International Journal of Computing and Digital Systems

A Goal Programming based Extremal Optimization Algorithm for Topology Design of Enterprise Networks

[1]Salman A. Khan

[1] *Computer Engineering Department, College of Information Technology, University of Bahrain, Bahrain*

E-mail address: sakhan@uob.edu.bh

Received 13 Jan. 2014, Revised 28 Feb. 2014, Accepted 14 Mar. 2014, Published 1 May. 2014

Abstract: Extremal optimization is an optimization technique that has been applied to a number of complex optimization problems. One such optimization problem is topology design of enterprise networks. The problem involves simultaneous optimization of a number of objectives, such as financial cost, network latency, maximum number of hops between communicating nodes in the network, and network reliability, while considering various design constraints. This paper presents an extremal optimization algorithm to efficiently solve the topology design problem of enterprise network. The multi-objective attribute of the problem is handled by incorporating goal programming in the extremal optimization algorithm. Two variants of the extremal optimization algorithm are proposed and mutually compared. Furthermore, the second variant of extremal optimization algorithm, namely, the modified extremal optimization algorithm, is also compared with the particle swarm optimization algorithm. Empirical results suggest that the modified extremal optimization algorithm produced results of higher quality than the basic extremal optimization algorithm. However, a lower level of performance was observed for the modified extremal optimization algorithm when compared with the particle swarm optimization algorithm.

Keywords: Computational Intelligence, Extremal Optimization Algorithm, Goal Programming, Multi-objective Optimization, Enterprise Networks, Particle Swarm Optimization

1. INTRODUCTION

Extremal Optimization (EO)[1][2] is a general-purpose local search heuristic for solving complex NP-hard combinatorial optimization problems. The algorithm is based on the concepts of *self-organized criticality* [3] that are concerned with the dynamics of non-equilibrium processes in ecosystems. The development of EO was inspired by algorithms such as simulated annealing [4] or genetic algorithms [5], which used physical intuition to optimize. EO operates in an iterative fashion, and attempts to replace the most undesirable members of a sub-optimal solution with new, random ones in an attempt to reach an optimal solution [2].

Being a relatively new optimization algorithm compared to its many counterparts such as simulated annealing, genetic algorithms, tabu search, simulated evolution, stochastic evolution, and many others, EO has not been exploited well. This aspect is more prominent for problems in the discrete domain, augmented by the limited applications in the area of multi-objective optimization. Multi-objective optimization (MOO) aims at simultaneous optimization of several conflicting objectives, and a possible compromise between these objectives is found by evaluating these objectives [6]. The literature has reported few applications of EO to MOO problems in the discrete domain, which include the Knapsack problem [7], bin packing [8], quadratic assignment [9], RFID antenna design [10], graph bi-partitioning [11], graph coloring [12], production scheduling [13], and heat pipe design [14].

Like the above examples, many real-world optimization problems are multi-objective in nature. One such problem is topology design of distributed local area networks. A specific case of this problem is the design of enterprise networks. An enterprise network is a computer network that interconnects together a large number of nodes of an enterprise. These nodes are classified as *end-user nodes* and *network active elements*. End-user nodes consist of access points representing personal computers, workstations, network printers, mainframe computers, etc. Network active elements represent devices such as gateways, routers, switches, and hubs. Together with links, the active elements are responsible for providing the physical communication paths between each and every pair of end-user nodes. Several constraints dictate the network topology. Since the number of end-users in an enterprise is generally huge, the end-users are grouped into

smaller parts or group of nodes, called a Local Area Network [15]. The formation of a local area network is governed by geographical, physical, and technical objectives and constraints, which are, naturally, are also applicable to the enterprise network. Some of these objectives are the financial cost, network latency, hop count between a communicating pair of nodes, and network reliability. The enterprise network topology design problem requires simultaneous optimization of all these objectives, under a given set of design constraints. Attempts have been made earlier [16] - [25] to solve different variations of the above problem using optimization techniques such as simulated evolution, simulated annealing, ant colony optimization, and particle swarm optimization. This provides a motivation to apply the EO algorithm for the enterprise network topology design problem to see how does the algorithm perform for the same problem.

