

On Routing IP Traffic Using Multi-constrained Genetic Optimization with Penalty Functions

Tayseer M. Fathelrahman¹ & Antoine B. Bagula^{1,2}

Abstract—We formulate the routing of traffic in emerging IP networks as a multi-constrained optimization problem solved using the well known penalty function strategies. These include the adaptive and the co-evolutionary penalty function strategies used in a hybrid genetic algorithm to find the optimal paths for the traffic offered to an IP network. We adopt the occurrence of links in the computed paths as a cost function used in a memetic algorithm to load balance an IP network by reducing the interference among competing flows. Using three test networks, we summarize and compare the quality of the paths found by the two strategies.

I. INTRODUCTION

The last decade have experienced an increased interest in the use of evolutionary optimization strategies to solve Nonlinear Programming problems. Evolutionary algorithms use concepts from real-world genetics to evolve solutions to problems according to the Darwin evolution theory. They are based on an evolutionary paradigm where each iteration of the algorithm transforms one population of individuals into a new generation, using some pre-determined fitness measure for an individual. In applications of evolutionary algorithms, potential solutions must be represented and encoded in terms of *genome*. Each problem generally has its own genome representation, and more than one representation could be used for a given problem. The fitness measure or *fitness function* determines how good the solution represented by some genome is. The appropriate fitness function is determined by the problem and genome representation. *Genetic Algorithms (GAs)* are based on a population selection where the evolution of one generation into another relies on the three main genetic operations: (1) *replacement* (2) *crossover* and (3) *mutation*. *Replacement* is a direct copying of a member of the current generation into the next generation. *Crossover* is the combination of two genomes from the current generation into two different genomes in the next generation. Crossover attempts to combine good solutions to find potentially better solutions. *Mutation* is the random permutation of one of the tokens in the Genome representation of a member of the current generation. By introducing new solutions at each stage of the algorithm, mutation ensures

that the evolution process does not get stuck at a local optimum. There are probabilities associated with the crossover, replacement and mutation operations. These probabilities are denoted P_c , P_r and P_m respectively, and $P_c + P_r + P_m = 1$. In general, $P_m \ll P_c$ and $P_c \approx P_r$. Candidates for the genetic operations are chosen randomly, but the selection is *fitness-proportionate* to ensure the survival of good solutions over generations. The conditions for the termination of the algorithm are problem-specific, although for practical reasons one often limits the number of iterations.

Evolutionary algorithms can find acceptably good solutions to problems by examining and manipulating a set of possible solutions from a set of designs but are not guaranteed to find the global solution to a problem. Evolutionary algorithms have been deployed in the literature to solve real-life problems using different strategies. The selection of the appropriate strategy to be used for a specific application is an important aspect upon which the usefulness of the evolutionary optimization process depends.

This paper addresses the problem of routing traffic in IP networks using evolutionary optimization. We formulate the routing in emerging IP networks as a multi-constraint routing problem solved using different genetic optimization strategies. These include the well-known penalty strategies using the *The Adaptive* and *The Co-evolutionary (Self-Adaptive)* penalty function methods. We compare the two strategies when computing paths for the flows offered to a 23- and 31-node test networks.

The reminder of this paper is organized as follows. Section II reviews the two penalty function methods while section III describes a multi-constraint optimization model. An application of the different strategies to compute paths for the flows offered to a 23- and 30-node networks is presented in section IV. We draw some preliminary conclusions and present guidelines for future work in section V.

II. PENALTY FUNCTION STRATEGIES

The penalty function methods are considered as one of the important methods to solve the nonlinear programming problems. They transform constrained optimization problems into unconstrained problems by joining both the equality and inequality constraints to the objective function. In Coello [5], various types of penalty function methods were discussed. They vary in the computational cost to attain the optimal solutions, handling the constraints, in addition to minimizing the objective functions. As presented in [5], the optimization

1. School of Mathematical Sciences, Computer Science Section, University of Stellenbosch, 7600 Stellenbosch, South Africa. Email: tfath, bagula@cs.sun.ac.za. Tel: +27 21 8084232 Fax: +27 21 8084416

2. Department of Electronics, Computer and Software Systems, Telecommunication Systems Laboratory, Royal Institute of Technology (KTH), Isafjordsgatan 39, SE-164 40 Kista, Sweden. Email: abbagula@kth.se Tel: +46 8 790 42 84 Fax: +46 8 751 17 93

This work is supported by grant numbers 2047362 and 2677 from the South African National Research Foundation, Siemens Telecommunications and Telkom SA Limited.

problem under consideration is of the form:

$$\min_{\mathbf{x} \in \mathbf{R}^n} f(\mathbf{x}) \quad (1)$$

subject to:

$$g_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m \quad (2)$$

$$h_j(\mathbf{x}) = 0, \quad j = 1, \dots, p \quad (3)$$

Where m is the number of inequality constraints, and p in the number of equality constraints.

