Evolutionary algorithms for the design of multi-overlay robust networks

L. Gómez, F. Casalongue, G. Lasalt, F. Robledo, S. Nesmachnow
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
{lgomez,fcasalongue,glasalt,frobledo,sergion}@fing.edu.uy

Abstract—**Nowadays, overlay networks are an essential tool in telecommunications. However, few works have addressed the optimization of overlay and multi-overlay networks. This work focuses on solving the problem of designing a minimum-cost and fault-tolerant multi-overlay network. The problem is NP-hard, and exact techniques are not appropriate to compute accurate solutions efficiently. This work explores the first advances in the application of a sequential and a parallel genetic algorithm to solve the problem. The experimental analysis is performed on real-world scenarios built from a MPLS data network mounted over a multiple technology (SDH/DWDM) transport network, by the Uruguayan national telecommunications company. The analysis show that the studied genetic algorithms are able to obtain promising results, while the parallel model significantly speeds up the problem resolution.**

Keywords; genetic algorithm, network design, multi-overlay

I. INTRODUCTION

Telecommunications have grown at a fast pace. The rapid development of network infrastructures has been driven by the growing demand for data communication over the last 20 years. As a consequence, the research community has shown a renewed interest in network design problems [1]. Overlay and multi-overlay networks are an essential tool in telecommunication. An overlay network is built on top of another network, and their nodes can be seen as connected by virtual links, built using many physical links in the underlying network. Overlay networks are nowadays present in telephony and data communication, such as broadband Internet access, voice over IP, etc; and distributed computing systems such as cloud computing, peer-to-peer networks, and client-server applications [2].

The multi-overlay robust network design problem (MORNDP) is NP-hard, since the Minimum Weight 2-Connected Network Problem [3] is reducible to it. Therefore, classic exact techniques are only applicable to solve very small instances of MORNDP. Among a broad set of modern heuristics and metaheuristics methods for optimization, evolutionary algorithms (EAs) have emerged as promising tools for solving network design problems, as they are able to compute accurate approximate solutions in acceptable execution times [4]. In this line of work, this article explores the application of sequential and parallel Genetic Algorithms (GA) to solve the MORNDP.

The experimental analysis of the proposed GAs is performed on real-world scenarios provided by the Uruguayan national telecommunications company (ANTEL). In these scenarios, the MORNDP proposes to find the optimum design of an MPLS network mounted over a multi-technology transport network (DWDM/SDH) [5]. The company wishes to design a robust MPLS data network, which can meet certain estimations of the future commercial demands, at the lowest cost possible. The transport layer is considered as fixed, as the company is not planning to change its topology in the near future. As the transport network is an expensive and limited resource, the maintenance costs of the transport network are transferred to the MPLS network. In fact, all other costs are considered as negligible in comparison. Therefore, the problem proposes to minimize the use of the transport layer by the designed MPLS network. In addition, the proposed network must fulfill some reliability constraints to be considered as a robust design: the network must support the routing of certain traffic demands between nodes, even in the case that a single link failure arises in the transport network.

The main contributions of this article are the proposal of a sequential and a parallel GA for solving the MORNDP and the experimental evaluation on real-world network scenarios. The GAs have been designed using simple operators, that allows the proposed methods to be used to solve realistic MORNDP scenarios. Promising results are reported for the sequential and the parallel version of the GA studied in this work, while the parallel model significantly speeds up the problem resolution. The proposed GAs are implemented using a well-known library of algorithms for optimization, which allows designing reusable algorithms that can be easily extended to solve other specific variants of the problem.

The rest of the article is organized as follows. Section 2 presents an overview of the problem and its mathematical formulation, along with a brief summary of related works. The main concepts about evolutionary algorithms and the sequential and parallel GA applied in this work are presented in section 3. The design and implementation details of the proposed GAs are described in section 4. The experimental evaluation of the algorithms on real-world scenarios is reported in section 6. Finally, section 7 comments the main conclusions of the research and summarizes the possible lines for future work.

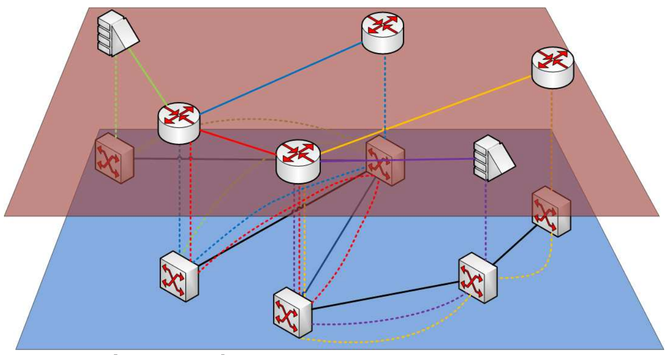
Fig. 1. Two layer network.