This paper is concerned with the design and analysis of a goal programming based multi-objective EO algorithm for topology design of enterprise networks. The rest of the paper is organized as follows. In Section 2, a brief background of the EO algorithm is provided. Section 3 discusses MOO with focus on goal programming approach for MOO. Section 4 briefly describes the multi-objective enterprise network topology design problem. A detailed description of the proposed goal programming based multi-objective EO algorithm is given in Section 5. A variant of the proposed multi-objective EO algorithm is also proposed in Section 5. Section 6 focuses on the performance evaluation of the two variants of the multi-objective EO algorithm, along with a comparison with a multi-objective particle swarm optimization algorithm [25] for the same problem. The paper ends with a conclusion in Section 7.

2. THE EXTREMAL OPTIMIZATION ALGORITHM

Extremal Optimization is a heuristic based on the Bak-Snappen model of co-evolution [26]. This model exhibits a phenomenon known as *self-organized criticality* which refers to as the capability of dynamical systems to organize themselves into optimal states. The EO algorithm iteratively operates on a single-solution and successively updates with the aim of removing the most undesirable element of the current solution, and replacing it with another random element. Furthermore, any change in the fitness value of an element results in a change in the fitness values of its neighboring element. Large fluctuations emerge dynamically, thus facilitating in efficient exploration of many local optima [2]. This signifies the strong local-search capability of the EO optimization algorithm.

A unique feature of the algorithm that makes it different from many other iterative optimization algorithms is the concept of fitness of an individual element. Many popular iterative heuristics such as genetic algorithms, simulated annealing, particle swarm optimization, ant colony optimization, tabu search, among many others, assess the fitness of a complete solution. In contrast, EO not only assesses the fitness of a solution, but also evaluates the fitness of all individual components (elements) within the solution. It is because of this individual component fitness that makes it possible for the EO algorithm to sort out the worst element and replace it with another one. Since the new element is chosen randomly, the updated solution may result in an inferior or a superior quality solution. This allows hill-climbing and valley-passing in the search space, thus avoiding pre-mature convergence, and allows convergence to optimal or near-optimal solutions. For further details on the EO algorithm, refer to [1][2].

3. MULTI-OBJECTIVE OPTIMIZATION AND GOAL PROGRAMMING

Multi-objective optimization is concerned with finding the best solution consisting of multiple objectives, from a pool of feasible solutions. A key feature of multi-objective optimization is that a tradeoff between several conflicting objectives needs to be found by evaluating these objectives [6], which makes multi-objective optimization problems very complex. This complexity is further intensified by the presence of a set of design constraints associated with the optimization problem. Therefore, it is essential for any optimization technique applied to solve these problems to ensure that all constraints are satisfied by the set of optimum solutions [27].

A number of approaches exist to solve multi-objective optimization problems. One such approach is known as scalarization (also called weighted aggregation), where the problem is converted into a single objective problem by aggregating all design objectives into a single objective function, which is then optimized by the underlying optimization algorithm. Some well-known scalarization methods are weighted sum method [28], fuzzy logic [29], and goal programming [6][30].

Goal programming is one of the earliest methods specifically developed for multi-objective optimization and decision-making [6]. In goal programming, the decision-maker specifies the targets, or goals, that need to be achieved. More specifically, the decision-maker sets aspiration levels (i.e. the ideal values of the objectives), Ti (i = 1,…, K), for the objective functions, and absolute

deviations from these aspirations levels are minimized as much as possible [6]. The simplest form of goal programming is formulated as [31]:

$$Minimize \sum_{i=1}^{K} |f_i(x) - T_i| \quad (1)$$

subject to $x \in S$

$g_m(x) \leq 0, \quad m = 1, \ldots, n$

$h_m(x) = 0, \quad m = n+1, \ldots, n+n_h$

In (1), the aim is to minimize the sum of the absolute deviations, subject to equality and inequality constraints (wherever applicable). Note that the deviations represent the differences between target values and actually achieved values [32]. Here, $x \in S$, where S is defined as the feasible region, $g_m(x)$ is the set of inequality constraints, and $h_m(x)$ is the set of equality constraints.