To solve this problem, Bean and Hadj-Alouane [3], [1] proposed a penalty function method in the form:

$$fitness(\mathbf{x}) = f(\mathbf{x}) + \lambda \left[\sum_{i=1}^m g_i^2(\mathbf{x}) + \sum_{i=1}^p |h_i(\mathbf{x})| \right] \quad (4)$$

where $\lambda(t)$ is updated at every generation t in the following way:

$$\lambda(t+1) = \begin{cases} \frac{1}{\beta_1} \lambda(t) & \text{if case 1} \\ \beta_2 \lambda(t) & \text{if case 2} \\ \lambda(t) & \text{otherwise} \end{cases} \quad (5)$$

Case 1 and case 2 denote situations where the best individual in the last k generations was always feasible (case 1) or was never feasible (case 2). This method is called *Adaptive Penalty*, and it has other approaches which can be found in [5].

Coello [4] developed another approach to compute fitness functions for problems with only inequality constraints. He named it *A Self-Adaptive penalty function* and in [5] he referred to it as *A Co-evolutionary Penalty*. It takes the form:

$$fitness(\mathbf{x}) = f(\mathbf{x}) - (coef \times w_1 + viol \times w_2) \quad (6)$$

The objective function $f(\mathbf{x})$ takes its value from a given set of variable encoded in a chromosome; w_1 and w_2 are 2 penalty factors (considered as integers); the sum of all the amounts by which the constraints are violated is given by *coef* where:

$$coef = \sum_{i=1}^m g_i^2(\mathbf{x}) \quad \forall g_i(\mathbf{x}) > 0 \quad (7)$$

The integer parameter *viol* takes the value 0 initially, and increases by one for each constraint that has been violated.

III. THE GENETIC OPTIMIZATION MODEL

Given a physical network which is viewed as a graph $G = (N, L)$ where N is the number of nodes and L is the number of links. The routing of traffic in this network is considered as a multi-constraint routing optimization problem consisting of computing a set of link weights used in path computation to find the optimum path which minimize the total delay in the network. The resulting routing objective is expressed by:

$$\min F = \sum_{l=1}^N F_l(f_l) = \sum_{l=1}^N \frac{f_l}{C_l - f_l} \quad (8)$$

subject to:

$$f_l - C_l \leq 0, \quad l = 1, 2, \dots, N \quad (9)$$

where C_l is the capacity of the link l , f_l is the total amount of traffic offered onto the link l , and N is the number of all links. $F_l(f_l)$ gives the delay at the link l . The function F described by equation (8) gives the total time delay for the network. Equation (9) put a constraint on maximum amount of flow for each link in the network.

In a previous work, Bagula and Wang [2] applied a hybrid genetic algorithm to find a set of optimal link weights leading to reduced maximum link utilization. In their model they expressed the population by a set of links represented by a vector where each position holds a link weight. They used a memetic algorithm illustrated by Figure 1 (b) where a classical genetic algorithm illustrated by Figure 1 (a) is complemented by a local search to improve the genetic individuals fitness through hill-climbing. The rest of this paper will address memetic algorithms but consider a different optimization model which minimizes the total queuing delay in a network subject to constraints on the occurrence of the links in the paths found when routing the traffic. Our changes includes changing their used genetic class to implement an new fitness function evaluation and other modifications on the their class to fit with the newly proposed genetic class.

Adaptive Penalty

For this method, the fitness function for the problem described by the equations (8) and (9) takes the formula:

$$fitness(i) = \frac{f_i}{C_i - f_i} + \lambda(t)(f_i - C_i)^2 \quad (10)$$

The initial population is generated randomly. We set $\lambda(0) = 100$, and $\lambda(t+1)$ is updated subject to the law (5). We set $\beta_1 = 1$, and $\beta_2 = 2$.

