A Two-Phase Algorithm for Off-line Inter-domain Traffic Optimization

Manuel Pedro[1,2], Edmundo Monteiro[2], and Fernando Boavida[2]

[1] Polytechnic Institute of Leiria, ESTG, Morro do Lena, Alto do Vieiro, 2411-901 Leiria, Portugal
[2] University of Coimbra, Pólo II, Pinhal de Marrocos, 3030-290 Coimbra, Portugal
{macpedro, edmundo, boavida}@dei.uc.pt

Abstract. Inter-domain traffic engineering is a key issue when QoS-aware resource optimization is concerned. Mapping inter-domain traffic flows into existing service level agreements is, in general, a complex problem, for which some algorithms have recently been proposed in the literature. In this paper a two-phase algorithm to optimize the utilization of domain resources both from a technical perspective and a monetary costs perspective is proposed. The first phase is carried out by a greedy random algorithm, which returns a feasible solution. This is followed by an improvement phase performed by a genetic algorithm, which returns the optimal solution. Results show that the first phase produces quasi-optimal resource assignments for regional ISPs and outer core autonomous systems (AS). For transit core and dense core ASs, where the number of aggregate flows is considerably higher, the second phase leads to significant improvement.

1 Introduction

The main purpose of inter-domain resource optimization is to map incoming inter-domain traffic flows into inter-domain network resources, satisfying quality of service (QoS) requirements, while aiming at optimizing the use of network resources across autonomous systems (AS) boundaries. Network resources usage is, in any case, conditioned by existing Service Level Specifications (SLSs) that, in turn, result from the Service Level Agreements (SLAs) established between each domain and its neighbours. For the purpose of this paper, the terms 'domain' and 'autonomous system' are synonyms.

In order to describe the inter-domain relationships of an autonomous system, one can use a simple model, as shown in Figure 1. An autonomous system is interconnected with other autonomous systems by means of its ingress and egress interfaces.

The service offerings between autonomous systems as well as their mutual responsibilities are described by means of Service Level Agreements. In general, each SLA defines a set of contractual, administrative and technical requirements. The latter are called Service Level Specifications. An SLS comprises several items or clauses, including identification, application scope, flow identification, traffic conformance, excess treatment, and performance guarantees.

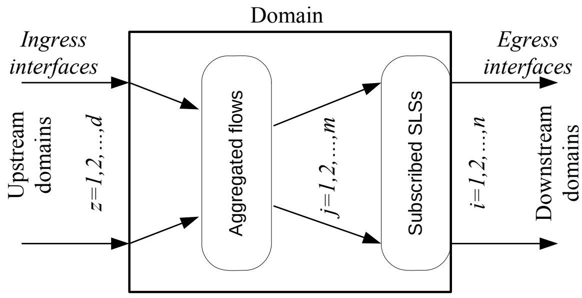

Figure 1 – Inter-domain relationship model

In the context of the present work an SLS is characterized by an egress interface, an inter-domain QoS class *i-QC* as proposed in [24], a destination prefix, the corresponding maximum bandwidth requirements *b*, and the monetary cost per unit of bandwidth, *mc*. The latter component reflects the monetary cost associated with the established SLA. An SLS entry for a domain with destination prefix *D* has, then, the following format:

SLS entry = [egress interface, i-QC, D, b, mc]

On the other hand, a domain receives from upstream domains a collection of *d* data flows towards other domains. Depending on the domain policy and on their common characteristics, such as destination and QoS class, these flows may be aggregated into *m* inter-domain traffic flows. The flows' common characterization includes the inter-domain class mapping and the destination prefix. That is, an aggregated flow entry for a domain with destination prefix *D* has the following format:

Aggregate flow entry = [ingress interface, i-QC, D, r]

where *r* is the bandwidth requirement of the aggregated flow. The flow will be mapped into one of the existing SLSs. The appropriate selection of the SLSs for the inter-domain traffic flows benefits the domain by improving the network resources utilization. This task is executed today in a trial-and-error fashion.

The optimization of inter-domain network resources falls into the Generalized Assignment Problem (GAP) category, where the objective is to find a minimum cost assignment of $m > 0$ jobs to $n > 0$ agents, subject to the agents' available capacity. In the specific problem at hand, jobs are aggregated traffic flows and agents are the established SLSs.

