4] [2, 1, 3, 4]	[2, 3, 5] **[2, 4, 5]** [2, 3, 4, 5]
3			[]	[3, 4] [3, 2, 4] **[3, 5, 4]**	[3, 5] **[3, 4, 5]** [3, 2, 4, 5]
4				[]	**[4, 5]** [4, 3, 5] [4, 2, 3, 5]
5					[]

We use this set of paths to generate individuals for the initial population. An individual in the algorithm is a vector of dimension D (where D is the problem's dimension) that represents a specific solution to the problem.

To encode our individuals we generate a vector of dimension D, for example:

$$X = [k_{1\to 2}, k_{1\to 3}, k_{1\to 4}, ..., D] \tag{3.11}$$

where D is equal to the number of connections between nodes in the network. If N is the number of nodes on the network then the number of connections among all nodes will be $N * (N-1)/2$, and $k_{i\to j}$ represents the selection of an available path between nodes i and j.

The paths are considered to be bidirectional. For example the path connecting node 2 to node 3 is the inverse of the path connecting node 3 to node 2. This reduces the dimension of the problem by half saving computation time.

Depending on the application of a given DE algorithm, one can establish upper and lower limits for the number of elements in the X vector. In our instance, the elements in the X vector must take on an integer value ranging in $[1, k]$ (k=maximum number of paths considered between nodes).

At the initialization step of the algorithm, a population of individuals Pop is created randomly taking into account the aforementioned limits.

As stated in subsections 3.2.1 and 3.2.2, the wavelength assignment process depends on whether the network has a wavelength conversion capability or not. For a network with full wavelength conversion capability the NWR is calculated with Eq. (3.4). On the other hand, when the network does not have a wavelength conversion capability wavelength assignments are necessary and the NWR is calculated with Eq. (3.3).

At each generation, all individuals in the Pop are evaluated in turn, the individual being evaluated is called the target vector. For each target vector $x^i, i = 1, \ldots, NP$, a mutant individual m^i is generated according to:

$$m^i = x^{r1} + M(x^{r2} - x^{r3}) \tag{3.12}$$

where $x^{r1}, x^{r2}, x^{r3} \in Pop$: $x^{r1} \neq x^{r2} \neq x^{r3} \neq x^i$. x^{r1}, x^{r2} and x^{r3} are three random individuals from the Pop, mutually different and also different from the current target vector x^i, and the mutation constant M is a scaling factor in the range of $[0-1]$. The M operator is used to control the magnitude of the difference between two individuals in Eq. (5.6). This operator allows us to manage the trade-off between exploitation and exploration in the search process.

Then, the recombination operator RC is applied to increase the diversity in the mutation process. This operator is the last step in the creation of the trial vector t^i. To create the trial vector, the mutant individual, m^i, is combined with the current target vector x^i. Particularly, for each component j, where $j = \{1, 2, \ldots, D\}$, we choose the jth element of the m^i with probability RC, otherwise from the x^i. If we choose a random number $rand$ in the $[0, 1]$ interval, then the t^i is created as follows:

$$t^{i,j} = \begin{cases} m^{i,j} & \text{if } rand < RC \quad \forall j \\ x^{i,j} & \text{otherwise} \end{cases} \tag{3.13}$$

After we create the trial vector t^i, it is necessary to verify the boundary constraints of each element of t^i to avoid creating an infeasible solution. This could happen because any jth element created by Eq. (5.6) that is not in the range of $[1 - k]$ has an RC probability of being selected. If any element of the trial vector violates the constraints it is replaced with a random number in the range of $[1 - k]$. This assures that a feasible solution is obtained.

Finally, the selection operator is applied; this operator is a simple rule of elitist selection of the vectors that improve the objective function. This is done by comparing the fitness between the trial vector and the target vector in the objective function using:

$$pop_i = \begin{cases} t^i & \text{if } f(t^i) < f(x^i) \\ x^i & \text{otherwise} \end{cases} \tag{3.14}$$

where pop_i is the population of the next generation, that changes by accepting or rejecting new individuals. The best individual in the population and the global best individual are kept at the end of each generation, to keep track of the best solution found so far.

Under these considerations a pseudocode of the DE algorithm is as follows:

3.4 An Illustrative Example of the DE-RWA

We now present a more detailed explanation of our approach, with our DE-RWA algorithm, using the simple network of five nodes shown in Figure 3.1. Our approach works for both cases, networks with or without a wavelength conversion capability, being the only difference the formula used to evaluate the fitness function.

To simplify our analysis in this explanatory example, we use a uniform traffic, therefore

Algorithm 6 Pseudocode of the DE algorithm

```
Set the control parameters $M$, $RC$ and $NP$
Create an initial Pop
Evaluate the fitness of every individual
repeat
  for each individual $x \in Pop$ do
    Select three individuals from Pop
    Apply mutation Eq. (5.6)
    Apply recombination Eq. (5.7)
    Verify boundary constraints
    if Boundary constraints are violated then
      modify the infeasible elements
    end if
    Apply selection operator Eq. (5.8)
    Update Pop
  end for
until a satisfactory solution is obtained or a computational limit is exceeded
```

all traffic demand between any given node to all other nodes in the network is one lightpath. This is done because we wish to compare our results to the theoretical lower bound presented in subsection 3.2.3. The generalization to more than one lightpath between nodes is straightforward.

As mentioned before (Section 3.3), our approach assumes that a set of paths to meet demand is available. In an initialization step, we calculate k paths between all pairs of nodes. Table 5.1 shows $k = 3$ shortest paths for our example.

As an example, consider the individual X:

$$X = [1, 2, 2, 1, 3, 1, 2, 3, 2, 1] \tag{3.15}$$

The individual is explained next. Table 5.1, presents all k-shortest paths (with $k = 3$) for all node pairs in our example. We read the k-shortest paths table from left to right and from top to bottom, having $D = 10$ pairs of connections. Recall that we consider bidirectional links, therefore the bottom-left part of the table is empty. Because there are 10 pairs of source-destination nodes, the vector of individual X has 10 elements, each one indicating which route was selected for the corresponding pair of nodes. In Table 5.1, routes marked in bold are the ones selected by the DE-RWA algorithm from those three available for each pair of nodes. Routes are numbered from top to bottom, being the route number 1 the one on top and the route number 3 the one at the bottom in each box in the Table 5.1. Then, the elements in the vector of individual X indicate which routes are selected. For instance, the first element in the vector, which is a 1, indicates that for the connection between nodes 1 and 2 the first route is selected. Hence, the fifth element in the vector, which is a 3, indicates that the third route was selected for the pair $2 \rightarrow 3$ (which has the fifth position in the table), and so on.

Up to this point the application of our algorithm is the same for both cases, networks with and without a wavelength conversion capability. The difference between both cases, as mentioned in section 3.3, resides in how we quantify the NWR part of the objective function. With full wavelength conversion a wavelength assignment is not necessary and the NWR is obtained from Eq. (3.3). On the other hand, without a wavelength conversion capability, a $FF-MH$ algorithm explained in subsection 3.2.2 is used for the wavelength assignment and the NWR is obtained from Eq. (3.4).

After we have selected and assigned routes and wavelengths, we use Eq. (3.1). For simplicity, in the example we use $a_1 = a_2 = 0.5$ to determine the weighted sum of the NWR and the APL, which give us the fitness of an individual.

As stated in section 3.3, the iterative process is repeated until the termination condition is met.

Figure 3.2 schematically shows a comparison of results obtained from our DE-RWA algorithm with a wavelength conversion capability and from a shortest path first-fit assignment (SP/FF) algorithm on the NWR and APL values. Note that the SP/FF algorithm does not have the ability to choose alternate routes. In our approach, the only solution of the SP/FF algorithm would be represented as:

$$X = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1] \tag{3.16}$$

which indicates that only the shortest paths have been selected. On the contrary, after applying our DE-RWA algorithm, the optimal solution obtained was the individual:

$$X = [1, 1, 1, 1, 1, 1, 1, 1, \mathbf{2}, 1] \tag{3.17}$$

which indicates that the second shortest path, between nodes (3,5) has been selected (e.g. $k_{3\rightarrow5}$

is equal to 2). This solution represents a more efficient routing and wavelength assignment, as it reduces the NWR by one wavelength, and obtains a better distribution of the traffic load on the network, while only slightly increasing the APL. Figure 3.2 (a) is the result of the SP/FF to the problem; the ring is marking the link between nodes 3 and 5 because that is the one with the highest traffic (three wavelengths). Figure 3.2 (b) shows the new distribution of wavelengths obtained with the DE-RWA algorithm, note that all links have at most two wavelengths assigned to them. In this figure the rings show the new route that wavelength number 3 in Figure 3.2 (b) takes, which reduces the NWR by one.

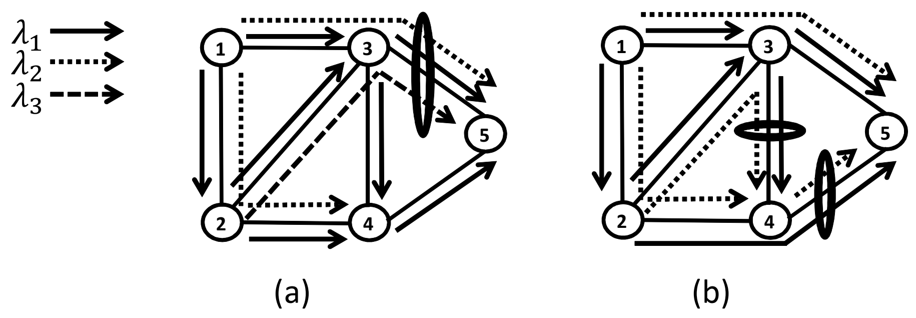


Figure 3.2: (a) RWA based on Shortestpath/first-fit (SP/FF). (b) Best RWA based on our approach.

Table 3.2 shows a numerical comparison of the NWR and the APL of the network in Figure 3.1 when we use our DE-RWA algorithm against a SP/FF algorithm. It clearly shows that the DE-RWA solution is superior to the one obtained by using SP/FF algorithm regarding the NWR parameter, it also shows that the value obtained for the APL from the two methods is not significantly different.

Table 3.2: DE-RWA vs. SP/FF results.

Algorithm	APL	NWR	$FitnessValue$
SP/FF	1.3	3	4.3
DE-RWA	1.4	**2**	3.4

3.5 Strategies to Improve the Efficiency of the Algorithm

We present two strategies to improve the efficiency of the algorithm: the introduction of a new routing algorithm which computes shortest disjoint paths through the cut-set links and the use of a $random - M$ parameter in the DE algorithm that improves the speed of convergence. These strategies are explained in the next two subsections.

3.5.1 The Disjoint Cut-set Paths Algorithm

We noticed that the links that form the main cut-set in a network are of paramount importance when distributing the traffic, and therefore one should be careful not to overload these links in order to get a successful routing and wavelength assignment.

To implement the Disjoint Cut-Set Paths (DCS-P) algorithm we first find the cut-set links using the algorithm in [128] and then by removing them we split the network into two subnets as shown in Figure 3.3 where the NSFnet (National Science Foundation Network) topology is divided in subnet 1 and subnet 2. We then apply the k-shortest paths [129] to each subnet and store the routes.

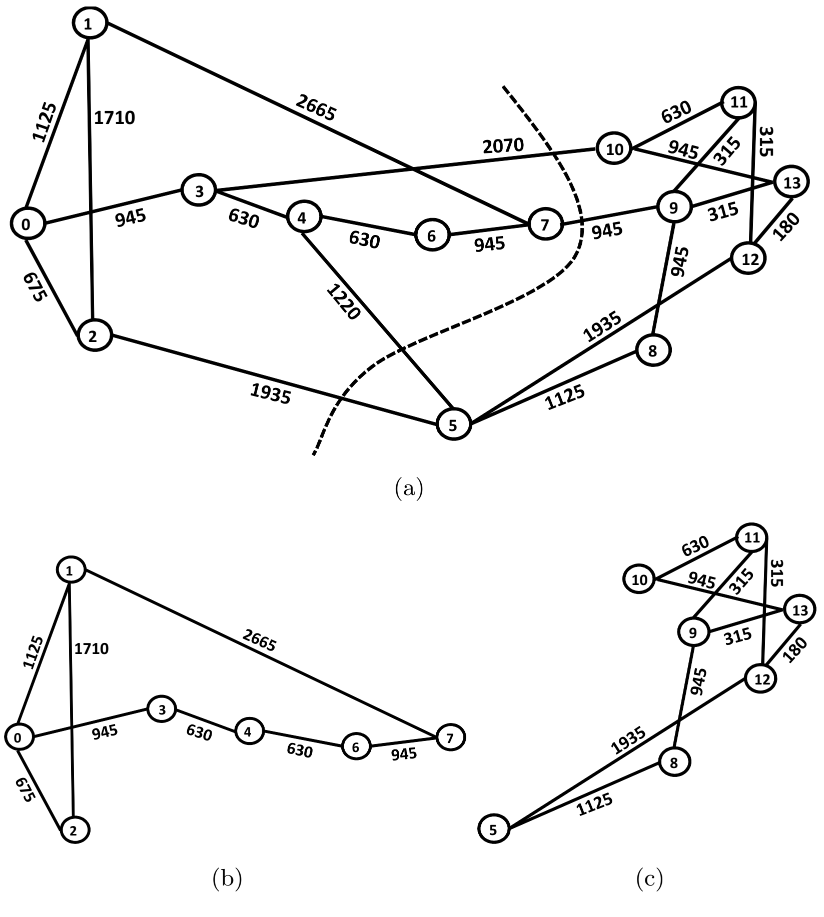


Figure 3.3: NSF network. (a) NSFnet topology and its main cut-set. 14 nodes, 21 links. (b) Subnet 1. (c) Subnet 2.

After this, we join the subnets and apply the Dijkstra's shortest path algorithm to find up to 4 shortest paths from nodes in subnet 1 to nodes in subnet 2. To do this, we first find the first shortest path between a node in subnet 1 to a node in subnet 2 and we identify which link of the cut-set was used, this link of the cut-set is removed from the topology and we proceed to find the second shortest path. We continue with this procedure to find the 4 paths from nodes in subnet 1 to nodes in subnet 2. Note that this specific network has 4 cut-set links and therefore the maximum number of disjoint cut-set paths we can find is 4. Using this methodology to obtain the paths we attain diversity in the routes that our DE-RWA algorithm can consider to choose from to obtain the best combination.

To better explain our algorithm, Table 3.3 shows the paths found with our DCS-P algorithm compared to the k-shortest path algorithm between nodes 6 and 13 (from subnets 1 and 2 respectively).

Table 3.3: DCS-P vs. k-shortest paths.

Disjoint Cut-Set Paths	k-Shortest Paths ($k = 5$)
$6 \rightarrow \mathbf{7} \rightarrow \mathbf{9} \rightarrow 13$	$6 \rightarrow \mathbf{7} \rightarrow \mathbf{9} \rightarrow 13$
$6 \rightarrow 4 \rightarrow \mathbf{3} \rightarrow \mathbf{10} \rightarrow 13$	$6 \rightarrow 4 \rightarrow \mathbf{3} \rightarrow \mathbf{10} \rightarrow 13$
$6 \rightarrow \mathbf{4} \rightarrow \mathbf{5} \rightarrow 12 \rightarrow 13$	$6 \rightarrow \mathbf{4} \rightarrow \mathbf{5} \rightarrow 12 \rightarrow 13$
$6 \rightarrow 7 \rightarrow 1 \rightarrow \mathbf{2} \rightarrow \mathbf{5} \rightarrow 12 \rightarrow 13$	$6 \rightarrow \mathbf{7} \rightarrow \mathbf{9} \rightarrow 11 \rightarrow 10 \rightarrow 13$
	$6 \rightarrow \mathbf{4} \rightarrow \mathbf{5} \rightarrow 8 \rightarrow 9 \rightarrow 13$

Although the first three paths are the same for both algorithms, path 4 of our algorithm uses the link between nodes 2 and 5, which is a disjoint cut-set link path. On the other hand, the k-shortest paths algorithm with $k = 5$ does not have a path that uses the link $2 \rightarrow 5$, instead the algorithm calculates two different paths but repeats the link $7 \rightarrow 9$ in the fourth path and $4 \rightarrow 5$ in the fifth path. Those links are already in paths 1 and 3 which reduce the diversity of paths from where the algorithm can choose from. This was the main reason to implement the DCS-P algorithm: it offers higher variety of routes and is relatively easy to implement. It is important to note that if we increase the value of k, the k-shortest paths algorithm will eventually find the fourth path (with link $2 \rightarrow 5$) obtained with the DCS-P algorithm, but as we observe in the results, increasing the k value has a negative impact in the convergence speed of the algorithm.

Also, notice that, even when the number of routes from one subnet to the other is limited by the number of cut-set links in the network, the value of k must be specified to obtain the routes that will be generated within each subnet. A value of k close to the number of cut-set links is a good choice.

3.5.2 The $Random - M$ Mutation Parameter

After a series of preliminary tests we noted that the algorithm is very sensitive to the M parameter, which, as explained before is a constant that controls the difference between two individuals in the main DE operator. This parameter is also responsible for the exploration and exploitation of the solution space. In preliminary tests we noticed that the convergence of the algorithm was somewhat unstable depending on the value of M as follows: if we chose an small value for M, in the range of $[0 - 0.2]$, the speed of convergence was high but it was also more probable to fall into stagnation in a local optimum. On the other hand, if the value of M was in the range of $[0.6 - 0.9]$ the convergence was slower.

To improve the convergence rate and avoid stagnation in some cases, we use a technique inspired in the term dither [130] called the $random - M$ parameter, which is a random number ($rand$) that multiplies the constant M at each evaluation of Eq. (5.6). Eq. (3.18) shows how the main operator changes with this addition:

$$m^i = x^{r1} + rand * M(x^{r2} - x^{r3}) \tag{3.18}$$

where $rand$ is a random number in the range of [0-1] that changes during the evolution process. By multiplying M by $rand$ we are randomly increasing or reducing its value, which allows the algorithm to obtain benefits from both, a high convergence speed and a strong reduction of the probability of stagnation.

Since dither is rotationally invariant and preserves the contour matching property this diversity enhancing method should always be used [130]. After introducing $rand$, we observed a substantial improvement in the convergence of our algorithm, as presented in the results section.

3.6 Numerical Results and Discussion

We applied our DE-RWA algorithm to four real sized networks, NSFNet, EON (European Optical Network), USA and Japan networks. Their topologies, as well as their cut-sets to determine the NWR's lower bound, are presented in Figures 3.3 (a), 3.4, 3.5 and 3.6 respectively.

To determine the performance measures of the proposed algorithm, lower bounds for the NWR as well as the APL of subsection 3.2.3 have been used. These lower bounds are theoretical bounds and may not be simultaneously achievable.

For the results presented in this section, we assume a uniform traffic demand, where all node pairs are assigned a lightpath consisting of a physical path and a unique wavelength for the case of not using wavelength conversion.

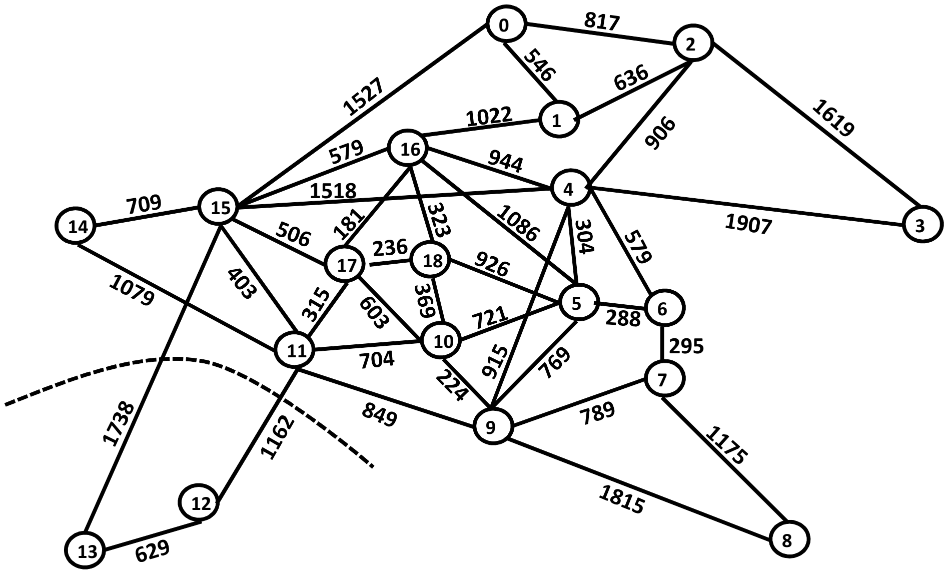


Figure 3.4: EON topology and its main cut-set. 19 nodes, 39 links.

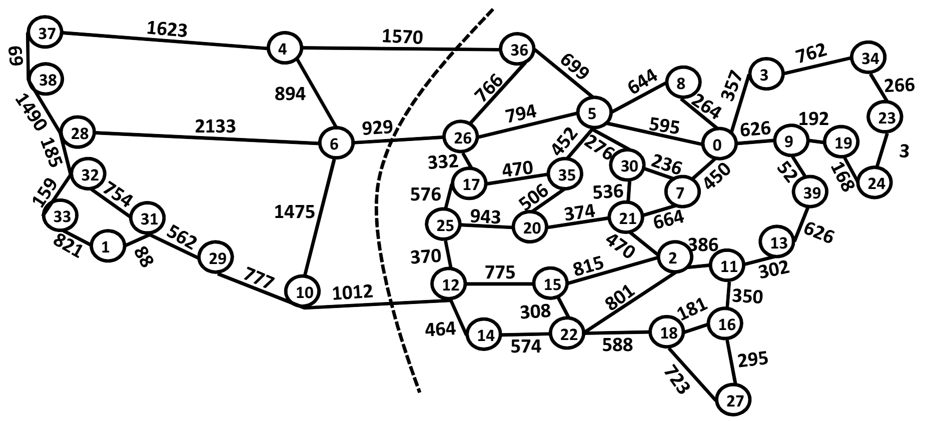


Figure 3.5: USA network topology and its main cut-set. 40 nodes, 58 links.

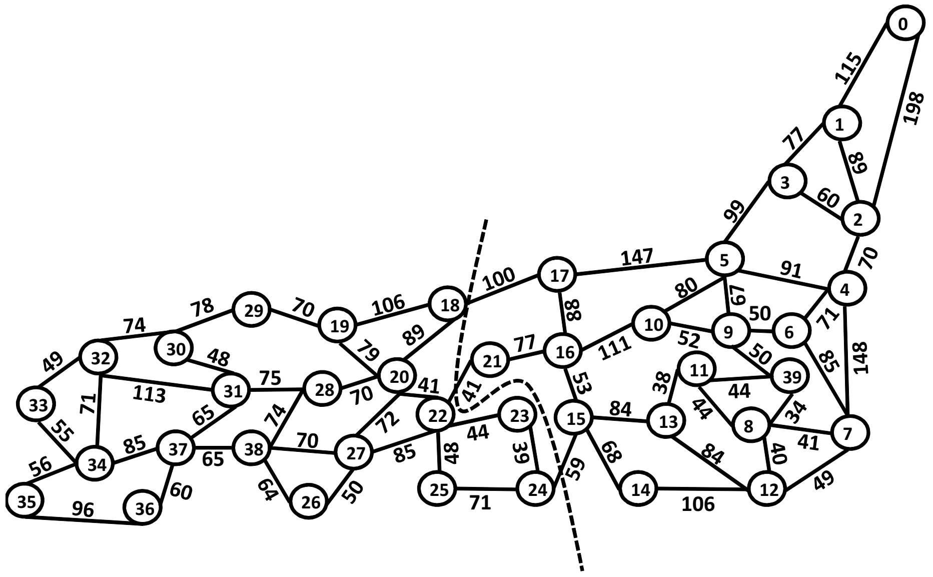


Figure 3.6: Japan network topology and its main cut-set. 40 nodes, 65 links.

It is worth noting that the DE-RWA algorithm requires the manipulation of few control parameters (NP, M, RC). In our application these parameters have a wide impact on the quality of the solutions. Therefore, we believe it is important to present an analysis on their effects. We concentrated our analysis in the 40-nodes USA network (Figure 3.5), because this network has the benefit of being complex enough to show the advantages of the DE-RWA algorithm.

For parametric analysis the weights of the objective function (Eq. (3.1)) a_1 and a_2 have been set to 0.5. The values of n_1 and n_2 for this network (USA) are obtained according to equations (5.4) and (5.5) in section 5.2.2, obtaining $n_1 = 319$ and $n_2 = 7.5167$. We use our DCS-P algorithm an set the value of $k = 4$ for the subnets. The values of the other parameters when doing parameter tuning are presented in their respective figures.

First, we present the parameters behavior when the network does not have a wavelength conversion capability. Figure 3.7 shows the best fitness value (Eq. (3.1)) against the number of generations, when varying the NP factor (population size). It is clear that the NP factor does not present a very marked impact on the speed of convergence, however with $NP = 25$ the algorithm presents a stagnation in a local optimum, therefore selecting a small NP value can lead to stagnation and hence to a non-optimal solutions, but at the same time a large NP requires more computation time. According to Figure 3.7 a NP value of 100 is suitable for this network.

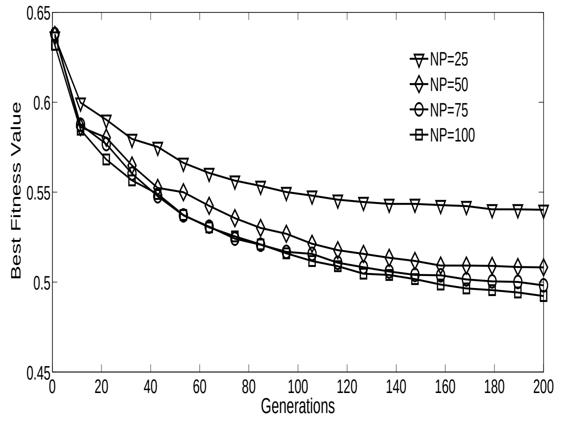


Figure 3.7: Best fitness value against number of iterations with $RC = 0.5$ and $M = 0.2$ for USA Network.

Figure 3.8 shows the impact of the M parameter. Note that, as previously mentioned, this parameter has a high impact on the speed of convergence or alternatively on the possibility of stagnation in local optimum, depending on the range of its value. Curves show that for values $M = 0.2$ and 0.5 the convergence speed and the quality of solution is better than for values of $M = 0.9$ and 0.7 after 200 iterations. Therefore we conclude that a value of $M = 0.2$ is suitable for our application.