In majority of multi-objective optimization problems, the units associated with different objectives, as well as the respective magnitudes of the objectives, are different. Therefore, it is sensible to assume that due to incommensurability, deviations measured in different units cannot be summed directly, unless they are mapped onto a uniform scale. Therefore, each deviation is normalized to allow direct comparison. One common approach for normalization is to turn all deviations into percentages. Another popular approach requires mapping all deviations onto a zero-one range. This is done by computing the obtained deviation between the best and the worst possible values [33].

4. ENTERPRISE NETWORK TOPOLOGY DESIGN PROBLEM

Variations of the enterprise network topology design problem considering three and four design objectives have been extensively studied earlier using different algorithms, as mentioned in Section 1. The enterprise network topology design problem is concerned with obtaining a quality feasible tree topology while considering a set of design objectives and constraints. The purpose of this tree topology is to interconnect all nodes, which represent individual local area networks in the enterprise network, thus establishing a backbone topology of an enterprise network. The term "feasible topology" represents a solution that satisfies all design principles and constraints. On the other hand, a solution that optimizes all design objectives is referred to as "quality topology". In this paper, the quality of a topology is assessed based on four design objectives as follows:

- *Financial cost* is concerned with the cost of equipment incurred in designing the network and needs to be minimized. Cost is mainly dependent on the cost of the cable used to interconnect various local area networks in the enterprise network, as well as the cost of network devices therein.
- *Network latency* measures the average time a packet takes to reach a destination node from a source node. This factor consists of delays due to communication links as well as delays due to network devices, and should be minimized.
- *Maximum number of hops between any source-destination pair* also requires consideration due to technical issues. A hop is counted if a packet passes through network device. For example, as per the specifications of Routing Information Protocol, if a packet has more than 15 hops, then the destination is considered unreachable, and the packet is dropped from the network. Therefore, it is important to minimize the hop count.
- *Network reliability* refers to the probability of occurrence of an event in which each node is able to communicate with every other node in the network. A highly reliable network guarantees smooth communication. Therefore, network reliability should be maximized.

In addition to aforementioned design objectives, three design constraints are also considered. The first constraint dictates that the number of nodes attached to a network device should not exceed the number of ports on that device. The second constraint states that the flow of traffic on a link should not exceed a certain defined threshold of the link's capacity. Finally, the last constraint directs that the hierarchy of network devices must be maintained. For example, a hub cannot be connected on the top of a router or gateway, etc. In view of the above objectives and constraints, it is therefore very crucial for the proposed algorithm to target only feasible topologies, and ignore all solutions which do not fulfil the conditions set for the objectives or constraints. A detailed description on computation of objective values and constraints can be found in [20][21].

5. GOAL PROGRAMMING BASED EXTREMAL OPTIMIZATION ALGORITHM FOR ENTERPRISE NETWORK TOPOLOGY DESIGN PROBLEM

This section discusses the steps of the goal programming based EO algorithm. The proposed algorithm has three steps, namely, initialization, element fitness evaluation, and perturbation. These are discussed below. Fig. 1 shows the pseudo-code of the proposed EO algorithm.

A. Initialization

Since EO is single-solution based algorithm, a feasible initial solution is generated randomly. Alternatively, an already existing feasible solution can be provided as a starting seed.

B. Element Fitness Evaluation

As discussed in Section 2, each element of the solution is evaluated based on an evaluation function, and the elements are sorted in a chronological order, with the worst element being at the top of list for it to be removed probabilistically. It therefore provides a strong motivation to design an efficient fitness evaluation function for an element that would reflect the problem specific knowledge and insight. Note that the four design criteria

ALGORITHM Extremal Optimization
NOTATION
Φ= Complete Solution.
l_i= Individual link in Φ.
O_i= Lower bound on cost of i^{th} link.
C_i= Current cost of i^{th} link in Φ.
f_i= Fitness of i^{th} link in Φ.
Begin
INITIALIZATION: Generate a valid Φ
Repeat

FITNESS EVALUATION: **ForEach** $l_i \in \Phi$ **DO**
begin
$f_i = \frac{O_i}{C_i}$
end
PERTURBATION: **For link with worst** $f_i \in \Phi$ **DO**
begin
$PERTURB(l_i, \Phi_i)$
end

Until *Stopping Criterion is met*
Return Best solution.
End $(ExtremalOptimization)$

FIG. 1 STRUCTURE OF THE EXTREMAL OPTIMIZATION ALGORITHM

(objectives) are in turn based on the individual element values, and all these objectives are calculated based on the cost, latency, hops and reliability associated with each element in the topology. A suitable approach has been developed and successfully utilized in [21][24] which can also be adapted after necessary changes for the work proposed herein.