Formally, the problem can be stated as follows. Let $I = \{1,2,...,n\}$ be the set of SLSs and $J = \{1,2,...,m\}$ the set of aggregated traffic flows. For each SLS *i* there is a given resource capacity, expressed in terms of bandwidth, $b_i > 0$. For each $i \in I$ and each $j \in J$ there is a given set of costs, $c_{i,j} > 0$, and resource requirements, $r_{i,j} > 0$, for assigning an aggregated traffic flow j to an SLS i. Additionally, $x_{i,j}$ is a variable that is set to 1 if the traffic flow j is assigned to SLS i and 0 otherwise. The mathematical formulation is as follows:

$$c(x) = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{i,j} \cdot x_{i,j} \qquad \textbf{(1)}\text{, subject to}$$

$$\sum_{j=1}^{m} r_{i,j} \cdot x_{i,j} \le b_i, \forall i \in I, \quad \textbf{(2)} \qquad \sum_{i=1}^{n} x_{i,j} = 1, \forall j \in J, \quad \textbf{(3)}$$

$$x_{i,j} \in \{0,1\}, \forall i \in I, \forall j \in J. \qquad \textbf{(4)}$$

The optimisation goal is to minimize the cost (1), where the capacity constraint (2) ensures that the total resource requirements of the traffic flows assigned to each SLS do not exceed the available capacity. The assignment constraint (3) guarantees that each traffic flow is assigned to exactly one SLS.

The objective of the work presented in this paper is to propose a two-phase algorithm to optimize the utilization of domain resources both from a technical perspective and a monetary costs perspective. The proposed algorithm is a modified version of a Greedy Randomised Adaptive Search Procedure (GRASP) [15] with only one trial. The first phase is supported by a greedy random algorithm which finds a first feasible solution with respect to domain resources. Then it is followed by an optional second phase carried out by a genetic algorithm to reach an optimal solution. The genetic algorithm includes the solution found in the first phase in its initial population, on which it improves towards an optimum assignment solution during the generation cycles. The improvement resulting from the second phase is achieved at the expense of a considerable increase in the processing time.

This work also proposes the inclusion of monetary costs in the traffic optimization strategy, in line with [25]. For this, an objective function is proposed, which includes the domain's resources technical cost and the monetary cost.

In section 2 of this paper an overview of related work is given. This is followed by a presentation of the proposed algorithms and proposed objective function in section 3. In order to validate our work, an evaluation framework comprising a set of test scenarios, each one representing a type of internet autonomous system [1], is presented in section 4. Section 5 presents and discusses the obtained results. The conclusions and guidelines for further work are presented in section 6.

2 Related Work

Several studies on intra-domain resource optimization, such as [4][6][7][8] and [14], can be found in the literature. In the case of inter-domain, references [2][5] and [9] constitute the framework for most of the current proposals. For inter-domain traffic optimization we can point the study developed in the MESCAL project [9] from which resulted the proposals presented in [5] and [26]. The first proposal uses greedy algorithms for selecting egress routers for inter-domain traffic with bandwidth guarantees in order to optimise the total bandwidth consumption in the network. Nevertheless, this proposal gives us a partial image of a domain, limiting the number of egress routers to 30 and the number of prefixes to 1000. On the other hand, the second proposal [26] presents a genetic algorithm to solve the same problem, using a more limited scenario with 100 destination prefixes only. Moreover it does not give any clue about the processing times.

In the proposal presented in this paper, we employ a two-phase algorithm strategy which uses a greedy random algorithm to generate a first feasible solution. This solution is next improved on a second phase through a genetic algorithm. The improvement of greedy-random-generated solutions to find locally optimum solutions has already been proposed by several authors [21][15] in what is called greedy randomized adaptive search procedures (GRASP). An extensive list of bibliography about GRASP can be found in [16]. GRASP procedures are iterative procedures in which each iteration consists of two phases: a construction phase, in which a feasible solution is produced, and a local search phase, in which a local optimum is searched in the neighbourhood of the solution. At the end of the process, the best overall solution is kept as the result.

GRASP procedures have already been applied in the networking area. In [19] they are used to design a survivable Wide Area Network backbone. In [20] they are used to take routing decisions about private virtual circuits in frame relay networks.

