2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

QoS Constrained Internet Routing with Evolutionary Algorithms

Miguel Rocha, Pedro Sousa, Miguel Rio, and Paulo Cortez

Abstract— **OSPFOSPF** *is the most common intra-domain routing protocol in* **Wide Area Networks.** *Thus, optimizing* **OSPF** *weights will produce tools for traffic engineering with* **Quality of Service** *constraints, without changing the network management model.* **Evolutionary Algorithms (EAs)** *provide a valuable tool to face this NP-hard problem, allowing flexible cost functions with several metrics of the network behavior. A novel framework is proposed that enriches current models for network congestion with delay constraints, setting the basis for* **EAs** *that allocate* **OSPF** *weights, guided by a bi-objective cost function. The results show that* **EAs** *make an efficient method, outperforming common heuristics and achieving effective network behavior under unfavorable scenarios.*

I. Introduction

Resource provisioning is a crucial task for any *Internet Service Provider (ISP)* network administrator, which is becoming more challenging due to the increases in demand and the onset of new types of applications. In this context, *ISPs* have *Service Level Agreements (SLAs)* [1] with their clients and/or with peered *ISPs* that have to be strictly obeyed to avoid strong financial penalties.

The convergence of telephony services with the Internet, in the form of *Voice over IP (VoIP)* services, raised extra challenges to network resource management tasks. Since the existing best-effort *TCP/IP* model does not have a connection admission control infra-structure, flows arriving to the network may disrupt existing traffic. Moreover, even if network capacity is sufficient, delay requirements that have to be enforced present extra constraints to be dealt by the administrator.

When modeling this class of problems, it is usual to assume that the administrator has access to a matrix representing traffic demands between each pair of nodes in the network. Optionally, delay constraints of part of the flows may also be available. Given this data, it is important to develop precise techniques to allocate network resources resorting to expedite administrative procedures.

To accomplish this aim, distinct *Quality of Service (QoS)* architectures and specific mechanisms were proposed by the research community, in order to provide distinct service levels to networked applications [2]. However, the provision of *QoS* differentiation capabilities in computer networks requires many components working together. One of such components is related to the ability of defining enhanced *QoS* aware mechanisms, which control the data path followed by packets that traverse a given *Wide Area Network (WAN)*. In a *TCP/IP WAN*, consisting of a single administrative domain, there are alternative strategies: Intra-domain routing protocols or *Multi-Protocol Label Switching (MPLS)* [3].

The most common routing protocol today is *Open Shortest Path First (OSPF)* [4][5]. Here, the administrator assigns weights to each link in the network, which are then used to compute the best path from each source to each destination using the well known Dijkstra algorithm [6]. The results of this method are then used to compute the routing tables in each node.

Since the weight setting process is the only way administrators can affect the network behavior, this choice is of crucial importance. Nevertheless, in practice, simple rules of thumb are typically used in this task, like setting the weights inversely proportional to the link capacity. This approach often leads to sub-optimal network resource utilization.

An alternative way to implement traffic engineering is to use *MPLS*. This is a more flexible approach since one can decide and configure the path of each individual flow. Hence, at least in theory, it is possible to use this technique to optimize network resource allocation. However, the use of *MPLS* presents significant drawbacks when used in the context of packet switching: firstly, it adds significant complexity to the IP model when compared with the simplicity of *OSPF*, since per-flow state has to be stored in every router of the path; secondly, it is not widely tested and deployed; finally, it represents a management overhead that incurs on extra costs for the organization.

An ideal alternative is to improve the process of *OSPF weight setting* to implement traffic engineering. This was the approach taken by Fortz et al [7] where this task was viewed as an optimization problem, by defining a cost function that measures the network congestion. The same authors proved that this task is a NP-hard problem and proposed some local search heuristics that compared well with the *MPLS* model. An alternative approach to this problem was the use of meta-heuristics such as *Evolutionary Algorithms (EAs)* to improve these results [8]. However, this approach did not accommodate delay based constraints that are crucial to implement QoS aware networking services in the Internet resorting to specific *QoS* architectures (e.g. as the *Differentiated Services Architecture* [9]).

In this work, *EAs* are employed to calculate link-state routing weights, that optimize traffic congestion, while si-

Miguel Rocha and Pedro Sousa are with the Department of Informatics/CCTC, University of Minho, Campus Gualtar, 4710-059 Braga, Portugal (phone: + 351-253604435; fax: + 351-253604471; email: {mrocha,pns}@di.uminho.pt).

Miguel Rio is with the Department of Electronic and Electrical Engineering, University College London, Torrington Place, WC1E 7JE, London, UK (email: m.rio@ee.ucl.ac.uk).

Paulo Cortez is with Department of Information Systems, University of Minho, 4800-058 Guimarães, Portugal (email: pcortez@dsi.uminho.pt).

multaneously complying to specific delay requirements[1]. To reach this goal, a novel analytical model of the problem was defined that accommodates both congestion and delay constraints. This model is used to define a proper cost function and therefore to develop fitness functions for the *EAs*, which are then used to calculate the optimal *OSPF* weights for each network link.