In the proposed EO algorithm, an element is a *link* that interconnects two network devices. The fitness of this link is defined based on three parameters, namely, financial cost associated with the link, optimum depth of the link (with respect to the root node), and the link's reliability. Consequently, the link's fitness measure has to be designed such that the above three parameters are efficiently accommodated. This can be done by finding the fitness of each parameter, and then aggregating them in a single fitness function. This can be achieved through the use of fuzzy logic [29] as follows. The three parameters, i.e., financial cost, optimum depth of a link (with respect to the root), and link reliability are considered fuzzy variables. Then, the fitness of a link is given by the following rule.

Rule 1: IF a link has *optimum cost* AND *optimum depth* AND *optimum reliability* THEN it has *ideal fitness*

Here, *optimum cost*, *optimum depth*, *optimum reliability*, and ideal fitness are linguistic values for the fuzzy variables cost, depth, reliability, and fitness, respectively. Using Werners aggregation function [34], Rule 1 translates to (2) for the fuzzy fitness measure, τ_i, of a link *i* as follows:

$$\tau_i = \gamma \min\{\mu_1(i), \mu_2(i), \mu_3(i)\} + \frac{1-\gamma}{3}\sum_{j=1}^{3}\mu_j(i) \qquad (2)$$

In Eq. (2), γ is a value between 0 and 1. $\mu_1(i)$, $\mu_2(i)$, and $\mu_3(i)$ represent memberships in the fuzzy sets optimum cost, optimum depth, and optimum reliability respectively. Further details about the formation of membership functions for each parameter can be found in [24].

C. Perturbation

In this phase, perturbations are made to the existing solution and fitness of the new solution is computed. This fitness is calculated based on the goal programming approach described in Section 3. For the work considered in this paper, the four objectives have different units and magnitudes. For example, the cost is in US Dollars, latency is in milliseconds, hops count is an integer, and reliability is a fraction between 0 and 1. Hence, it is necessary to normalize the four criteria to a uniform scale. For this purpose, the second normalization approach has been adopted, i.e., the criteria have been normalized to map onto a zero-one range, and the deviations from the ideal values were calculated and the overall deviation was found by adding the individual deviations. Table 1 shows the ideal values (targets) of each objective as used in previous studies [20] - [25] which are also adopted in this paper.

Once the elements (links) are sorted based on their fitness values, a link is selected probabilistically, and a new link is randomly introduced into the topology. Then, the fitness of the new resulting solution is assessed. If the fitness of this new solution is better than the fitness of the current best solution, then this new solution is accepted as the best solution found so far.

However, it is quite possible that the newly introduced link that replaced the worse link, may result in an infeasible solution (that is, the newly introduced link violates any of the design constraints). In this case, the old (worse) link is reinstated and no change happens to the current solution.

D. Modified extremal optimization algorithm

One potential drawback of the perturbation step in the basic EO algorithm discussed above is that if an infeasible solution is found in an iteration, then the algorithm will revert back to its previous state. If this phenomenon continues for a long time, then the algorithm will get stuck in local minimum, resulting in premature convergence. The probability of occurring of this phenomenon is high, since the enterprise network topology design problem is a highly constrained multi-objective optimization problem, with a very tight feasible search space. To avoid the above situation, a better approach is that, instead of trying one random link for each removed link, rather try multiple links randomly for each link selected for removal. More specifically, for each removed link, try as many multiple links as possible until a link that gives a feasible solution is found. This feasible link may be found after a single trial, or it may be found after multiple trials. A possible extremity is that a feasible link may never be found, in which case the removed link will be placed back in its position. This perturbation strategy will very likely increase the probability of getting out of local minima, thus enhancing the quality of the solution generated at the end.