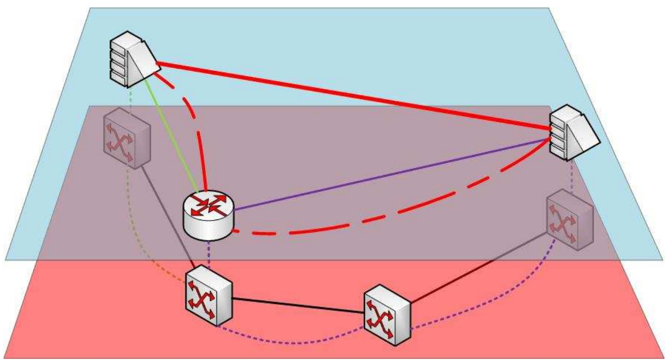
Fig. 2. A multi-overlay network.

II. MULTI-OVERLAY NETWORK DESIGN PROBLEM

This section introduces the MORNDP and presents the mathematical formulation of the problem and a brief summary of relevant related works.

A. *Problem formulation*

The MORNDP proposes to find a reliable design of an MPLS multi-overlay network over a fixed transport infrastructure, with minimum cost.

In the studied MORNDP, two different networks are present: the data network and the transport network. The data network is a MPLS virtual network built over a multiple technology (SDH/DWDM) transport network. An example is presented in Figure 1, where the black lines represent the physical links (transport links), the continuous colored lines represent the virtual links (data links) and the dotted lines represent physical paths used by the data links. The transport network is considered fixed in the model, i.e. the topology of the transport network is part of the problem input.

In contrast to the transport network, the design of the data network is part of the problem. Since MPLS networks are overlay networks, a second overlay is given by the MPLS network. This situation is shown in Fig. 2. As before, thin colored lines represent the data links. The continuous wide red line represents a MPLS virtual circuit. The path associated to this circuit is shown as a dotted red line.

Solving the MORNDP implies the following:

- Design the data network, which implies defining the links that will finally be included in the network and the capacity values assigned to each one.
- Define the transport paths of each data link included in the data network.
- The designed data network should be tolerant to single failures on the transport network. This means that the data network should be able of routing all the traffic demand, even in case of single failures of the transport links.
- The maintenance cost of the designed data network should be minimized.
- Define the routing on MPLS circuits for each failure scenario. In this work, a simple approach based on primary/alternative paths is used to solve this issue.

B. *Mathematical Model*

The mathematical model of the MORNDP considers the following elements [5]:

- $G_T = (V_T, E_T)$ is the graph that represents the (static) transport network.
- $G_D = (V_D, E_D)$ is the graph of candidate elements to include in the data network. All data edges in E_D are optional. V_D can be divided into two independent subsets: V_F (terminal nodes) and V_S (optional *Steiner* nodes).
- Transport and data graphs are linked by network stations, which always contain a transport node. The function $tns : V_D \to V_T$ returns the transport node of the station for a given data node.
- $M = \lfloor m_{i,j} \rfloor_{1 \le i,j \le \lfloor V_F \rfloor}$ is the demand matrix between terminal nodes.
- A data link cannot be dimensioned to an arbitrary capacity. $\hat{B} = \{\hat{b}_0, \hat{b}_1 ... \hat{b}_{\hat{B}}\}$ represents the capacities that a data link can hold. If a data link is not included in the solution, a null capacity ($\hat{b}_0 = 0$) is assigned to it.
- The vector $B = \{b_{i,j} \in \hat{B}, e_{i,j} \in E_D\}$ represents the dimension that will be assigned to each link of the data layer. The notation $b_{i,j}$ is used to refer to the dimension of edge $e_{i,j}$.
- Let P_D be the set of all possible paths in G_D and $g_D : V_D x V_D \to 2^{P_D}$ a function that returns all possible paths between two data nodes.
- The function $\Phi : E_T \to 2^{P_D}$ returns the routing to be used for each transport link failure scenario. The notation $\Phi(e_T)^{i,j}$ denotes the routing between v_i and v_j in G_D.

$$\Phi(e_T)^{i,j} = \Phi(e_T) \cap g_D(i,j). \qquad (1)$$

- Similarly, P_T and g_T are defined over the graph G_T. The function $\Psi : E_D \to P_T$ returns the transport routing for any data edge. It must meet the restriction in (2).

$$\Psi(e_{i,j}) \in g_T(tns(v_i), tns(v_j)) \forall e_{i,j} \in E_D. \qquad (2)$$

- $r : P_T \to \Re$ returns the length of a path (in kilometers).
- $T : \hat{B} \to \Re$ returns the cost per kilometer for a given data edge dimension.