Given their numerous and successful applications in real-world constrained problems, both in numerical and combinatorial optimization, *EAs* make the ideal tool to address this problem. They are typically capable of obtaining near-optimal results within an acceptable computational time, which may be critical in a real network scenario.

The paper is organized as follows: firstly, the problem is defined under the model developed; next, the *EAs* designed to tackle this problem are described; the following section presents the experiments and corresponding results; finally, conclusions are drawn and the future work is revealed.

II. PROBLEM DESCRIPTION

The general routing problem [10], that underpins our work, represents routers and transmission links by a set of nodes (N) and a set of arcs (A) in a directed graph $G = (N, A)$. In this model, c_a represents the capacity for each link $a \in A$. Additionally, a demand matrix D is available, where each element d_{st} represents the traffic demand between each pair of nodes s and t from N.

Let us assume that, for each arc a, the variable $f_a^{(st)}$ represents how much of the traffic demand between s and t travels over arc a. The total load on each arc a (l_a) can be defined in the following way:

$$l_a = \sum_{(s,t)\in N\times N} f_a^{st} \tag{1}$$

while the link utilization rate u_a is given by $\frac{l_a}{c_a}$. It is then possible to define a congestion measure for each link (Φ_a), using a cost function p that has small penalties for values near 0; however, as the values approach the unity becomes more expensive and exponentially penalizes values above 1 (Figure 1):

$$p(x) = \begin{cases} x, & x \in [0, 1/3) \\ 3x - 2/3, & x \in [1/3, 2/3) \\ 10x - 16/3, & x \in [2/3, 9/10) \\ 70x - 178/3, & x \in [9/10, 1) \\ 500x - 1468/3, & x \in [1, 11/10) \\ 5000x - 16318/3, & x > 11/10 \end{cases} \tag{2}$$

Given this function, the congestion measure for a given arc can be defined by $\Phi_a = p(u_a)$ [8]. Under this framework,

[1]Note that the end-to-end delay of a given network path includes the propagation delays and the queueing delays. However, in the context of this work, we have relaxed the terminology and the term delay is from now on used to express the propagation delay of a given network path. The queueing delay part of the end-to-end delay is expected to be controlled/differentiated by the use of specific mechanisms operating at each node, such as queueing and scheduling mechanisms.

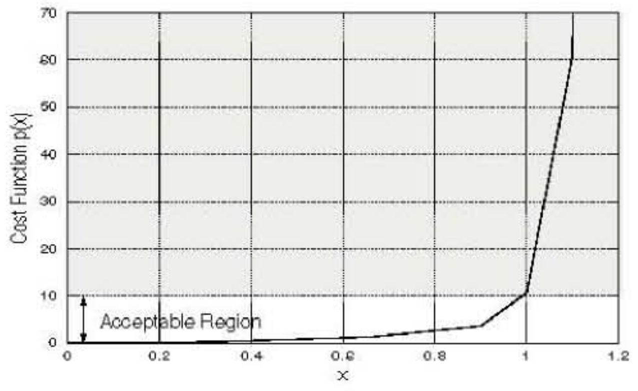

Fig. 1. Graphical representation of the penalty function p.

it is possible to define a linear programming instance [7]. where the purpose is to set the value of the variables f_a^{st} that minimize the following objective function:

$$\Phi = \sum_{a\in A} \Phi_a \tag{3}$$

subject to:

$$\sum_{u:(u,v)\in A} f_{(u,v)}^{(s,t)} - \sum_{u:(v,u)\in A} f_{(u,v)}^{(s,t)} = \begin{cases} -d_{st}, & \text{if } v = s \\ d_{st}, & \text{if } v = t \\ 0, & \text{otherwise,} \end{cases} \quad v, s, t \in N \tag{4}$$

$$l_a = \sum_{(s,t)\in N\times N} f_a^{st}, \quad a \in A \tag{5}$$

$$\phi_a \geq l_a, \quad a \in A \tag{6}$$

$$\phi_a \geq 3l_a - 2/3c_a, \quad a \in A \tag{7}$$

$$\phi_a \geq 10l_a - 16/3c_a, \quad a \in A \tag{8}$$

$$\phi_a \geq 70l_a - 178/3c_a, \quad a \in A \tag{9}$$

$$\phi_a \geq 500l_a - 1468/3c_a, \quad a \in A \tag{10}$$

$$\phi_a \geq 5000l_a - 16318/3c_a, \quad a \in A \tag{11}$$

$$f_a^{(s,t)} \geq 0, \quad a \in A, s, t \in N \tag{12}$$

In the following the optimal solution to this problem is denoted by Φ_{Opt}.

In *OSPF*, all arcs are associated with an integer weight. Every node uses these weights in the Dijkstra algorithm [6] to calculate the shortest paths to all other nodes in the network, with themselves as the root. Each of these paths has a length equal to the sum of the weights of each arcs in this route. All the traffic from a given source to a destination travels along the shortest path. If there are two or more paths with the same length, between a given source and a destination, traffic is evenly divided among the arcs in these paths (load balancing) [11].