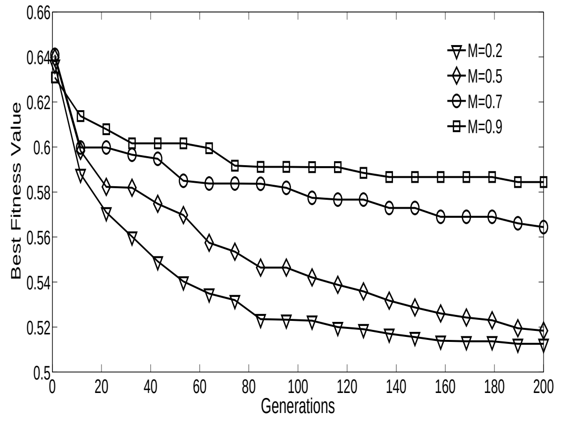


Figure 3.8: Best fitness value against number of iterations with $RC = 0.5$ and $NP = 40$ for USA Network.

Figure 3.9 presents results for the RC parameter. We found two cases of stagnation when $RC = 0.9$ and $RC = 0.7$. Note that a very small value of $RC = 0.2$ requires more generations to converge, therefore we conclude that a mean value of $RC = 0.5$ is the most suitable for this application.

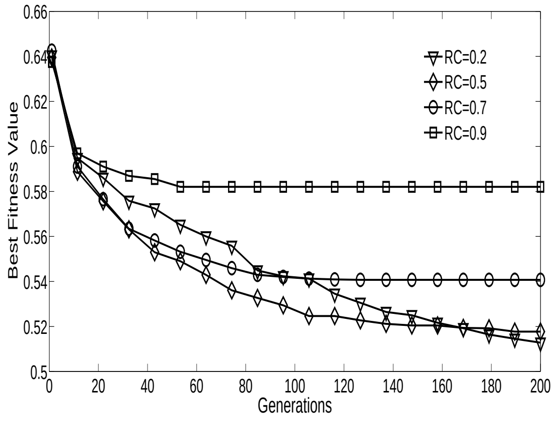


Figure 3.9: Best fitness value against number of iterations with $M = 0.2$ *and* $NP = 40$ *for USA Network.*

After doing the same parametric analysis for the case when the network has a wavelength conversion capability, we conclude that the selection of the algorithm's parameters affects the solution procedure similarly when a wavelength conversion capability exists or not, nevertheless a careful selection of these values should be made, rendering an analysis of the parameters as very important. This analysis can be done through preliminary tests on the network.

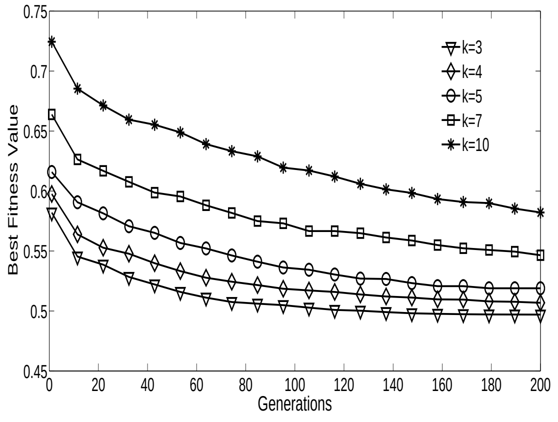


Figure 3.10: Number of iterations vs. best fitness value with $RC = 0.5$*,* $M = 0.2$ *and* $NP = 40$ *for USA Network.*

We also studied the effect of the k parameter, which determines the number of paths between nodes (in the k-shortest paths algorithm). This parameter has a direct relation to the solution space, since the solution space becomes larger as k increases, even when the dimension of the problem D remains the same. Figure 3.10 shows the variation of fitness when the parameter k is modified. Notice that an increase in the value of k causes a decrease in the convergence rate, which makes sense given that the solution space increases as well, and the search for the global optimum requires more effort. Another important issue is that when the value of k is increased, to for example above 10, the quality of the solutions decreases, which indicates that the value of k should not be increased excessively. A set of preliminary tests, according to the networks size, is advisable to set an acceptable value for k.

However it is important to note that by using our DCS-P algorithm we avoid having to select a value for k, since the number of paths directly depends on the number of cut-set links in the network. We only set this k value for the routes within subnets.

Figure 3.11 present results comparing the best fitness value when the DCS-P algorithm is used and when it is not for the USA network. Figure 3.11 shows that the DCS-P algorithm outperforms k-shortest paths when $k = 8$.

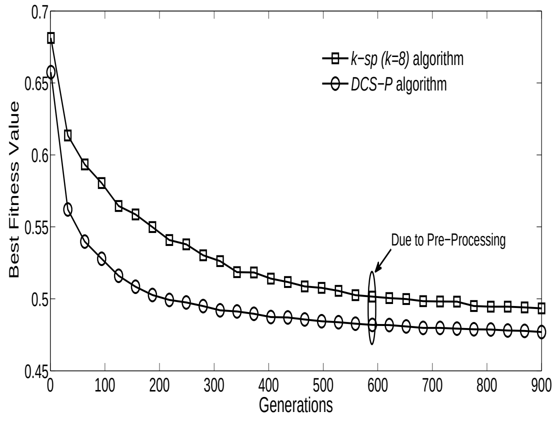


Figure 3.11: Comparison between k-shortest paths with $k = 8$ and DCS-P for USA Network.

Similarly, Figure 3.12 present results comparing the best fitness value when random-M parameter is used and when it is not. The weights a_1 and a_2 were fixed to 0.5, $n1 = 319$ and $n2 = 7.5167$ and $k = 6$ for each subnet. We noticed that when the $random - M$ parameter is used, the fitness value exceeds the one obtained when a fixed value of $M = 0.2$ is considered. Specifically, we observed that when using the $random - M$ parameter, the lower bound for the NWR is reached (equal to 107 wavelengths) opposed to a NWR equal to 109 wavelengths when the value of $M = 0.2$ is fixed (in a simulation run of 900 iterations).

After performing the parameter tuning and having showed the importance of the DCS-P

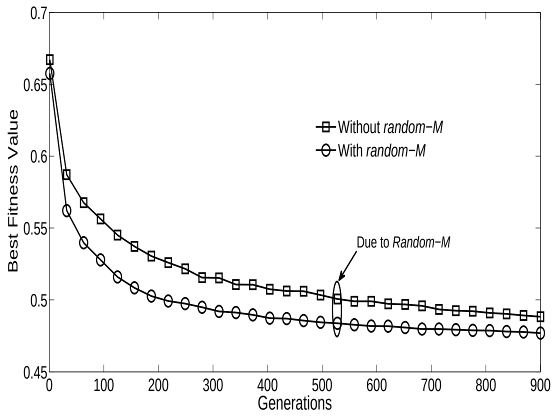


Figure 3.12: Comparison between $fix-M$ and $random-M$ parameter for USA network without a wavelength conversion capability.

algorithm, and the $random-M$ parameter we present an analysis on the weights a_1 and a_2 in the objective function, and how this values affect the convergence speed to the optimal NWR or APL values. Our main interest in this work is to minimize the NWR value, however, we also wish to reduce the APL in as much as possible. For this, we have performed an analysis varying a_1 from 0 to 1 in increments of 0.1, while at the same time varying the value of a_2 from 1 to 0. The results presented are the average of three independent tests with every pair of a_1 and a_2 values.

Figure 3.13 shows the NWR and APL values obtained. We can see that when $a_1 = 0$ and $a_2 = 1$ the algorithm minimizes the APL, as it should since the NWR does not appear in the objective function, and when $a_1 = 1$ and $a_2 = 0$ the algorithm minimizes the NWR. From our analysis we observed that values of $a_1 = 0.5$ and $a_2 = 0.5$ render the best performance in the objective function since the NWR_{lb} is reached and at the same time the APL value obtained is relatively low.

Table 3.4 shows the NWR, APL_{hops} and APL_{dist} lower bounds of the analized topologies. It presents also the values of the normalizing constants n_1 and n_2 for each network.

Table 3.4: Summary of the lower bounds.

Network	NWR	APL_{hops}	APL_{dist}	n_1	n_{2hops}	n_{2dist}
NSFNet	13	2.1429	2264.8	49	4.4066	4592.4
EON	17	2.2047	1851.9	34	3.1988	2452.7
USA	107	4.2859	2648.5	319	7.5166	4719.8
Japon	134	4.4731	330.93	400	6.6231	464.39

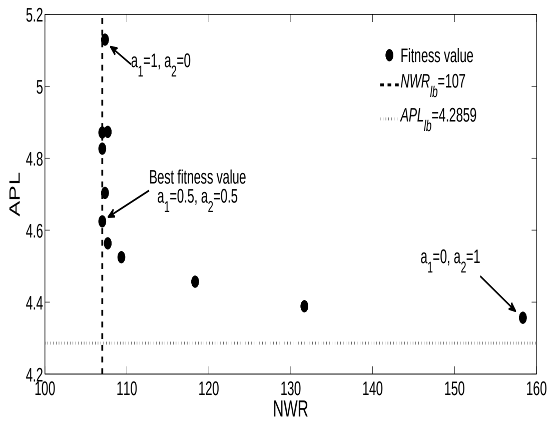


Figure 3.13: NWR vs. APL points obtained varying the weights a_1 and a_2 on the USA network.

Tables 3.5 and 3.6 summarize the best results obtained with our algorithm, considering networks without wavelength conversion capability and with wavelength conversion capability respectively. Results are for uniform traffic of one lightpath between all node pairs, this in view of being able to compare against the APL_{lb} and NWR_{lb} obtained as explained in Section 3.2.3 and presented in Table 3.4. Columns $Shortest-path$ and $DE-RWA$ of Tables 3.5 and 3.6 show the APL and NWR values obtained when using a shortest-paths without any kind of optimization and these same values obtained with our algorithm respectively. $I-reach$ is the iteration number in which the best value reported was obtained.

The parameter values considered for Table 3.5 and 3.6 are: $ramdon - M = 0.2$ and $CR = 0.5$. The population size NP is specified below the name of each network on the tables. The weights a_1 and a_2 have been set to 0.5 and the normalizing constants n_1 and n_2 for each network are specified in Table 3.4. Also, the pre-calculated set of paths were obtained with our DCS-P algorithm not only based on hops $+dist(s,d)/large_{number}$, but also based on distances (both cases presented in Tables 3.5 and 3.6). This shows that our algorithm is versatile enough to be used to minimize the APL based on distances as well.

We can see from table 3.5 that under the assumption of no conversion capability in the network, our DE-RWA algorithm reaches the lower bounds (explained in subsection 3.2.3) for the NWR not only for small networks, but also for larger and more complex networks like USA and Japan (65 and 58 links respectively), with 40 nodes each. This is an important result since indicates that our algorithm obtains the minimum NWR value without the use

Table 3.5: Summary of the best DE-RWA results without a wavelength conversion capability based on hops and distances.

		Shortest-paths		DE-RWA		
Network	Type	APL	NWR	APL	NWR	I-reach
NSFNet	hops:	2.1429	16	2.2418	13	76
$NP = 40$	dist:	2264.8	21	2435.6	13	94
EON	hops:	2.2047	22	2.3158	17	98
$NP = 25$	dist:	1851.9	22	1950.3	18	94
USA	hops:	4.2859	157	4.5936	107	897
$NP = 150$	dist:	2648.5	157	2924.5	107	888
Japon	hops:	4.4731	192	4.6423	134	866
$NP = 120$	dist:	330.93	174	341.59	134	873

Table 3.6: Summary of the best DE-RWA results with a wavelength conversion capability based on hops and distances.

		Shortest-paths		DE-RWA		
Network	Type	APL	NWR	APL	NWR	I-reach
NSFNet	hops:	2.1429	16	2.2418	13	82
$NP = 40$	dist:	2264.8	21	2389.5	13	91
EON	hops:	2.2047	22	2.2865	17	129
$NP = 25$	dist:	1851.9	31	1996.8	17	87
USA	hops:	4.2859	144	4.4962	107	847
$NP = 150$	dist:	2648.5	157	2830.9	107	891
Japon	hops:	4.4731	192	4.6321	134	770
$NP = 120$	dist:	330.93	174	339.46	134	808

of wavelength converters in the network, which can significantly reduce its cost. At the same time, the value of APL is very much acceptable with respect to its lower bound.

To assess the degree of the quality solutions of our DE-RWA, we present a comparison of our results to those reported in [68] and [69], in which two PSO algorithms are used, for the 9-nodes (fig. 3.14) [69], NSFNet (fig. 4.1(a)) and a 20-nodes EON (fig. 3.15) [131] topologies.

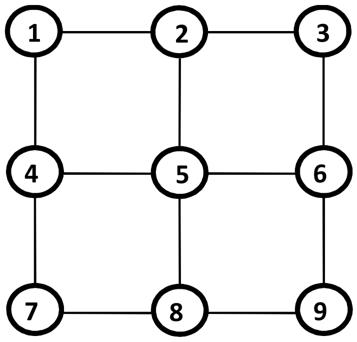


Figure 3.14: Nine node topology.

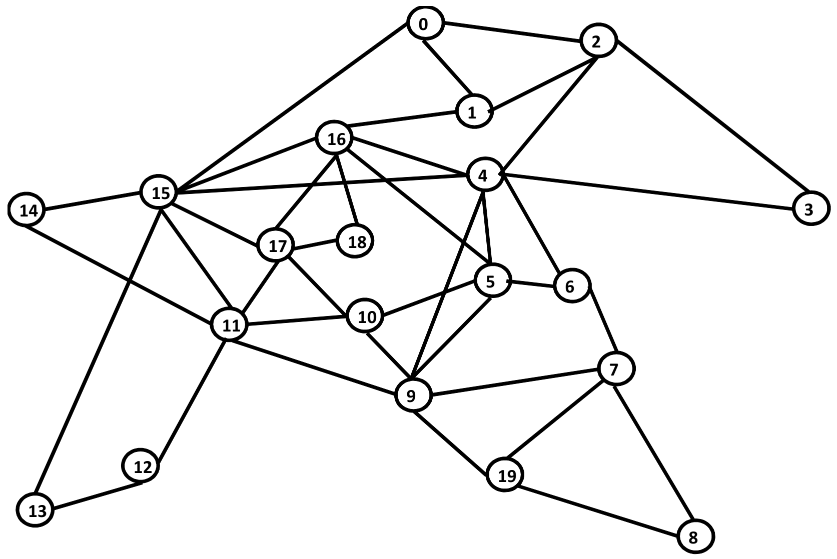


Figure 3.15: 20 Nodes EON topology.

The two PSO algorithms from [68, 69] have been modeled to minimize the NWR and APL, as in this DE-RWA approach. Table 3.7 shows the Swarm Size (SS) and the Neighborhood Size (NS) used by the PSO approaches as well as the NP used by DE-RWA for the different networks used in the comparison, and are presented as information. Weights a_1 and a_2 are set to equal to 1 and the normalizing constants are removed as in [68, 69]. We have used the parameter values obtained from the parametric analysis ($CR = 0.5$ and $random - M = 0.2$), and we have used the routes obtained with our DCS-P algorithm with $k = 3$ for the subnets, except for the network with 9 nodes in which we have used a k-shortest path directly with $k = 3$ because it is a network in which the cut-set is not unique.

Table 3.7: Swarm Size (SS), Neighborhood Size (NS) and Population Size (NP) for PSO and DE-RWA algorithms.

Networks	PSO [68]		PSO [69]		$DE - RWA$
	SS	NS	SS	NS	NP
Nine Nodes Networks	10	5	10	3	30
NSFnet	25	10	14	3	30
EON, 20N	30	10	20	3	30

Moreover, the results presented in [68] were carried out on a 3.0GHz, Pentium4 PC and the time taken to produce their solutions was approximately 15 seconds for 100 iterations. Instead, our simulations were carried out on a 2.35 GHz, Pentium R Dual-Core. The time taken to evaluate 100 iterations was approximately 10 seconds for the NSFnet and 23 seconds for the EON.

Table 3.8 presents a comparison of our results to those reported in [68] and [69]. Columns $I - reached$ show the iteration in which the algorithms achieve their best value. Note that DE-RWA achieves a better NWR for the 9 node network and a slightly better APL for the other networks compared to the results of [68, 69]. In addition, the time taken to evaluate just 100 iteration in [68] was approximately 15 seconds (they not specified for which network) whereas in our approach we need less than 100 iterations to get the best solution, and the time taken to evaluate 100 iterations for the NSFnet and EON is 10 and 23 seconds respectively.

Table 3.8: DE-RWA vs. PSO results.

Networks		9N, 12E	NSFnet	EON
Lower Bounds	APL	2	2.14	2.36
	NWR	6	13	18
DE-RWA	APL	**2**	**2.32**	**2.5**
	NWR	**6**	**13**	**18**
	I-reached	**35**	**67**	**95**
PSO [68]	APL	2.02	2.54	2.86
	NWR	7	13	18
	I-reache	4280	9299	4797
PSO [69]	APL	2.05	2.39	2.77
	NWR	7	13	18
	I-reached	2993	3640	11238

Similarly, we compared our algorithm against a genetic algorithm (GA) proposed in [67]. The parameters used for both algorithms are given in Table 3.9. We use the same number of iterations and presents results for different (s, d) pairs-connections for all the networks.

Table 3.9: GA and DE-RWA Parameters.

GA [67]		DE-RWA	
Population size	200	NP	30
Crossover probability	0.01	CR	0.5
Mutation probability	1/no. of bits	$rand - M$	0.2
Iterations	100	Gen	100

Table 3.10 presents the comparison of our DE-RWA against the GA for the NSFnet and EON networks presented in [67]. The 20 links NSFnet topology from [67] is similar to Fig. 4.1(a) but without the link 12 $\rightarrow$ 13. The 18 Nodes 33 Links EON topology is presented in Fig. 3.16.

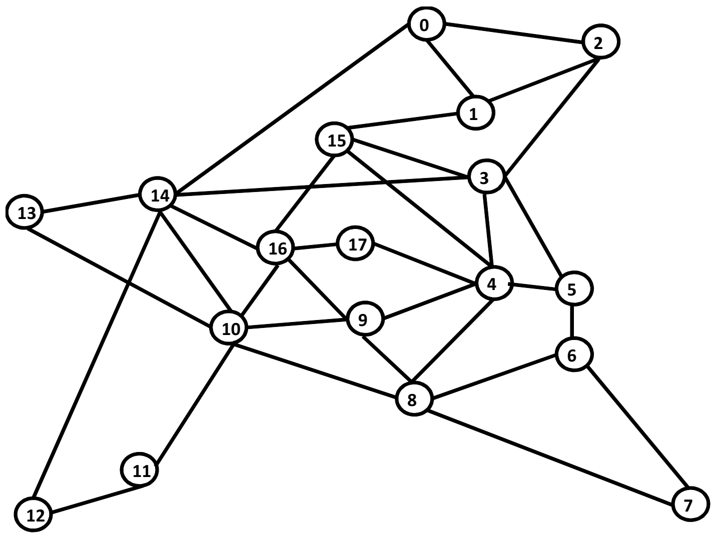


Figure 3.16: 18 Nodes, 33 Links, EON topology.

Values for the weights and constants are as follows: $a1 = 0.6$, $a2 = 04$, $n1 = 49$, $n2 = 4.8022$ for NSFNet and $a1 = 0.2$, $a2 = 0.8$, $n1 = 32$, $n2 = 3.3464$ for EON. We ran the simulations for 20, 40, 60, 80 and 100 connections. We created 10 random independent trials for each connection (s, d) presented in the table to show an average value as well as the best value. Column $(s, d)pairs$ is the number of random generated connections, column FF is the number of wavelengths when the first-fit algorithm is used and is presented as information. Column GA is the result obtained in [67]. DE_{best} is the best result from the 10 random independent runs while DE_{prom} is the average obtained from those runs.

Table 3.10: DE-RWA vs. GA and First-Fit results.

Networks [67]	$(s, d)pairs$	FF	GA	DE_{best}	DE_{prom}
NSFnet 14 Nodes 20 Links	20	6	5	**3**	3.9
	40	11	9	**6**	6.9
	60	21	14	**8**	9.3
	80	31	23	**12**	12.1
	100	41	23	**14**	15.8
EON 18 Nodes 33 Links	20	5	4	**2**	3.2
	40	9	6	**5**	5.8
	60	13	10	**7**	7.7
	80	15	11	**8**	9.7
	100	18	11	**10**	11.2

Table 3.10 shows the NWR obtained from the different algorithms when setting up different (s, d) pair requests. For the NSFNet and EON, the DE-RWA algorithm best result

obtains a better NWR in all cases. As the random creation of virtual topologies has an impact in the wavelength distribution, we have also presented the average value obtained when considering 10 random independent topologies, we can notice that even when the result is not always the same, in average, the value of the NWR obtained with our algorithm is better for practically all cases (except for the case of 100 connections in the EON network) compared to the one obtained by GA, showing our algorithms ability to cope with random generated connections.

3.7 Conclusions

In this chapter we propose the application of a differential evolution (DE) algorithm to the RWA problem. We confirm the effective capabilities in terms of convergence speed and quality of the solutions obtained, minimizing the NWR as well as the APL. We present an illustrative experiment to demonstrate the methodology of our DE-RWA. Despite the solutions are somewhat sensible to variations of the DE-RWA algorithm parameters, the computed results show that a good combination of these parameters leads to a system performance's improvement and to a superior convergence speed. In addition, we have developed strategies that improve the performance of the algorithm for this particular application. Results show that by introducing the $random - M$ parameter and the disjoint cut-set links paths leads to an improvement in convergence and quality of the solutions making the algorithm more robust. When the network does not have a wavelength conversion capability, results indicate that for networks with up to 40 nodes, the DE-RWA algorithm obtains results that equal the NWR lower bound. For networks with a wavelength conversion capability all the NWR lower bounds were reached and the APL is improved with our DE-RWA algorithm compared with the case of networks without wavelength conversion capability, however this wavelength conversion capability implies a higher investment in the networks having to be equipped with wavelength converters. We have provided a practical DE model to solve the RWA problem that is simple to implement and could also be extended by adding other features in the objective function. As further work we propose to investigate the use of the DE-RWA algorithm including other features, such as impairment-aware, energy efficient, survivability, among others in optical networks and the use of intensification procedures like path-relinking to perform local search.

Chapter 4

DE APPLIED TO THE SURVIVABLE VTM PROBLEM

In IP-over-WDM networks, a virtual topology is placed over the physical topology of the optical network. Given that a simple link failure or a node failure on the physical topology can cause a significant loss of information, an important challenge is to make the routing of the virtual topology on to the physical topology survivable. This problem is known as survivable virtual topology mapping (SVTM) and is known to be an NP-complete problem. So far, this problem has been optimally solved for small instances by the application of integer linear programming and has been sub-optimally solved for more realistic instances by heuristic strategies such as ant colony optimization, and genetic algorithms. In this chapter we introduce the application of differential evolution (DE) to solve the SVTM problem and enhancements based on DE are proposed as well. Three algorithms based on DE are developed. The enhanced variants have better convergence rate, get better quality of solutions and require few control parameters. We present the impact of these parameters on the system's performance improvement. Algorithms are evaluated in different test bench optical networks, as NSFnet and USA, demonstrating that the enhanced DE algorithm overcomes the other two, for small instances. The three algorithms reach a 100% survivable mapping for small instances. The three algorithms also find positive survivable mappings and reduce the network wavelength links (NWL). Results show the effectiveness and efficiency of the proposed algorithms.

4.1 Introduction

Nowadays it is evident that telecommunications are developing almost exponentially worldwide in response to the ever-increasing bandwidth demand and transmission distances required in communication networks. The systems fit to cope with this exponential growth are led by optical systems using wavelength division multiplexing (WDM) technology.

In WDM networks the high bandwidth capacity can be divided into some non-overlapping channels operating at a different wavelength; through which the upper layers (IP, Ethernet, etc.) can transmit data [77]. Since every physical fiber link carries a huge amount of data, a simple link failure or a node failure, can cause a significant loss of information.

In view of this, optical networks have to be designed insuring their resilience in case of failures to avoid this loss of information. In this scenario, survivability, i.e. the ability of a network to withstand and recover from failures or attacks, is one of the most important requirements in today's networks. Its importance is magnified in fiber optic networks with throughputs in the order of gigabits and terabits per second [13].

More specifically, in IP-over-WDM networks, a virtual topology is placed over the physical topology of the optical network. The virtual topology consists of virtual links which are in fact the lightpaths in the physical topology. There are two main approaches to protect the IP-over-WDM networks: WDM protection level and IP restoration level [26]. In WDM protection level, backup paths for the virtual topology are reserved; this provides faster recovery for time-critical applications but is less resource efficient since the resources are reserved without knowledge of the failure. In IP restoration level, failures are detected by the IP routers, which adapt their routing tables and therefore no action is taken at the optical layer.

Considering the IP restoration level scenario, an important challenge is to make the routing of the virtual topology on to the physical topology survivable. Each virtual link (IP link) should be mapped on the physical topology as a lightpath, and usually a fiber physical link is used for more than one lightpath. Therefore a single failure of a physical link can disconnect more than one IP link in the virtual topology. To achieve the IP restoration level the virtual topology needs to remain connected after a failure occurs. Failures can be of many types: node failure, link failure or multiple link failures. In order to call a mapping survivable, we need to specify the type of failure that it has to survive. A single link failure is one of the most common failures in optical networks [132]. The problem of routing virtual links into a physical topology in such a way that the virtual topology (lightpaths set up on the physical network) remains connected after a physical link failure occurs is known as the Survivable Virtual Topology Mapping (SVTM) problem. This combinatorial problem is NP-complete [7].

The importance of the SVTM problem can be assessed by the number of approaches proposed in the literature to obtain optimal solutions for small instances and sub-optimal solutions for more realistic cases. Integer linear programming (ILP) models [7, 81] have been successfully used to solve the SVTM problem in small optical networks. However, as the network's size increases so does the dimension of the ILP model, whose solution typically requires an extensive computation effort (and execution time) which renders them impractical for medium to large scale networks. Therefore, different heuristic-based algorithms have been proposed. Among them are the evolutionary algorithms (EA), ant colony optimization (ACO), etc. Ducatell and Gambardela [79] present a local search algorithm which can provide survivable routing in the presence of physical link failures. Their algorithm can easily be extended for the cases of node failures and multiple simultaneous link failures. Another approach is presented by Kurant and Thiaran [78], in which the problem or task is divided into a set of independent and simple subtasks. The combination of solutions of these subtasks is a survivable mapping. Finally, nature-inspired heuristics have been used to solve the SVTM. Ergin et al. [82] and Kaldirim et al. [77] present an efficient EA and a suitable ACO algorithm respectively to find a survivable mapping of a given virtual topology while minimizing the resource usage. In [133] a comparison between both algorithms, ACO and EA, is presented.