The goal of the MORNDP is to determine (B,Φ,Ψ), such that the total cost function F (in (4)) is minimized, subject to the restrictions in (5) and (6).

$$F(B,\Phi,\Psi)=\sum_{1<i<j<|V_D|} r\left(\Psi\left(e_{i,j}\right)\right).T\left(b_{i,j}\right) \tag{4}$$

$$\begin{aligned} & e_{k,l}\in\Phi\left(e_T\right)^{i,j}\rightarrow e_t\notin\Psi\left(e_{k,l}\right) \\ & \forall e_T\in E_T, v_i, v_j, v_k, v_l\in V_E / m_{i,j}\neq 0 \end{aligned} \tag{5}$$

$$B\left(e_{p,q}\right)\geq\sum_{1\leq i\leq j\leq|V_E|} m_{i,j}\left|\Phi\left(e_T\right)^{i,j}\cap e_{pq}\right| \forall e_T\in E_T, e_{pq}\in E_D \tag{6}$$

Restriction (5) ensures that for any pair of terminal nodes, the routing path in the failure scenario of a transport edge e_T does not contain e_T. Restriction (6) states that the dimension of data links is not exceeded by transport links in a failure scenario.

C. *Related work*

To the best of our knowledge, there are no references of previous work on the MORDNP outside our workgroup. Risso [5] recently presented a mathematical formulation of the problem and a specific GRASP algorithm to solve it; and Corez [6] applied GRASP and VNS to the MORNDP.

However, there has been an extensive research on simpler MORDNP variants, like the Generalized Steiner Problem (GSP) [7] and other overlay network optimization problems. Grötschel, Monma, and Stoer [8] introduced several versions of the survivable/robust network design problem, while they also studied greedy heuristics and integer programming techniques to solve them. Alevras, Grötschel, Jonas, Paul, and Wessäly [9] addressed an overlay network design problem in mobile telecommunication and they proposed applying integer programming techniques to solve it. Sridhar and Park [10] tackled the survivable capacitated networks design problem using an approach that combined the application of Lagrangian relaxations and heuristics. The problem of designing survivable capacitated networks was also addressed by Pickavet, Poppe, Luystermans, and Demeester [11] using genetic algorithms, while Leitner [12] and Leitner, Raidl, and Hu [13] applied exact and heuristic methods to the 2 connected minimum network generalized problem. Nesmachnow, Cancela, and Alba [14] explored the application of parallel metaheuristics (including genetic algorithms, CHC, simulated annealing and hybrids) to solve the GSP. The optimization of capacitated service overlay networks was addressed by Capone, Elias and Martignon [15] using a mixed integer linear programming approach.

III. EVOLUTIONARY ALGORTIHMS

EAs are non-deterministic methods that emulate the evolutionary process of species in nature, in order to solve optimization, search, and other related problems [4]. In the last thirty years, EAs have been successfully applied for solving optimization problems underlying many real applications of high complexity. This section introduces the GA and the parallel GA proposed in this work for solving the MORNDP.

A. *Genetic algorithm*

GA is a population-based optimization technique inspired in the natural evolution [16][17]. A GA is an iterative technique (each iteration is called a *generation*) that applies stochastic operators on a pool of individuals (the *population*) in order to improve their *fitness*, a measure related to the objective function. Every individual in the population is the encoded version of a tentative solution of the problem. The initial population is generated at random or by using a specific heuristic for the problem. An evaluation function associates a fitness value to every individual, indicating its suitability to the problem. Iteratively, the probabilistic application of *evolutionary operators* like the *recombination* of parts of two individuals or random changes (*mutations*) in their contents are guided by a selection-of-the-best technique to tentative solutions of higher quality. The stopping criterion usually involves a fixed number of generations or execution time, a quality threshold on the best fitness value, or the detection of a stagnation situation. Specific policies are used to select the groups of individuals to recombine (the *selection* method) and to determine which new individuals are inserted in the population in each new generation (the *replace* criterion). The GA returns the best solution ever found in the iterative process, taking into account the fitness function considered.

The generic schema of a GA is shown in Algorithm 1.

```
initialize (Population(0))
generation = 0
while (not stopcriteria)
    evaluate (P(generation))
    parents = selection(P(generation))
    offspring = evolutionary operators(parents)
    newpop = replace(offspring,P(generation))
    generation++
    P(generation) = newpop
end
return best solution ever found
```

Algorithm 1. Schema of a genetic algoritm.

B. *Parallel genetic algorithm*

Parallel implementations became popular in the last decade as an effort to improve the efficiency of GAs. By splitting the population into several processing elements, parallel genetic algorithms (PGAs) allow reaching high quality results in a reasonable execution time, even for hard-to-solve optimization problems [18]. The parallel GA used in this work to solve the MORNDP is categorized within the *distributed subpopulations* model [19]: the original population is divided into several subpopulations, separated geographically from each other. Each subpopulation runs a sequential EA, so individuals are able to interact only with other individuals in the subpopulations. An additional *migration* operator is defined: within a certain number of generations some selected individuals are exchanged between subpopulations, introducing a new source of diversity in the evolutionary search mechanism.