Let us assume a given solution, i.e. a weight assignment (w), and the corresponding utilization rates on each arc. In this case, the total routing cost is expressed by

$$\Phi(w) = \sum_{a \in A} \Phi_a \quad (13)$$

for the loads calculated based on the given OSPF weights. In this way, the *OSPF weight setting problem* (as defined in [7], [8]) is equivalent to finding the optimal weight values for each link (w_{opt}), in order to minimize the function $\Phi(w)$.

The congestion measure can be normalized over distinct topology scenarios, by using a scaling factor defined as [7]:

$$\Phi_{UNCAP} = \sum_{(s,t) \in N \times N} d_{st} h_{st} \quad (14)$$

where h_{st} is the minimum hop count between nodes s and t.

Finally, the scaled congestion measure cost is defined as:

$$\Phi^*(w) = \frac{\Phi(w)}{\Phi_{UNCAP}} \quad (15)$$

and the following relationships hold:

$$1 \leq \Phi^*_{OPT} \leq \Phi^*_{OptOSPF} \leq 5000 \quad (16)$$

where $\Phi^*_{OptOSPF}$ is the normalized congestion imposed by the optimal solution to the *OSPF weight setting problem*.

It is important to note that when Φ^* equals 1, all loads are below $1/3$ of the link capacity; on the other hand, when all arcs are exactly full the value of Φ^* is 10 2/3. This value will be considered a threshold that bounds the acceptable working region of the network.

In order to include *QoS* constraints in this model, it is necessary to include delay constraints in the optimization framework. These requirements were modeled as a matrix DR, that for each pair of nodes $(s,t) \in N \times N$ (where $d_{st} > 0$) gives the delay target for traffic between origin s and destination t (denoted by DR_{st}).

In a way similar to the congestion model presented before, a cost function was developed to evaluate the delay compliance for each scenario (a given solution defined by the set of weights in the *OSPF*). This function takes into account the average delay of the traffic between the two nodes (Del_{st}), a value calculated by considering all paths between s and t with minimum cost and averaging the delays in each (the delay in each path is the sum of the delays in its arcs).

The delay compliance ratio for a given pair $(s,t) \in N \times N$ is, therefore, defined as

$$dc_{st} = \frac{Del_{st}}{DR_{st}} \quad (17)$$

As before, a penalty for delay compliance can be calculated using function p. So, the γ function is defined according to the following equation:

$$\gamma_{st} = p(dc_{st}) \quad (18)$$

This, in turn, allows the definition of a delay minimization cost function, given a set of *OSPF* weights (w):

$$\gamma(w) = \sum_{(s,t) \in N \times N} \gamma_{st} \quad (19)$$

This function can be normalized dividing the values by the sum of all minimum end-to-end delays (for each pair of nodes the minimum end-to-end delay ($minDel_{st}$) is calculated as the delay of the path with minimum possible overall delay):

$$\gamma^*(w) = \frac{\gamma(w)}{\sum_{(s,t) \in N \times N} minDel_{st}} \quad (20)$$

It is now possible to define the optimization problem addressed in this work, that is clearly multiobjective. Indeed, given a network represented by a graph G of nodes and arcs A, a demand matrix D and a delay requirements matrix DR, the aim is to find the set of *OSPF* weights that simultaneously minimizes the functions $\Phi^*(w)$ and $\gamma^*(w)$.

III. EVOLUTIONARY ALGORITHMS FOR OSPF WEIGHT SETTING

In this work, *Evolutionary Algorithms (EAs)* are proposed to address the above formulated problems, either by considering the multiobjective formulation, or by taking the two distinct aims described in the previous section separately.

In the proposed *EA*, each individual encodes a solution as a vector of integer values, where each value (gene) corresponds to the weight of an arc in the network, whose values range from 1 to w_{max}. Therefore, the size of the individual equals the number of arcs in the graph (links in the network). The individuals in the initial population are randomly generated, with the arc weights taken from a uniform distribution in the allowed interval.

In order to create new solutions, several reproduction operators were used, more specifically two mutation and two crossover operators:

- *Random Mutation*, replaces a given gene by a new randomly generated value, within the allowed range $[1, w_{max}]$;
- *Incremental/decremental Mutation*, replaces a given gene by the next or by the previous value (with equal probabilities) and constrained to respect the range of allowed values;
- *Uniform crossover* and *Two-point crossover*, two standard crossover operators, applied in the traditional way [12].

In each generation every operator is used to create new solutions with equal probabilities (all operators are used in every run). The selection procedure is done by converting the fitness value into a linear ranking in the population, and then applying a roulette wheel scheme. In each generation, 50% of the individuals are kept from the previous generation, and 50% are bred by the application of the genetic operators.

The evaluation process, for each individual in the population, measures the quality of the *OSPF* weights in the optimization aims defined in the previous section. When a single objective is considered the fitness of an individual (encoding weight set w) is calculated using functions $\Phi^*(w)$ for congestion and $\gamma^*(w)$ for delays.

For multiobjective optimization a quite simple scheme was devised. The fitness ($f(w)$) of the individual is, in this case, derived by the expression:

$$f(w) = \alpha\Phi^*(w) + (1-\alpha)\gamma^*(w) \tag{21}$$

This scheme, although simple, can be effective since both cost functions are normalized in the same range and use a similar penalty function.