Our proposal employs a genetic algorithm during the local search phase of the GRASP procedure. GRASP hybridization with a genetic algorithm has already been proposed by other authors like in [22][23] and has been applied, for example, in the Unmanned Air Vehicles area [17] to take routing decisions.

In what concerns genetic algorithms, they already have been used to solve network optimization problems [3][4][6][7][8][14]. These algorithms, belonging to the class of evolution strategies used in optimisation, resemble the process of biological evolution, where each individual is described by its genetic code, called a chromosome. On the other hand each chromosome is composed of individual genes. In the problem at hand, a gene is the assignment of a single aggregate traffic flow to an SLS, and an individual (i.e., a chromosome) is a potential solution.

To the best of the authors' knowledge, there is a clear lack of a comprehensive study about traffic optimization for the different types of existing domains as defined in [1]. On the other hand, there is also no proposal comparing the processing times of genetic algorithms with other types of algorithms for the inter-domain traffic optimization problem.

As in [26], our proposal contemplates both technical costs and monetary costs for traffic optimization. The latter are usually only considered at a high level of domain management. The proposed approach extends the use of monetary cost to low level decisions of traffic engineering.

3 Proposal

This section presents the proposed two-phase traffic engineering optimization algorithm. The first phase is carried out by a greedy random algorithm, with the purpose of determining feasible solution after a very short computation time. The second phase is performed by a genetic algorithm that, without the constraints of short response time, improves on the solution returned by the first phase. This section also presents the proposed objective function for cost determination.

3.1 Objective cost function

The resource usage optimisation goal is the minimisation of the cost function (5) or objective function $\psi_{i,j}$. This function has two components, one that measures the egress interfaces bottleneck, or technical cost (6), and one that measures the monetary cost of the assignment (7). The weight of each component is regulated by the parameter *α*. For example, when *α=0* only the monetary cost component is considered and when *α=1* only the technical cost component is taken into consideration. The cost function and its components are as follows:

$$\psi_{i,j} = \alpha \cdot R_{i,j} + (1-\alpha) \cdot M_{i,j} \quad \text{with } \alpha \in [0,1] \tag{5}$$

and

$$R_{i,j} = \frac{1}{(b_i - b_j + 0.1)^2} \tag{6}$$

$$M_{i,j} = c_i \cdot b_j \tag{7}$$

where b_i and b_j are the SLS *i* available bandwidth and the aggregated flow *j* required bandwidth, respectively. c_i is the SLS *i* monetary cost. In the denominator of (6) the value 0.1 was added in order to limit the value of the technical cost to 100.

3.2 First phase – Greedy Random Algorithm

The algorithm used in the fist phase is a variation of the greedy random algorithm proposed in [18]. On a limited number of trials the algorithm tries to find a feasible solution. Each trial a cycle starts by finding the restricted candidate list (RCL) of the current flow. The RCL contains the set of feasible SLSs. The selection of these SLSs is based on information such as destination address prefix, QoS class, and available bandwidth, and the resource cost returned by equation (7) (see section 3.1). Then it randomly assigns the flow to one of the SLSs in the RCL.

This algorithm fails if no solution is found after *M* trials. In our tests a solution was found after 20 trials on average, and the algorithm never failed for the studied scenarios. This algorithm has a computational complexity of $O(Mmn)$, where *m* is the number of flows and *n* the number of SLSs.

3.3 Second phase - Solution refinement

The algorithm used to perform a refinement of the solution returned in the first phase is a variation of the proposal in [13] and different from the one presented in [26]. As shown in Figure 2, the algorithm starts by building the restricted candidate gene list (RCGL), in the same way as the construction of the RCL list of the first phase. Then the flows are randomly and uniformly distributed by the RCGL elements, leading to the initial population. The solution returned by the algorithm in the first phase is also inserted into this population. Then the population is submitted to an evolution for a predefined number of generations.