IV. Evolutionary algorithms for the MORNDP

This section introduces the software library used to implement the proposed evolutionary algorithms, and it also present the specific sequential and parallel GAs designed in this work to solve the MORNDP.

A. *The MALLBA library*

MALLBA [20] is a library of algorithms for optimization that can deal with parallelism -on a Local Area Network (LAN) or on a Wide Area Network (WAN)-, in a user-friendly and, at the same time, efficient manner. The EAs described in this section are implemented as generic templates on the library as *software skeletons*, to be instantiated with the features of the problem by the user. They incorporate all the knowledge related to the resolution method, its interactions with the problem, and the parallel considerations. Skeletons are implemented by a set of *required* and *provided* C++ classes that represent an abstraction of the entities participating in the resolution method:

- The provided classes implement internal aspects of the solver in a problem-independent way. The most important provided classes are Solver (the algorithm) and SetUpParams (setup parameters)
- The required classes specify information specifically related to the problem. Each solver includes the required classes Problem and Solution, which encapsulate the problem-dependent entities needed by the resolution method. Depending on the algorithm, other classes may be required.

The MALLBA library is available at University of Málaga website `http://neo.lcc.uma.es/mallba/easy-mallba`. Using MALLBA allowed an efficient and reusable implementation of the GA and the parallel GA applied to the MORNDP in this work.

B. *Genetic Algorithm*

The main details of the specific GAs implemented to solve the MORNDP are described in the following paragraphs.

1) *Solution representation*

Using traditional encodings to represent MORNDP solutions will require to design specific evolutionary operators to maintain the solutions feasibility, thus a problem-dependant encoding was used.

A solution is implicitly encoded by the routings of its terminal nodes. Two logical sub-encodings are used: the *routing encoding* and the *mapping encoding*. Fig. 3 (routing encoding) and Fig 4. (mapping encoding) present a graphical representation of the two proposed sub-encodings to represent MORNDP solutions.

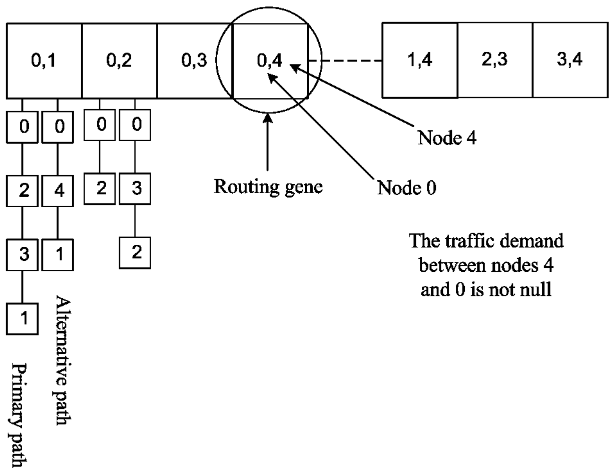

Fig. 3. Routing encoding.

The routing encoding is divided in "routing genes" (see Fig. 3). Each routing gene is associated with a pair of terminal nodes with a non-null demand between them. A routing gene is composed by the primary path (i.e., the default routing) and the alternative path (i.e., the routing applied when any of the transport links associated to the primary path fail). Both paths are represented as lists of data nodes. The data network is implicitly defined by the paths in the routing genes. Each data link present in any path should be also present in the data network. The capacity of each link is calculated taking the minimum bandwidth available that supports the maximum data rate achieved for all failure scenarios.

The mapping encoding defines the transport mapping of each data edge present in the solution (see Fig. 4). Each gene in the mapping encoding is associated to a data link present in the solution. The gene contains a list of transport nodes that represents the transport path associated to the data link.

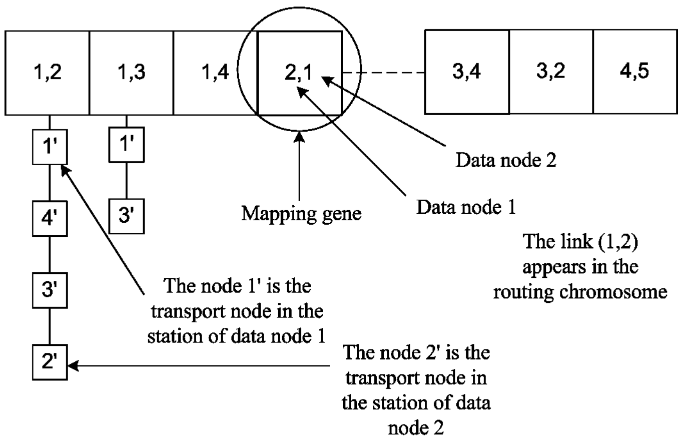
1,2
1,3
1,4
2,1
3,4
3,2
4,5
Data node 2
Mapping gene
Data node 1
The link (1,2) appears in the routing chromosome
The node 1' is the transport node in the station of data node 1
The node 2' is the transport node in the station of data node 2

Fig. 4. Mapping encoding.