IV. Experiments and Results

In order to evaluate the effectiveness of the proposed *EAs*, a number of experiments was conducted. For this purpose, a set of 12 networks was generated by using the Brite topology generator [13], varying the number of nodes ($N = 30, 50, 80, 100$) and the average degree of each node ($m = 2, 3, 4$). This resulted in networks ranging from 57 to 390 links (graph edges). The link bandwidth (capacity) was generated by an uniform distribution between 1 and 10 Gbits/s. The network was generated using the Barabasi-Albert model, using a heavy-tail distribution and an incremental grow type (parameters HS and LS were set to 1000 and 100, respectively).

Next, the demand and delay constraints matrices (D and DR) were generated. For each of the twelve instances a set of three distinct D matrices was generated, varying a parameter (D_p) which determined the expected mean of the congestion in each link (u_a) (values for D_p in the experiments were 0.1, 0.2 and 0.3). For the generation of the DR matrix, the strategy was to calculate the average of the minimum possible delays, over all pairs of nodes. A parameter (DR_p) was considered, this time representing the multiplier applied to the previous value to get the matrix DR (values for DR_p in the experiments were 3, 4 and 5). Overall, a set of $12 \times 3 \times 3 = 108$ instances of the optimization problem was considered.

A number of heuristic methods was considered [7], for a comparison with the results obtained by the *EA*:

- **Unit** - sets all arc weights to 1 (one);
- **InvCap** - sets arc weights to a value inversely proportional to capacity of the link;
- **L2** - sets arc weights to a value proportional to the physical Euclidean distance (L2 norm) of the link;
- **Random** - a number of randomly generated solutions are analyzed and the best is selected. The number of solutions considered is always equal to the number of solutions evaluated by the *EA* in each problem.

The proposed *EA* and heuristics were implemented by the authors using the *Java* programming language. The *EA* was run for a number of generations ranging from 1000 to 6000, a value that was incremented proportionally to the number of variables optimized by the *EA*. The running times varied from a few minutes in the small networks to a few hours in the larger ones. So, in order to perform all the tests, a computing cluster with 46 dual Xeon nodes was used.

The population size was kept in 100 and w_{max} was set to 20. In multiobjective optimization all the results shown in this paper consider α to be 0.5, thus considering each aim to be of equal importance. Since the *EA* and the *Random* heuristic are stochastic methods, R runs were executed in each case (R was set to 10 in the experiments).

For a better understanding, the results are grouped into three sets according to the cost function used. The first two consider single objective cost functions, for the optimization of congestion and delays respectively. These are used mainly as baselines for the comparison with the results obtained with the last group, that presents the results using the multiobjective cost function. In all figures the data was plotted in a logarithmic scale, given the exponential nature of the penalty function adopted.

A. Congestion

Since the number of performed experiments is quite high, it was decided to present all the results for just one of the networks (out of the 12), to explain the experimental methodology, and then to show some aggregate results that can be used to draw conclusions. This strategy was also used in the presentation of the results of the following sections.

Therefore, in Table I we show the results for the optimization of the congestion, for one of the networks (with 100 nodes and 197 links). Both the results obtained by the proposed *EA* and by the set of heuristic methods described before are shown. In this table, the first column represents the demand generation parameter D_p (higher values for this parameter indicate higher mean demands, thus harder optimization problems). The remaining columns indicate the congestion measure ($\Phi^*(w)$) for the best solution (w) obtained by each of the methods considered in this study. In the case of the *EAs* and *Random* heuristic the results represent the mean value of the results obtained in the set of runs.

TABLE I
RESULTS FOR THE OPTIMIZATION OF CONGESTION (FUNCTION Φ^*) IN ONE EXAMPLE NETWORK WITH 100 NODES AND 197 LINKS.

D_p	**Unit**	**L2**	**InvCap**	**Random**	**EA**
0.1	3.62	190.67	1.68	12.05	1.02
0.2	136.75	658.66	135.07	280.27	1.25
0.3	264.02	874.89	488.53	551.65	1.49

Table II shows the results for all available networks, averaged by the demands levels (value of D_p), including in the last line the overall mean value for all problem instances. It is clear that the results for all the methods get worse with the increase of D_p, as would be expected.

The comparison between the methods shows an impressive superiority of the *EA* when compared to the heuristic methods. In fact, the *EA* achieves solutions which manage a very reasonable behavior in all scenarios (worse case is 1.49), while the other heuristics manage very poorly. Even $InvCap$, an heuristic quite used in practice, gets poor results when D_p is 0.2 or 0.3 (Figure 2), which means that the optimization with the *EAs* assures good network behavior in scenarios where demands are at least 200% larger than the ones where $InvCap$ would assure similar levels of congestion.