On each generation the fitness of every chromosome is evaluated by (1) using the objective function given in equation (5). The chromosomes are ranked and the generations' best fitted individual is saved. Then the fraction of the more fitted individuals is retained for the next generation and the fraction of the less fitted is discarded. After that, two chromosomes from the higher fitness fraction of the population are randomly selected and submitted to a crossover process to produce an offspring. In this process a set of genes of the higher fitness chromosome are randomly selected with probability *pc*, and combined with a set of genes of the lower fitness chromosome, in order to compose a new chromosome for the next generation. This offspring is then submitted to a mutation process where its genes are randomly changed with a mutation probability *pm*. These new genes belong to the RCGL initially created.

```
Build the restricted candidate gene list (RCGL);
Generate a population of N chromosomes;
For each generation
        Calculate the fitness of each chromosome;
        Rank the population by fitness;
        Save the generation's best chromosome;
        Pass a fraction of the higher fitness individuals to next generation;
        Discard a fraction of lower fitness individuals
        Crossover the remaining chromosomes to generate new individuals
        Mutate the new individuals
Endfor.
```

Figure 2 – Proposed genetic algorithm

Empirical trials gave us the most suitable values for *N, pc* and *pm*, as 60, 0.7, and 0.07 respectively. For the number of generations we found the value of 40 suited, higher values leading to no improvement.

The computational complexity of this algorithm is $O(GNmn)$, where G is the number of generations.

4 Evaluation Framework

In order to evaluate each algorithm, several scenarios were built, each representing a typical autonomous system.

The characterisation of autonomous systems has been studied in several pieces of work in the recent past [1][10][11]. For the present study, the characterisation presented in [1] was used as a basis, as this represents the most recent work.

According to [1], ASs can be classified as costumer ASs, small regional ISPs, outer core, transit core, and dense core. The existing numbers of ASs, for each of these categories, are presented in Table 1.

Table 1. Hierarchical distribution of ASs [1]

Level	Layer	# of ASs
0	Dense Core	20
1	Transit core	129
2	Outer core	897
3	Small regional ISPs	971
4	Customers	8898

Table 2. Interconnectivity between levels [1]

Level	0	1	2	3	4
0	312	626	1091	958	6732
1	183	850	1413	665	3373
2	29	145	1600	543	3752
3	0	0	0	212	2409

Still according to [1], there are certain numbers of ingress and egress interfaces to ASs of the same or different class. These numbers are reproduced in Table 2, where the columns represent ingress interfaces and the rows represent egress interfaces. For instance, there are 312 ingress interfaces between all dense core ASs (i.e., level 0 ASs), 183 ingress interfaces between dense core and transit core ASs, and 29 ingress interfaces between dense core and outer core ASs.

Using the values of Tables 1 and 2 as a basis, four different evaluation scenarios were constructed, each one representing a typical non stub autonomous system of type ‘small Regional ISPs’, ‘outer core’, ‘transit core’, and ‘dense core’. ASs of type ‘customer’ as stub ASs were not considered in the tests.

Each of these four scenarios (i.e., typical AS) is characterised by a given number of ingress interfaces, egress interfaces, supported QoS classes, maximum number of destination prefixes, aggregate bandwidth and number of aggregate flows, as presented in Table 3.

Table 3 – Test scenarios characterization

AS level	Dense core	Transit core	Outer core	Small Regional ISP
Test scenario	#0	#1	#2	#3
Ingress Interfaces	26	13	5	2
Egress Interfaces	486	50	7	3
QoS classes	3	3	3	3
Destinat. prefixes	10915	10915	10915	10915
Bandwidth (aggr.)	0..100	0..100	0..100	0..100
Aggregate flows	454997	227576	65168	42869

In this table, the number of destination prefixes corresponds to the total number of ASs, which can be found by adding the number of ASs given in Table 1.

The number ingress interfaces in Table 3 was found by dividing the total number of ingress interfaces presented in Table 2 by the number of corresponding ASs presented in Table 1 (for instance, in the case of dense core ingress interfaces, 26 = (312 + 183 + 29) / 20). The number of egress interfaces was given applying the same process to egress interfaces of the table.

A maximum of 3 inter-domain QoS classes was selected. The maximum bandwidth per aggregate flow is expressed by a number between 0 and 100 (like a percentage).

The domain characteristics defined above were used by a flow generator algorithm in order to determine the number of aggregate flows for each of the scenarios. This algorithm consists of the following five steps:

Step 1: Create a set of ingress traffic flows;
Step 2: Aggregate the flows with the same class requirements and the same destination;
Step 3: Randomly and uniformly distribute *m* aggregate flows by *n* egress interfaces;
Step 4: Create some spare bandwidth summing a random, uniformly-distributed value between 0 and 10 to each SLS;
Step 5: Set a monetary cost to each SLS adding a random, uniformly-distributed value between 1 and 10. This value is not correlated with the SLSs' bandwidth.