Differential evolution (DE) is a very simple but very powerful stochastic global optimizer for a continuous search domain. It was proposed by Storn and Price [14], to represent a very complex process of evolution. Intelligently using the differences between populations (a population is a set of candidate solutions) and with the manipulation of few control parameters, they created a simple but fast linear operator called differentiation, which makes DE unique. Additionally, studies show that DE in many instances outperforms other evolutionary algorithms [10, 21, 126]. More specifically DE exploits a population of potential solutions to effectively probe the search space. The algorithm is initialized with a population of random candidate solutions, conceptualized as individuals. For each individual in the population, a descendant is created from three parents. One parent, the main, is disturbed by the vector of the difference of the other two parents. If the descendant has a better performance resulting in the objective function, it replaces the individual. Otherwise, the individual is retained and is passed onto the next generation of the algorithm. This process is repeated until it reaches the termination condition. A complete theoretical analysis of the algorithm is presented in [14].

In this chapter we study the application of a differential evolution (DE) algorithm to the SVTM problem. This paper extends the results of a former work [18]. In [18] a simple example on how to apply DE to the SVTM problem was presented for a small network. With respect to that former work, in this paper three algorithms based on DE are presented, two basic algorithms named BI-DE-VTM and BII-DE-VTM, and an enhanced one named E-DE-VTM. A careful analysis on the impact of DE parameters was carried out. Then, results are presented for networks with up to 40 nodes, showing that the three algorithms reach a 100% survivable mapping for small instances. The three algorithms find positive survivable mappings and reduce the network wavelength links (NWL). Moreover, the enhanced algorithm E-DE-VTM overcomes the other two, for small instances. To the best of our knowledge, our work is the first application of a DE algorithm to the SVTM problem.

The chapter layout is as follows, the SVTM problem formulation is introduced in Sect. 4.2. The proposed search algorithm is discussed in Sect. 4.3. Section 4.4 introduces the modification to the basic DE algorithm BI-DE-VTM in order to create BII-DE-VTM and E-DE-VTM algorithms. Sect. 4.5 presents and explains an illustrative example of the BI-DE-VTM. Parameter tuning and results are then presented in Sect. 4.6, and conclusions are addressed in Sect. 4.7.

4.2 Problem Formulation

The physical network is modelled as an undirected graph $G_p = (V_p, E_p)$, where V_p is the set of physical nodes numbered $\{1, 2, ..., |V_p|\}$, and E_p is the set of physical links $\{e^p_{i,j}, i, j \in V_p\}$ with cardinality $|E_p|$. In a similar way, the virtual topology is modelled as an undirected graph $G_v = (V_v, E_v)$ in which V_v is a subset of V_p, and E_v is a set of virtual links representing lightpaths on the physical topology.

In the physical topology, $e^p_{i,j}$ is in E_p if there is a link between nodes i and j. On the other hand, the virtual topology has a set of edges (lightpaths) E_v, where an edge $e^v_{s,d}$ exists in E_v if both nodes s and d are in V_v and there is a lightpath between them.

There is not a SVTM that protects the network against all component failures, therefore we need to define what kind of failure our network is able to survive, i.e. single link failures, node failure, multiple link failure, etc. Our analysis considers single link failure which is one of the most common in optical networks [132] and it is also less cumbersome to analyze.

To illustrate the SVTM, consider the physical topology (WDM network) presented in Fig. 4.1(a). Figure 4.1(b) presents a virtual topology with virtual links $e^v_{1,3}$, $e^v_{3,4}$, $e^v_{4,5}$, $e^v_{2,5}$ and $e^v_{1,2}$, which in fact are lightpaths (IP links) that need to be mapped on the WDM network. Figure 4.1(c) presents a SVTM against a single link failure. Observe that a single link failure disconnects at most one virtual link of the virtual topology, so the virtual topology remains connected achieving the IP restoration level. To show a not survivable mapping, in Fig. 4.1(d) we have routed the lightpath $e^v_{2,5}$ through the physical links $e^p_{2,4}$ and $e^p_{4,5}$. It can be seen that a failure in the physical link $e^p_{4,5}$ disconnects the virtual links $e^v_{2,5}$ and $e^v_{4,5}$ of the virtual topology (dotted lines in Fig. 4.1(b)), leaving the virtual node 5 isolated in the virtual topology, which clearly indicates that the mapping is not survivable.

Lightpaths can be mapped taking into account different metrics along with the requirement of survivable mapping, such as the reduction of wavelengths to connect the lightpaths, or the number of wavelength links. Based on the ILP formulations previously done in [7], we have chosen to reduce the number of wavelength links (NWL), since this metric gives a

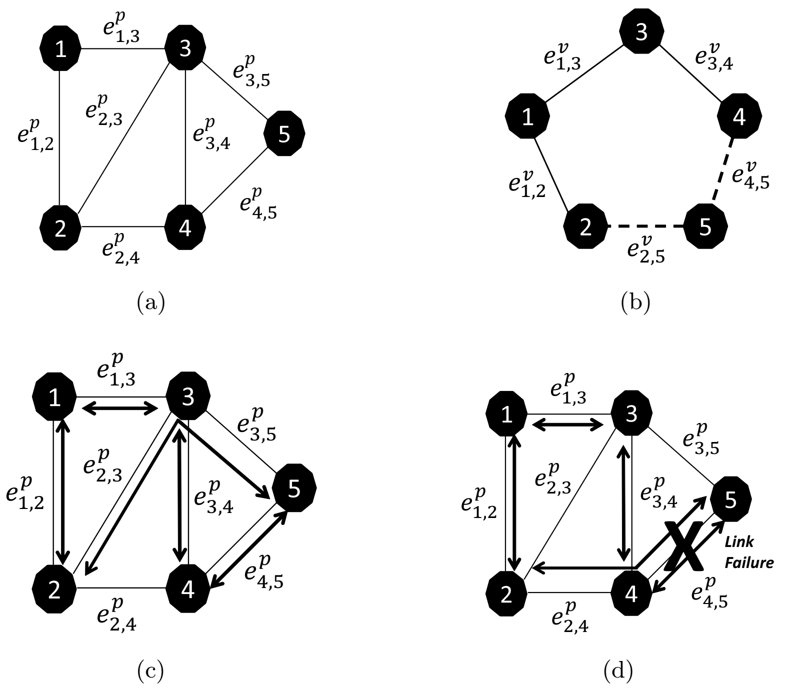


Figure 4.1: Illustrative survivable and unsurvivable virtual topology mapping for a simple 5-nodes network. (a) Physical topology. (b) Virtual topology. (c) Survivable mapping. (d) Unsurvivable mapping. Solid arrows in Figs. 4.1(c) and 4.1(d) represent virtual links mapped onto the physical topology.

better idea of the use of the resources in the network.

A wavelength link is defined as a physical link that a lightpath traverses in the virtual topology. The NWL is defined as:

$$NWL = \sum lp_{s,d} \quad \forall s \neq d \in V_v \tag{4.1}$$

where $lp_{s,d}$ is the length of the path in number of hops that connects nodes (s, d) in the virtual topology.

4.3 Differential Evolution (DE) Algorithm

We applied the DE algorithm to find a SVTM in the network. As mentioned before, DE algorithm uses a population (Pop) of individuals and iterates by creating new populations until an optimal or near optimal solution is obtained. The individuals represent specific solutions to the problem, so that, an encoding that is well-suit to it is necessary. Our approach, inspired in [67], assumes that a set of paths to meet demand is available. Therefore, in an initialization step we calculate k paths between all pairs of nodes, using the k-shortest paths algorithm [129]. The pre-calculation of paths could be based on hops or distance. For simplicity, the paths are pre-calculated based on hops and are considered to be bidirectional.

To encode our individuals we generate a vector of dimension D, for example:

$$X = [k_{1\to2}, k_{1\to3}, k_{1\to4}, ..., k_{N-1\to N}] \tag{4.2}$$

where D is equal to the number of virtual links (lightpaths connections) between virtual nodes in the virtual network, N is the number of virtual nodes (i.e. $N = |V_v|$) and $k_{i\to j}$ represents the selection of an available path between nodes i and j. Each virtual link is assigned a position in the individual, starting with virtual link $e^v_{1,2}$ (if it exists), and proceeds (in a row major order) to virtual link $e^v_{N-1,N}$.

DE algorithms are run for a limited number of iterations (generations). At the beginning of the algorithm, assuming we do not have information about the optimum, the initial population is created randomly. DE employs repeated cycles of recombination and selection to guide the population towards the vicinity of a global optimum. We apply the probability operators which are crossing and mutation to each individual in a population to obtain new individuals (children). These new individuals have some properties of their ancestors. These ancestors are kept or deleted by selection. The term generation is used to designate the conversion of all individuals into new ones, i.e. to move from one population to another.

DE has three crucial control parameters: the mutation constant (M), which controls the mutation strength, the recombination constant (RC) to increase the diversity in the mutation process and the population size (NP). Throughout the execution process, the user defines M and RC values in the range of [0,1], and the Pop size NP.

At each generation, all individuals in the Pop are evaluated in turn, the individual being evaluated is called the target vector. For each target vector $x^i, i = 1, \ldots, NP$, a mutant individual m^i is generated according to:

$$m^i = x^{r1} + M * (x^{r2} - x^{r3}) \tag{4.3}$$

where $x^{r1}, x^{r2}, x^{r3} \in Pop$: $x^{r1} \neq x^{r2} \neq x^{r3} \neq x^i$. x^{r1}, x^{r2} and x^{r3} are three random individuals from the Pop, mutually different and also different from the current target vector x^i, and the mutation constant M is a scaling factor in the range of $[0, 1]$. The M operator is used to control the magnitude of the difference between two individuals in Eq. (5.6). This operator allows us to manage the trade-off between exploitation and exploration in the search process.

Note from Eq. (5.6), that the elements of m^i could be non-integer values. For this reason, the round operation is applied to all elements of m^i to guarantee integer values.

Then, the recombination operator RC is applied to increase the diversity in the mutation process. This operator is the last step in the creation of the trial vector t^i. To create the trial vector, the mutant individual, m^i, is combined with the current target vector x^i. Particularly, for each component j, where $j = \{1, 2, \ldots, D\}$, we choose the jth element of the m^i with probability RC, otherwise from the x^i. If we choose a random number $rand$ in the $[0, 1]$ interval, then the t^i is created as follows:

$$t^{i,j} = \begin{cases} m^{i,j} & \text{if } rand < RC \\ x^{i,j} & \text{otherwise} \end{cases} \tag{4.4}$$

After we create the trial vector t^i, it is necessary to verify the boundary constraints of each element of t^i to avoid creating an infeasible solution. This could happen because any jth element created by Eq. (5.6) that is not in the allowed range of the specification of a problem has an RC probability of being selected. If any element of the trial vector violates the constraints it is replaced with a random number in the allowed range.

Finally, the selection operator is applied; this operator is a simple rule of elitist selection of the vectors that improve the objective function. This is done by comparing the fitness between the trial vector and the target vector in the objective function using:

$$pop_k = \begin{cases} t^i & \text{if } f(t^i) < f(x^i) \\ x^i & \text{otherwise} \end{cases} \tag{4.5}$$

where pop_k is the population of the next generation, that changes by accepting or rejecting new individuals. The best individual in the population and the global best individual are kept at the end of each generation, to keep track of the best solution found so far.

4.3.1 Fitness of an Individual

Our objective is to minimize the NWL (Eq. (4.1)), which means that the length of the chosen paths must be as short as possible while considering the survivability requirement. This objective is used to evaluate the fitness of an individual. We include penalties in the objective function when the survivability requirement is not met. To count the penalties added to a solution, we erase each physical link one by one and count how many of these removed links disconnect the virtual topology. According to this, the fitness of an individual is given by:

$$Fitness = NWL + w * p \tag{4.6}$$

Where NWL is the sum of the paths's length, p is the number of links which disconnects the virtual network and w is a weighted factor multiplying p.

4.3.2 Pseudocode of the DE Algorithm

Under these considerations a pseudocode of the basic DE algorithm BI-DE-VTM is presented in algorithm 9:

Algorithm 7 BI-DE-VTM pseudocode

```
Set the control parameters $M$, $RC$ and $NP$.
Create an initial Pop.
Evaluate the fitness of every individual.
repeat
  for each individual $x \in Pop$ do
    Select three individuals from Pop.
    Apply mutation Eq. (5.6).
    Apply round operation over the mutant individual.
    Apply recombination Eq. (5.7).
    Verify boundary constraints.
    if Boundary constraints are violated then
      modify the infeasible elements.
    end if
    Apply selection operator Eq. (5.8).
    Update Pop.
  end for
until a satisfactory solution is obtained or a computational limit is exceeded.
```

4.4 Two Enhanced DE-VTM Algorithms

We develop two modifications of the DE-VTM algorithm that make it more efficient and robust. We refer to the application of the algorithm without any modification as basic DE algorithm BI-DE-VTM. The two algorithms that we propose, which are modifications of the BI-DE-VTM are called BII-DE-VTM and enhanced DE algorithm E-DE-VTM, both are explained in Sects. 4.4.1 and 4.4.2 respectively.

4.4.1 BII-DE-VTM Algorithm

In this algorithm we define a variation of the DE algorithm's main operator that will suit our purpose better. In Eq. (5.6), the new mutant individual is created over a continuous space based only in the addition of the scaled difference between two individuals to another one ($x^{r1}+M*(x^{r2}-x^{r3})$). We observed that the "arithmetic operation" may lead to infeasible solutions and the single operator "+" limits the diversity of the new possible mutant vector. So we redefine Eq. (5.6) as follows:

$$m^i = \lfloor x^{r1} \pm M * (x^{r2} - x^{r3}) \rfloor \tag{4.7}$$

Given the nature of the problem and the allowed range of elements of m^i (i.e. in the range of $[1-k]$), if we just used the operator "+", then when the scale difference between x^{r2} and x^{r3} is positive and x^{r1} has a big value (near k), the result of the mutant operation may lead to an infeasible element in m^i. On the contrary, if in the same case we use the operator "-", it is more probably that a feasible element will be obtained. That is the reason why the operator "±" could lead to more diversity in the creation process. The use of either the "+" or the "-" operator is decided randomly for simplicity.

4.4.2 E-DE-VTM Algorithm

In this algorithm we use some features of the VTM problem. Specifically, we modified the BI-DE-VTM to obtain shorter paths and achieve a lower NWL.

First, we have made a slight modification to the main operator of the classic DE:

$$m^i = \lfloor x^{r1} \pm M * |(x^{r2} - x^{r3})| \rfloor \tag{4.8}$$

By introducing the absolute value in the difference $|(x2 - x3)|$ we limit the outcome of this difference to positive numbers. Then from Eq. (4.7) it is clear that we can get two possible individuals depending on which operator we use. For instance, if m^i_{o1} is the mutant individual obtained with the "+" operator and m^i_{o2} is the one obtained with the "-" operator, then:

$$m^i_{o1} = \lfloor x^{r1} + M * |(x^{r2} - x^{r3})| \rfloor \tag{4.9}$$

$$m^i_{o2} = \lfloor x^{r1} - M * |(x^{r2} - x^{r3})| \rfloor \tag{4.10}$$

We noticed that m^i is statistically related to the hop length of the path; the shorter the value of m^i the shorter the length of the path. Instead of randomly choosing between these two mutant individuals, as we do in BII-DE-VTM, we calculate a probability that reflects the length of the path each represented as: $h_1 = 1/m^i_{o1}$ and $h_2 = 1/m^i_{o2}$, where h_1 and h_2 are the probabilities for individuals m^i_{o1} and m^i_{o2} respectively. The individual that represents a shortest path accounts for higher probability and is, therefore, preferred over the others, as it improves the NWL.

For the second and third modifications of the original algorithm, we need an array of costs (measured as number of hops) of each pre-calculated path between nodes. For instance, let's say we have an array K_{cost} of dimension NxN. $K_{cost(i,j)}$ will have a vector of dimension k (number of pre-calculated paths) with the costs of those pre-calculated paths between nodes (i, j). This is:

$$K_{cost(i,j)} = [cost_{K_1}, cost_{K_2}, ..., cost_{K_k}] \tag{4.11}$$

With this information, the second modification is done if boundary constraints are violated. Instead of randomly generating a path, we can calculate probabilities for the feasible paths. Suppose that we have k paths available for every pair of nodes. Then, we will have an available vector $K_{cost(i,j)} = [cost_{K_1} cost_{K_2} ... cost_{K_k}]$, representing the cost of the k shortest path between nodes (i, j). If we compute the reciprocal of the elements in $K_{cost(i,j)}$, we get different probabilities for every shortest path based on their lengths. By using these probabilities to select the elements that will replace those that are unfeasible, the shorter paths are more likely to be chosen.

The third modification is included to avoid stagnation, by preventing the creation of an individual that already exists. If we detect that an identical individual has been created, we randomly choose one of its elements and modify this single element by using the method of probabilities based on costs as explained before.

By including these three simple modifications the algorithm becomes more efficient and robust, as we will show in the results section. A pseudocode including these three modifications is presented in algorithm 8.

4.5 DE for the Survivable VTM Problem

We present an illustrative example of our BI-DE-VTM algorithm, using the simple physical network of five nodes shown in Fig. 4.1(a), and the virtual topology from Fig. 4.1(b).

As mentioned in Sect. 4.3, our approach assumes that a set of pre-calculated paths is available. In an initialization step, we calculate k shortest paths between all pairs of nodes. Table 5.1 shows $k = 4$ shortest paths for the 5-node network.

Now, each virtual link from Fig. 4.1(b) must be represented with an element of an individual in DE. So, with a row major order, for this illustrative example an individual

Algorithm 8 E-DE-VTM Pseudocode

```
Set the control parameters $M$, $RC$ and $NP$
Create an initial Pop.
Evaluate the fitness of every individual.
repeat
  for each individual $x \in Pop$ do
    Select three individuals from Pop.
    Apply mutation Eq. (4.8).
    Apply recombination using probabilities based on length of the paths Eq. (5.7).
    Verify boundary constraints.
    if Boundary constraints are violated then
      modify the infeasible elements using probabilities based on length of the paths.
    end if
    while Individual is not different from the rest do
      modify one random element using probabilities based on length of the paths.
    end while
    Apply selection operator Eq. (5.8).
    Update Pop.
  end for
until a satisfactory solution is obtained or a computational limit is exceeded.
```

will have the form of $X = [k_{1\to2}, k_{1\to3}, k_{2\to5}, k_{3\to4}, k_{4\to5}]$, representing selected routes for the virtual topology. Now, as stated in algorithm 9, we first set the control parameters. For this illustrative example the values are set to $NP = 5$, $M = 0.2$ and $CR = 0.5$. Then, an initial random population is created and the fitness of each particle is calculated as:

$$pop = \begin{cases} x^1 = [4,3,1,3,2] & fitness = 213 \\ x^2 = [4,1,2,1,4] & fitness = 162 \\ x^3 = [1,4,3,2,3] & fitness = 263 \\ x^4 = [3,3,2,1,1] & fitness = 110 \\ x^5 = [3,2,4,1,3] & fitness = 312 \end{cases} \tag{4.12}$$

To explain how the fitness of each individual is calculated, consider the individual x^4:

$$x^4 = [3,3,2,1,1] \tag{4.13}$$

The elements in the vector of individual x^4 indicate which routes (marked in bold) from Table 5.1 are selected. For instance, the first element in the vector, which is a 3, indicates that for the virtual link $e^v_{1,2}$ the third route ($k_{1\to2} = [1-3-4-2]$) is selected. The second

Table 4.1: k-shortest paths for a 5-node net. The selected paths for the individual in the example are in bold.

Node	1	2	3	4	5
1	[]	[1, 2] [1, 3, 2] **[1, 3, 4, 2]** [1, 3, 5, 4, 2]	[1, 3] [1, 2, 3] **[1, 2, 4, 3]** [1, 2, 4, 5, 3]	[1, 2, 4] [1, 3, 4] [1, 2, 3, 4] [1, 3, 2, 4]	[1, 3, 5] [1, 2, 3, 5] [1, 3, 4, 5] [1, 2, 4, 5]
2		[]	[2, 3] [2, 1, 3] [2, 4, 3] [2, 4, 5, 3]	[2, 4] [2, 3, 4] [2, 1, 3, 4] [2, 3, 5, 4]	[2, 4, 5] **[2, 3, 5]** [2, 3, 4, 5] [2, 1, 3, 5]
3			[]	**[3, 4]** [3, 2, 4] [3, 5, 4] [3, 1, 2, 4]	[3, 5] [3, 4, 5] [3, 2, 4, 5] [3, 1, 2, 4, 5]
4				[]	**[4, 5]** [4, 3, 5] [4, 2, 3, 5] [4, 2, 1, 3, 5]
5					[]

element in the vector, which is also a 3, indicates that for the virtual link $e^v_{1,3}$ also the third route ($k_{1\to3} = [1-2-4-3]$) is selected, and so on.

For this individual, considering Eq. (4.1), we get an $NWL = 10$, which is the total length of the five selected routes. Moreover, with this solution if a failure occurs in physical links $e^p_{2,4}$ and $e^p_{3,4}$ the virtual topology becomes disconnected, which give us a penalty of 2. Finally, the fitness of the individual using Eq. (4.6), with an arbitrary weight factor of $w = 50$, is given by:

$$Fitness = NWL + w * p = 10 + 50 * 2 = 110 \tag{4.14}$$

Using the same methodology, the fitness of each individual is calculated.

We illustrate the application of the mutation, recombination and selection operators of DE. As stated in Sect. 4.3, from the initial *Pop*, we select a target individual x^i. Suppose that our target vector in the first generation is x^1. Then, three random individuals x^{r1}, x^{r2} and x^{r3} mutually different and also different from the current target vector are selected from the *Pop* as well. For this example, let $r1 = 4$, $r2 = 3$ and $r3 = 5$. Figure 4.2 shows the application of the mutation operator (Eq. (4.7)) to get the mutant individual m^i.

Once we have the mutant individual m^i, we apply the recombination operator (Eq. (5.7))

$$\begin{array}{ccccccccc}
\lfloor x^4 & \pm & M & * & (x^3 & - & x^5)\rfloor & = & m^i \\
\downarrow & & & & \downarrow & & \downarrow & & \downarrow \\
\lfloor 3 & \pm & 0.2 & * & (1 & - & 3)\rfloor & = & 2 \\
\lfloor 3 & \pm & 0.2 & * & (4 & - & 2)\rfloor & = & 4 \\
\lfloor 2 & \pm & 0.2 & * & (3 & - & 4)\rfloor & = & 2 \\
\lfloor 1 & \pm & 0.2 & * & (2 & - & 1)\rfloor & = & 2 \\
\lfloor 1 & \pm & 0.2 & * & (3 & - & 3)\rfloor & = & 1
\end{array}$$

Figure 4.2: Mutant individual generation.

to combine the target individual and the mutant individual. Figure 4.3 illustrates how the trial individual t^i is formed with elements chosen from the target individual x^1 and the mutant individual m^i. The elements are chosen with probability RC.

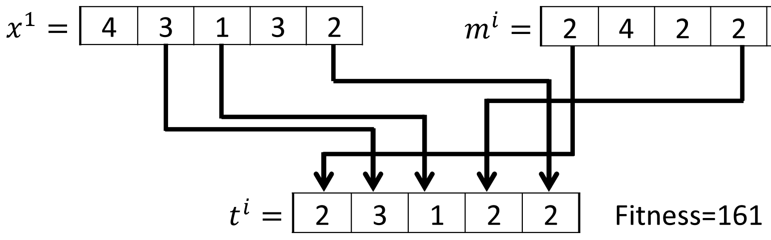
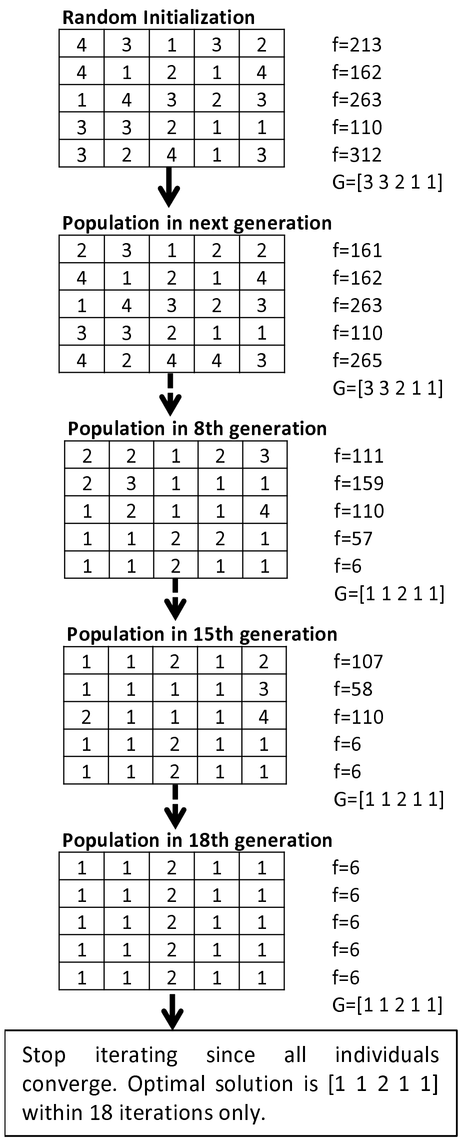


Figure 4.3: Trial individual generation. Elements to create t^i are taken from the target individual and the mutant individual with probability RC.

Finally, the selection operator (Eq. (5.8)) is applied between the target individual x^1 and the trial individual t^i. This operator is a simple rule of elitist selection. The individual with the best fitness value will survive to the next generation. In this example, the trial vector t^i will replace x^1 in the next generation because $f(t^i) = 161 < f(x^1) = 213$.

The process is repeated for each individual in *Pop* to form the population of the next generation. The algorithm stops when the termination condition is met.

The capability of the applied BI-DE-VTM is illustrated in Fig. 4.4. The DE algorithm finds the optimal SVTM within 18 iterations.