The proposed two-vector encoding defines all attributes of a MORNDP solution:

- data links and their capacity are implicitly defined by the routing encoding;
- data routings are defined by the routing encoding;
- transport routings are defined by the mapping encoding.

2) *Initialization*

A specific procedure was proposed to generate feasible initial solutions in the GA. It is a kind of greedy randomized algorithm that can delete (previously incorporated) elements in the solution, if the construction process gets stuck.

The algorithm follows a cyclic procedure that tries to generate a routing gene in each step. The construction of the mapping genes is subordinated to the construction of the routing genes. Each routing gene is built in an iterative process: at each step the algorithm randomly selects a new data link to append to the existing routing gene. The selection is performed using the roulette wheel technique: data links that add high costs to the network are rated with low probability, and data links that add low costs or are already present on the solution are rated with high probability. If a selected data link is not mapped, the mapping gene generation is applied. It is also an iterative process that selects transport links using the roulette wheel technique. The aptitude function that is used to rate transport links is inversely proportional to the kilometers of a transport link.

The routing gene generation can fail, mainly by capacity overflow issues or primary-alternative path collision. If the routing gene construction fails after twenty attempts, a selected routing gene already incorporated to the solution is deleted and re-built in a different way. This process is repeated until the routing gene that is being constructed is completed.

The described greedy constructor is also used to repair unfeasible solutions after applying the evolutionary operators.

3) *Crossover*

Traditional crossover operators do not assure to generate feasible solutions when applied to the MORNDP. To deal with this problem, an ad-hoc crossover operator was designed.

The child is generated step by step, by copying a routing gene and all dependant mapping genes from a randomly selected parent. The crossover verifies that the primary and the alternative paths are independent in the child (they could intersect in the child, as some transport mappings in the child can be different to the original ones in the parent). The capacity restrictions are also checked and corrected if necessary. This process is repeated until the routing gene is successfully copied or no more parents are left. If the routing gene insertion fails, the routing encoding will remain incomplete and the new solution will not be feasible. The greedy randomized solution generator/corrector is then applied to complete the missing parts of the individual.

The behavior of the crossover operator is shown in Fig. 5. The routing gene [2,4] is marked with orange because it could not be inserted from any of the parent individuals. The routing genes [0,4] and [2,4] were created using the generator/corrector greedy algorithm described in the previous subsection.

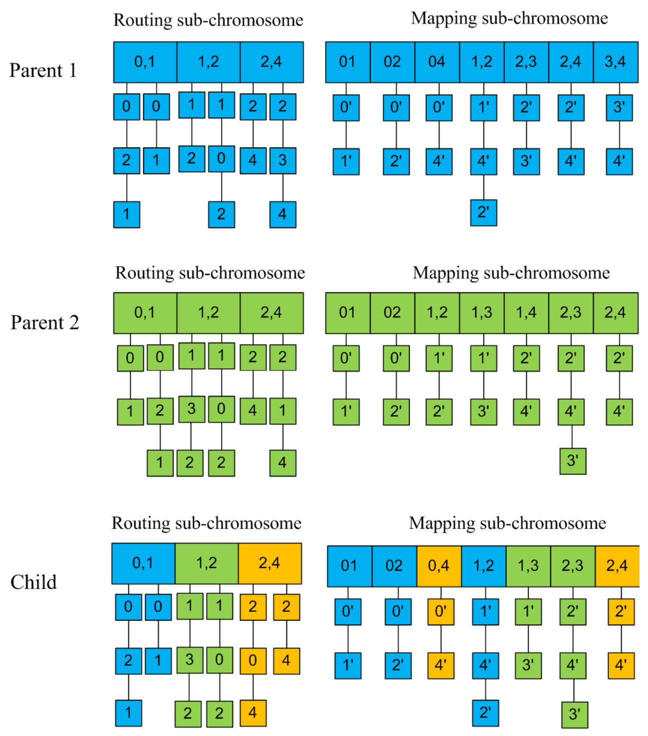

Fig. 5. Crossover example.

4) *Mutation*

Five mutation operators were proposed to introduce diversity and improve the solutions in the GA:

- *Data layer mutation*: it rebuilds a random gene of the routing encoding, mainly modifying the paths in the data layer. This mutation can change the mapping of a data link if this link is used only by one routing gene. The gene rebuilding task is delegated to the greedy randomized solution generator/corrector.
- *Transport layer mutation*: it changes the mapping of data links by randomly selecting a set of mapping genes and modifying a portion of the path in each selected gene.
- *Tabu link mutation*: it randomly selects a “tabu” data link which is eliminated along with its associated genes from the solution. The randomized solution generator/corrector is then used to rebuild the missing parts of the solution, without using the tabu link.
- *Best data layer mutation*: it is a local search operator that uses the data layer mutation as a base for searching in the neighborhood of a solution. The operator modifies the current solution 30 times and the best result is returned.
- *Best transport layer mutation*: it is another local search operator that uses the transport layer mutation for the neighborhood search. Like the previous operator, it performs 20 changes and returns the best result found.

