A Memetic Algorithm for OSPF Routing

Luciana S. Buriol, *UNICAMP, Campinas, SP, Brazil*
Mauricio G. C. Resende, *AT&T Labs Research, Florham Park, NJ, USA*
Celso C. Ribeiro, *Catholic University, Rio de Janeiro, RJ, Brazil*
Mikkel Thorup, *AT&T Labs Research, Florham Park, NJ, USA*

Keywords: OSPF routing, Internet, metaheuristics, genetic algorithm, memetic algorithm.

1. Introduction

With increasing Internet traffic demand, service providers have adopted new technologies to improve the utilization of existing network resources. Data packet routing algorithms can have a major impact on network utilization. Packets are sent along network paths from source to destination following a protocol. Open Shortest Path First (OSPF) is one of the most commonly used intra-domain Internet routing protocols. In OSPF, traffic is routed along shortest paths, splitting flow at nodes with more than one outgoing link on a shortest path to the destination IP address. Integer link weights (in the range 1 to 65535) are assigned by the network operator. A path length is the sum of the weights of the links in the path. We consider the problem of determining OSPF weights, given a set of projected demands, with the objective of minimizing network congestion. The weight assignment problem is NP-hard. Ericsson, Resende, and Pardalos [2] proposed a genetic algorithm (GA) for OSPF weight setting. In this paper, we extend the GA by adding to it a local improvement procedure, resulting in a memetic algorithm (MA). We compare the MA with an optimized version of the GA and show that the MA not only finds better solutions, but also converges to a given solution in less time.

2. Genetic and Memetic Algorithms

We summarize the detailed description of the GA given in [2]. Let N and A denote, respectively, the sets of nodes (routers) and arcs (transmission links). Solutions are represented by a vector of weights $w = \langle w_1, w_2, \ldots, w_{|A|} \rangle$, where $w_i \in [1, 65535]$ for each arc $i = 1, \ldots, |A|$. We use the piecewise linear objective function proposed by Fortz and Thorup [3] to minimize network congestion.

The initial population, of $|P|$ elements, is randomly generated. The population is partitioned into three sets $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ according to each element's objective function value. The best solutions are kept in $\mathcal{A}$, while the worst are in $\mathcal{C}$. All solutions in $\mathcal{A}$ are promoted to the next generation. Solutions in $\mathcal{B}$ are replaced by crossover of one parent from $\mathcal{A}$ with another from $\mathcal{B} \cup \mathcal{C}$ using the *random keys* crossover scheme of Bean [1]. All solutions in $\mathcal{C}$ are replaced by randomly generated solutions.

The MA (see [5]) consists of the GA described above with an additional local improvement procedure applied to each solution obtained during crossover. The local improvement procedure seeks to increase the weights of k links with highest objective function component, thus relieving the traffic on highly loaded links by rerouting it through less loaded links.

3. Dynamic Shortest Path Algorithm

In the local improvement procedure, the computational bottleneck consists of the cost evaluations for different weight settings. We use the fact that when evaluating costs with consecutive weight settings only a few shortest paths are affected and typically only a small portion of the shortest path

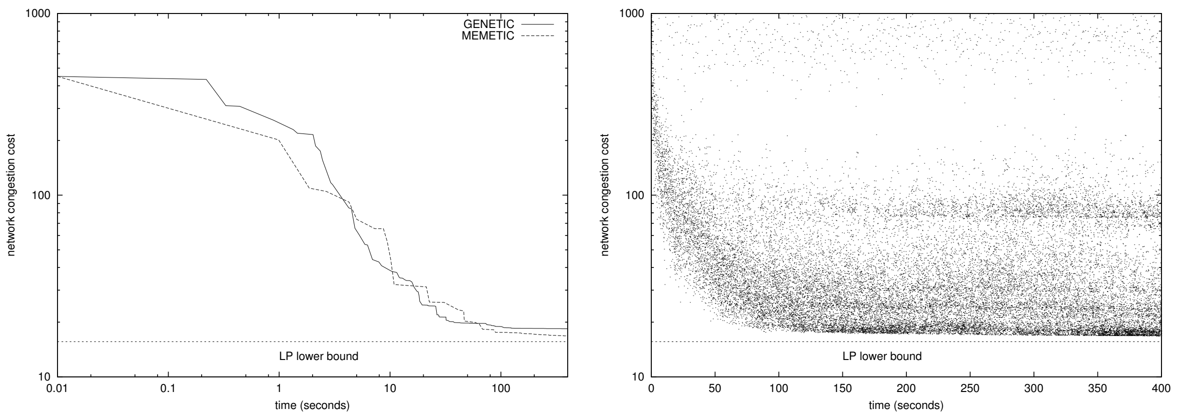

Figure 1: Left figure compares MA and GA on AT&T Worldnet backbone network. Right figure shows all solution values produced by the MA. LP lower bound is shown on both figures.

changes. For this reason it is sensible to avoid recomputation of all shortest paths. Ramalingam and Reps [6] propose an efficient algorithm [4] for these dynamic computations. We specialize their procedure tailoring it to our problem.

4. Computational Results

Figure 1 illustrates the typical behavior of the MA on a real-world network, a realistic version of the AT&T Worldnet backbone network. The plot on the left shows that the solution found by the MA is better than the one produced by its genetic counterpart. The plot on the right shows that as the algorithm progresses, the concentration of good solutions produced by the MA increases and the best solution is near the linear programming lower bound.

References

[1] Bean, J. C., "Genetic algorithms and random keys for sequencing and optimization," *ORSA J. on Comp.*, vol. 6, pp. 154–160, 1994.

[2] Ericsson, M., M. G. C. Resende, and P. M. Pardalos, "A genetic algorithm for the weight setting problem in OSPF routing," *J. of Comb. Opt.*, forthcoming.

[3] Fortz, B. and M. Thorup, "Increasing Internet capacity using local search," Technical report, AT&T Labs Research, 2000.

[4] Frigioni, D., M. Ioffreda, U. Nanni, and G. Pasqualone, "Experimental analysis of dynamic algorithms for the single source shortest path problem," *ACM J. of Exp. Alg.*, vol. 3, article 5, 1998.

[5] Moscato, P., "Memetic Algorithms," in *Handbook of Applied Optimization*, P.M. Pardalos and M.G.C. Resende (Eds.), Oxford University Press, New York, pp. 157–168, 2002.

[6] Ramalingam, G. and T. Reps, "An incremental algorithm for a generalization of the shortest-path problem," *J. of Algorithms*, vol. 21, pp. 267–305, 1996.