Random Initialization

4	3	1	3	2	f=213
4	1	2	1	4	f=162
1	4	3	2	3	f=263
3	3	2	1	1	f=110
3	2	4	1	3	f=312

G=[3 3 2 1 1]

Population in next generation

2	3	1	2	2	f=161
4	1	2	1	4	f=162
1	4	3	2	3	f=263
3	3	2	1	1	f=110
4	2	4	4	3	f=265

G=[3 3 2 1 1]

Population in 8th generation

2	2	1	2	3	f=111
2	3	1	1	1	f=159
1	2	1	1	4	f=110
1	1	2	2	1	f=57
1	1	2	1	1	f=6

G=[1 1 2 1 1]

Population in 15th generation

1	1	2	1	2	f=107
1	1	1	1	3	f=58
2	1	1	1	4	f=110
1	1	2	1	1	f=6
1	1	2	1	1	f=6

G=[1 1 2 1 1]

Population in 18th generation

1	1	2	1	1	f=6
1	1	2	1	1	f=6
1	1	2	1	1	f=6
1	1	2	1	1	f=6
1	1	2	1	1	f=6

G=[1 1 2 1 1]

Stop iterating since all individuals converge. Optimal solution is [1 1 2 1 1] within 18 iterations only.

Figure 4.4: Illustration of differential evolution simulation.

4.6 Numerical Results and Discussion

The numerical results section is divided in two parts. First we present the parameter tuning and then the performance of the DE algorithms in different network topologies. To gauge the effect of different parameters in the quality of the solutions, we analyzed three metrics, which are the success rate (SR), which is the percentage of the number of times the algorithm was able to find a survivable mapping for a given virtual topology (VT). The NWL which is being optimized and represents the number of wavelength links used on the mapping,

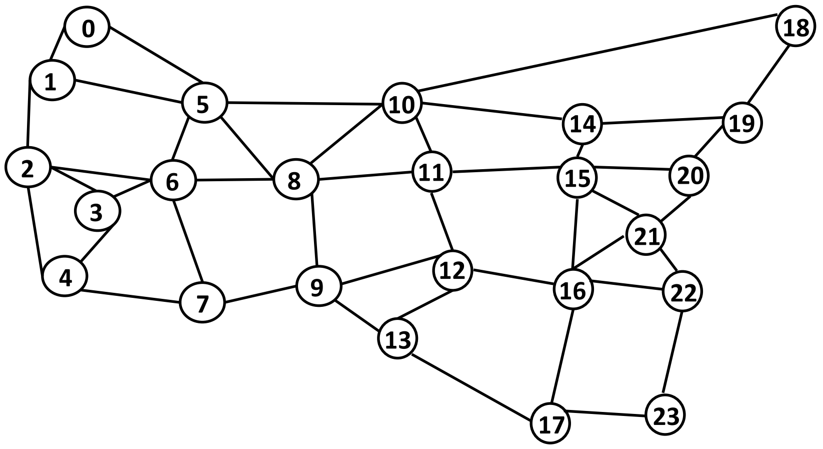


Figure 4.5: Telco Net topology with 24 nodes and 43 links.

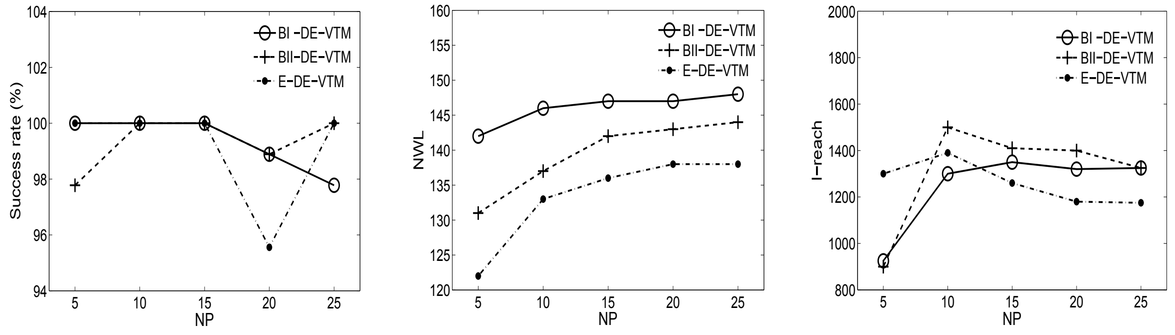
and finally the iteration reach ($I - reach$) indicating the objective function evaluation (OFE) number of iteration in which algorithms achieve their best values. In the three developed DE-VTM algorithms, the number of OFEs per generation is proportional to the population size NP, so that $I - reach = (G - reach) * NP$, where $G - reach$ is the generation in which the algorithms achieve their best results.

4.6.1 Tuning the Parameters

We randomly generated 30 VTs of degree 3 and performed 3 tests on each VT over the physical Telco-Net topology shown in Fig. 4.5. Results in this section are the average of those 90 tests.

Using a set of default values of $NP = 15$, $M = 0.5$ and $CR = 0.5$, we use a sweeping technique to assess their impact on the solutions, one at a time. The first $k = 10$ shortest paths are calculated based on hops for simplicity. We also have limited the algorithms to run for 300 generations or a maximum of 300 seconds. This number of generations and time value was chosen after some preliminary test, because we observed that the solutions do not present a big change around those values. The algorithms also stop when the best so far solution does not change after 100 generations. Our experimentation was carried out on a 2.35 GHz, Pentium R Dual-Core PC.

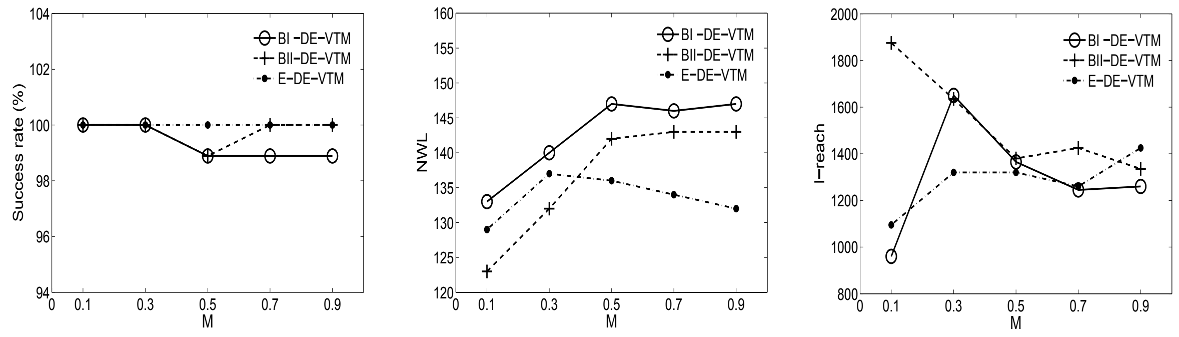
Figure 4.6 shows results of SR, NWL and $I - reach$ respectively for the three DE-VTM algorithms when the NP parameter varies from 5 to 25. These three metrics have to be considered simultaneously since a good solution is obtained not only when a small NWL value is reached but also when a high SR value is obtained. Figure 4.6(a), shows that a success rate of 100% is reached for the three algorithms when NP is equal to 10 and 15. However, when $NP = 20$, E-DE-VTM gets its lowest value of about 95.5 % that means only 4 out of 90 tests did not obtain a survivable mapping. Also, Fig. 4.6(b) shows that the greater the NP value the greater the NWL. This is because the greater the population the greater the number of iterations to minimize NWL. However, E-DE-VTM obtains the lowest NWL of the three DE algorithms. Second best results regarding NWL are obtained by BII-DE-VTM. Finally, Fig. 4.6(c) shows that the $I - reach$ is worst for the BI-DE-VTM for populations equal and greater than 15. The E-DE-VTM has a stable $I - reach$ value even when the population grows. We also noticed that the NP parameter has to be small to obtain a good value for the NWL, however caution must be taken since in the BII-DE-VTM algorithm, small values can lead to stagnation.



(a) Success rate against NP. (b) NWL against NP. (c) $I - reach$ against NP.

Figure 4.6: SR, NWL and $I - reach$ varying NP parameter with $RC = 0.5$ and $M = 0.5$ for Telco-Net.

Figure 4.7 shows SR, NWL and $I - reach$ when the M parameter is varied. Figure 4.7(a) shows that the M parameter does not affect the success rate for E-DE-VTM and in general a small value for M is preferred for the three algorithms. Also from Fig. 4.7(b) we conclude that a small value of M is more suitable to obtain a better NWL for the three algorithms. Finally, Fig. 4.7(c) shows that for small values of M the $I - reach$ value is better for the E-DE-VTM. When M has a value over $M = 0.5$ the three algorithms have a similar behavior. When $M = 0.1$ and $M = 0.3$ the algorithms perform very differently, this is expected due that lower the value of M lower the convergence of the algorithms. However, when $M \geq 0.5$, the algorithms perform similarly regarding $I - reach$ but NWL results are worst. Contrary of what M parameter represent (stride of change), it is evident from the analysis that a big value of M leads to a worse performance. It seems that a big value of M may lead to stagnation in sub-optimal solutions. For this particular problem, it looks that small steps in the exploration of the solution space are preferred to obtain better solutions. Note that the priority of the different metrics is not the same, it is more important first to maximize the SR, then minimize the NWL and after that minimize the $I-reach$.



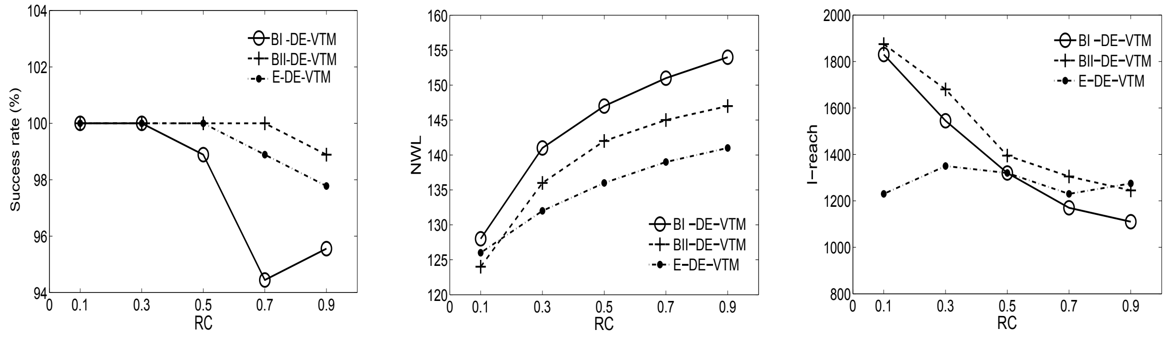
(a) Success rate against M. (b) NWL against M. (c) $I-reach$ against M

Figure 4.7: SR, NWL and $I-reach$ varying M parameter with $RC = 0.5$ and $NP = 15$ for Telco-Net.

Figure 4.8 shows the results when the parameter RC varies from 0.1 to 0.9. From Fig. 4.8(a), it can be observed that the three algorithms are affected when RC grows. BI-DE-VTM is the most sensible to variations of the RC parameter with respect to the SR metric. Note that a $SR = 94$ % is not a bad result, means that only 5 out of 90 tests did not find a survivable mapping. However, BII-DE-VTM and E-DE-VTM are more robust for greater values of RC improving the SR as observed in the figure. Figure 4.8(b) clearly shows that a small value of RC is needed to get a small NWL. Finally, in Fig. 4.8(c) it can be observed that E-DE-VTM is very stable to the variation of the RC parameter with respect to $I-reach$.

After doing the sweeping of the parameters with the default configuration, we noticed that small values of NP, M and RC led us to better results. So, we decided to test some of the parameters not included in the first tuning in order to get a more suitable set for the different DE-VTM algorithms.

Table 4.2 presents the results for the three metrics using different groups of NP, M and RC parameters. Considering the BI-DE-VTM column, it can be seen that the BI-DE-VTM is very sensible to small NP values, which leads to a small SR value in most of the cases when



(a) Success rate against RC. (b) NWL against RC. (c) $I - reach$ against RC.

Figure 4.8: SR, NWL and $I - reach$ varying RC parameter with $M = 0.5$ and $NP = 15$ for Telco-Net.

$NP = 5$. Also, the best NWL value is obtained when $NP = 15$ (marked in bold). Therefore, for the BI-DE-VTM the best values of SR and NWL is obtained when $NP = 15$, $M = 0.1$ and $RC = 0.1$.

Table 4.2: Summary of the parameter tuning.

		BI-DE-VTM			BII-DE-VTM			E-DE-VTM		
		SR	NWL	$I - reach$	SR	NWL	$I - reach$	SR	NWL	$I - reach$
$NP = 5$	$M = 0.1$ $RC = 0.1$	86.66	138	400	98.88	126	710	**100**	**112**	**925**
	$M = 0.5$ $RC = 0.1$	100	125	1165	98.88	118	1140	**100**	**112**	**1220**
	$M = 0.1$ $RC = 0.5$	57.77	152	110	93.33	140	265	98.88	116	800
$NP = 10$	$M = 0.1$ $RC = 0.1$	100	124	1130	**100**	**113**	**1770**	100	118	1110
	$M = 0.5$ $RC = 0.1$	100	124	1640	**100**	**113**	**2570**	100	119	1400
	$M = 0.1$ $RC = 0.5$	97.77	141	440	100	118	1430	100	132	1240
$NP = 15$	$M = 0.1$ $RC = 0.1$	**100**	**118**	**1770**	100	115	1903	100	124	1110
	$M = 0.5$ $RC = 0.1$	100	128	1830	100	125	1860	100	126	1230
	$M = 0.1$ $RC = 0.5$	100	133	960	100	122	1965	100	129	1095

Considering the BII-DE-VTM and E-DE-VTM columns, the best values of SR, NWL and $I - reach$ is obtained when $RC = 0.1$ and $M = 0.1$. Also, it is interesting to note that E-DE-VTM reaches a $SR = 100$ with $NP = 5$, which means that stagnation is avoided even when considering a small population.

4.6.2 DE-VTM Application on Different Topologies

After performing the parametric analysis, we applied the algorithms to two real sized networks, NSFnet and USA. Their topologies are shown in Figs. 4.9 and 4.10 respectively.

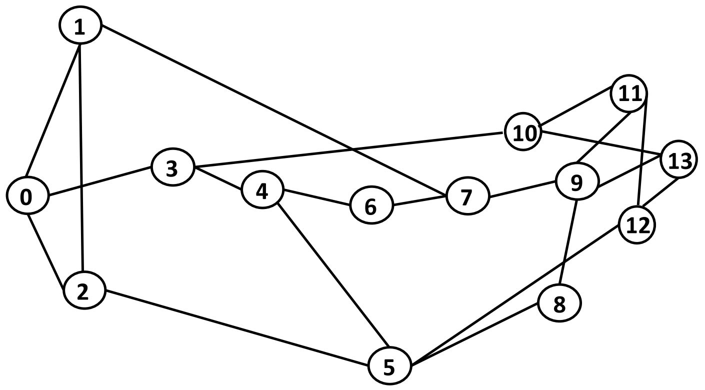


Figure 4.9: NSFnet topology. 14 nodes, 21 links.

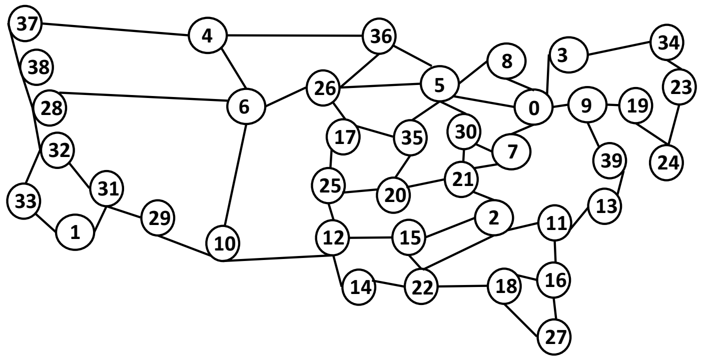
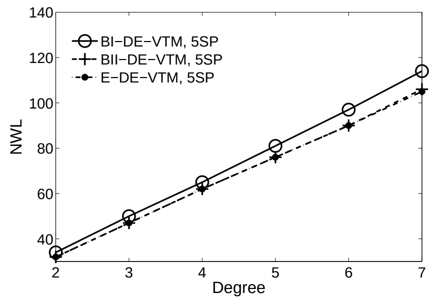


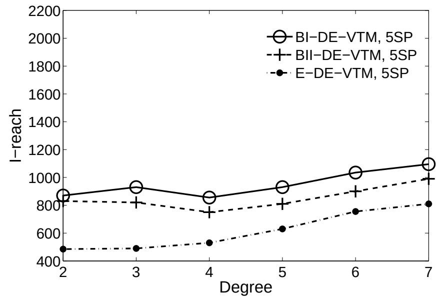
Figure 4.10: USA network topology. 40 nodes, 58 links.

We randomly generated 25 VTs of degree 2 up to 7 and perform 3 tests for each VT over the physical topologies NSFnet and USA. Results in this section are the average of 75 tests for VTs of the same degree. We used a set of values found from our parameter tuning methodology. $M = 0.1$ and $RC = 0.1$ were used in the three algorithms. For the BI-DE-VTM, BII-DE-VTM and E-DE-VTM, we used a value of $NP = 15$, $NP = 10$ and $NP = 5$ respectively. Results reported consider a different number of shortest-paths in order to appreciate the effect of this variation. To be fair with the algorithms, we again limited the algorithm to run up to 300 generations or 300 seconds. The algorithm also stops when the best solution does not change after 100 generations.

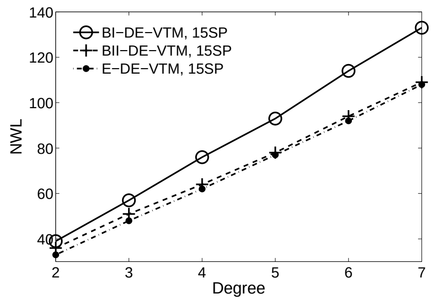
The three algorithms reach a SR of 100 % for the NSFnet topology when the random



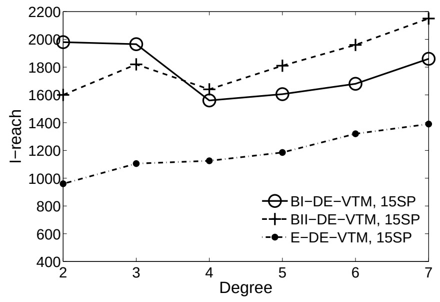
(a) NWL with 5 shortest-paths.



(b) $I - reach$ with 5 shortest-paths.



(c) NWL with 15 shortest-paths.



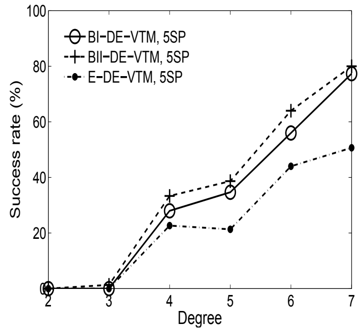
(d) $I - reach$ with 15 shortest-paths.

Figure 4.11: NWL and $I - reach$ mapping VTs from 2 up to 7 degree for NSFnet when 5 and 15 shortest-paths are used.

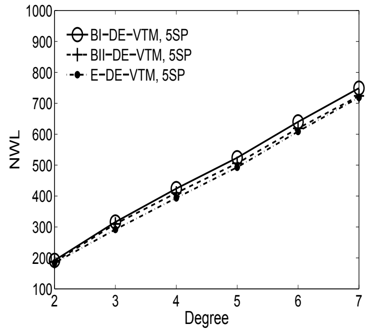
VTs were greater than degree 2. On the contrary, the algorithms were not able to find any survivable mapping for VTs of degree 2.

Figure 4.11 shows the NWL and $I - reach$ when 5 and 15 shortest paths are used for the NSFnet. As we expected, Fig. 4.11(a) shows that the BI-DE-VTM has the worst performance out of the three algorithms regarding the NWL, given NWL values greater than the obtained by the other two algorithms. Also, from Fig. 4.11(c) it can be observed that when the number of shortest paths changes from 5 to 15, the performance of the BI-DE-VTM is even worse, increasing the NWL values compared with the NWL of the other two algorithms. However, BII-DE-VTM and E-DE-VTM have a similar performance in both cases. Nevertheless, the E-DE-VTM is slightly more stable than the BII-DE-VTM when the number of shortest paths changes.

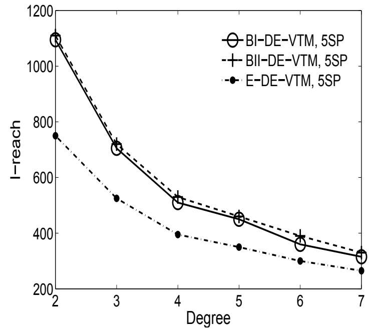
The advantage of the E-DE-VTM algorithm over the BII-DE-VTM is more evident when the $I - reach$ was analyzed. Figure 4.11(b) shows that E-DE-VTM has a better performance than the other two algorithms when 5 shortest-path are used. Also, from Fig. 4.11(d), it can be observed that the number of iterations required to reach the best fitness value is



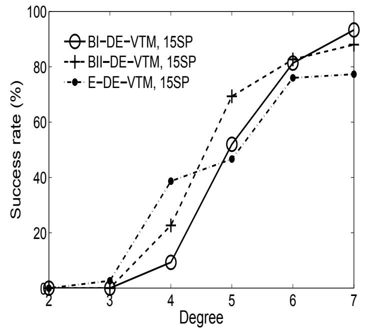
(a) SR with 5 shortest-paths.



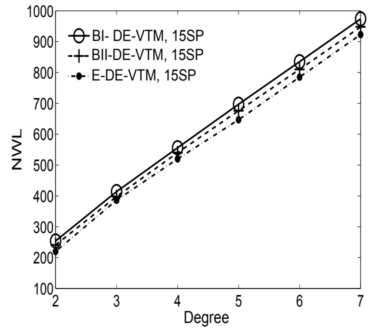
(b) NWL with 5 shortest-paths.



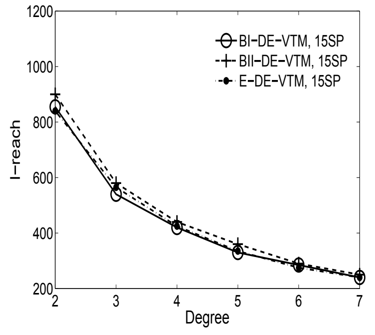
(c) $I - reach$ with 5 shortest-paths.



(d) SR 15 shortest-paths.



(e) NWL 15 shortest-paths.



(f) $I - reach$ 15 shortest-paths.

Figure 4.12: SR, NWL and $I - reach$ mapping VTs from 2 up to 7 degree for USA network when 5 and 15 shortest-paths are used.

approximately the double comparing to Fig. 4.11(b) for the three algorithms, so that the best performance of the E-DE-VTM over the BII-DE-VTM and BI-DE-VTM is stressed.

We tested our algorithms in the more realistic 40 nodes USA network. This is a more complex network, so the metric SR has to be included in the analysis. Figure 4.12 presents the SR, NWL and $I - reach$ by the three algorithms when 5 and 15 shortest paths are used respectively. In Fig. 4.12(a), we observe that the E-DE-VTM has the worst performance. This indicates that the algorithm has a drawback when the complexity of the network grows. The bad performance of the E-DE-VTM is because it has a preference for the shortest paths and stagnation occurs. In order to keep the NWL as low as possible, the algorithms does not reach a good SR. Figure 4.12(d) shows a similar behavior for the three algorithms. For VTs of degree 7, the algorithms reach a SR around 80 % - 90 % when the shortest paths available grows to 15.

From Figs. 4.12(b) and 4.12(e) it can be observed that the three algorithms have a similar performance. Again, the E-DE-VTM has slightly better performance than the other two for both cases. Also, it can be noticed that the NWL gets worse for the three algorithms when the number of shortest path increases from 5 to 15. This behavior is explained by the growth of the solution space. When the number of shortest paths and the degree of the VTs grows, the three algorithms have more trouble finding a near optimal solution. Besides, since the algorithms are limited to run for a fixed number of generations or time, it is expected that the value reached for a bigger solution space will be worse.

Finally, Figs. 4.12(c) and 4.12(f) show the iteration in which the best fitness value was reached. In this metric, the E-DE-VTM presents a similar behavior for the two values of shortest paths. On the contrary, the other two algorithms present an improvement on this metric when the shortest path increases to 15.

In summary, for small networks, the three algorithms have a good performance related to the SR metric. The three algorithms reach a 100 % SR when the degree of the nets is greater than 2. Nevertheless, the E-DE-VTM has a better performance for the NWL metric; this is because by modifying it, the algorithm has a preference for the shortest paths. However, when the complexity of the networks grows, these improvements affect the SR metric for the E-DE-VTM.

Nevertheless, the three algorithms work well to find survivable mappings reducing the NWL. These algorithms can be considered as a good tool for optimization of the SVTM problem.

4.7 Conclusions

In this chapter, we propose the application of a differential evolution (DE) algorithm to the SVTM problem. We presented three algorithms based in DE, named BI-DE-VTM, BII-DE-VTM and E-DE-VTM. We confirm the effective capabilities in terms of convergence speed and quality of the solutions obtained, minimizing the NWL and reaching a good SR. We present an illustrative experiment to demonstrate the methodology of our BI-DE-VTM algorithm and show the effectiveness and efficiency of the proposed evolutionary algorithm. Despite the solutions are somewhat sensible to variations of the DE-VTM algorithm's parameters, the computed results show that a good combination of these parameters leads to a system performance improvement and to a superior convergence rate. For small networks, the three algorithms have a good performance related to the SR metric, reaching a 100 % SR when the degree of the VTs nets is greater than 2. We have provided a practical DE model to solve the SVTM problem that is simple to implement and could also be extended by adding other features into the objective function. As further work, we propose to investigate the use of the VTM-DE algorithms including other features, such as survivability to failures in nodes or multiple link failures, among others in optical networks and the use of intensification procedures like path-relinking to perform local search.

Chapter 5

DE APPLIED TO THE RSA PROBLEM

Flexible optical network (FON) architectures are considered a very promising solution where spectrum resources are allocated within flexible frequency grids. Routing and spectrum allocation (RSA) in FON is an NP-complete problem. So far, this problem has been optimally solved for small instances with integer linear programming and has been sub-optimally solved for more realistic instances by heuristic strategies. In this chapter, we introduce the application of differential evolution (DE) to the RSA problem in flexible optical networks. Comparative studies show that in many cases DE outperforms many other well-known evolutionary computational approaches. Furthermore, the method typically requires few control parameters. An illustrative example is presented showing the effectiveness and efficiency of the proposed algorithm. Different heuristics are compared against the DE-RSA algorithms.To the best of our knowledge, our work is the first application of a DE algorithm to the RSA problem.