In Step 1, the creation of the ingress traffic flows is done by randomly and uniformly distributing a maximum of *F* flows by each of the *q* inter-domain QoS classes and by *m* ingress interfaces, for *F* destinations prefixes, where *F* is the total number of autonomous systems previously defined. On the other hand, in step 2, the flows are aggregated, and on the remaining steps 3 to 5 the traffic flow and the SLS matrices are created.

For each generated scenario this algorithm guarantees that there exists at least one global assignment solution, that is, that it is possible to assign all traffic flows to SLSs.

The algorithms under study were coded using the MATLAB language, and the tests were performed in a machine with a Pentium 4 processor at 1.7 GHz, 1GBytes of RAM, with the MS Windows XP operating system.

The comparison of the algorithms was carried out using the following parameters:

1. total cost of the solution, given by equation (1), with the cost given by (5) for $\alpha=\{0, 0.5, 0.98, 1\}$;
2. total processing time spent by the test machine in order to run the algorithms for each test scenario.

5 Results

Each of the four scenarios presented in Table 3 was used for testing the assignment algorithms presented in Section 3. The results of the tests are presented below.

Figure 3 shows the comparison between the processing times of both algorithms. The processing time of the genetic algorithm varies from 36 to 155 times the processing time of the greedy random algorithm. The processing time of the genetic algorithm includes the time taken by the greedy random algorithm in order to find the first-phase solution.

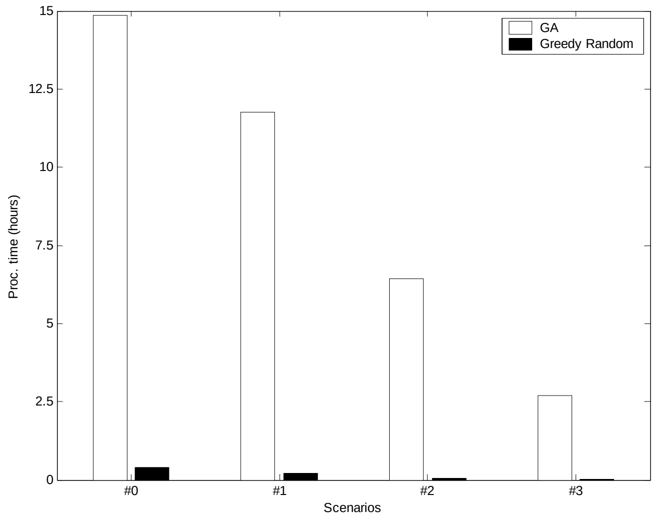

Figure 3 – Processing time comparison between the genetic algorithm (GA) and the greedy random algorithm.

Figure 4 presents the technical costs comparison having by reference a set of "optimal" values that resulted from the best values of the overall test trials. This figure shows that for scenarios #2 and #3 the values returned by the greedy random algorithm are almost optimal and little improvement is achieved with the genetic algorithm. Additionally, the figure also shows that the genetic algorithm produces considerable improvement for scenarios #0 and #1. The technical costs are reduced nine times for scenario #0 and six times for scenario #1. The optimal values of the technical costs are reached with $\alpha = 0.98$.

Figure 5 presents a similar comparison, this time in terms of monetary costs. Again, the greedy random algorithm returns quasi optimal results for scenarios #2 and #3, while the genetic algorithm leads to considerable improvement in scenarios #0 and #1. The optimal values of the monetary costs are reached when α is lower than 0.98.