TABLE II
RESULTS FOR THE OPTIMIZATION OF CONGESTION (Φ^*) - AVERAGED RESULTS BY DEMAND LEVELS

D_p	Unit	L2	InvCap	Random	EA
0.1	8.03	215.94	1.50	75.75	1.02
0.2	99.96	771.87	57.70	498.74	1.18
0.3	227.30	1288.56	326.33	892.87	1.73
Overall	111.76	758.79	128.51	489.12	1.31

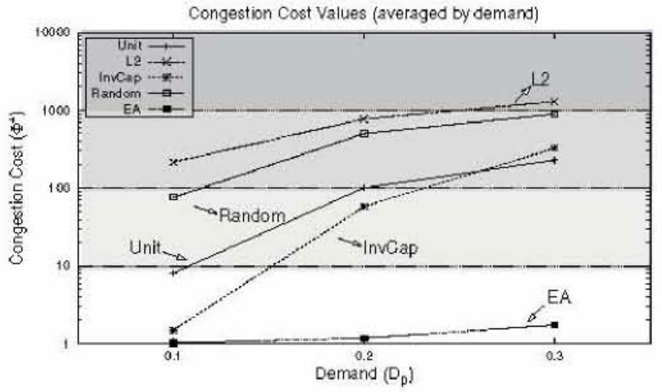

Fig. 2. Graphical representation of the results obtained by the different methods in congestion optimization (averaged by D_p).

Table III shows the results for congestion, averaged by the number of nodes in the network. Figure 3, on the other hand, represents the same data, but aggregated by the number of arcs (links). It is clear from both results that the results obtained by the *EAs* are quite scalable, since the quality levels are not affected by the number of nodes or edges in the network graph.

TABLE III
RESULTS FOR THE OPTIMIZATION OF CONGESTION(Φ^*) - AVERAGED RESULTS BY THE NUMBER OF NODES.

Nodes	Unit	L2	InvCap	Random	EA
30	98.90	598.35	95.71	263.69	1.29
50	121.08	815.08	104.92	418.59	1.28
80	111.62	730.71	157.50	594.37	1.31
100	115.45	891.00	155.90	679.82	1.36

The results obtained in this section show that the *EA* makes an effective method for the optimization of *OSPF weights*, in order to minimize the congestion of the network.

These results confirm the findings of Ericsson et al [8], although a precise comparison of the approaches is impossible since the original data is not available

B. Delays

Regarding the optimization of delays (cost function γ^*), a similar methodology was adopted. Indeed, in Table IV the results for the same example network are shown. The methods used in the optimization are the same as in the

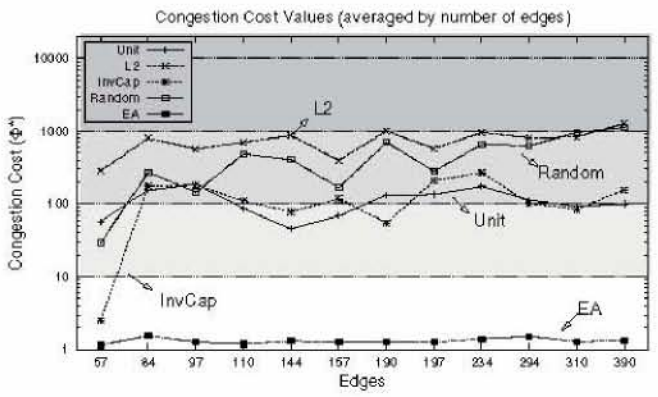

Fig. 3. Graphical representation of the results obtained by the different methods in congestion optimization (averaged by the number of network links).

previous section. In this case, the first column represents the parameter used for the generation of delay requirements (DR_p).

TABLE IV
RESULTS FOR THE OPTIMIZATION OF DELAYS (FUNCTION γ^*) IN ONE EXAMPLE NETWORK WITH 100 NODES AND 197 LINKS.

DR_p	Unit	L2	InvCap	Random	EA
3	13.50	1.38	201.62	4.36	1.38
4	2.00	1.13	18.33	1.82	1.13
5	1.47	1.04	3.62	1.54	1.04

On the other hand, Table V and Figure 4 represent the results obtained for the delay optimization averaged by the parameter used in the generation of delays requirements (DR_p). In this case, the results of all methods improve when the value is higher, since this means the optimization problem is easier (higher delay requirements are easier to comply).

The relative performance of each method shows a good behavior of the *EA*, as before, but now there is a simpler heuristic method - the *L2* - that achieves very similar results. This is not a surprise, since in the proposed model only propagation delays were considered and these are proportional to the length of each link. The *L2* heuristic considers the *OSPF* weights to be proportional to the arc length, which means they are also directly proportional to the delays. So, it is clear that the *L2* heuristic exhibits a near-optimal behavior in this problem.

It is important to notice that in the context of network management, the delay minimization, unlike the congestion, is not typically an optimization aim by itself. So, the results in this section will be used mainly as a basis for comparison with the results of multiobjective optimization.

As before, the results for the delay minimization are also shown aggregated by the number of nodes (Table VI) and by the number of links (Figure VI). The scalability of both *L2* and the *EAs* prevails in these results.