5.1 Introduction

Nowadays, telecommunications are developing almost exponentially in response to the ever-increasing internet traffic combined with emerging high-rate applications such as high definition TV, cloud computing, video on demand, among others [4]. The systems fit to cope with this exponential growth are led by optical systems which have superior features over other wired systems; these characteristics are, for example, a higher bandwidth (in the order of Tbps), lower signal attenuation, lower distortion, lower power consumption, among others. Worldwide networking and communication systems and applications use high-speed optical transport networks as appropriate backbones for connecting buildings, cities, and countries [123]. Wavelength division multiplexing (WDM) networks which are connection-oriented networks, have led to substantial research that has eventually emphasized the modifications required in the optical network architectures to achieve their full potential. Despite the fact that WDM networks seem a good cost-effective solution, they have a drawback in their rigid granularity which can lead to inefficient capacity utilization. This problem is expected to become more significant given the necessity of higher capacity systems (i.e. systems of 40 and 100 Gbps channels) [134].

Therefore, a new more agile network infrastructure is needed to provide flexibility and efficiency in the use of resources. Optical Packet and Burst switching (OPS/OBS) have been proposed in the literature as suitable candidates for providing this flexibility and efficiency. However, OPS and OBS are regarded as long-term solutions since they require enabling technologies which are not yet mature [135]. Thus, using the intrinsic scalability and flexibility characteristics of OFDM (optical frequency division multiplexing), a novel flexigrid optical network (FON) architecture has been proposed possessing the capability to manage different data rates and variable bandwidth [4].

In FONs, the optical spectrum is divided into frequency slots of finer size than the fixed ITU-T WDM grid (50 Ghz) providing more flexibility. Some proposals for the slots'sizes include 25 Ghz, 12.5 Ghz and 6.25 Ghz. The connections can occupy multiples of these slots according to the transmission rate, the modulation format and the distance required [5]. The well-known routing and wavelength assignment (RWA) problem in WDM networks became the routing and spectrum allocation (RSA) in FONs. However, new challenges arise at the networking level since the previous WDM algorithms can no longer be applied directly.

The RSA is an NP-complete problem [9] and has recently gained wide interest within the optical communications research community. Integer Linear Programming (ILP) models [5, 134] have been successfully used to solve the RSA problem in small sized optical networks. However, as the network's size increases so does the dimension of the ILP model, whose solution typically requires an extensive computational effort (and execution time) which renders them impractical for medium to large scale networks. Therefore, heuristic-based algorithms become a good tool to cope with more realistic instances of the problem [136–138].

DE is a very simple but very powerful stochastic global optimizer for a continuous search domain. It was proposed by Storn and Price [14] to represent a very complex process of evolution. Intelligently using the differences between populations (a population is a set of candidate solutions) and with the manipulation of few control parameters, they created a simple but fast linear operator called differentiation, which makes DE unique. Additionally, studies show that DE in many instances outperforms other evolutionary algorithms [10, 22]. A complete theoretical analysis of the algorithm is presented in [14].

In this chapter we study the application of a differential evolution (DE) algorithm to the off-line RSA problem. This paper extends the results of former work [19]. In [19], a simple example on how to apply DE to the RSA problem was presented for a small network. In this paper our objective function is modified to optimize not only the spectrum utilization (SU), but also the average path length (APL). Two permutation-based DE algorithms are developed for this problem. An illustrative example is presented, and afterwards the DE-RSA algorithm is compared to different heuristics in test bench mark networks showing that DE outperforms those heuristics. To the best of the authors knowledge, we provide a novel practical DE model to solve the off-line RSA problem that is simple to implement. Our model could also be extended to include other features, such as impairment-aware, energy efficiency, modulation formats, survivability, among others in optical networks.

The chapter layout is as follows. In Sect. 5.2 the problem formulation is presented. The proposed search algorithm is discussed in Sect. 5.3. Two permutation-based approaches for the DE algorithm which can be applied to the RSA problem are introduced in Sect. 5.4. An illustrative example on how to apply the DE algorithm to the RSA problem is presented in Sect. 5.5. The results are then explained in Sect. 5.6. Finally, conclusions are addressed in Sect. 5.7.

5.2 Problem formulation

The RSA problem in OFDM networks is very similar to the classical RWA problem in WDM networks. Similar to the RWA, the RSA problem can be categorized into planning (off-line) RSA or dynamic (on-line) RSA. Due to its complexity, the RSA can be divided into two sub-problems: a routing sub-problem and an allocation sub-problem. The physical network is modelled as an undirected graph $G = (V, E)$, where V is the set of physical nodes numbered $\{1, 2, , |V|\}$, and E is the set of physical links $\{e_{i,j}, i, j, \in V\}$ with cardinality $|E|$. $e_{i,j}$ is in E if there is a link between nodes i and j. Each link has a capacity C split into spectrum slots of size F, so that the number of slots in every link is equal to $N = C/F$. To route the paths a guard band of G subcarriers (slots) has to separate adjacent spectrum paths. The RSA can be formally stated as follow: given a list of traffic demands R in which a request is defined as $r_i = s_i, d_i, n_i$, where s_i is the source node, d_i is the destination node and n_i is the number of slots to transport the requested bandwidth of demand r_i, establish paths and allocate the spectrum requested for every transported demand, so that all demand is met and the spectrum utilization (i.e. the max index of the utilized slot in a link) is minimized, subject to the continuity constraints and the contiguity constrain.

The objective function comprises the minimization of the SU and the APL. The SU has an economic implication: the lower the spectrum utilization, the lower the cost of the network. At the same time, it is also expected that the paths established will have the shortest path length in order to minimize the APL of the network, which is defined as the average number of links used by all selected routes. The reduction of the APL has a primary impact on the delays and transmission impairments of the signal; it also helps to reduce the network's resource wastage.

Therefore, we must establish a criterion for measuring the performance of the algorithm. Our objective function or fitness function is to minimize the weighted sum of the SU and the APL, as follows:

$$min\ f(X) = a_1 * (SU/b_1) + a_2 * (APL/b_2) \tag{5.1}$$

where a_1 and a_2 are the weights used to vary the relative importance of either term in the objective function. $a_1 = 1 - a_2$, takes values from 0 to 1. b_1 and b_2 are normalizing constants that are used to maintain the SU and the APL values in the range of $[0, 1]$, since we do not wish to have one term significantly dominating the other. The way in which these normalizing constants are calculated is described in Sect. 5.2.2.

5.2.1 Spectrum utilization and average path length

Optimizing the spectrum utilization (SU) in the planning phase brings along different benefits to the resources in a network. An optimized SU will not only save spectrum through a more efficient management of the resources, but it will also balance the load in its links. SU can be defined as [136]:

$$SU = \max\{S(e)\}, \forall e \in E \tag{5.2}$$

where $S(\bullet)$ is a function that returns the maximum index of the utilized slots in link $e_{i,j}$. In this work, as in [134], we assume that each link $e_{i,j} \in E$ is characterized by a subcarrier binary vector $\bar{U_l} = [u_{li}] = [u_{l1}, u_{l2}, ..., u_{lN}]$ of length N (i.e. the number of slots in the link), with an ith element equal to 1 if the ith slot is available and 0 if it has already been used for a path or connection.

The second part of the objective function is related to the average number (or the average length) of links used by all selected routes. The APL, for a given set of demands R, can be defined as:

$$APL = \frac{\sum lp_{s,d}}{|R|} \quad \forall s, d \in R \tag{5.3}$$

where $lp_{s,d}$ is the length of the path that connects nodes (s, d) and $|R|$ is the number of requested connections in R.

5.2.2 The normalizing constants

We now explain how to calculate the normalizing constants (in Eq. (5.1)) b_1 and b_2. b_1, that is an upper bound for SU, is calculated considering the worst case scenario for the SU in a network, which occurs when all the requests from R shared at least one common link. Then:

$$b_1 = \sum n_i \quad \forall n_i \in R \tag{5.4}$$

where n_i is the number of slots to transport the requested bandwidth of demand r_i.

Similarly, b_2 is an upper bound for APL; to compute it we first find k-shortes paths that can be calculated using a k-shortes path algorithm. Then, the worst case scenario for the APL is when all longest routes are considered for the connections between node pairs that have already been calculated. Therefore, b_2 is calculated by:

$$b_2 = \frac{\sum sp_{max(s,d)}}{|R|} \quad \forall s, d \in R \tag{5.5}$$

where $sp_{max(s,d)}$ is the longest path from the set of calculated paths for the connection between nodes (s, d).

Note that since b_1 is the upper bound for the SU and b_2 is the upper bound for the APL, then, by dividing both terms (SU and APL) by their respective upper bounds, we guarantee that the number obtained is in the range of $[0, 1]$.

5.2.3 Single demand RSA heuristic algorithms

A RSA single demand heuristic algorithm to solve the planning problem that can be scaled to networks of larger size is presented in [134]. In that algorithm a set P of pre-calculated paths is available. Then a pre-ordering phase is used to finally serve the demands from the list R one by one sequentially.

The pre-ordering phase is quite important, since a different spectrum utilization is achieved depending on the ordering policy used. In this paper we evaluate three ordering policies proposed in [134]:

1.-First-Fit (FF) without ordering: The demands are served one by one as they appear in the list R.

2.-Most Subcarriers First (MSF) ordering: The demands are ordered in decreasing order of the number of their requested subcarriers (slots), and the demand served first is the one that requires the highest number of subcarriers.

3.- Longest Path First (LPF) ordering: The demands are ordered in decreasing order of the number of links their shortest path utilizes, and the demand served first is the one whose shortest path utilizes the highest number of links.

In order to assess the efficiency of the proposed DE-based algorithms, we compared our results against these three simple policies.

5.3 Differential evolution (DE) algorithm

The basic DE algorithm uses a population (Pop) of individuals (solutions to the problem), and iterates by creating new populations until a satisfactory solution is obtained or a computational limit is exceeded. At the beginning of the algorithm, assuming we do not have information about the optimum, the initial population is created randomly.

DE has three crucial control parameters: the mutation constant (M), which controls the mutation strength; the recombination constant (RC), which increases the diversity in the mutation process; and the population size (NP). Throughout the execution process, the user defines M and RC values in the range of $[0, 1]$ and the *Pop* size NP, which is an integer that depends on the dimension of the problem. These parameters are maintained fixed throughout the execution of the algorithm.

At each generation, all individuals in the *Pop* are evaluated in turn. The individual being evaluated is called the target vector (x^i). For each target vector $x^i, i = 1, \ldots, NP$, a mutant individual m^i is generated according to:

$$m^i = x^{r1} + M * (x^{r2} - x^{r3}) \tag{5.6}$$

where $x^{r1}, x^{r2}, x^{r3} \in Pop$: $x^{r1} \neq x^{r2} \neq x^{r3} \neq x^i$. x^{r1}, x^{r2} and x^{r3} are three random individuals from the *Pop*, mutually different and also different from the current target vector x^i.

Then, the recombination operator RC is applied to create the trial vector (t^i). In this opeator, the mutant individual, m^i, is combined with the current target vector x^i. Particularly, for each component j, where $j = \{1, 2, \ldots, D\}$, we choose the jth element of the m^i with probability RC, otherwise from the x^i. Moreover, a random integer value Rnd is chosen from the interval $[1, D]$ to guarantee that at least one element is taken from m^i. Choosing a random number $rand$ in the $[0, 1]$ interval, then the t^i is created as follows:

$$t^{i,j} = \begin{cases} m^{i,j} & \text{if } (rand < RC) \vee (Rnd = j) \\ x^{i,j} & \text{otherwise} \end{cases} \tag{5.7}$$

After we create the trial vector t^i, it is necessary to verify the boundary constraints of each element of t^i to avoid creating infeasible solutions. This could happen because any jth element created by Eq. (5.6) that is not in the allowed range of the specification of a problem has an RC probability of being selected. If any element of the trial vector violates the constraints it is replaced with a random number in the allowed range.

Finally, the selection operator is applied; this operator is a simple rule of elitist selection of the vectors that improve the objective function. This is done by comparing the fitness between the trial vector and the target vector in the objective function using:

$$pop_k = \begin{cases} t^i & \text{if } f(t^i) < f(x^i) \\ x^i & \text{otherwise} \end{cases} \tag{5.8}$$

where pop_k is the population of the next generation, that changes by accepting or rejecting new individuals. The best individual in the population and the global best individual are kept at the end of each generation, to keep track of the best solution found so far.

Under these considerations a pseudocode of the basic DE algorithm is presented in algorithm 9:

Algorithm 9 BI-DE-VTM pseudocode

```
Set the control parameters $M$, $RC$ and $NP$.
Create an initial Pop.
Evaluate the fitness of every individual.
repeat
  for each individual $x \in Pop$ do
    Select three individuals from Pop.
    Apply mutation Eq. (5.6).
    Apply recombination Eq. (5.7).
    Verify boundary constraints.
    if Boundary constraints are violated then
      modify the infeasible elements.
    end if
    Apply selection operator Eq. (5.8).
    Update Pop.
  end for
until a satisfactory solution is obtained or a computational limit is exceeded.
```

5.3.1 Encoded and fitness of the individuals

We applied the DE-based algorithm to solve the off-line RSA problem. The individuals must represent specific solutions to the problem, so that, an encoding that is well-suit to it is necessary. To encode our individuals we generate a vector of dimension D, for example:

$$X = [rs_1, rs_2, rs_3, ..., rs_{|R|}] \tag{5.9}$$

where D is equal to the number of requests in R and rs_i represents the turn in which the request i from R is served. These encoding means that the vector is a permutation from 1 to $|R|$ representing the order in which the demands from R will be served.

To evaluate the fitness of an individual, as stated in Eq. (5.1), our objective is to minimize the spectrum utilized as well as the average path length. The encoding represents the order in which the demands are served, but it is also necessary to specify a method through which the spectrum and routes are assigned, in order to calculate the fitness of an individual.

Our approach, as in [134], assumes that a set of pre-calculated paths is available. Therefore, in an initialization step we calculate k paths between all pairs of nodes, using the k-shortest paths algorithm [129]. The pre-calculation of paths could be based on hops or distance. The pre-calculation of routes will speed up the procedure of routes selection in the algorithm. As stated in Sect. 5.2.1, we assume that each link $e_{i,j} \in E$ is characterized by a subcarrier binary vector $\bar{U_l}$ of length N (i.e. the number of slots in the link), with an ith element equal to 1 if the ith slot is available and 0 if it has already been used for a connection. A subcarrier binary vector of a path can be calculated using the subcarrier binary vectors of the links that the path traverses as:

$$\bar{U_p} = [u_p i] = [\wedge_{l \in p} \quad u_{li}] \tag{5.10}$$

where $\wedge$ denotes the Boolean AND operation over all the binary links in the path.

Then, to serve a conecction r_i that requires n_i slots we first use Eq. (5.10) to calculate the spectrum availability $\bar{U_p}$ of all candidate paths. We search each spectrum availability vector $\bar{U_p}$ for the first possible placement of n_i subcarriers (along with the required G guardbands). After that, we select the path with the lowest indexed starting subcarrier and store it as the selected path for that connection. We update the spectrum availability of the links that comprises the selected path by setting 0's to the corresponding spectrum slots. We repeat this procedure with the next connection r_i in the order specified by the individual X. After all demands in R have been served, we calculate SU (Eq. (5.2)) and the APL (Eq. (5.3)) to obtain the fitness of the individual X (Eq. (5.1)).

In this work we select the path with the minimum starting subcarrier for simplicity. Nevertheless, we can use other approaches such as void filling or random filling to get different results [24].

5.4 Permutation-based approaches for DE algorithm

The DE algorithm is originally applicable to continuous optimization problems because its search mechanism is based on perturbations built with differences between vectors. However, due to the encoding of the individuals in this problem (which are permutations), we realized that we are dealing with a combinatorial optimization problem with symbolic variables. This makes the arithmetic operations of the original DE neither applicable nor meaningful [139].

For this reason, in Sects. 5.4.1 and 5.4.2, two permutation-based approaches of DE that can be applied to the off-line RSA problem are introduced.

5.4.1 DE-based relative position indexing approach

The Relative Position Indexing (RPI) approach is applicable for permutation-based problems only. This approach transforms the elements of the integer permutation vector into the floating-point interval [0, 1]. Then, the mutation and the recombination operators can be applied using the transformed values in the continuous domain. The resulting values are then converted back into the integer domain using a relative position indexing, as described in [140].

To illustrate the transformation of an individual with the RPI approach, consider the individual $X = [2, 3, 1, 5, 4]$. The transformation into floating-point values is achieved by dividing each element of the vector by the largest one of them, in this case 5, resulting in $X_t = [0.4, 0.6, 0.2, 1, 0.8]$. Then, the basic DE algorithm can be applied to obtain the trial individual. In order to convert the trial individual back into the integer domain, using RPI, the smallest floating-point value is replaced by the smallest integer value, and then the next smallest floating-point value is replaced by the next integer value, and so on until all elements have been converted. Note that this approach always yields a feasible solution, except when two or more floating-point values are the same. When such an event occurs, the trial vector must be repaired as explained before.

5.4.2 DE-based combinatorial approach

The RPI approach maps the integer values to the continuous domain before applying the DE mutation and crossover operators. Although the RPI approach preserves the continuous DE operators, these operators do not capture the essence of the DE's search mechanism. Because of this, a more general approach for combinatorial optimization was proposed in [139]. This general combinatorial (GC) approach based on DE aims at preserving the search mechanism for discrete domains by defining the difference between two candidate solutions as a differential list of movements in the search space.

The key idea of the GC approach is to define the difference between two individuals as a list of movements. Suppose that the search space SS is defined by the set of all valid combinations of values for the variables (i.e. all the permutations that can represent a solution for a problem). A differential list of movements $DL_{j\to i}$ is a list containing a sequence of valid movements vm_k such that the application of these movements to a solution $X_j \in SS$ leads to the solution $X_i \in SS$.

In this way, the difference between two individuals is defined as being the list of movements:

$$DL_{j\to i} \doteq X_i \ominus X_j \tag{5.11}$$

where $\ominus$ is a special operator that returns a list of movements that represents a path from X_j towards X_i. The application of a list of movements to a given solution is defined as:

$$X_i' = X_i \oplus DL_{a\to b} \tag{5.12}$$

where the operator $\oplus$ receives a valid solution and a list of movements, returning another solution. With these definitions the following relation is valid:

$$X_i = X_j \oplus DL_{j\to i} = X_j \oplus (X_i \ominus X_j) \tag{5.13}$$

With this relation, the main operator of DE (Eq. (5.6)) can be re-written to apply the GC approached as:

$$m^i = x^{r1} \oplus M \otimes (x^{r2} \ominus x^{r3}) \tag{5.14}$$

where M is a scaled factor to control how many movements resulting from $(x^{r2} \ominus x^{r3})$ will be applied to the x^{r1} individual. In [139], three alternatives for applying the multiplication of the mutant constant M were proposed. The multiplication of a list of movements $DL_{i\to j}$ by a constant M (in the range $[0, 1]$) denoted as $M \otimes DL_{i\to j}$ returns:

1.- A list $DL'_{i\to j}$ with the first $\lceil M * |DL_{i\to j}|\rceil$ movements selected from $DL_{i\to j}$.

2.- A list $DL'_{i\to j}$ with a random $\lceil M * |DL_{i\to j}|\rceil$ movements selected from $DL_{i\to j}$.

3.- A list $DL'_{i\to j}$ which is formed by selecting each element from $DL_{i\to j}$ with probability M.

After applying the mutation operator, the recombination operation must be applied to perform recombination between the target vector and the mutant vector. This operator is also different for a permutation-based problem, since the elements cannot be repeated and Eq. (5.7) is not applicable as it is. Different crossover operators for a permutation representation have been proposed before [141].

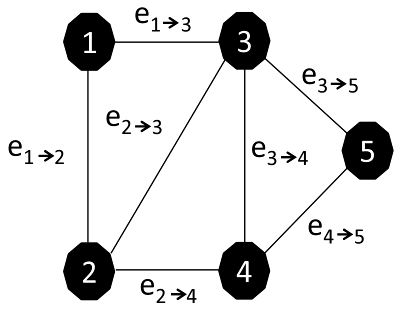
To be consistent with the format of the basic DE, a new recombination operator for the GC approach has been developed. This new operator forms the trial vector (t^i) taking information from the mutant individual (m^i) and the current target individual (x^i). The crossover operator takes elements from the m^i with probability RC and copies them to the t^i individual in its absolute position. Then, the elements that have not yet being assigned are copied to t^i using the relative order that they have in the individual x^i. In this way, the t^i individual is formed with information of m^i and x^i. Moreover, a random integer value Rnd is chosen from the interval $[1, D]$ to guarantee that at least one element is taken from m^i.

Once we established the mutant and crossover operators for the GC approach, the algorithm can be applied as in pseudocode 9 by just replacing Eq. (5.6) for Eq. (5.14) and performing the aforementioned crossover operator instead of that of Eq. (5.7).

5.5 DE for the RSA problem

We present an Illustrative example of our two DE-RSA algorithms (DE-RPI and DE-GC), using the simple physical network of five nodes shown in Fig. 5.1(a). Since both algorithms are based in DE, they share part of the methodology. We start by explaining the common methodology that both algorithms use. Then, we explain how to apply the mutation operator and the recombination operator for the DE-RPI (which is the main difference between the two approaches), followed by the application of these parameters to the DE-GC. Finally, the selection operator, which is the same for both algorithms, is explained.

A request list R with 16 requests is presented in Fig. 5.1(b), showing the tag of the request, the source node (s), the destination node (d) and the number of slots requested (n).



(a)

Request	***s***	***d***	***n***	**Request**	***s***	***d***	***n***
1	1	2	1	**9**	3	2	2
2	1	3	1	**10**	3	4	2
3	1	5	4	**11**	3	5	1
4	2	1	3	**12**	4	1	2
5	2	3	2	**13**	4	3	4
6	2	4	2	**14**	5	2	3
7	2	5	2	**15**	5	3	3
8	3	1	3	**16**	5	4	3

(b)

Figure 5.1: Illustrative example. (a) Simple five-nodes network. (b) Traffic demand set list R.

For simplicity, we consider single bidirectional fiber links with capacity of 20 slots and a guardband $G = 1$ slot is also considered.

As mentioned in Sect. 5.3.1, our approach assumes that a set of pre-calculated paths is available. In an initialization step, we calculate k shortest paths between all nodes. Table 5.1 shows $k = 4$ shortest paths for the 5-node network.

Each individual will have the form of Eq. (5.9), representing the order in which the requests of the list $|R|$ will be served. That means that all the individuals are permutations of size $|R|$.

As stated in algorithm 9, we first set the control parameters. For this illustrative example, and for both algorithms (i.e. DE-RPI and DE-GA), the values are set to $NP = 5$, $M = 0.5$ and $CR = 0.5$. Then, an initial random population is created, e.g.:

Table 5.1: k-shortest paths for a 5-node net. The selected paths for the individual in the example are in bold.

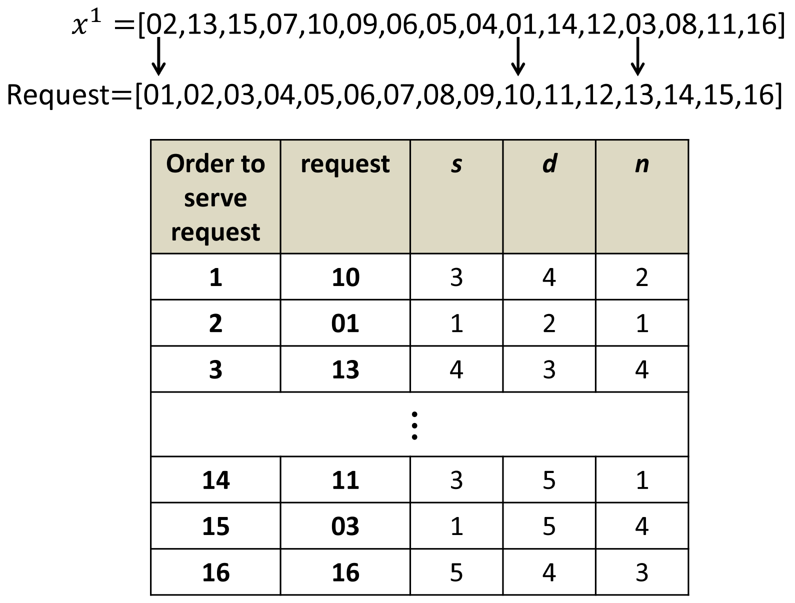
Node	1	2	3	4	5
1	$[]$	$[1,2]$ $[1,3,2]$ $[1,3,4,2]$ $[1,3,5,4,2]$	$[1,3]$ $[1,2,3]$ $[1,2,4,3]$ $[1,2,4,5,3]$	$[1,2,4]$ $[1,3,4]$ $[1,2,3,4]$ $[1,3,2,4]$	$[1,3,5]$ $[1,2,3,5]$ $[1,3,4,5]$ $[1,2,4,5]$
2		$[]$	$[2,3]$ $[2,1,3]$ $[2,4,3]$ $[2,4,5,3]$	$[2,4]$ $[2,3,4]$ $[2,1,3,4]$ $[2,3,5,4]$	$[2,4,5]$ $[2,3,5]$ $[2,3,4,5]$ $[2,1,3,5]$
3			$[]$	$[3,4]$ $[3,2,4]$ $[3,5,4]$ $[3,1,2,4]$	$[3,5]$ $[3,4,5]$ $[3,2,4,5]$ $[3,1,2,4,5]$
4				$[]$	$[4,5]$ $[4,3,5]$ $[4,2,3,5]$ $[4,2,1,3,5]$
5					$[]$

$$pop = \begin{cases} x^1 = [02, 13, 15, ..., 08, 11, 16] \\ x^2 = [06, 04, 08, ..., 10, 03, 01] \\ x^3 = [07, 12, 04, ..., 05, 11, 09] \\ x^4 = [08, 10, 14, ..., 02, 12, 11] \\ x^5 = [14, 04, 09, ..., 15, 11, 05] \end{cases} \tag{5.15}$$

Then, according to Eq. (5.1), to calculate the fitness of each particle we stablish the values of the weights constants a_1 and a_2 and the normalizing constants b_1 and b_2. For this example, we set the values of the weights constants as $a_1 = 0.5$ and $a_2 = 1 - a_1 = 0.5$. On the other hand, the normalizing constants depend on the upper bounds for the total spectrum (or number of slots) requested by the list R and for the size of the pre-calculated paths. Those upper bounds can be calculated as in Sect. 5.2.2. For this example, $b_1 = 38$ and $b_2 = 2.75$. With this information, the fitness of each particle is calculated as:

$$\begin{cases} f(x^1) = 0.5 * (17/38) + 0.5 * (1.56/2.75) = 0.51 \\ f(x^2) = 0.5 * (17/38) + 0.5 * (1.50/2.75) = 0.50 \\ f(x^3) = 0.5 * (15/38) + 0.5 * (1.37/2.75) = 0.45 \\ f(x^4) = 0.5 * (13/38) + 0.5 * (1.43/2.75) = 0.43 \\ f(x^5) = 0.5 * (17/38) + 0.5 * (1.31/2.75) = 0.42 \end{cases} \tag{5.16}$$

Before explaining how the fitness of each individual is calculated, the permutation representation of an individual adopted in this work has to be explained. Figure 5.2 shows the individual x^1, which is a permutation of size $|R|= 16$ (i.e. the number of request to be served). In this representation, each position of x^1 represents a request in R, and the number in that position is the turn in which that request is to be served. For instance, the number 01 in the 10th position of the individual x^1 indicates that request 10 will be the first one to be served. Then, the number 02, in the first position, which is a 01, of the individual x^1 indicates that request 01 will be the second one to be served, and so on.



x^1 =[02,13,15,07,10,09,06,05,04,01,14,12,03,08,11,16]

Request=[01,02,03,04,05,06,07,08,09,10,11,12,13,14,15,16]

Order to serve request	request	*s*	*d*	*n*
1	**10**	3	4	2
2	**01**	1	2	1
3	**13**	4	3	4
⋮				
14	**11**	3	5	1
15	**03**	1	5	4
16	**16**	5	4	3

Figure 5.2: Permutation individual representation.