Co-evolutionary Penalty

For this method, the fitness function for the problem described by the equations (8) and (9) takes the formula:

$$fitness(i) = \frac{f_i}{C_i - f_i} - (coef \times w_1 + viol \times w_2)$$

Both of w_1 and w_2 were set to the value 100. The parameter *coef* is given by the relation $coef = f_i - C_i$. *viol* was initiated to 0 as described by the method ([5]) and takes the values 0 and 1 according to whether there was no or there was a violation on the constraint (9) accordingly.

IV. PERFORMANCE EVALUATION

This section presents simulation experiments to compare the performance of the two different genetic penalty algorithms (1) the adaptive penalty and (2) the co-evolutionary penalty algorithms. We considered three test networks in our experiments, two 23-node 76-link networks but with different number of routes, refer to the USA network, and a 31-node 64-link PAREN (Pan Africa Research and Education Network). These networks are illustrated by figures 1 and 2 respectively.

Our comparison based on several criteria, the total delay,

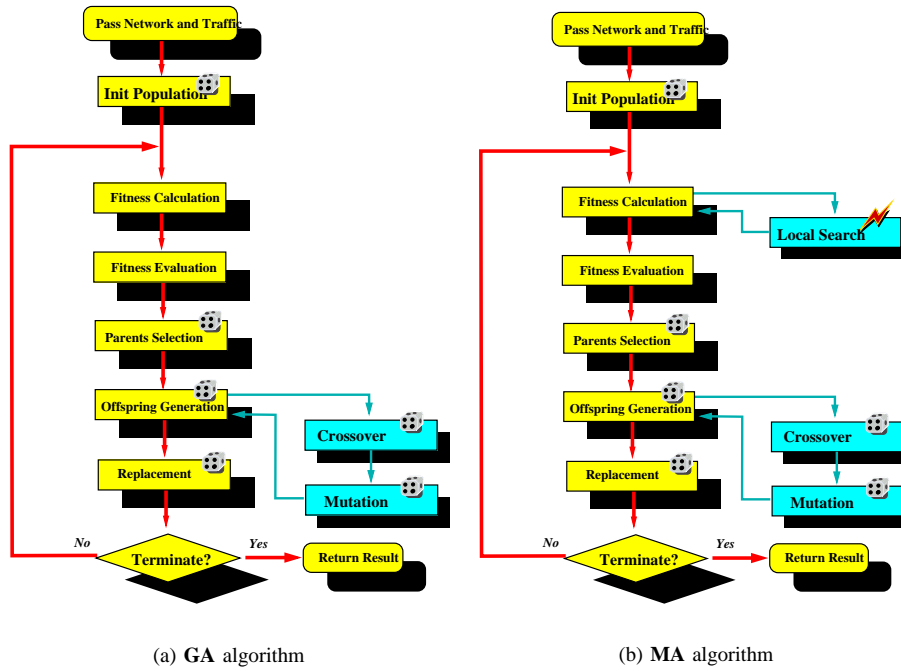


Fig. 1. "Evolutionary algorithms"

TABLE I

TABLE SHOWING STATISTICS FOR BOTH PENALTY METHODS, DISPLAYING THE TOTAL DELAY, THE MAXIMUM LINK DELAY WITH THEIR STANDARD DEVIATIONS AND THE TIME

Net.	Penalty Method	Tot Delay	StdDiv Delay	Max Link	Time
Net1	Adaptive	3.0584	0.1626	0.3496	60
	S-Adaptive	4.1354	0.4603	0.4717	59.8
Net2	Adaptive	7.9122	0.7795	0.7839	133.1
	S-Adaptive	9.7554	1.793	0.8067	132.6
Net3	Adaptive	0.1657	0.0052	0.0150	333
	S-Adaptive	0.1857	0.0117	0.0199	331

TABLE II

TABLE SHOWING STATISTICS FOR BOTH PENALTY METHODS, DISPLAYING THE PATH LENGTH, THE MAXIMUM PATH LENGTH, THE STANDARD DEVIATION AND THE STRONG CORRESPONDENCE

Net.	Penalty Method	Path Length	Max Path Length	StdDiv	Strong Corresp
Net1	Adaptive	3.000	7.300	1.613	26.316%
	S-Adaptive	3.500	9.100	1.950	
Net2	Adaptive	3.000	8.000	1.670	37.225%
	S-Adaptive	3.200	8.900	1.990	
Net3	Adaptive	3.00	9.100	2.144	45.913%
	S-Adaptive	3.300	10.400	2.505	

the maximum link delay, the path length, the maximum path length and the computational cost. We used a route analysis program to compare between the results given by the two methods. The route analysis programme does comparisons between two methods, according to the routes obtained by them. It gives information about the average path lengths, the multiplicity and the path usage for each method.