TABLE V

RESULTS FOR THE OPTIMIZATION OF DELAYS (γ^*) - AVERAGED RESULTS BY THE DELAY REQUIREMENTS PARAMETER (DR_p)

DR_p	Unit	L2	InvCap	Random	EA
3	152.37	2.94	577.94	156.62	2.85
4	28.78	1.25	158.85	24.35	1.25
5	6.59	1.10	44.13	4.29	1.10
Overall	62.58	1.76	260.30	61.75	1.73

TABLE VI

RESULTS FOR THE OPTIMIZATION OF DELAYS (γ^*) - AVERAGED RESULTS VALUES BY THE NUMBER OF NODES

Nodes	Unit	L2	InvCap	Random	EA
30	60.75	1.84	296.22	16.84	1.82
50	115.32	2.04	417.88	74.75	1.96
80	57.16	1.69	187.45	97.53	1.66
100	17.08	1.48	139.67	57.88	1.50

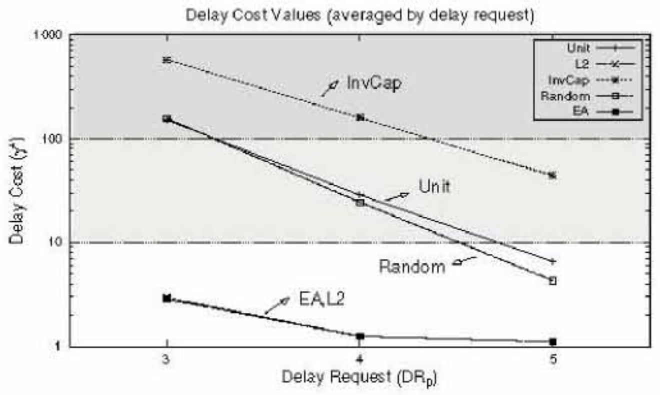

Fig. 4. Graphical representation of the results obtained by the different methods in delay optimization (averaged by DR_p).

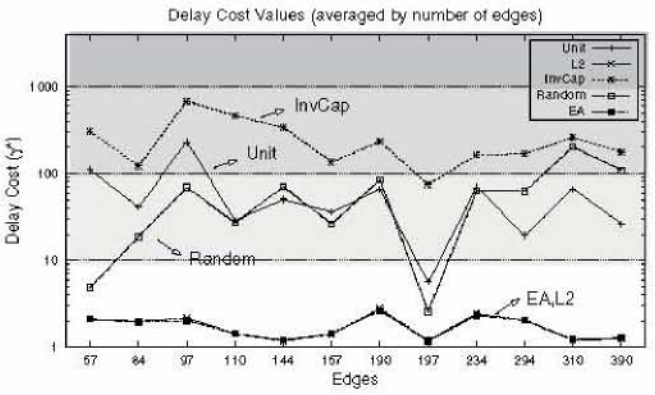

Fig. 5. Graphical representation of the results obtained by the different methods in delay optimization (averaged by the number of links).

C. Multi-objective optimization

In this section, the results for the multiobjective optimization are discussed. From the set of methods discussed before, only the *EA* and the *Random* heuristic can be used to perform multiobjective optimization by considering function f (Equation 21) as the cost/ fitness function. In all other heuristic methods the solution is built disregarding the cost function, so the results for multiobjective optimization can be copied from the ones obtained in its single objective counterpart.

The results of both *EAs* and *Random* methods are presented in terms of the values for the two objective functions (Φ^* and γ^*), since the value of f for these solutions can be easily obtained and is not relevant to the analysis (it does not represent any measure for the network behavior).

Table VII represents the results obtained in the example network, for the the multi-objective optimization obtained by the *EAs* and *Random* heuristics. The first two columns represent the parameters for demand and delay requirements; the next two indicate the results for the *Random* heuristic in both aims and, finally, the last two give the results of the *EA* for both congestion and delay, each with an extra information indicating the percentage by which this results exceed the ones obtained by the corresponding *EA* under the corresponding single objective cost function.

In Table VIII the results are aggregated averaging by the demand level (D_p) being shown, in the last row, the overall mean results. The overall results show that, in average, there a 25% decrease in the congestion performance and around 44% in the delays minimization. These values are quite good, since in this case both aims have to be simultaneously obbeyed, even if they are contradictory. In fact, a decrease in the performance, when compared to single objective optimization would always be expected. If the absolute average values for both cost functions are taken into account this indicates a quite acceptable network performance, well within the defined working region.

It is clear that when the problem gets harder in terms of congestion, both optimization aims are affected, both in absolute terms and when comparing to the results of single objective optimization in the previous sections. However, even in the worst case (when D_p equals 0.3) the network still manages an acceptable behavior. It is important to notice that in this scenario, and even when the D_p equals 0.2, all heuristics behave quite badly.

A similar picture is found looking at Table IX, where the results are averaged by the delay requirement parameter DR_p. In fact, with the increase of DR_p the results improve on both aims, both in absolute terms and considering the percentage of deviation from single objective optimization. Still, and as before, the results are quite acceptable in terms of network behaviour and the deviation from single objective results are within reasonable ranges.

Table X, on the other hand, confirms the good scalability properties of the *EA*. In fact, and as seen in the previous sections for both congestion and delay optimization, the results are almost constant for the different network sizes (in this case, measured by the number of nodes).