6. RESULTS AND DISCUSSION

The proposed EO algorithm was tested on five test scenarios which have been used in previous studies [20]-[25]. In these test cases, the number of nodes (representing local area networks) consisted of 15, 25, 33, 40, and 50 nodes, denoted by N15, N25, N33, N40, and N50, respectively. For each test scenario, thirty independent runs were done, each starting with the same predefined initial solution. The average of the best solution found in each of these 30 runs was taken. Each run was made for 4000 solution evaluations. Statistical testing using Wilcoxon rank-sum test was used to validate the significance of the results at a confidence level of 95%.

A. Comparison of Extremal Optimization and Modified Extremal Optimization Algorithms

Table 2 shows the average overall deviation for the goal programming as obtained for MEO optimization and basic EO. It is very obvious from the table that, with the exception of N33, MEO was able to achieve lower deviations as compared to EO. However, statistical testing revealed that only the differences for N15 and N33 were statistically significant. Yet, it can be fairly claimed that MEO was able to show better performance than basic EO. This better performance of the modified version can be attributed to its better search capabilities than the basic version of extremal optimization, as explained in Section V-D.

Table 3 provides a more detailed insight of the performance of MEO and EO considering each individual objective. Columns 2, 3, and 10 of the table show that EO was able to achieve lower financial cost for four cases, with the exception of N50. However, only the difference in N33 was statistically significant. Even for the other three cases where EO got better results, the difference was minimal, less than 2.7%. Thus, it can be fairly said the EO had a slightly better performance than MEO with regard to the *cost* objective.

As far as *network latency* is concerned, columns 4, 5, and 11 of Table 3 shows that MEO was able to achieve exceptionally high improvements for majority of cases, which were also statistically significant, with one insignificant improvement for test case N25. The only exception where EO was able to achieve better results was for the test case N50. Therefore, it can be clearly noted that MEO had a superior performance over EO as far as network latency objective was concerned.

TABLE I. IDEAL VALUES OF TEST CASES USED IN EXPERIMENTS. COST IS IN US$, LATENCY IS IN MILLISECONDS, AND TRAFFIC IS IN MBPS.

TEST CASE	MINIMUM COST	MINIMUM LATENCY	MINIMUM HOPS	MINIMUM REL.	TRAFFIC
N15	4640	2.143	1	1.0	24.63
N25	5120	2.151	1	1.0	74.12
N33	8158	2.154	1	1.0	117.81
N40	9646	2.088	1	1.0	144.76
N50	11616	2.090	1	1.0	164.12

TABLE II. GOAL PROGRAMMING DEVIATIONS FOR EO AND MODIFIED EO FOR THE FIVE TEST CASES. % IMP SHOWS THE PERCENTAGE IMPROVEMENT ACHIEVED BY MODIFIED EO (MEO) COMPARED TO EO. STATISTICALLY SIIGNIFICANT IMPROVEMENTS ARE IN BOLDFACE.

TEST CASE	MEO	EO	%IMP
N15	4.418	4.509	**2.08**
N25	12.602	12.781	1.42
N33	18.081	17.457	**-3.45**
N40	28.013	28.483	1.67
N50	41.093	41.178	0.21

With regard to the *maximum number of hops between a source-destination pair*, columns 6, 7, and 12 of Table 3 reveals that both MEO and EO were able to get better results for two cases each, with MEO having better improvements for N15 and N25, and EO with N33 and N50. Although all four improvements were statistically significant, the extent of improvement achieved by MEO was higher than that of EO. There was one instance of N40 where both variants produced same results. Considering the aforementioned arguments, it can be comfortably claimed that MEO showed a better performance than EO for the *hops* objective.

Finally, for the *Reliability* objective, the results in columns 8, 9 ,and 13 of Table 3 suggest that EO was able to produce statistically significantly better results than EO for test cases N25, N40 and N50, while for the other two cases, MEO was able to produce statistically significantly better results. Therefore, it can be fairly claimed that EO had a slightly better performance than MEO.