Using both the adaptive and the self-adaptive penalty function methods we got results illustrated in tables (I) and (II) (in average).

The week correspondence is equal to zero for all the networks in all the simulations. The multiplicity and the path usage are equal for all the simulations between the adaptive and the co-evolutionary penalty functions.

V. CONCLUSION

In this work we used the adaptive and the co-evolutionary penalty function methods to solve the problem of minimizing the total time delay in the network. Preliminary experimental results revealed that the adaptive method is superior to the co-evolutionary method in terms of minimizing the total delay and the path length. But the co-evolutionary methods performs slightly better than the adaptive method in terms of the computational costs.

These preliminary results do not agree with what Coello stated about the superiority of the co-evolutionary method over the other penalty methods when solving optimization problems with only inequality constraints. Future research work using different test network models and investigating different other performance indexes will be achieved to asses the relevance

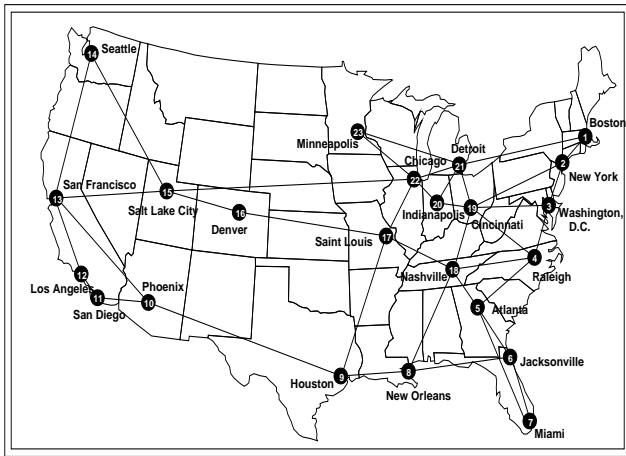


Fig. 2. USA (net1&net2)

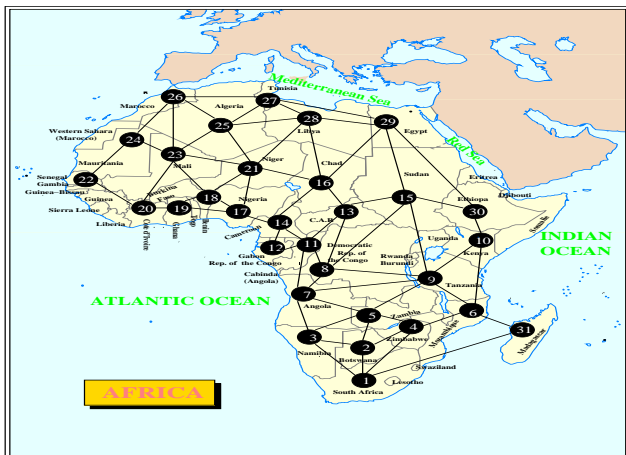


Fig. 3. PAREN (net3)

of our preliminary results.

REFERENCES

- [1] A.B. Hadj Alouane and J.C.Bean, "A genetic Algorithm for the Multiple Choice Integer Program", *Operations Research*, 45, 92-101, 1997.
- [2] A.B.Bagula and H.F.Wang,"On the Use of Genetic Algorithms to Fine-tune OSPF Routing", *South African Telecommunication Networks and Applications Conference*, Kuazulu Natal, South Africa, September 2005.
- [3] J.C.Bean and A.B.Hadj Alouane, "A Dual Genetic Algorithm for Bounded Integer Programs", technical Report TR 92-53, Department of Industrial and Operations Engineering, The University of Michigan, 1992.
- [4] C.A.Coello,"Use of a Self-Adaptive Penalty Approach for Engineering Optimization Problems", *Computers in Industry* ,41(2):113-127, January 2000.
- [5] C.A.Coello, "Theoretical and Numerical Constraint-Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art", *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245-1287, January 2002.