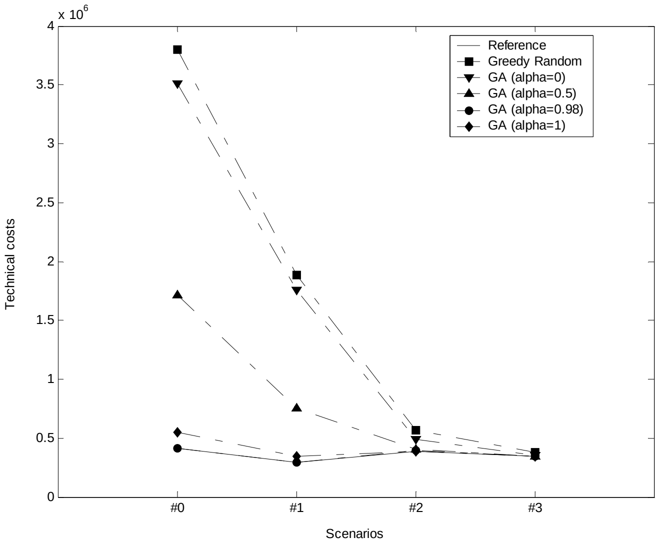

Figure 4 – Technical costs comparison between the reference values and the greedy random and the genetic algorithms

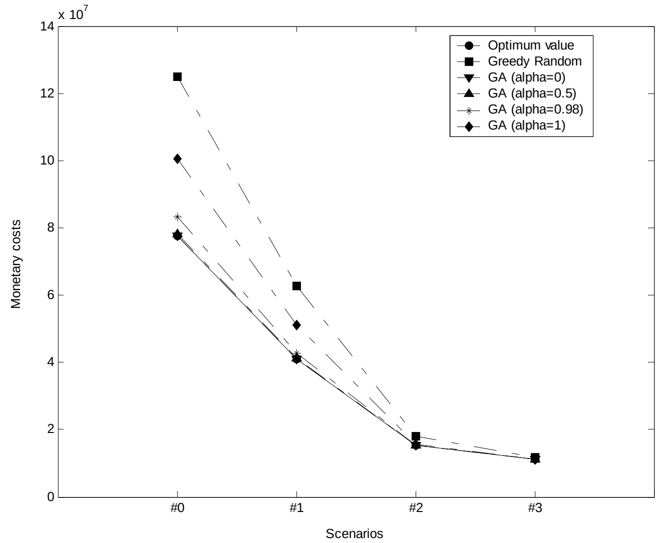

Figure 5 – Monetary costs comparison between the reference values and the greedy random and the genetic algorithms

6 Conclusions

Inter-domain QoS-aware resource optimisation is one of the main challenges of current traffic engineering. In this paper, a two-phase algorithm for the optimisation of inter-domain traffic assignment was proposed. This optimisation can be done with respect to two different and independent variables, the technical cost and the monetary cost of the used resources.

The evaluation of the proposed algorithm has shown that the use of a two-phase approach has several benefits. The first phase leads to a feasible solution within a relatively short time. Moreover, the solution provided by this phase is quasi-optimal for Outer Core and Small Regional ISP scenarios. The second phase leads to optimised results, at the expense of processing power and time. The benefits of this phase are apparent in the Dense Core and Transit Core scenarios, for which it leads to an improvement of technical cost up to 9 times and a monetary improvement of more than 1.5 times.

The work has also shown that it is possible optimize traffic flows assignment not only in terms of technical cost but also in terms of monetary cost.

The work presented in this paper has unveiled several other issues for future work. The inclusion of other QoS parameters in the decision process is an obvious field for further research. In this respect, ways of efficiently linking network operation information (e.g. QoS routing information) with high-level traffic engineering decisions will be explored in the context of on-going projects. Another line of research will be the proposal and evaluation of new assignment algorithms, in order to significantly reduce the processing time.

References

1. Subramanian, L., Agarwal, S., Rexford, J., Katz, R.: Characterizing the Internet hierarchy from multiple vantage points. In: Proceedings of IEEE INFOCOM. (2002)
2. Morand, P., et al.: D1.1: Specification of Business Models and a Functional Architecture for Inter-domain QoS Delivery. IST-2001-37961, unpublished. (June 2003)
3. Lin, X., Kwok, Y., Lau, V.: A genetic algorithm based approach to route selection and capacity flow assignment. Computer Communications, Vol. 26. (2003) 961-974
4. Ericsson, M., Resende, M., Pardalos, P.: A genetic algorithm for the weight setting problem in OSPF routing. Technical report, ATT Shannon Laboratory (2001)
5. Ho, K., Wang, N., Trimintzios, P., Pavlou, G., Howarth, M.: On Egress Router Selection for Inter-domain Traffic with Bandwidth Guarantees. In: Proceedings of IEEE Workshop in High Performance Switching and Routing (HPSR'2004), Phoenix, Arizona, USA (April 2004)
6. Buriol, L., Resende, M., Ribeiro, C., Thorup, M.: A memetic algorithms for OSPF routing. In: Proceedings of 6th INFORMS Telecom (2002) 187--188
7. Buriol, L., Resende, M., Ribeiro, C., Thorup, M.: A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. Unpublished [Online]. Available: www.optimization-online.org/DB_FILE/2003/06/674.pdf (2003)
8. Riedl, A.: A hybrid genetic algorithm for routing optimization in IP networks utilizing bandwidth and delay metrics. In: Proceedings of IEEE Workshop on IP Operations and Management (IPOM), Dallas (October 2002)