TABLE III. RESULTS FOR EO AND MODIFIED EO FOR THE FIVE TEST CASES. % IMP. SHOWS THE PERCENTAGE IMPROVEMENT ACHIEVED BY MODIFIED EO (MEO) COMPARED TO EO. STATISTICALLY SIGNIFICANT IMPROVEMENTS ARE IN BOLDFACE.

TEST CASE	COST (US $)		LATENCY (MILLI SECONDS)		HOPS		RELIABILITY		% IMP COST	% IMP LATENCY	% IMP HOPS	% IMP RELIABILITY
	MEO	EO	MEO	EO	MEO	EO	MEO	EO				
N15	10070	9840	3.00	3.71	6	7	0.284	0.306	-2.28	**23.36**	**16.67**	**7.95**
N25	16956	16504	4.65	4.71	9	10	0.143	0.127	-2.67	1.32	**11.11**	**- 11.11**
N33	31279	28108	6.19	9.63	11	10	0.065	0.104	**-10.14**	**55.53**	**-9.09**	**60.57**
N40	34610	34220	5.52	8.43	10	10	0.051	0.037	-1.13	**52.58**	0.00	**-27.70**
N50	52372	52964	7.09	6.24	11	10	0.041	0.031	1.13	**-12.02**	**-9.09**	**-24.92**

Fig. 2 further elaborates on the results discussed above by providing the trends of EO and MEO for the four objectives with reference to the target (ideal) value. It can be seen that although both EO and MEO result in values much away from the target values, but the extent of these "far away" results in somewhat higher for EO compared to MEO. This is more obvious for the Latency objectives, and somewhat for the Hops objective, illustrated in Figs. 2(B) and 2(C) respectively.

Based on results in Tables 2 and 3, the results in Fig. 2, and the underlying observations and discussions, it can be concluded that, overall, MEO produced results of higher quality than EO. This is very obvious from the overall deviation, as well as from the results of *Latency* and *Hops* objectives, while for *Cost* and *Reliability* objectives, EO exhibited a slightly better performance than MEO.

B. Comparison of Extremal Optimization and Particle Swarm Optimization Algorithms

A comparison of EO was also done with the particle warm optimization (PSO) algorithm. PSO is a well-known optimization algorithm that has been applied to variety of optimization problems in various domains. PSO was originally proposed by Kennedy and Eberhart [35], and is inspired by the sociological behavior of flock of birds. The algorithm operates on a population of individuals, called particles, where each particle represents a valid solution. A particle has an adaptable velocity (position change), according to which it moves through in the search space. Each particle has the capability to remember the best position (solution) it has reached in its travel span. This behavior is termed as the cognitive component of the algorithm. In addition, all particles can also share their information about the search space, so there exists a
global best solution, whereas this particular behavior is known as the social component of the algorithm. The performance of the algorithm is also affected by a number of parameters, such as the swarm size, inertia weight w, acceleration coefficients c_1 and c_2, and velocity clamping V_{max}.

Swarm size, which represents the number of valid solutions per iteration, is an important factor in PSO. On the one hand, increasing swarm size generally results in increased computational complexity per iteration. On the other hand, higher swarm size favors higher diversity, and therefore, may require less iterations to converge [36]. Generally, an inverse relationship exist between the size of the swarm and the number of iterations needed to find the optimum of an objective function [36].

The inertia weight *w* is used to control the impact of the previous history of velocities on the current velocity [37]. This is done in order to manage the trade-off between global exploration and local exploitation carried out by the particles. A high value of *w* favors exploration which increasing diversity since new search areas are explored. A small value of *w* facilitates exploitation resulting in fine-tuning of the current search area.

The acceleration coefficients, c_1 and c_2, associated with the cognitive and social components have a strong impact on the convergence ability of the PSO. Variation in these parameters affect the pull towards the two best positions (i.e. personal best and neighborhood best). A detailed discussion on the effect of c_1 and c_2 can be found in [38].