C. *Parallel GA*

The parallel GA follows the distributed subpopulation model, using the same encoding and operators than the sequential GA. A migration operator exchanges individuals between subpopulations, considering them connected in a unidirectional ring. Four subpopulations were used in the PGA.

V. EXPERIMENTAL ANALYSIS

This section presents the experimental evaluation of the proposed GAs for solving the MORNDP.

A. *Development and execution platform*

The GAs were implemented in C++, using the MALLBA library and the standard GNU compiler, version 4.1.2. The parallel GA uses the MPICH (version 1.2.7) implementation of MPI [21] for the distributed execution.

The experimental analysis was performed in the Cluster FING high performance computing infrastructure [22], using Quad core Xeon E5430 servers at 2.66 GHz, with 8 GB RAM and Linux CentOS operating system.

B. *Test instances*

Two different sets of test scenarios were used in the experimental evaluation. A set of simple heterogeneous scenarios was constructed for the parameters calibration of the proposed algorithms. These scenarios were created using a random generator program. In addition, a set of realistic and more complex scenarios, much more appropriate to test the efficiency and quality of the proposed algorithms, was used in the experimental evaluation. A subset of three scenarios presented by Risso [5] was used for these proposes. These scenarios were constructed using the estimated traffic demand data over the MPLS network of ANTEL for 2013. The main characteristics of the used scenarios are presented in Table I.

C. *Parameters calibration*

A calibration analysis was performed in order to determine the best parameter values for the GA operators. The parameter setting analysis was performed using a set of six randomly generated MORNDP instances with reduced size. The studied parameters included: population size (#pop), crossover probability (p_C), data layer mutation probability (p_{dM}), transport layer mutation probability (p_{tM}), best data layer mutation probability (p_{bdM}), best transport layer mutation probability (p_{btM}), number of generations (#gen) and number of subpopulations (#I) in the parallel GA. The candidate values for the parameters were #pop: 30, 50, and 70 individuals; p_C : 0.6, 0.8, 0.9; p_{dM} : 0.01, 0.05; p_{tM} : 0.01, 0.05; p_{bdM} : 0.30, 0.70; p_{dM} : 0.30, 0.70, #gen: 1000, 2000, and #I: 2, 4, 8.

In the parameters setting experiments, the best results were obtained when using the following parameter configuration: #pop: 50 individuals; $p_C = 0.6$; $p_{dM} = 0.01$; $p_{tM} = 0.01$; $p_{bdM} = 0.70$; $p_{btM} = 0.70$, #gen = 2000. #I = 4.

D. *Evaluation on real-world scenarios*

The final evaluation of the proposed algorithms was performed using the set of three real-world scenarios from ANTEL, whose details are presented in Table I.

Table II presents the best and average cost values, and the standard deviation results obtained in 15 independent executions of the sequential and parallel GA. The time required to execute the algorithm, (*avg. time*), in hours, and the time to achieve the best solution (t_{BEST}) are also reported.

TABLE I. CHARACTERISTICS OF THE REAL-WORLD ANTEL SCENARIOS

Scenario	#data nodes	# traffic requisites	# traffic links	# candidate data links	traffic demand (Gb)
02	56	58	164	286	39465
06	56	58	164	286	26543
10	56	58	164	286	28182

TABLE II. EXPERIMENTAL RESULTS FOR REAL-WORLD SCENARIOS

	Scenario 02				
	best cost	*avg. cost*	*st. dev.*	*avg. time (h)*	t_{BEST} *(h)*
GA	562549	625795	43688	109.19	97.04
PGA	566856	612832	34285	46.04	42.45
	Scenario 06				
	best cost	*avg. cost*	*st. dev.*	*avg. time (h)*	t_{BEST} *(h)*
GA	538079	576988	23569	115.00	98.91
PGA	486894	550120	42203	33.95	29.30
	Scenario 10				
	best cost	*avg. cost*	*st. dev.*	*avg. time (h)*	t_{BEST}. *(h)*
GA	544673	575028	20830	125.20	103.13
PGA	515785	557720	26207	39.83	37.32

Table II shows that the sequential GA took a long time to execute the 2000 generations, mainly due to the inherent complexity of the large real-world scenarios tackled (but it is actually a reasonable time for hard network design problems). However, the efficiency results also demonstrate that the distributed subpopulation model allows the parallel GA to significantly reduce the execution times. Additionally, the parallel model outperformed the sequential GA regarding the best cost for two out of the three studied scenarios, and regarding the average cost in all of them best cost.

Table III summarizes the performance metrics for the parallel model. It reports the values for the *speedup* and *computational efficiency* metrics. The speedup (S_P) is defined as the ratio between the time required by the sequential GA and the parallel GA when using P computational resources. The computational efficiency (e_P) is a normalized value of the speedup, considering the number of computational resources ($e_P = S_P / P$). Table III reports the speedup and the computational efficiency of the parallel GA when using four computing resources.