After the request list has been served with the methodology explained in 5.3.1, using Eq. (5.2) for the individual x^1 we get an $SU = 17$, which is the max index occupied after serving all demands. Also, using Eq. (5.3), we get an $APL = 1.5625$, which is the average length of the selected routes. Finally, the fitness of the individual using Eq. (5.1), with $a_1 = a_2 = 0.5$

and $b_1 = 38$ and $b_2 = 2.75$ is given by $f(x^1)$ in Eq. (5.16). Using the same methodology, the fitness of each individual is calculated.

In the next step, from the initial Pop, we select a target individual x^i. Suppose that our target vector in the first generation is x^1. Then, three random individuals x^{r1}, x^{r2} and x^{r3} mutually different and also different from the current target vector are selected from the Pop as well. For this example, let $r1 = 3$, $r2 = 4$ and $r3 = 2$.

Up to this point, the methodology used is the same for both algorithms, DE-RPI and DE-GA, the difference, as mentioned in Sect. 5.4, resides on how we apply the mutation and recombination operators on each.

5.5.1 Mutation and recombination operators for the DE-RPI

As stated in Sect. 5.4.1, we need to transform the elements of the individuals (integer permutations) into floating-point values by dividing each element of the individual by the largest one of them, in this case $D = |R|= 16$. This gives:

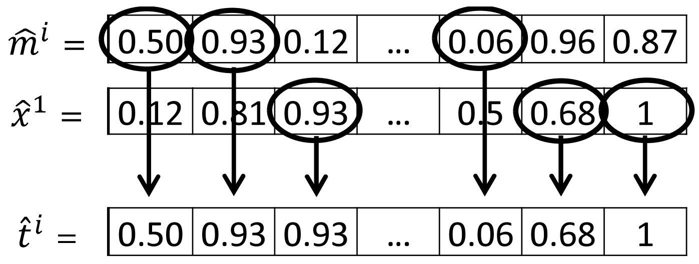
$$\begin{cases} \hat{x^1} = x^1/D = [0.12, 0.81, 0.93, ..., 0.50, 0.69, 1] \\ \hat{x^{r1}} = x^3/D = [0.43, 0.75, 0.25, ..., 0.31, 0.68, 0.56] \\ \hat{x^{r2}} = x^4/D = [0.50, 0.62, 0.25, ..., 0.12, 0.75, 0.68] \\ \hat{x^{r3}} = x^2/D = [0.37, 0.25, 0.50, ..., 0.62, 0.18, 0.06] \end{cases} \tag{5.17}$$

With these values we can apply the original DE mutation operator (Eq. (5.6)). Figure 5.3 shows the resulting mutant individual m^i.

$$\begin{array}{ccccccccc} \hat{x}^3 & + & M & * & (\hat{x}^4 & - & \hat{x}^2) & = & \hat{m}^i \\ \downarrow & & & & \downarrow & & \downarrow & & \downarrow \\ 0.43 & + & 0.5 & * & (0.50 & - & 0.37) & = & 0.5 \\ 0.75 & + & 0.5 & * & (0.62 & - & 0.25) & = & 0.93 \\ 0.25 & + & 0.5 & * & (0.25 & - & 0.50) & = & 0.12 \\ & & & \vdots & & & & & \vdots \\ 0.68 & + & 0.5 & * & (0.75 & - & 0.18) & = & 0.96 \\ 0.56 & + & 0.5 & * & (0.68 & - & 0.06) & = & 0.87 \end{array}$$

Figure 5.3: Mutant individual generation for DE-RPI.

Once we have the mutant individual m^i, we apply the recombination operator (Eq. (5.7)) to combine the target individual and the mutant individual. Figure 5.4 illustrates how the trial individual $\hat{t^i}$ is formed with elements chosen from the target individual $\hat{x^1}$ and the mutant individual $\hat{m^i}$. The elements are chosen with probability RC. Also recall that a random integer value Rnd is chosen from the interval $[1, D]$ to guarantee that at least one element is taken from $\hat{m^i}$ and copied into $\hat{t^i}$.



$\hat{m}^i$ = 0.50 0.93 0.12 … 0.06 0.96 0.87
$\hat{x}^1$ = 0.12 0.81 0.93 … 0.5 0.68 1
$\hat{t}^i$ = 0.50 0.93 0.93 … 0.06 0.68 1

Figure 5.4: Trial individual generation. Elements to create $\hat{t^i}$ are taken from the target individual and the mutant individual with probability RC.

The trial individual $\hat{t^i}$ is then transformed back into the integer domain. To do so, we replace the smallest floating-point value by the smallest integer value of the permutation (i.e. 1), and then replace the next smallest floating-point value by the next integer value (i.e. 2), and so on until all elements have been converted. Note that this approach always yields a feasible solution, except when two or more floating-point values are the same. When such an event occurs, one viable option is two break ties randomly. For this example, after performing the back transformation, we get the trial individual and its fitness as:

$$\begin{aligned} t^i &= [09, 13, 14, 07, 15, 01, 08, ...12, 05, 04, 11, 16] \\ (t^i) &= 0.5 * (15/38) + 0.5 * (1.5/2.75) = 0.47 \end{aligned} \tag{5.18}$$

Finally, the selection operator (Eq. (5.8)) is applied between the target individual x^1 and the trial individual t^i. This operator is a simple rule of elitist selection. The individual with the best fitness value will survive to the next generation. In this example, the trial vector t^i will replace x^1 in the next generation because $f(t^i) = 0.47 < f(x^1) = 0.51$. The process is repeated for each individual in Pop to form the population of the next generation. The algorithm stops when the termination condition is met, which in this case is a number of 1000 generations.

5.5.2 Mutation, recombination and selection operators for the DE-GC

As stated in Sect. 5.4.2, to apply Eq. (5.14), we first need to find a list of movements as $DL_{r3 \to r2} = x^{r2} \ominus x^{r3}$. For this example $r1 = 3$, $r2 = 4$ and $r3 = 2$. To generate $DL_{x^2 \to x^4}$ we need to iteratively find the movements that lead x^2 closer to x^4. Figure 5.5 shows the first two steps in building the list $DL_{x^2 \to x^4}$. In a first step (Fig. 5.6(a)) we swap the elements 1 and 3 of x^2 bringing solution x^2 closer to x^4. The second step (Fig. 5.6(b)) is to swap the elements 2 and 14 of $\hat{x^2}$. The procedure is repeated until $\hat{x^2}$ is transformed into x^4.

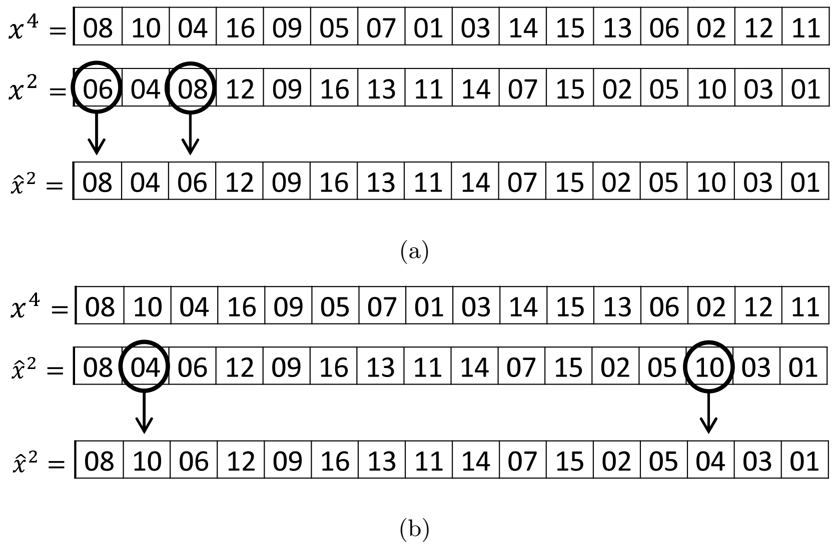

x^4 = 08 10 04 16 09 05 07 01 03 14 15 13 06 02 12 11
x^2 = 06 04 08 12 09 16 13 11 14 07 15 02 05 10 03 01
$\hat{x}^2$ = 08 04 06 12 09 16 13 11 14 07 15 02 05 10 03 01
(a)
x^4 = 08 10 04 16 09 05 07 01 03 14 15 13 06 02 12 11
$\hat{x}^2$ = 08 04 06 12 09 16 13 11 14 07 15 02 05 10 03 01
$\hat{x}^2$ = 08 10 06 12 09 16 13 11 14 07 15 02 05 04 03 01
(b)

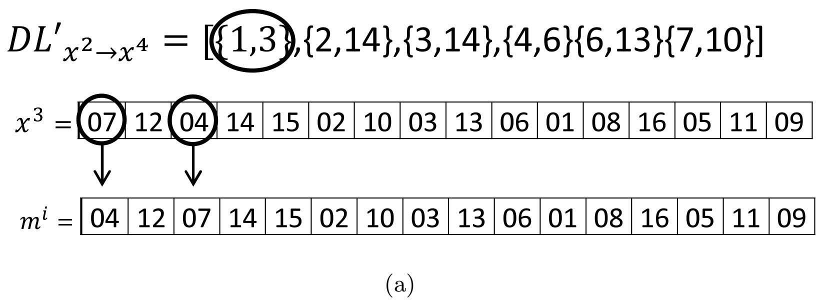
Figure 5.5: Generating the differential list $DL_{2 \to 4}$. (a) First swap: $Mov = \{1, 3\}$. (b) Second swap: $Mov = \{2, 14\}$

As a result, a DL list containing the index pair of swapting moves is obtained as:

$$DL_{x^2 \to x^4} = \begin{cases} (1,3)(2,14)(3,14)(4,6)(6,13)(7,10) \\ (8,16)(9,15)(10,15)(12,15)(13,14) \\ (14,15) \end{cases} \tag{5.19}$$

Then, we need to scale the list using the parameter M. As stated in Sect. 5.4.2, we built a scaled list by taking the first $\lceil M * |DL_{i \to j}| \rceil$ movements from $DL_{x^2 \to x^4}$. If $M = 0.5$ and $|DL_{x^2 \to x^4}| = 12$, then $\lceil M * |DL_{i \to j}| \rceil = 6$ leading to a $DL'_{x^2 \to x^4}$ list containing the first 6 pairs of movements of Eq. (5.19).

Next, according to Eq. (5.14), we need to apply these movements to x^{r1}. Figure 5.6 shows the application of the first two movements of $DL'_{x^2 \to x^4}$. The process continues for all pair of movements of $DL'_{x^2 \to x^4}$ to generate the mutant individual m^i.


$DL'_{x^2 \to x^4} = [\{1,3\},\{2,14\},\{3,14\},\{4,6\}\{6,13\}\{7,10\}]$
x^3 = 07 12 04 14 15 02 10 03 13 06 01 08 16 05 11 09
m^i = 04 12 07 14 15 02 10 03 13 06 01 08 16 05 11 09
(a)

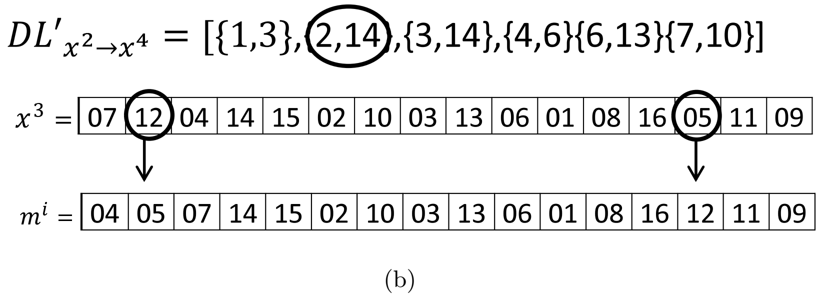
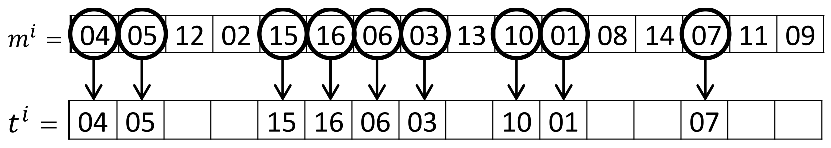

$DL'_{x^2 \to x^4} = [\{1,3\},\{2,14\},\{3,14\},\{4,6\}\{6,13\}\{7,10\}]$
x^3 = 07 12 04 14 15 02 10 03 13 06 01 08 16 05 11 09
m^i = 04 05 07 14 15 02 10 03 13 06 01 08 16 12 11 09
(b)

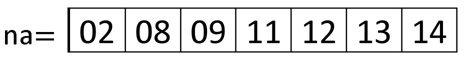
Figure 5.6: Mutant individual generation. (a) First move applied to x^{r1}. (b) Second move applied to x^{r1}

Once we have m^i, we apply the recombination operator to combine the target individual and the mutant individual. Figure 5.7 illustrates how the trial individual t^i is formed with elements chosen from the target individual x^1 and the mutant individual m^i in just three steps. In the first step (Fig. 5.7(a)), we copy elements from m^i to t^i with probability RC in its absolute position. Also recall that a random integer value Rnd is chosen from the interval $[1, D]$ to guarantee that at least one element is taken from m^i and copied into t^i. In the second step, we need to identify what elements of the permutation have not yet being assigned. Figure 5.7(b) shows those unassigned values. Finally, in the third step (Fig. 5.7(c)), we copy the unassigned elements in the relative order that they have in x^1.

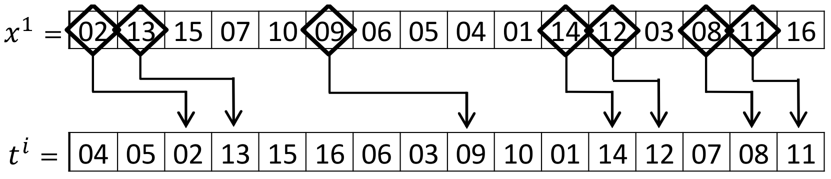
Finally, the selection operator (Eq. (5.8)) is applied between the target individual x^1 and the trial individual t^i. In this example, the trial vector t^i will replace x^1 in the next generation because $f(t^i) = 0.42 < f(x^1) = 0.51$. The process is repeated for each individual in Pop to form the population of the next generation. The algorithm stops when the termination condition is met, which can be a number of generations (1000 for this example).


m^i = 04 05 12 02 15 16 06 03 13 10 01 08 14 07 11 09
t^i = 04 05 15 16 06 03 10 01 07

(a)



(b)

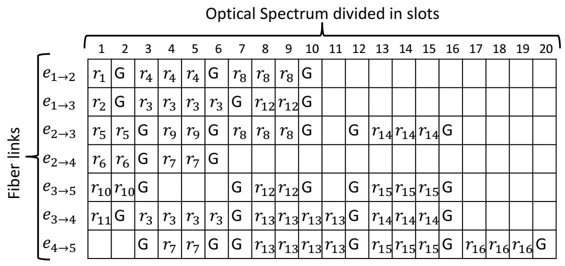

x^1 = 02 13 15 07 10 09 06 05 04 01 14 12 03 08 11 16
t^i = 04 05 02 13 15 16 06 03 09 10 01 14 12 07 08 11

(c)

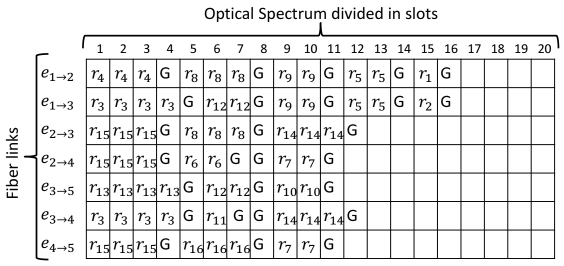
Figure 5.7: Trial individual generation. (a) Step 1. (b) Step 2. (c) Step 3.

5.5.3 Comparision between heuristics and the DE-RPI and DE-GC approaches

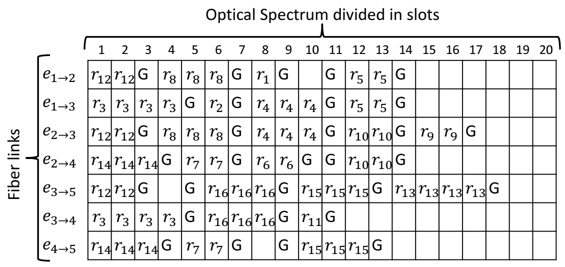
Results for the three single demand heuristics of Sect. 5.2.3 and the two DE approaches of Sect. 5.4 are presented in Fig. 5.8. The figure shows the state of the links after completely serving the request list $|R|$ with each algorithm. Figures 5.8(a), 5.8(b) and 5.8(c) show the network state achived using the single demand heuristic algorithm with FF, MSF and LPF ordering policies respectively. Then, using the parameters for the illustrative example of Sect. 5.5, Figs. 5.8(d) and 5.8(e) show the network state results for the DE-RPI and DE-GC respectively. Both solutions render most efficient allocations since they reduce the SU to 11 slots while at the same time reduce the APL. Moreover, both algorithms balance the traffic in the network links. From ten independent test, the DE-RPI solution requires on average 9 generations to find the solution while the DE-GA requires on average 13 generations. Both algorithms use a small population of $NP = 5$ and no parameter tuning was performed for this example. A parameter tuning may improve the convergence rate of the DE-RSA algorithms. Finally, Fig. 5.8(f) sumarizes the SU and APL obtained for this example, showing the superiority of the solutions of the DE-RSA approaches. This simple methodology can be applicable to larger networks.



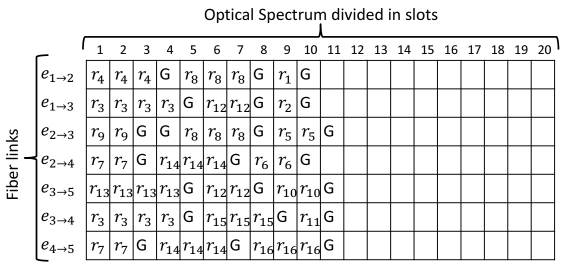
(a)



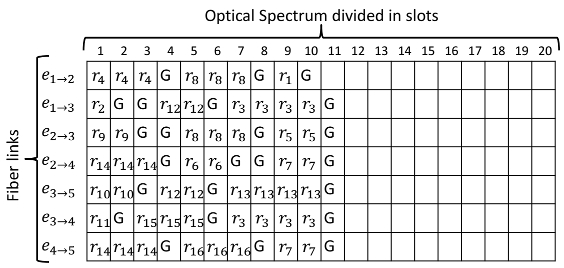
(b)



(c)



(d)



(e)

Algorithm	SU	APL	Generation
FF	20	1.4375	Na
MSF	16	1.5625	Na
LPF	18	1.6875	Na
DE-RPI	11	1.3125	9
DE-GC	11	1.3125	13

(f)

Figure 5.8: Network state after served request list $|R|$ with each algorithm. (a) FF without ordering. (b) MSF ordering. (c) LPF ordering. (d) DE-RPI approach. (e) DE-GC approach. (f) Spectrum utilization and average path length for each algorithm.

5.6 Numerical results and discussion

We applied our DE-RSA algorithms to four real-sized networks, NSFNet, EON (European Optical Network), USA, and Japan networks. Their topologies with link distances in km are presented in Figs. 5.9, 5.10, 5.11 and 5.12, respectively.

The numerical results section is divided into two parts. First we present the parameter tuning and then the performance of the DE algorithms in different network topologies.

5.6.1 Tuning the parameters

It is worth noting that the DE-RSA algorithms require the manipulation of few control parameters (NP, M, RC, k). In our applications, these parameters have an impact on the quality of the solutions. Therefore, we believe it is important to perform an analysis on their effects. This analysis can be done through preliminary tests on a network.

We perform a parametric analysis over the NSFnet (Fig. 5.9). To do this, we generate a random request list R with $N * N - 1$ requests (where N is the number of nodes of the network). The number of slots requested by every pair of nodes in every list is taken from an uniform distribution from $[0...S_{max}]$ with $S_{max} = 4$ for the parametric analysis. We use the average of 10 independent test as a solution value. The weights of the objective function (Eq. (5.1)) a_1 and a_2 have been set to 0.5. The values of b_1 and b_2 for the NSFnet are obtained according to Eqs. (5.4) and (5.5) in Sect. 5.2.2.

We observed that the M and RC parameters do not have a wide impact in the quality of the solutions. For that reason, as in [22], we decide to use the values $M = 0.2$ and $RC = 0.5$ which lead to good solutions. On the other hand, the NP parameter has a much marked impact in the quality of the solutions and it is related to the size of the network. A population value of around $NP = 20$ and $NP = 50$ is adecuate for the networks'sizes that we are analyzing, to render good solutions in both algorithms.

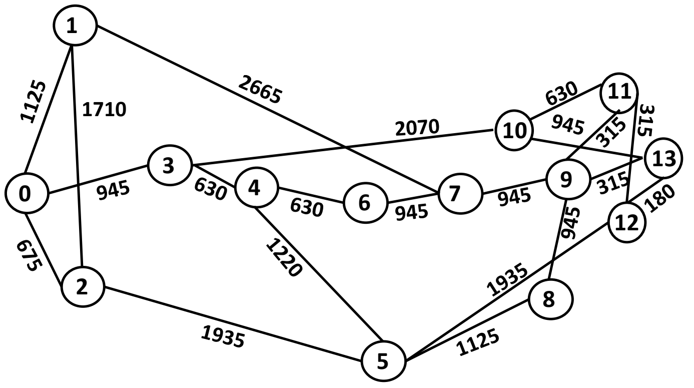


Figure 5.9: NSFnet topology, 14 nodes, and 21 links.

The objective function is formed by two objectives: SU and APL. So that, it is expected that the parameter k has a stronger influence in the solutions since it is related to the number of pre-computed available paths. Figure 5.13 shows the SU and APL when the parameter

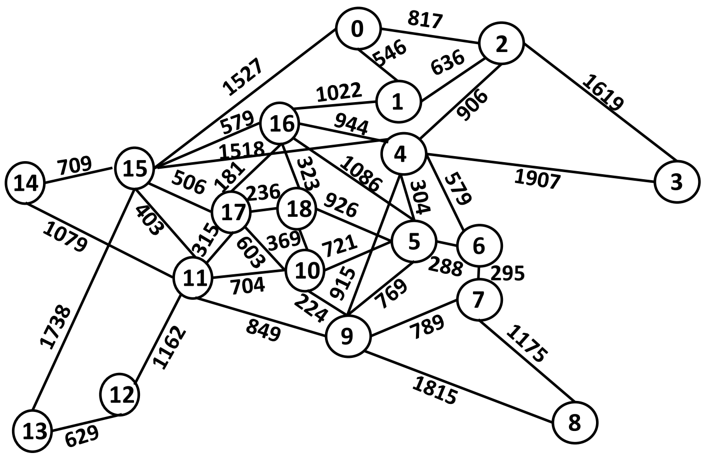


Figure 5.10: EON topology, 19 nodes, and 39 links.

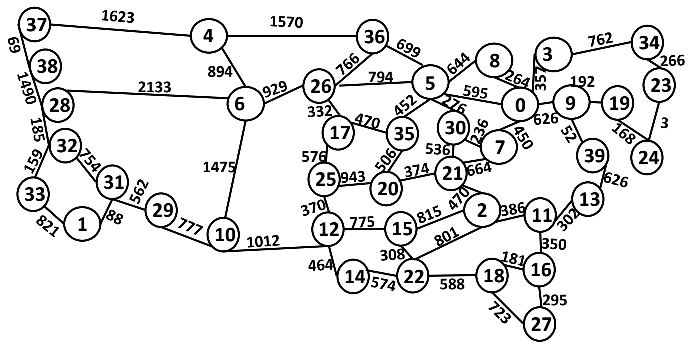


Figure 5.11: USA topology, 40 nodes, and 58 links.

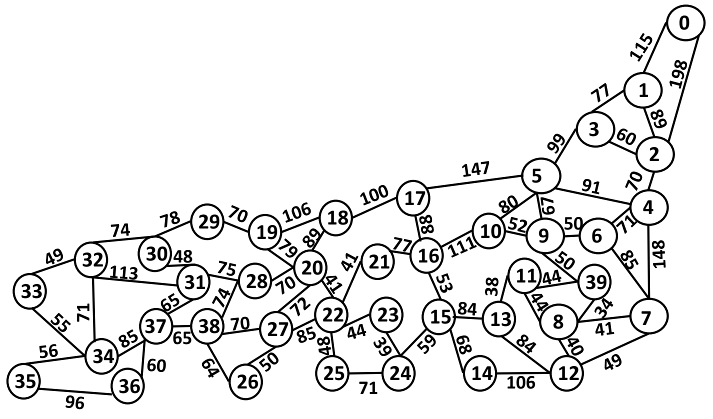


Figure 5.12: Japan topology, 40 nodes, and 65 links.

k varies from 1 to 7 over the Japan network. It can be observed that as the SU improves its value, the APL worsens with the increase of k, showing that the two objectives are in conflict. This means that the value of k has to be choose according to the size of the network that is under analysis, trying to avoid an unnecessary increase in the APL objective. Figure 5.13 also shows that for values of k greater than 4 the SU does not improve significantly and the APL continues to increase. So a value of around $k = 5$ is preferred, which renders a good SU and maintains the APL in a reasonable value. The test was repeated over the four network topologies, using both DE-RPI and DE-GC algorithms, showing a similar behavior. This indicates that a value of k around 5 gives a good SU with a reasonable increase in the APL. For small networks, as the NSFnet even a value of $k = 3$ is enough to obtain good solutions.