9. MESCAL project [website]: www.mescal.org.
10. Gao, L.: On inferring autonomous system relationships in the Internet. IEEE/ACM Trans. Networking, vol. 9, no. 6 (December 2001)
11. Ge, Z., Figueiredo, D., Jaiwal, S., Gao, L.: On the hierarchical structure of the logical Internet graph. In. Proceedings of SPIE ITCOM (August 2001)
12. Bressoud, T., Rastogi, R., Smith, M.: Optimal Configuration for BGP Route Selection. In: Proceedings of IEEE INFOCOM' 2003, San Francisco (March/April 2003)
13. Mitchell, M.: An Introduction to Genetic Algorithms. The MIT Press (1999)
14. Reichert, C., Magedanz, T.: A Fast Heuristic for Genetic Algorithms in Link Weight Optimization. In: Solé-Pareta, J. et al. (eds.): Quality of service in the Emerging Networking Panorama. Springer (2004) 144-153
15. Resende, M.: Greedy Randomized Adaptive Search Procedures (GRASP). In: Floudas, C., Pardalos (eds.). Encyclopedia of Optimization. Kluwer Academic Press, vol. 2 (2001) 373-382
16. Festa, P., Resende, M.: An annotated bibliography. In: Hansen, P., Ribeiro, C.(eds.): Essays and Surveys on Metaheuristics. Kluwer Academic Publishers (2001)
17. Juell, P., Perera, A., Nygard, K.: Application of a Genetic Algorithm to Improve an Existing Solution for the Generalized Assignment Problem. In: Proceedings of the 16th International Conference on Computer Applications in Industry and Engineering, Las Vegas (November 2003)
18. Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to Algorithms. The MIT Press, 2nd ed. (2001)
19. Cancela, H., Robledo, F., Rubino, G.: A GRASP algorithm for designing a Wide Area Network backbone. In: Proceedings of International Network Optimization Conference (INOC), Évry/Paris, France (2003)
20. Resende, M., Ribeiro, C.: A GRASP with path-relinking for private virtual circuit routing. In. Networks, vol. 41 (2003) 104-114
21. Li, Y., Pardalos, P., M., Resende, M.: A greedy randomized adaptive search procedure for the quadratic assignment problem. In: Pardalos, P., Wolkowicz, H. (eds.): Quadratic assignment and related problems, volume 16 of DIMACS Series on Discrete Mathematics and Theoretical Computer Science, American Mathematical Society (1994) 237-261
22. Ahuja, R., Orlin, J., Tiwari, A.: A greedy genetic algorithm for the quadratic assignment problem. Computers and Operations Research 27 (2000) 917–934
23. Areibi, S. and A. Vannelli.: Efficient hybrid search techniques for circuit partitioning. In: IEEE 4th World Multiconference on Circuits, Systems, Communications & Computers. (2000)
24. Levis, P., Boucadair, M., Morand, P., Trimintzios, P.: The Meta-QoS-Class concept: a step towards global QoS inter-domain services. In: proc. of IEEE International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2004) (October 2004)
25. Aiber, S. et al.: Autonomic Self-Optimization According to Business Objectives. In: Proceedings of the International Conference on Autonomic Computing (ICAC'04) (2004)
26. Ho, R., Pavlou, G., Howarth, M., Wang, N.: An Incentive-based Quality of Service Aware Algorithm for Inter-AS Traffic Engineering. In: Proceedings of the IEEE International Workshop on IP Operations and Management (IPOM'2004), Beijing, China (October 2004)