The results in Table III indicate that the parallel GA showed a sub-linear speedup behavior, but very good efficiency values (i.e. near 0.8) when using four computing resources were achieved for the two more complex scenarios. These efficiency values suggest that using a parallel GA model is an efficient and effective approach to solve the MORNDP.

TABLE III. PERFORMANCE METRICS FOR THE PARALLEL GA

Scenario	speedup (S_4)	Efficiency (e_4)
02	2.37	0.59
06	3.38	0.85
10	3.14	0.79

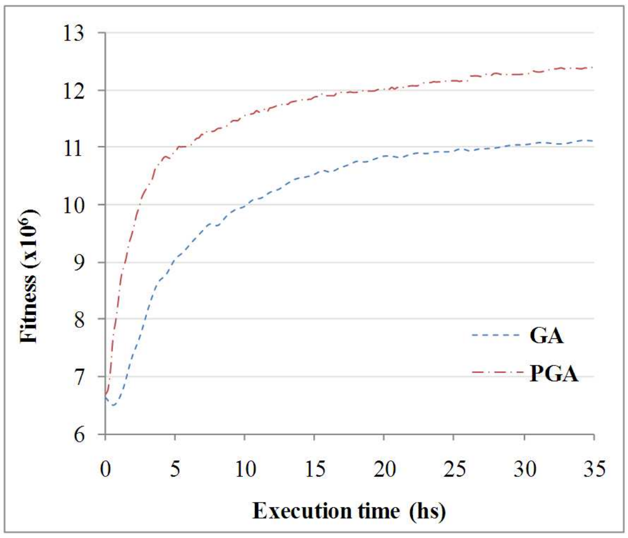

Fig. 6. Fitness evolution example (Scenario 10).

Figure 6 presents an example of the graphical analysis for an important feature in EAs and other metaheuristics: the evolution of the average fitness values along the generations. The comparative analysis shows that the parallel GA is able to compute more accurate solutions than the sequential GA in significantly lower execution times, as it is presented in the sample evolution for the Scenario 10 problem instance. This behavior was consistently detected in every execution of GA and PGA for the three studied scenarios.

Table IV presents a comparison of the best results obtained using GA and PGA with the results achieved with the GRASP method by Risso [5]. The comparative analysis shows that the best results obtained with the evolutionary techniques are still far away from the best results previously presented for the problem. However, the results by Risso are only taken into account as a relative reference baseline to compare our results, due to two important considerations that have to be remarked: i) a less restrictive routing algorithm is used by Risso, allowing more than two routes between each pair of nodes, instead of the much more simple routing approach based on primary/alternative paths used in the GAs proposed in this work; and ii) the proposed GAs were designed to follow a simple search approach, without using much problem-specific information in the evolutionary operators, while the GRASP method by Risso heavily relies on a sophisticated construction scheme for the candidate solutions.

Another important consideration is that all the best results obtained with the parallel GA represent a significant improvement in the network design cost with respect to the actual design by ANTEL.

Taking into account the previously presented arguments, the results obtained with the parallel GA can be considered as a promising first step for solving the MORNDP with a non-specific metaheuristic for the problem.

TABLE IV. COMPARISON WITH PREVIOUS RESULTS

Scenario	best cost GA/PGA	best cost [5]	gap
02	562549	532896	5.5 %
06	486894	451360	7.8 %
10	515785	451360	14.2 %

VI. CONCLUSIONS AND FUTURE WORK

This work presented the first advances on applying a sequential and a parallel evolutionary algorithm to solve the multi-overlay robust network design problem. The MORNDP is a complex NP-hard optimization problem that models the structural design of important modern telecommunication infrastructures, and few previous works have addressed its resolution using generic metaheuristics.

The GA and the parallel GA proposed in this work were designed using simple and intuitive evolutionary operators, without including much problem-specific information or sophisticated optimization methods for the search.

The algorithms were evaluated over a set of three real-world network design scenarios built with data provided by the Uruguayan national telecommunications company (ANTEL). The experimental analysis allowed us to conclude that promising results were obtained with the proposed GAs, especially with the parallel GA, despite the simple approach followed to design the evolutionary operators.

Regarding the computational efficiency, the parallel version of the proposed GA significantly improved over the execution times of the sequential GA. The speedup and computational efficiency values were almost-linear for two out of the three problem instances solved. In addition, the fitness evolution analysis showed that the parallel GA was able to compute more accurate solutions than the sequential GA in significantly lower execution times. These two previous results suggest that the use of parallel implementations of GA is a promising idea to efficiently solve the MORNDP.