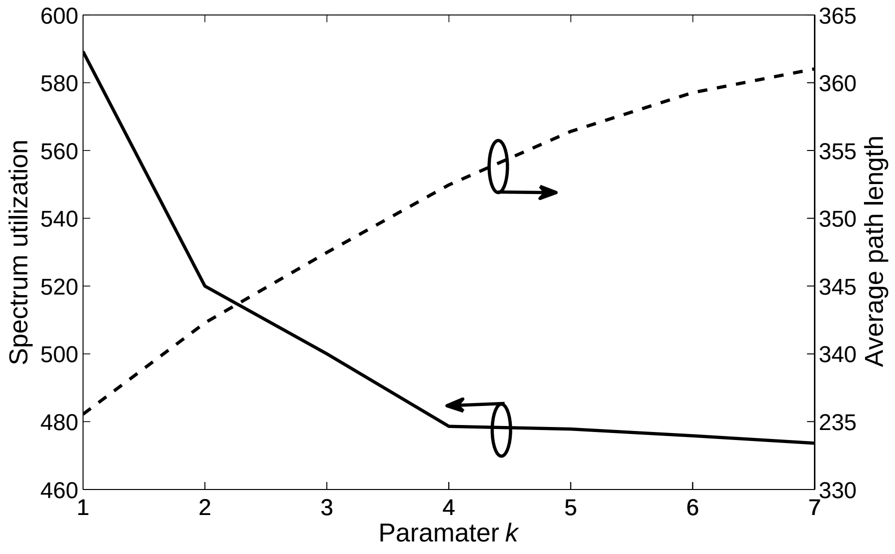


Figure 5.13: SU and APL variying k parameter over the Japan topology.

We now present an analysis on the weights a_1 and a_2 in the objective function, to gauge how this values affect the quality of the solutions in the SU or APL values. Our main interest in this work is to minimize the SU value; however, we also wish to reduce the APL in as much as possible. Having this in mind, we have performed an analysis varying a_1 from 0 to 1 in increments of 0.1, while at the same time varying the value of a_2 from 1 to 0. The results presented are the average of five independent tests of 500 generations with every pair of a_1 and a_2 values.

Figure 5.14 shows the SU and APL values obtained. We can see that when $a_1 = 0$ and $a_2 = 1$, the algorithm minimizes the APL, as it should since the SU does not appear in the objective function, and when $a_1 = 1$ and $a_2 = 0$, the algorithm minimizes the SU. From our analysis, we observed that values of a1 = 0.5 and a2 = 0.5 render the best performance in the objective function. Also, this gives us an idea on how this two metrics are in conflict since reducing the SU does not garantee a reduction of the APL.

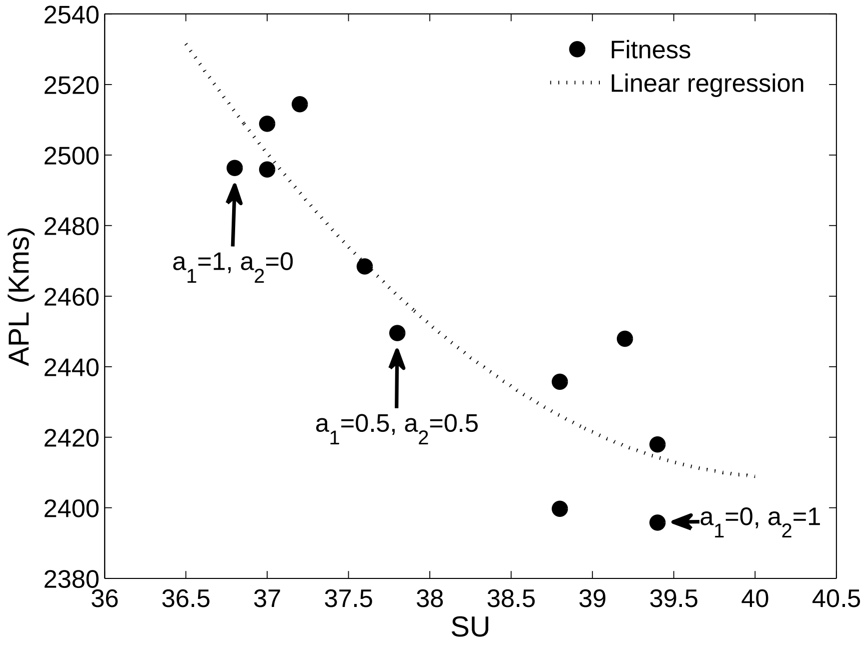


Figure 5.14: SU versus APL points obtained varying the weights a_1 and a_2 on the NSF network.

5.6.2 DE-RSA application on different topologies

We applied the two DE-RSA algorithms to four real-sized networks (Figs. 5.9, 5.10, 5.11 and 5.12). For the simulation, 10 R lists with $N * N - 1$ (where N is the number of nodes) requests were randomly generated for each network. The number of slots required for each connection of the lists was chosen uniformly from 0 to 4 for the NSFnet and EON network and from 0 to 5 for the USA and Japan networks. A guardband of one slot (i.e. G=1) was assumed. Results in this section are the average of 10 independent tests over those 10 request lists.

For these tests, after performing a quick parametric analysis, we used a set of values $M = 0.2$, $CR = 0.5$ and $NP = 20$ for the DE-RSA algorithms. We ran both algorithms for 1000 generations. a_1 and a_2 have been set to 0.5 to optimize both metrics. The normalizing constants b_1 and b_2 are obtained according to Eqs. (5.4) and (5.5) in Sect. 5.2.2 for each network and request list. The pre-calculated paths were set to $k = 3$ for the NSF and EON networks and $k = 5$ for the USA and Japan networks.

Figures 5.15 up to 5.18 show the results of SU and APL for the FF, MSF, LPF ordering policies and both DE-RPI and DE-GC algorithms. Both DE-based algorithms improve the SU and APL demonstrating its superiority. Moreover, it can be observed that DE-RPI performs a little better than the DE-GC algorithms regarding both metrics in the four analyzed networks. In Figs. 5.17 and 5.18 it can also be observed that MSF performs very well in terms of SU compared to DE-RPI and DE-GC.

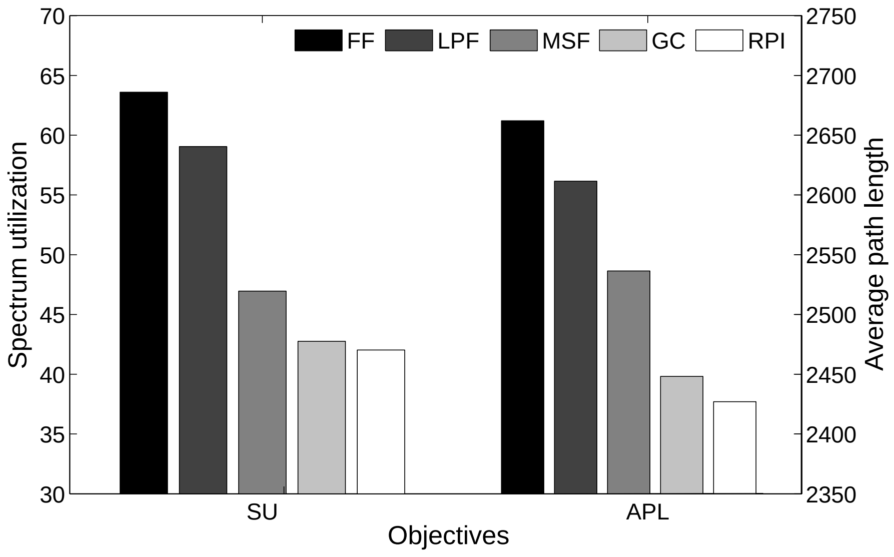


Figure 5.15: Comparison between heuristics and DE-RSA algorithms in the NSFnet.

Table 5.19 summarizes the best results obtained with the DE-based algorithms against the ordering policies. Here, it is clear that the DE-based algorithms overcome other ordering policies in terms of the fitness function. Moreover, DE-RPI presents the best results.

It is important to recall that since both metrics are in conflict (SU and APL) and the weight values (a_1 and a_2) are set to 0.5, the DE-based algorithms improve the fitness value even if that implies a greater SU value. If we are interested in just optimizing a specific metric (i.e. SU ignoring the APL metric), then the weights have to be set accordingly.

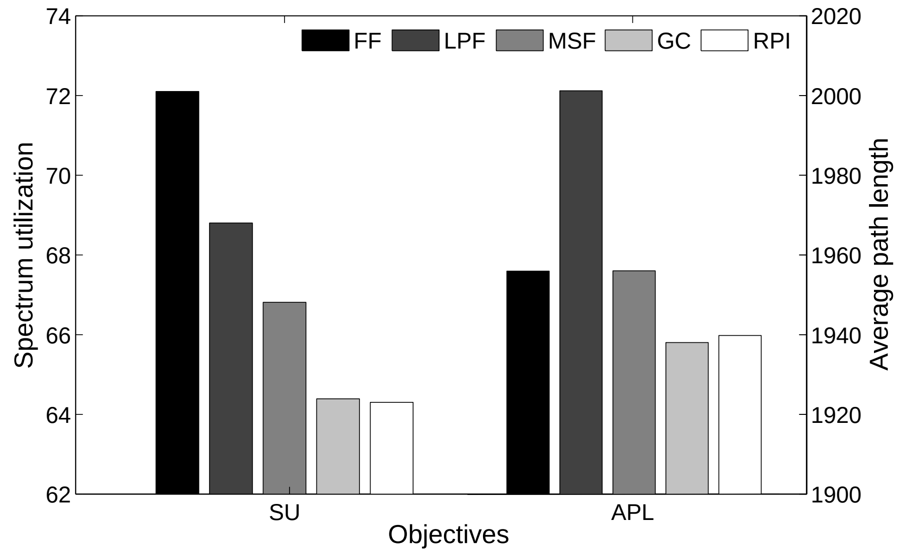


Figure 5.16: Comparison between heuristics and DE-RSA algorithms in the EON.

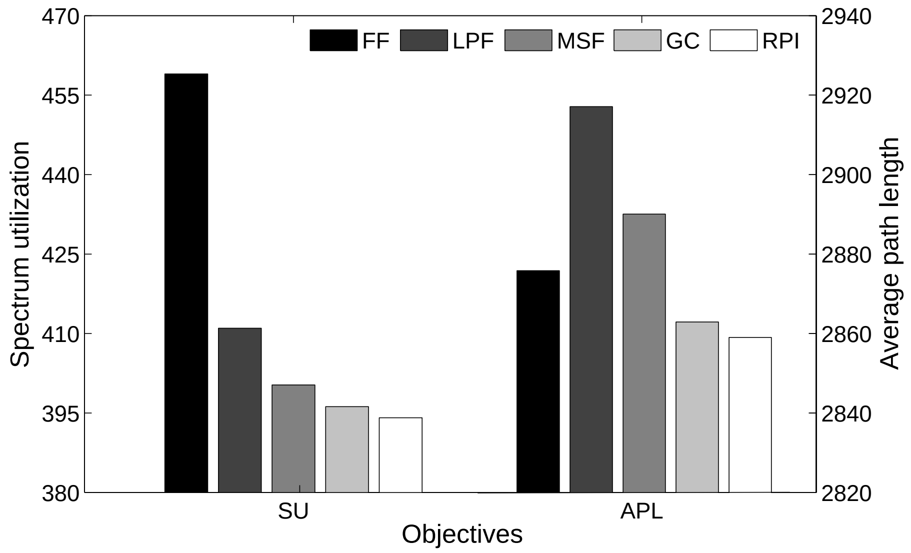


Figure 5.17: Comparison between heuristics and DE-RSA algorithms in the USA network.

5.7 Conclusions

In this chapter, we propose the application of a differential evolution (DE) algorithm to the off-line RSA problem. For this combinatorial problem, we applied two DE permutation-based approaches named DE-RPI and DE-GC. We confirm the effective capabilities in terms of the quality of the solutions obtained, minimizing the SU as well as the APL. We present an

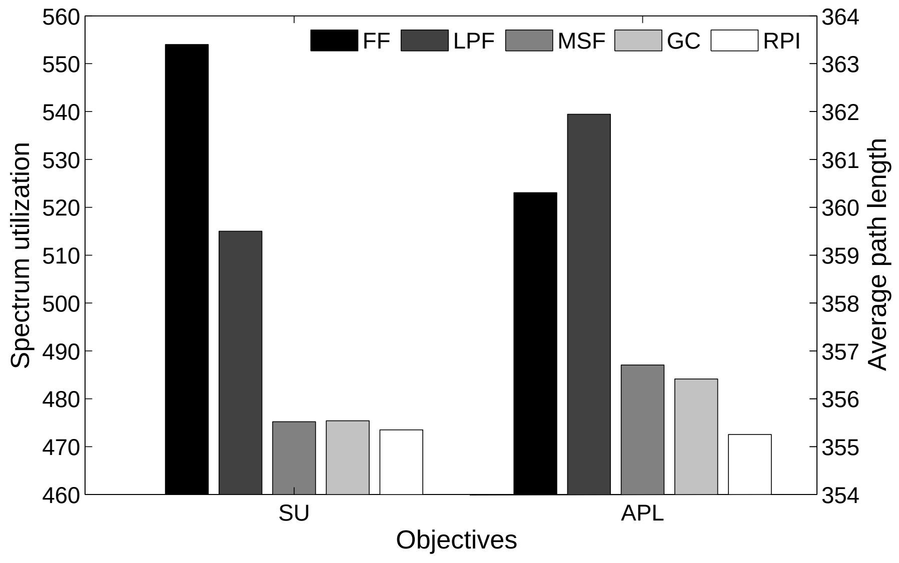
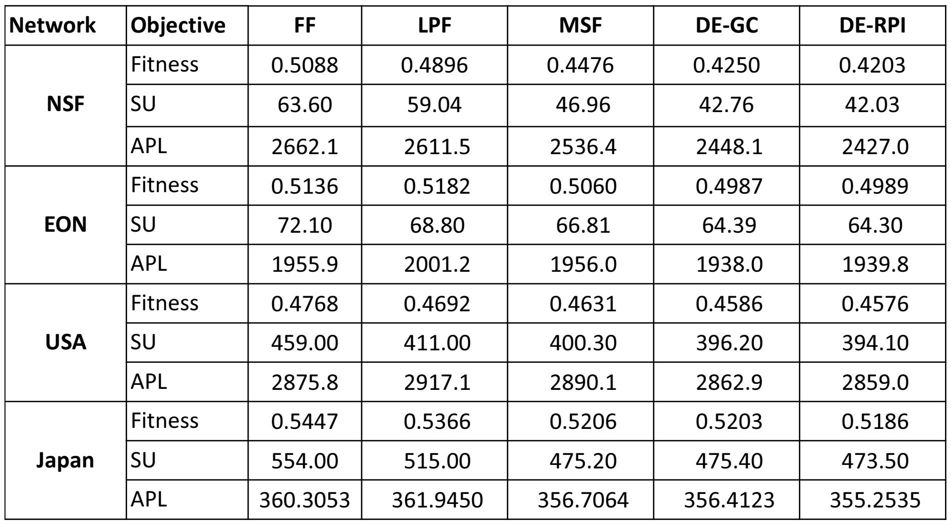


Figure 5.18: Comparison between heuristics and DE-RSA algorithms in the Japan network.



Network	**Objective**	**FF**	**LPF**	**MSF**	**DE-GC**	**DE-RPI**
NSF	Fitness	0.5088	0.4896	0.4476	0.4250	0.4203
	SU	63.60	59.04	46.96	42.76	42.03
	APL	2662.1	2611.5	2536.4	2448.1	2427.0
EON	Fitness	0.5136	0.5182	0.5060	0.4987	0.4989
	SU	72.10	68.80	66.81	64.39	64.30
	APL	1955.9	2001.2	1956.0	1938.0	1939.8
USA	Fitness	0.4768	0.4692	0.4631	0.4586	0.4576
	SU	459.00	411.00	400.30	396.20	394.10
	APL	2875.8	2917.1	2890.1	2862.9	2859.0
Japan	Fitness	0.5447	0.5366	0.5206	0.5203	0.5186
	SU	554.00	515.00	475.20	475.40	473.50
	APL	360.3053	361.9450	356.7064	356.4123	355.2535

Figure 5.19: Fitness comparison between heuristics and DE-RSA algorithms.

illustrative experiment to demonstrate the methodology of our DE-based algorithms and show the effectiveness and efficiency of the proposed evolutionary algorithm. Despite the solutions are somewhat sensible to variations of the DE-based algorithm parameters, the computed results show that a good combination of these parameters leads to a system performance's improvement. In addition, the computed results show that a more efficient RSA is obtained with the DE-based algorithms for realistic flexgrid scenarios when compared to other ordering policies. We have provided a practical DE model to solve the RSA problem that is simple to implement and could also be extended by adding other features into the objective function. As further work, we propose to investigate the use of the DE-based algorithms including other features, such as to choose the modulation level (BPSK, QPSK, 8-QAM, etc.) depending e.g. on the length of the path or even to consider physical impairments. It will also be interesting to analyze the use of intensification procedures like path-relinking to perform local search.

Chapter 6

CONCLUSIONS AND FURTHER WORK

Through the doctoral dissertation, we presented some of the most important nature-inspired algorithms, and also the important role they play in the solution of optimization problems. We have focused our attention in the optical networking field, in which the applications of optimization algorithms are vast due to the high complexity of the problems that arises during the evolution of the technologies.

The field of optical networking optimization as well as the growing diversity of approaches in the literature opens a window for research opportunities. For this reason, we consider of paramount importance to identify these research opportunities and propose methods or tools to solve some of the problems in optical networks.

Even when there have been applied many algorithm to these problems, due to their importance it is a necessity to developed new optimization tools to find efficient solutions in less amount of time. In the category of evolutionary algorithm, DE is arguably one of the most powerful stochastic real-parameter optimization algorithms in current use.

We applied DE not only to the problems of RWA and SVTM in WDM networks, but also to the RSA problem in elastic networks. Through the different chapters of this doctoral dissertation, we have demonstrated that DE can achieve good solutions in an acceptable amount of time for very complex problems in optical networks.

We have presented illustrates examples to understand clearly the application of the DE algorithm to each problem. Moreover, the results show the effectiveness and efficiency of the proposed DE-based algorithms.

A good number of publications validate the results of the research done through this doctoral dissertation. At the end of this doctoral dissertation, a detailed list of the different published papers obtained with the research of this work is presented.

Finally, it is clear that the application of computational intelligence has relevant importance in solving optimization problems. Many directions can be followed to continue with this research. The most important thing to recall is that optical networking is a very rich field with many combinatorial optimization problems. The entire field of optical networks is huge. For that reason, we focused on design and network planning issues, and the use of differential evolution algorithms. Nevertheless, there are plenty algorithms under developing, such as such as evolutionary game theory (EGT) [121] among other heuristics [122]. Moreover, many other areas in optical networks, such as the design of equipment, physical layer, green networks, among others can be and must be analyzed in the future.

Bibliography

[1] C. Saradhi and S. Subramaniam, “Physical layer impairment aware routing (PLIAR) in WDM optical networks: issues and challenges,” *IEEE Communications Surveys and Tutorials*, vol. 11, no. 4, pp. 109–130, 2009.

[2] R. Ramaswami and K. Sivarajan, “Routing and wavelength assignment in all-optical networks,” *IEEE/ACM Transactions on Networking*, vol. 3, pp. 489–500, oct 1995.

[3] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, “Survivable WDM mesh networks,” *Journal of Lightwave Technology*, vol. 21, pp. 870–883, april 2003.

[4] G. Zhang, M. De Leenheer, A. Morea, and B. Mukherjee, “A survey on OFDM-based elastic core optical networking.,” *IEEE Communications Surveys and Tutorials*, vol. 15, pp. 65–87, 2013.

[5] L. Velasco, M. Klinkowski, M. Ruiz, and J. Comellas, “Modeling the routing and spectrum allocation problem for flexgrid optical networks.,” *Photonic Network Communications*, vol. 24, pp. 177–186, 2012.

[6] M. Gagnaire, M. Koubaa, and N. Puech, “Network dimensioning under scheduled and random lightpath demands in all-optical WDM networks,” *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 9, pp. 58–67, 2007.

[7] E. Modiano and A. Narula-Tam, “Survivable lightpath routing: a new approach to the design of WDM-based networks,” *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 800–809, may 2002.

[8] I. Chlamtac, A. Ganz, and G. Karmi, “Lightpath communications: an approach to high bandwidth optical WAN’s,” *IEEE Transactions on Communications*, vol. 40, pp. 1171–1182, July 1992.

[9] K. Christodoulopoulos, I. Tomkos, and E. Varvarigos, "Elastic bandwidth allocation in flexible OFDM-based optical networks," *Journal of Lightwave Technology*, vol. 29, pp. 1354–1366, 2011.

[10] S. Das and P. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, pp. 4–31, feb. 2011.

[11] C. Riziotis and A. V. Vasilakos, "Computational intelligence in photonics technology and optical networks: A survey and future perspectives," *Information Sciences*, vol. 177, no. 23, pp. 5292–5315, 2007.

[12] H. Zang and J. P. Jue, "A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks," *Optical Networks Magazine*, vol. 1, pp. 47–60, 2000.

[13] D. Zhou and S. Subramaniam, "Survivability in optical networks," *IEEE Network*, vol. 14, pp. 16–23, nov/dec 2000.

[14] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[15] J. S. Choi, N. Golmie, F. Lapeyrere, F. Mouveaux, and D. Su, "A functional classification of routing and wavelength assignement schemes in DWDM networks: Static case," in *In proceedings of the International conference on Optical Communication and Networks*, 2000.

[16] G. Castañón, F. Lezama, and A. M. Sarmiento., "Wavelength converters placement in all optical networks using differential evolution optimization," in *Proceedings of the International Conference on Evolutionary Computation*, pp. 340–343, 2010.

[17] F. Lezama, F. Callegati, W. Cerroni, and L. H. Bonani, "Trunk reservation for elastic optical networks," in *Proceedings of the International Conference on Transparent Optical Networks*, pp. 1–4, 2013.

[18] F. Lezama, G. Castanon, and A. Sarmiento, "Survivable virtual topology mapping in IP-over-WDM networks using differential evolution optimization," in *Proceedings of the International Conference on Transparent Optical Networks*, pp. 1–4, 2013.

[19] F. Lezama, G. Castañón, A. Sarmiento, and I. Martins, "Routing and spectrum allocation in flexgrid optical networks using differential evolution optimization," in *Proceedings of the International Conference on Transparent Optical Networks*, pp. 1–4, 2014.

[20] I. Martins, G. Castan, F. Lezama, and I. Aldaya, "Impact of spectral width on signal transmission distance in elastic optical network," in *Proceedings of the International Conference on Transparent Optical Networks*, 2014.

[21] F. Lezama, G. Castañón, and A. M. Sarmiento, "Differential evolution optimization applied to the wavelength converters placement problem in all optical networks," *Computer Networks*, vol. 56, pp. 2262–2275, June 2012.

[22] F. Lezama, G. Castañón, and A. M. Sarmiento, "Routing and wavelength assignment in all optical networks using differential evolution optimization," *Photonic Network Communications*, vol. 26, no. 2-3, pp. 103–119, 2013.

[23] F. Lezama, G. Castan, A. M. Sarmiento, F. Callegati, and W. Cerroni, "Survivable virtual topology mapping in ip-over-wdm networks using differential evolution optimization," *Photonic Network Communications*, vol. (DOI) 10.1007/s11107-014-0455-1, 2014.

[24] F. Callegati, L. Bonani, F. Lezama, W. Cerroni, A. Campi, and G. Castañón, "Trunk reservation for fair utilization in flexible optical networks," *IEEE Communications Letters*, vol. 18, no. 5, pp. 889–892, 2014.

[25] A. M. Sarmiento and F. Castañón, G.and Lezama, *Wavelength and Routing Assignment in All Optical Networks Using Ant Colony Optimization*. Intelligent Systems for Optical Networks Design: Advancing Techniques. IGI-Global, 2013.

[26] L. Sahasrabuddhe, S. Ramamurthy, and B. Mukherjee, "Fault management in ip-over-wdm networks: Wdm protection versus ip restoration," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 1, pp. 21–33, 2002.

[27] Y. Yamada, H. Hasegawa, and K. Sato, "Survivable hierarchical optical path network design with dedicated wavelength path protection," *Journal of Lightwave Technology*, vol. 29, pp. 3196–3209, nov.1, 2011.

[28] V. S. Kartalopoulos, *DWDM: Networks, Devices and Technology.* Jhon Wiley and Sons, Hoboken/New Jersey, 2002.

[29] X.-S. Yang, *Nature-inspired Metaheuristics Algorithms.* Luniver Press. Frome, BA11 6TT, United Kingdom, 2010.

[30] X.-S. Yang, *Nature-inspired Optimization Algorithms.* Elsevier Science, 2014.

[31] J. H. Holland, *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.

[32] D. E. Goldberg and K. Deb, *A comparative analysis of selection schemes used in genetic algorithms.* Morgan Kaufmann, 1991.

[33] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[34] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[35] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *proceedings of the IEEE International Conference on Evolutionary Computation (CEC)*, pp. 69–73, 1998.

[36] Z.-L. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1187–1195, 2003.

[37] B. Zhao, C. Guo, and Y. Cao, "Improved particle swam optimization algorithm for OPF problems," in *Proceedings of the IEEE Power Systems Conference and Exposition*, vol. 1, pp. 233–238, 2004.

[38] C.-M. Huang, C.-J. Huang, and M.-L. Wang, “A particle swarm optimization to identifying the ARMAX model for short-term load forecasting,” *IEEE Transactions on Power Systems*, vol. 20, no. 2, pp. 1126–1133, 2005.