A different view is offered by Figures 6 and 7 where the results are plotted with the two objective functions in each axis. The former shows the results averaged by the demand levels and the latter by the delay requirements parameter.

TABLE VII

RESULTS FOR MULTIOBJECTIVE OPTIMIZATION IN ONE EXAMPLE NETWORK WITH 100 NODES AND 197 LINKS.

D_p	DR_p	Random		EA	
		Φ^*	γ^*	Φ^* (%)	γ^* (%)
0.1	3	27.36	39.97	1.14 (11.4%)	1.52 (10.2%)
0.1	4	7.22	16.06	1.09 (6.9%)	1.26 (11.8%)
0.1	5	8.82	2.28	1.08 (6.1%)	1.13 (8.9%)
0.2	3	356.25	29.42	1.47 (17.4%)	1.75 (26.2%)
0.2	4	274.06	2.37	1.40 (11.9%)	1.42 (25.9%)
0.2	5	339.06	1.96	1.38 (9.8%)	1.29 (23.7%)
0.3	3	587.51	48.72	1.76 (18.4%)	2.04 (47.8%)
0.3	4	495.32	7.08	1.61 (8.2%)	1.56 (38.4%)
0.3	5	601.00	2.34	1.56 (5.0%)	1.37 (31.3%)

TABLE VIII

RESULTS FOR THE MULTIOBJECTIVE OPTIMIZATION - AVERAGED BY D_p

D	Random		EA	
	Φ^*	γ^*	Φ^* (%)	γ^* (%)
0.1	88.00	106.79	1.17 (14.5%)	1.92 (12.8%)
0.2	481.50	136.68	1.47 (25.1%)	2.32 (35.2%)
0.3	949.85	148.96	2.41 (37.5%)	3.23 (83.3%)
Overall	506.45	130.81	1.68 (25.7%)	2.49 (43.8%)

In these graphs, the good overall network behavior of the solutions provided by the *EA* is clearly visible, both in absolute terms, regarding the network behavior in terms of congestion and delays, and when compared to all other alternative methods. In fact, it is easy to see that no single heuristic is capable of acceptable results in both aims simultaneously. *L2* behaves well in the delay minimization but fails completely in congestion; *InvCap* is better on congestion (although in a very limited range) but fails completely in the delays. *EAs*, on the other hand, are capable of a good compromise between both optimization targets.

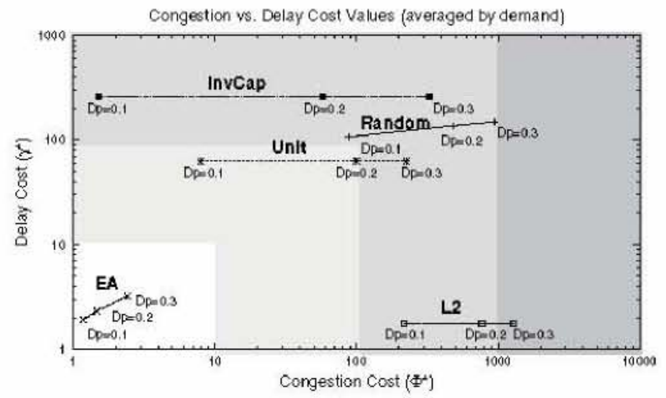

Fig. 6. Graphical representation of the results obtained by the different methods in the multiobjective optimization (averaged by D_P).

V. CONCLUSIONS AND FURTHER WORK

The optimization of *OSPF* weights brings important tools for traffic engineering in *WANs*, without demanding any modifications on the basic network management model. This work presented an optimization scheme based on *Evolutionary Algorithms* with an integer representation for the purpose of multiobjective routing in the Internet.

TABLE IX

RESULTS FOR THE MULTIOBJECTIVE OPTIMIZATION - AVERAGED BY DR_p.

DR	Random		EA	
	Φ^*	γ^*	Φ^* (%)	γ^* (%)
3	535.28	283.16	1.95 (42.8%)	4.22 (55.2%)
4	505.69	82.04	1.59 (20.3%)	1.78 (41.8%)
5	478.37	27.23	1.51 (14.2%)	1.48 (34.4%)

TABLE X

RESULTS FOR THE MULTIOBJECTIVE OPTIMIZATION - AVERAGED BY THE NUMBER OF NODES.

Node	Random		EA	
	Φ^*	γ^*	Φ^* (%)	γ^* (%)
30	283.32	74.77	1.58 (19.9%)	2.25 (24.3%)
50	442.16	165.63	1.78 (36.0%)	2.96 (51.9%)
80	619.14	170.75	1.62 (22.8%)	2.37 (42.7%)
100	681.17	112.09	1.75 (24.3%)	2.38 (56.2%)

To achieve this aim, an analytical model was developed allowing the performance evaluation of several *QoS* constrained *OSPF* routing scenarios of a given *ISP*. Resorting to a large set of network topology configurations, each one constrained by several bandwidth and end-to-end delay requirements, it was shown that the proposed *EAs* were able to provide *OSPF* weight settings able to satisfy the users demands. Moreover, the performance of *EAs* was compared with several heuristics, some of them rules of thumb typically used by network administrators, clearly showing the superiority of the proposed optimization approach in this specific multiobjective problem.