Two main lines remain to be tackled as future work, both already in progress: improving the efficiency of the proposed methods, and improve the search mechanism in order to obtain better network designs. Regarding the first line for future work, the computational efficiency of the proposed GAs should be increased by applying a more effective parallel strategy that allow increasing the number of subpopulations in the parallel GA without reducing the quality of the solutions. On the other hand, the comparison with the previous results obtained for the problem using other techniques showed that there is still room to improve the evolutionary search in order to obtain better results with GAs. Regarding this issue, additional simple evolutionary operators have to be analyzed, and other variants of the problem (i.e., those that include the routing method described in [5]) should be studied. We are currently working in these commented lines right now.

References

[1] P. Oppenheimer. Top-Down Network Design. Cisco Press, 2004

[2] V. Alwayn, Optical Network Design and Implementation, John Kane, Ed. Indianapolis, USA: Cisco Press, 2004.

[3] L. Monma, S. Munson, and R. Pulleyblank, Minimum-weight two-connected spanning networks. Mathematical Programming vol. 46, num 1-3, pp. 153-172, 1990.

[4] T. Back, D. Fogel, and Z. Michalewicz (Eds.). Handbook of Evolutionary Computation. IOP Publ. Ltd., Bristol, UK, 1997.

[5] C. Risso, Optimización de Costos en Redes Multicapas Robustas, Master Thesis, Universidad de la República, Uruguay, 2010. Available Online at http://premat.fing.edu.uy/IngenieriaMatematica/archivos/tesis_claudio_risso.pdf, accessed March 2011 (text in Spanish).

[6] A. Corez, Overlay Network Planning by applying a Variable Neighbourhood Search Approach. Master Thesis, Universidad de la República, Uruguay, 2010. Available online at http://premat.fing.edu.uy/IngenieriaMatematica/archivos/tesis_andres_corez.pdf, accessed March 2011.

[7] V. Kann and P. Crescenzi. (2003) A compendium of NP optimization problems. Available online at http://www.nada.kth.se/~viggo/problemlist/compendium.html, accessed March 2011.

[8] M. Grötschel, C. Monma, and M. Stoer, Design of Survivable Networks in Network Models, Handbooks in Operations Research and Management Science 7. Amsterdam: North-Holland, 1995.

[9] D. Alevras, M. Grötschel, P. Jonas, U. Paul, and R. Wessäly. Survivable Mobile Phone Network Architectures: Models and Solution Methods. IEEE Communications Magazine, 36:3 (1998) 88-93.

[10] V. Sridha and J. Park, An Approach to Solving the survivable capacitated network design problem International Journal of Business Data Communications and Networking (2): 1-16, 2005.

[11] M. Pickavet, F. Poppe, J. Luystrermans, and P. Demeester, A generic algorithm for solving the capacitated survivable network design problem, en Proc. of Fifth International Conference on Telecommunication Systems, Brussels, Belgium, 1997, pp. 71-76.

[12] M. Leitner, Solving two generalized network design problems with exact and heuristic methods, Master Thesis. University of Technology, Institute of Computer Graphics and Algorithms, Vienna, 2006. Available online at http://publik.tuwien.ac.at/files/pub-inf_4483.pdf, accessed March 2011.

[13] M. Leitner, G. Raidl, and B. Hu, Variable Neighborhood Search for the Generalized Minimum Edge Biconnected Network Problem, Networks, Vol. 55, No. 3, pp. 256-275, 2010.

[14] S. Nesmachnow, H. Cancela, and E. Alba, Evolutionary algorithms applied to reliable communication network design, Engineering Optimization, Vol. 39, No. 7, pp. 831-855, 2007.

[15] A. Capone, J. Elias, and F. Martignon, Routing and resource optimization in service overlay networks. Computer Networks 53(1):180–190, 2009.

[16] D. Goldberg, Genetic algorithms in search, optimization, and machine learning. New York: Addison-Wesley Longman Publishing Co, 1989.

[17] M. Mitchell, An introduction to genetics algorithms. Cambridge: MIT Press, 1996.

[18] E. Alba (Ed.). Parallel Metaheuristics: A New Class of Algorithms, Wiley, 2005

[19] E. Alba and M. Tomassini, Parallelism and Evolutionary Algorithms. IEEE Transactions on Evolutionary Computation, IEEE Press, 6(5):443-462, 2002.

[20] E. Alba, F. Almeida, M. Blesa, C. Cotta, M. Díaz, I. Dorta, J.Gabarró, C. León, G. Luque, J. Petit, C. Rodríguez, A. Rojas, and F. Xhafa, Efficient Parallel LAN/WAN Algorithms for Optimization. The MALLBA Project, Parallel Computing 32(5-6):415-440, 2006.

[21] W. Gropp, E. Lusk, and A. Skjellum. Using MPI: Portable Parallel Programming with the Message-Passing Interface. MIT Press, Cambridge, MA, USA, 1994.

[22] Cluster FING. High Performance Scientific Computing at Universidad de la República. Available online at http://www.fing.edu.uy/cluster, accessed March 2011.