[39] A. Colorni, M. Dorigo, and V. Maniezzo, “Distributed optimization by ant colonies,” in *proceedings of the European Conf. on Artificial Life (ECAL)*, pp. 134–142, 1991.

[40] M. Dorigo, V. Maniezzo, and A. Colorni, “The ant system: An autocatalytic optimizing process,” Tech. Rep. (Technical Report TR91-016), Politecnico di Milano, Milano, Italy., 1991.

[41] M. Dorigo, G. Di Caro, and L. M. Gambardella, “Ant algorithms for discrete optimization,” *Artificial Life*, vol. 5, pp. 137–172, Apr. 1999.

[42] B. C. Mohan and R. Baskaran, “A survey: Ant colony optimization based recent research and implementation on several engineering domain,” *Expert Systems with Applications*, vol. 39, no. 4, pp. 4618–4627, 2012.

[43] E. Bonabeau, “Editor’s introduction: Stigmergy,” *Artificial Life*, vol. 5, no. 2, pp. 95–96, 1999.

[44] J. Pasteels, J. L. Deneubourg, and S. Goss, “Self-organization mechanisms in ant societies (I): Trail recruitment to newly discovered food sources,” *Experientia Supplementum*, pp. 155–175, 1987.

[45] S. Goss, S. Aron, J. Deneubourg, and J. Pasteels, “Self-organized shortcuts in the argentine ant,” *Naturwissenschaften*, vol. 76, no. 12, pp. 579–581, 1989.

[46] J.-L. Deneubourg, S. Aron, S. Goss, and J. Pasteels, “The self-organizing exploratory pattern of the argentine ant,” *Journal of Insect Behavior*, vol. 3, no. 2, pp. 159–168, 1990.

[47] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: optimization by a colony of cooperating agents,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.

[48] M. Dorigo and L. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[49] D. Karaboga, “An idea based on honey bee swarm for numerical optimization,” tech. rep., Technical report, Engineering Faculty, Erciyes University, Kayseri, Trkiye, 2005.

[50] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, “A comprehensive survey: artificial bee colony (ABC) algorithm and applications,” *Artificial Intelligence Review*, vol. 42, no. 1, pp. 21–57, 2012.

[51] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems.* New York, NY: Oxford University Press, Santa Fe Institute Studies in the Sciences of Complexity, 1999.

[52] V. Tereshko and A. Loengarov, “Collective decision making in honey-bee foraging dynamics,” *Computing and information systems*, vol. 9, no. 3, pp. 1–7, 2005.

[53] K. V. Price, “Genetic annealing,” *Dr. Dobbs Journal*, vol. 19, no. 11, pp. 127–132, 1994.

[54] R. Storn and K. Price, “Minimizing the real functions of the ICEC’96 contest by differential evolution,” in *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 842–844, 1996.

[55] R. Storn, “On the usage of differential evolution for function optimization,” in *Proceedings of the Biennial Conference of the North American Fuzzy Information Processing Society*, pp. 519–523, 1996.

[56] K. Price and R. Storn, “Differential evolution: a simple evolution strategy for fast optimization,” *Dr. Dobbs Journal*, vol. 22, pp. 18–24, 1997.

[57] H. Afaq and S. Saini, “On the solutions to the travelling salesman problem using nature inspired computing techniques,” *Journal of Computer Science Issues*, vol. 8, no. 4, pp. 326–334, 2011.

[58] S. Azodolmolky, M. Klinkowski, E. Marin, D. Careglio, J. S. Pareta, and I. Tomkos, “A survey on physical layer impairments aware routing and wavelength assignment algorithms in optical networks,” *Computer Networks*, vol. 53, no. 7, pp. 926–944, 2009.

[59] R. Randhawa and J. Sohal, “Static and dynamic routing and wavelength assignment algorithms for future transport networks,” *Optik - International Journal for Light and Electron Optics*, vol. 121, no. 8, pp. 702–710, 2010.

[60] D. Banerjee and B. Mukherjee, “A practical approach for routing and wavelength assignment in large wavelength-routed optical networks,” *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 903–908, June 1996.

[61] G. Varela and M. Sinclair, “Ant colony optimisation for virtual-wavelength-path routing and wavelength allocation,” in *Proceedings of the Congress of Evolutionary Computation*, pp. 1809–1816, 1999.

[62] S.-H. Ngo, X. Jiang, S. Horiguchi, and M. Guo, “Dynamic routing and wavelength assignment in WDM networks with ant-based agents,” in *Embedded and Ubiquitous Computing* (L. Yang, M. Guo, G. Gao, and N. Jha, eds.), vol. 3207, pp. 829–838, Springer Berlin Heidelberg, 2004.

[63] J. Triay and C. Cervelló-Pastor, “An ant-based algorithm for distributed routing and wavelength assignment in dynamic optical networks,” *IEEE Journal on Selected Areas in Communications*, vol. 28, pp. 542–552, May 2010.

[64] K. Bhaskaran, J. Triay, and V. Vokkarane, “Dynamic anycast routing and wavelength assignment in WDM networks using ant colony optimization (ACO),” in *proceedings of the IEEE International Conference on Communications (ICC)*, pp. 1–6, 2011.

[65] M. Al-Momin, J. Cosmas, and S. Amin, “Enhanced ACO based RWA on WDM optical networks using requests accumulation and re-sorting method,” in *Proceedings of the Computer Science and Electronic Engineering Conference*, pp. 97–102, Sept 2013.

[66] D. Bisbal, I. de Miguel, F. Gonzalez, J. Blas, J. C. Aguado, P. Fernandez, J. Duran, R. Duran, R. M. Lorenzo, E. J. Abril, and M. Lopez, “Dynamic routing and wavelength assignment in optical networks by means of genetic algorithms,” *Photonic Network Communications*, vol. 7, no. 1, pp. 43–58, 2004.

[67] N. Banerjee and S. Sharan, “A evolutionary algorithm for solving the single objective static routing and wavelength assignment problem in WDM networks,” in *Proceedings*

of International Conference on Intelligent Sensing and Information Processing, pp. 13 – 18, 2004.

[68] T. Rao and V. Anand, “Particle swarm optimization for routing and wavelength assignment in optical networks,” in *proceedings of the IEEE Sarnoff Symposium.*, pp. 1–4, march 2006.

[69] A. Hassan and C. Phillips, “Static routing and wavelength assignment inspired by particle swarm optimization,” in *Proceedingsof the International Conference on Information and Communication Technologies*, pp. 1 –6, april 2008.

[70] A. Hassan and C. Phillips, “Chaotic particle swarm optimization for dynamic routing and wavelength assignment in all-optical WDM networks,” in *proceedings of the International Conference on Signal Processing and Communication Systems (ICSPCS).*, pp. 1–7, Sept 2009.

[71] T. F. Noronha, M. G. Resende, and C. C. Ribeiro, “A biased random-key genetic algorithm for routing and wavelength assignment,” *Journal of Global Optimization*, vol. 50, pp. 503–518, July 2011.

[72] R. Barpanda, A. Turuk, B. Sahoo, and B. Majhi, “Genetic algorithm techniques to solve routing and wavelength assignment problem in wavelength division multiplexing all-optical networks,” in *proceedings of the International Conference on Communication Systems and Networks (COMSNETS).*, pp. 1–8, Jan 2011.

[73] A. Rubio-Largo, M. A. Vega-Rodriguez, J. A. Gomez-Pulido, and J. M. Sanchez-Perez, “A differential evolution with pareto tournaments for solving the routing and wavelength assignment problem in WDM networks,” in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, july 2010.

[74] Y. S. Kavian, A. Rashedi, A. Mahani, and Z. Ghassemlooy, “Routing and wavelength assignment in optical networks using artificial bee colony algorithm,” *Optik - International Journal for Light and Electron Optics*, vol. 124, pp. 1243–1249, June 2013.

[75] A. Rubio-Largo, M. A. Vega-Rodriguez, J. A. Gomez-Pulido, and J. M. Sanchez-Perez, “A multiobjective approach based on artificial bee colony for the static routing and wavelength assignment problem,” *Soft Computing*, vol. 17, no. 2, pp. 199–211, 2013.

[76] K. Li, "Heuristic algorithms for routing and wavelength assignment in WDM optical networks," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing*, pp. 1 –8, 2008.

[77] E. Kaldirim, F. Ergin, S. Uyar, and A. Yayimli, "Ant colony optimization for survivable virtual topology mapping in optical WDM networks," in *Proceedings of the International Symposium on Computer and Information Sciences*, pp. 334 –339, sept. 2009.

[78] M. Kurant and P. Thiran, "Survivable mapping algorithm by ring trimming (SMART) for large IP-over-WDM networks," in *Proceedings of the International Conference on Broadband Networks*, BROADNETS '04, (Washington, DC, USA), pp. 44–53, IEEE Computer Society, 2004.

[79] F. Ducatelle and L. Gambardella, "A scalable algorithm for survivable routing in IP-over-WDM networks," in *Proceedings of the Conference on Broadband Networks*, pp. 54 – 63, oct. 2004.

[80] Y. Liu, D. Tipper, and P. Siripongwutikorn, "Approximating optimal spare capacity allocation by successive survivable routing," *IEEE/ACM Transactions on Networking*, vol. 13, pp. 198–211, feb. 2005.

[81] A. Todimala and B. Ramamurthy, "A scalable approach for survivable virtual topology routing in optical WDM networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, pp. 63–69, august 2007.

[82] A. Y. Fatma Corut Ergin and S. Uyar, "An evolutionary algorithm for survivable virtual topology mapping in optical WDM networks," in *Applications of Evolutionary Computing*, vol. 5484 of *Lecture Notes in Computer Science*, pp. 31–40, Springer Berlin / Heidelberg, 2009.

[83] H. Chong and S. Kwong, "Optimization of spare capacity in survivable WDM networks," in *Genetic and Evolutionary Computation (GECCO)* (E. Canta-Paz, J. Foster, K. Deb, L. Davis, R. Roy, U.-M. O'Reilly, H.-G. Beyer, R. Standish, G. Kendall, S. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. Potter, A. Schultz, K. Dowsland, N. Jonoska, and J. Miller, eds.), vol. 2724, pp. 2396–2397, Springer Berlin Heidelberg, 2003.

[84] C. S. Ou and B. Mukherjee, *Survivable Optical WDM Networks.* Springer US, 2005.

[85] J. Skorin-Kapov and D. Skorin-Kapov, *Handbook of Research on Telecommunications Planning and Management for Business*, ch. Trends, Optimization and Management of Optical Networks, pp. 941–956. IGI Global, 2009.

[86] R. Barry and P. Humblet, "Models of blocking probability in all-optical networks with and without wavelength changers," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 858–867, jun 1996.

[87] A. Birman, "Computing approximate blocking probabilities for a class of all-optical networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 852–857, jun 1996.

[88] S. Subramaniam, M. Azizoglu, and A. Somani, "All-optical networks with sparse wavelength conversion," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 544–557, aug 1996.

[89] K. R. Venugopal, M. Shiva Kumar, and P. Sreenivasa Kumar, "A heuristic for placement of limited range wavelength converters in all-optical networks," *Computer Networks*, vol. 35, no. 2-3, pp. 143–163, 2001.

[90] C. Vijayanand, M. Shiva Kumar, K. R. Venugopal, and P. Sreenivasa Kumar, "Converter placement in all-optical networks using genetic algorithms," *Computer Communications*, vol. 23, no. 13, pp. 1223–1234, 2000.

[91] J. Siregar, H. Takagi, and Y. Zhang, "Optimal wavelength converter placement in optical networks by genetic algorithm," *IEICE Transactions on Communications*, vol. E85-B(6), pp. 1075–1082, 2002.

[92] L. Li and A. K. Somani, "Efficient algorithms for wavelength converter placement," *Optical Networking Magazine*, vol. 3, no. 2, pp. 54–62, 2002.

[93] S. Thiagarajan and A. K. Somani, "Optimal wavelength converter placement in arbitrary topology wavelength-routed networks," *Computer Communications*, vol. 26, no. 9, pp. 975–985, 2003.

[94] C. Teo, Y. C. Foo, S. Chien, A. Low, A. You, and G. Castañon, "Wavelength converters placement in all optical networks using particle swarm optimization," *Photonic Network Communication*, vol. 10, pp. 23–37, julio 2005.

[95] M. Alicherry, H. Nagesh, and V. Poosala, "Constraint-based design of optical transmission systems," *Journal of Lightwave Technology*, vol. 21, pp. 2499–2510, Nov 2003.

[96] M. O'Mahony, "Semiconductor laser optical amplifiers for use in future fiber systems," *Journal of Lightwave Technology*, vol. 6, pp. 531–544, April 1988.

[97] G. Agrawal, *Nonlinear Fiber Optics.* Academic Press. San Diego, USA, 1995.

[98] B. Ramamurthy, J. Iness, and B. Mukherjee, "Optimizing amplifier placements in a multiwavelength optical LAN/MAN: the equally powered-wavelengths case," *Journal of Lightwave Technology*, vol. 16, pp. 1560–1569, Sep 1998.

[99] A. F. Fumagalli, G. Balestra, and L. Valcarenghi, "Optimal amplifier placement in multiwavelength optical networks based on simulated annealing," in *Proceedings of SPIE: All-Optical Networking: Architecture, Control, and Management Issues*, vol. 3531, pp. 268–279, 1998.

[100] T. Deng and S. Subramaniam, "Amplifier placement in transparent DWDM ring networks," in *Proceedings of SPIE: Opticomm*, vol. 5285, pp. 246–257, 2003.

[101] A. Tran, R. Tucker, and N. Boland, "Amplifier placement methods for metropolitan WDM ring networks," *Journal of Lightwave Technology*, vol. 22, pp. 2509–2522, Nov 2004.

[102] L. Zhong and B. Ramamurthy, "Optimization of amplifier placements in switch-based optical networks," in *Proceedings of the IEEE International Conference on Communications*, vol. 1, pp. 224–228, Jun 2001.

[103] A. Hamad and A. Kamal, "Optical amplifiers placement in WDM mesh networks for optical multicasting service support," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 1, pp. 85–102, June 2009.

[104] S. Singh and R. Kaler, “Placement of optimized semiconductor optical amplifier in fiber optical communication systems,” *Optik - International Journal for Light and Electron Optics*, vol. 119, no. 6, pp. 296–302, 2008.

[105] G. Ferreira, S. P. N. Cani, M. Pontes, and M. E. V. Segatto, “Optimization of distributed raman amplifiers using a hybrid genetic algorithm with geometric compensation technique,” *IEEE Photonics Journal*, vol. 3, pp. 390–399, June 2011.

[106] Z. Ye and G. Auner, “Raman spectrum baseline identification and its optimization using genetic algorithm,” in *Proceedings of the IEEE International Symposium on Intelligent Control*, pp. 489–494, Sept 2004.

[107] A. Jukan and G. Franzl, “Path selection methods with multiple constraints in service-guaranteed WDM networks,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 1, pp. 59–72, 2004.

[108] Z.-X. Wang, Z.-Q. Chen, and Z.-Z. Yuan, “QoS routing optimization strategy using genetic algorithm in optical fiber communication networks,” *Journal of Computer Science and Technology*, vol. 19, no. 2, pp. 213–217, 2004.

[109] J. Wang, X. Wang, P. Liu, and M. Huang, “A differential evolution based flexible QoS multicast routing algorithm in NGI,” in *proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, pp. 250–253, 2006.

[110] W. Xing-Wei, Y. Hai-Quan, H. Min, and L. Guo, “ABC supporting QoS unicast routing scheme with particle swarm optimization,” in *Proceedings of the Asian Conference on Intelligent Information and Database Systems*, pp. 426–429, 2009.

[111] H. Cheng, X. Wang, S. Yang, M. Huang, and J. Cao, “QoS multicast tree construction in IP/DWDM optical internet by bio-inspired algorithms,” *Journal of Network and Computer Applications*, vol. 33, no. 4, pp. 512–522, 2010.

[112] E. Guillen, Y. Camargo, and P. Estupiñán, “Hybrid approaches in network optical routing with QoS based on genetic algorithms and particle swarm optimization.,” *International Journal on Network Security*, vol. 2, no. 4, 2011.

[113] Z. Zheng, H. Wang, and L. Yao, "An artificial bee colony optimization algorithm for multicast routing," in *Proceedings of the International Conference on Advanced Communication Technology*, pp. 168–172, 2012.

[114] I. Flores-Llamas, F. Moumtadi, and J. Delgado-Hernandez, "Gain equalization for erbium-doped fiber amplifiers by the synthesis of a long-period fiber grating," in *Proceedigns of the IEEE conference on Electronics, Robotics and Automotive Mechanics*, pp. 333–337, 2011.

[115] I. Abdelaziz, F. AbdelMalek, S. Haxha, H. Ademgil, and H. Bouchriha, "Photonic crystal fiber with an ultrahigh birefringence and flattened dispersion by using genetic algorithms," *Journal of Lightwave Technology*, vol. 31, no. 2, pp. 343–348, 2013.

[116] 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.

[117] F. Callegati, L. Bonani, and W. Cerroni, "Service fairness in flexible optical networks," in *proceedigns of the Optical Fiber Communication Conference and Exposition and the National Fiber Optic Engineers Conference (OFC/NFOEC)*, p. OTh1H.4, 2013.

[118] H. Tode, K. Hamada, and K. Murakami, "ORGAN: Online route and wavelength design based on genetic algorithm for OPS networks," in *proceedings of the Conference on Optical Network Design and Modeling (ONDM)*, pp. 1–6, 2010.

[119] A. Belbekkouche, A. Hafid, M. Tagmouti, and M. Gendreau, "A novel formulation for routing and wavelength assignment problem in OBS networks," in *proceedings of the IEEE International Conference on Communications (ICC)*, pp. 1–6, 2010.

[120] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 414–454, 2014.

[121] M. Anastasopoulos and A. Tzanakaki, "Adaptive virtual infrastructure planning over interconnected IT and optical network resources using evolutionary game theory," in *proceedings of the International Conference on Optical Network Design and Modeling (ONDM)*, pp. 1–5, 2012.

[122] L. Guo, X. Wang, Y. Liu, L. Zhang, and W. Hou, "Game theory based inter-domain protection in multi-domain optical networks," in *proceedings of the International Conference on Information Networking and Automation (ICINA)*, vol. 1, pp. 480–485, 2010.

[123] S. K. S. G. Ramaswami, R., *Optical Networks: A Practical Perspective.* Morgan Kaufmann, Burlington., 2009.

[124] A. Somani and M. Azizoglu, "Wavelength assignment algorithms for wavelength routed interconnection of LANs," *IEEE Journal of Lightwave Technology*, vol. 18, no. 12, pp. 1807 –1817, 2000.

[125] N. Skorin-Kapov, "Routing and wavelength assignment in optical networks using bin packing based algorithms," *European Journal of Operational Research*, vol. 177, no. 2, pp. 1167 – 1179, 2007.

[126] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *proceedings of the Congress on Evolutionary Computation*, vol. 2, pp. 1980–1987, 2004.

[127] N. Nagatsu, Y. Hamazumi, and K.-i. Sato, "Optical path accommodation designs applicable to large scale networks," *IEICE Transactions on Communications*, vol. E78-B, no. 4, pp. 597–607, 1995.

[128] D. Wischik, *Routing and wavelength assignment in optical networks.* University of Cambridge, May 1996.

[129] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[130] R. Storn, "Differential evolution research - trends and open questions," in *Advances in Differential Evolution* (U. Chakraborty, ed.), vol. 143 of *Studies in Computational Intelligence*, pp. 1–31, Springer Berlin Heidelberg, 2008.

[131] S. Baroni and P. Bayvel, "Wavelength requirements in arbitrarily connected wavelength-routed optical networks," *Journal of Lightwave Technology*, vol. 15, pp. 242–251, Feb. 1997.

[132] B. Mukherjee, *Optical WDM Networks*. New York: Springe, 2006.

[133] A. Y. F. C. Ergin, E. Kaldirim and A. Uyar., "Ensuring resilience in optical WDM networks with nature-inspired heuristics," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 2 (8), pp. 642–652, 2010.

[134] K. Christodoulopoulos, I. Tomkos, and E. Varvarigos, "Routing and spectrum allocation in OFDM-based optical networks with elastic bandwidth allocation.," in *Proceedings of the IEEE Global Telecommunications Conference*, pp. 1–6, 2010.

[135] J. Jue, W.-H. Yang, Y.-C. Kim, and Q. Zhang, "Optical packet and burst switched networks: a review," *IET Communications*, vol. 3, no. 3, pp. 334–352, 2009.

[136] L. Gong, X. Zhou, W. Lu, and Z. Zhu, "A two-population based evolutionary approach for optimizing routing, modulation and spectrum assignments (RMSA) in O-OFDM networks," *IEEE Communications Letters*, vol. 16, pp. 1520–1523, 2012.

[137] Y. Wang, X. Cao, and Y. Pan., "A study of the routing and spectrum allocation in spectrum-sliced elastic optical path networks.," in *Proceedings of the IEEE INFOCOM* (1503-1511, ed.), 2011.

[138] M. Klinkowski and K. Walkowiak, "Routing and spectrum assignment in spectrum sliced elastic optical path network.," *IEEE Communications Letters*, vol. 15, pp. 884–886, 2011.

[139] R. Prado, R. Silva, F. Guimarães, and O. Neto, "Using differential evolution for combinatorial optimization: A general approach," in *Proceedings of the IEEE International Conference on Systems Man and Cybernetics*, pp. 11–18, Oct 2010.

[140] D. Lichtblau, "Relative position indexing approach," in *Differential Evolution: A Handbook for Global Permutation-Based Combinatorial Optimization* (G. Onwubolu and D. Davendra, eds.), vol. 175 of *Studies in Computational Intelligence*, pp. 81–120, Springer Berlin Heidelberg, 2009.

[141] A. Moraglio and R. Poli, "Geometric crossover for the permutation representation," *Intelligenza Artificiale*, vol. 5, no. 1, pp. 49–63, 2011.

Published Work

by

Fernando Lezama Cruzvillasante

Journal Publications:

- Lezama, F., Castañón, G., Sarmiento, A. M.: **Differential evolution optimization applied to the wavelength converters placement problem in all optical networks**. *Computer Networks*, 56(9), pp. 2262-2275, Elsevier (2012).

- Lezama, F., Castañón, G., Sarmiento, A. M.: **Routing and wavelength assignment in all optical networks using differential evolution optimization**. *Photonic Network Communications*. 26(2-3), pp. 103-119, Springer (2013).

- Lezama, F., Castañón, G., Sarmiento, A. M., Callegati, F., Cerroni, W.: **Survivable virtual topology mapping in IP-over-WDM networks using differential evolution optimization**. *Photonic Network Communications*, Springer (2014). (DOI) 10.1007/s11107-014-0455-1.

- Callegati, F., Bonani, L., Lezama, F., Cerroni, W., Campi, A., Castañón, G.: **Trunk Reservation for Fair Utilization in Flexible Optical Networks**. *IEEE Communications Letters*. 18(5), pp. 889-892, (2014).

- Lezama, F. and Castañón, G.: **A Survey of Nature-Inspired Algorithms Applications in Optical Networks.** Submitted to: *Artificial Intelligence Review*, Springer (September 2014).

- Lezama, F., Castañón, G., Sarmiento, A. M., and Martins, I. B.: **Differential evolution optimization applied to the routing and spectrum allocation problem in flexgrid optical networks.** Submitted to: *Photonic Network Communications*, Springer (December 2014).

Proceedings Publications:

- Castañón, G., Lezama, F., Sarmiento., A. M.: Wavelength converters placement in all optical networks using differential evolution optimization. In: Proc. of the International Conference on Evolutionary Computation, pp. 340343, Valencia, Spain, (2010).
- Lezama, F., Callegati, F., Cerroni, W., Bonani, L. H.: Trunk reservation for elastic optical networks, in: Proc. of International Conference on Transparent Optical Networks, paper Tu.C1.1, Cartagena, Spain, June 2013.
- Lezama, F., Castañón, G., Sarmiento, A. M.: Survivable virtual topology mapping in IP-over-WDM networks using differential evolution optimization, in: Proc. of International Conference on Transparent Optical Networks, paper Th.A3.4, Cartagena, Spain, June 2013.
- Lezama, F., Castañón, G., Sarmiento, A. M., and Martins, I. B.: Routing and spectrum allocation in flexgrid optical networks using differential evolution optimization, in: Proc. of International Conference on Transparent Optical Networks, paper Th.B3.5, Graz, Austria, July 2014.
- Martins, I., Castañón, G., Lezama, F., and Aldaya, I.: Impact of Spectral Width on Signal Transmission Distance in Elastic Optical Network, in: Proc. of International Conference on Transparent Optical Networks, paper Tu.P.19, Graz, Austria, July 2014.

Chapter Books:

- Sarmiento, A. M., Castañón, G., Lezama, F.: Wavelength and Routing Assignment in All Optical Networks Using Ant Colony Optimization. In: Intelligent Systems for Optical Networks Design: Advancing Techniques, ed. Yousef S. Kavian and Z. Ghassemlooy, pp. 217-234, IGI-Global (2013).
- Lezama, F., Castañón, G., Sarmiento, A. M.: On the impact of Differential Evolution parameters solving the Survivable virtual topology mapping problem in IP-over-WDM networks. Submitted to: Bio-Inspired Computation in Telecommunications. Ed. Xin-She Yang, S. F. Chien, T. O. Ting, Elsevier (2014). Status: Accepted: September, 2014.

Vita

Fernando Lezama was born in Córdoba, Veracruz, México, in 1986. He received the B.S. degree in Electronic Engineering from the Minatitlán Institute of Technology (ITM), Veracruz, México in 2009 and the M.Sc. degree (with honors) in Electronic Engineering (Telecommunications) from the Monterrey Institute of Technology and Higher Education (ITESM), México, in 2011.

He obtained the Ph.D degree in Information Technology and Communications at ITESM, México, in 2014. During his doctoral studies, he was a research visitor at the University of Bologna, Italy, from 2012 to 2013 where his research was related with optimization in elastic optical networks. He has been author and co-author of journal papers for editorials such as Elsevier, Springer and IEEE and also has participated in different international conferences as expositor. He was a member of the research chair of Optical Communications in the Center of Electronics and Telecommunications at ITESM since August of 2009.

His research interests include Optical Networks Optimization and Evolutionary Computation.

e-mail: ing.flezama@gmail.com

Typeset with LaTeX by Fernando Lezama Cruzvillasante.