The research results presented in this work give ground to the idea that it is possible to develop network management tools which automatically provide network administrators with optimal configurations for a given network topology and corresponding sets of *QoS* demands. In this way, *ISP* resource provisioning management tasks can be now simplified, while providing better results and, consequently, strong financial improvements can be achieved by organizations using the proposed *OSPF* optimization scheme.

The proposed optimization framework, although requiring some computational effort, can be achieved in useful time, since a change in the *OSPF* weights in reply to a change in traffic is a rare event. If very distinct traffic profiles occur in different times of day (e.g. night and day) the corresponding matrices should be used to optimize distinct *OSPF* weights. Furthermore, the adaptation to a new solution is always faster than running from scratch, since a good solution is available to boost the search. Given all these facts, we can say that the proposed framework would be implemented in a straightforward way in a real world scenario.

Although a simple weighting method was used to face the multiobjective nature of the problem, the results were of high quality. This is probably due to the effort of normalizing

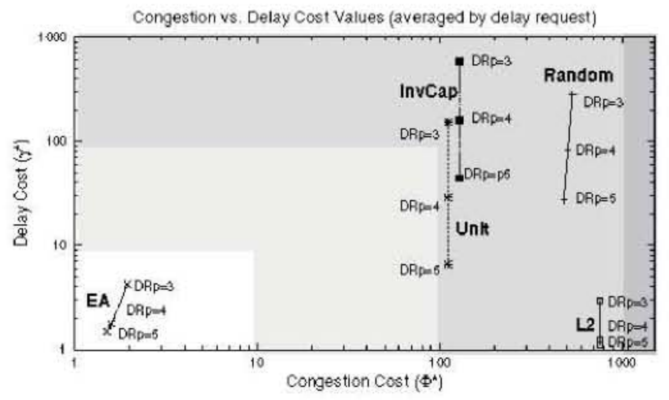

Fig. 7. Graphical representation of the results obtained by the different methods in the multiobjective optimization (averaged by DR_p).

both cost functions in a coherent manner. Nevertheless, the consideration of more specific *EAs* to handle multiobjective problems [14][15] will be taken into account in future work.

Memetic Algorithms, that consider local optimization procedures embedded in the *EA*, have also been attempted in the congestion optimization problem [16]. Their application in this bi-objective scenario is also a research direction that has a strong potential to improve these results.

Another topic for future work is the integration of priority *QoS* demands in the proposed optimization model. This will allow to provide *QoS* guarantees to specific flows without the overhead of network signaling.

ACKNOWLEDGMENT

The authors wish to thank the Portuguese National Conference of Rectors (CRUP)/British Council Portugal (B-53/05 grant), the Nuffield Foundation (NAL/001136/A grant), the Engineering and Physical Sciences Research Council (EP/522885 grant). and Project SeARCH (Services and Advanced Research Computing with HTC/HPC clusters), funded by FCT under contract CONC-REEQ/443/2001, for the computational resources made available.

REFERENCES

[1] D. Verma, *Supporting Service Level Agreement on IP Networks*. McMillan Technical Publishing, 1999.

[2] Z. Wang, *Internet QoS: Architectures and Mechanisms for Quality of Service*. Morgan Kaufmann Publishers, 2001.

[3] B. Davie and Y. Rekhter, *MPLS: Multiprotocol Label Switching Technology and Applications*. USA: Morgan Kaufmann, 2000.

[4] J. Moy, "RFC 2328: OSPF version 2," Apr. 1998.

[5] T. ThomasII, *OSPF Network Design Solutions*. Cisco Press, 1998.

[6] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 269-271, 1959.

[7] B. Fortz and M. Thorup, "Internet Traffic Engineering by Optimizing OSPF Weights," in *Proceedings of IEEE INFOCOM*, 2000, pp. 519–528.

[8] M. Ericsson, M. Resende, and P. Pardalos, "A Genetic Algorithm for the Weight Setting Problem in OSPF Routing," *J. of Combinatorial Optimization*, vol. 6, pp. 299–333, 2002.

[9] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "RFC 2475: An architecture for differentiated services," Dec. 1998.

[10] R. K. Ahuja, T. L. Magnati, and J. B. Orlin, *Network Flows*. Prentice Hall, 1993.

[11] J. Moy, *OSPF, Anatomy of an Internet Routing Protocol*. Addison Wesley, 1998.

[12] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd ed. USA: Springer-Verlag, 1996.

[13] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: Universal Topology Generation from a User's Perspective," http://citeseer.ist.psu.edu/article/medina01brite.html, Tech. Rep. 2001-003, Jan. 2001.

[14] C. Fonseca and P. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computation*, vol. 3, no. 1, pp. 1–16, 1995.

[15] C. C. Coello, *Recent Trends in Evolutionary Multiobjective Optimization*. London: Springer-Verlag, 2005, pp. 7–32.

[16] L. Buriol, M. Resende, C. Ribeiro, and M. Thorup, "A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing," *Networks*, 2003.