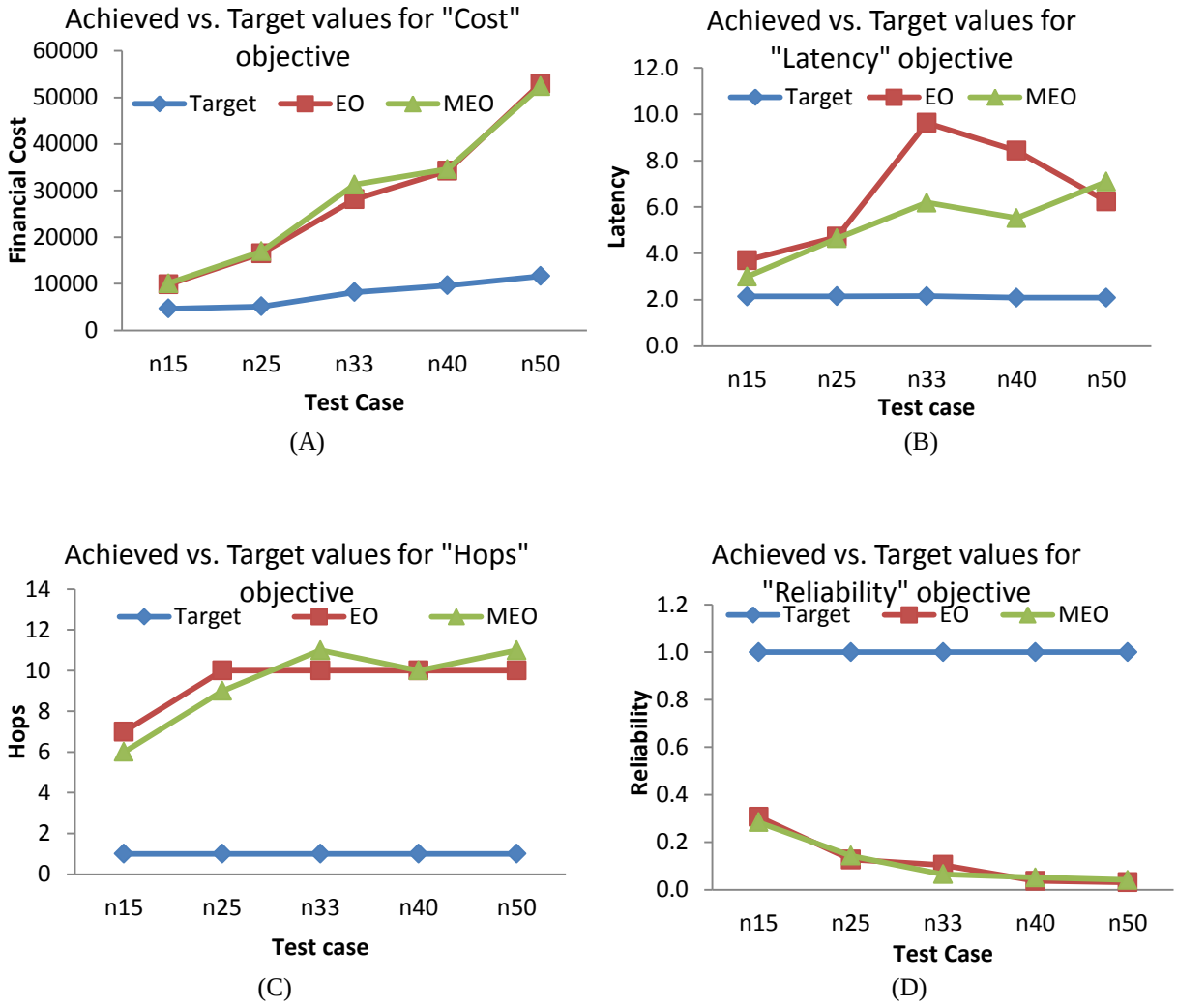

FIGURE 2. TARGET VERSUS ACHIEVED VALUES FOR EO AND MEO FOR (A) COST (B) LATENCY (C) HOPS AND (D) RELIABILITY.

Velocity clamping V_{max} is used to control the velocity of a particle by imposing a maximum value on it [39]. V_{max} restricts the step size which corresponds to the amount by which velocity is updated. This upper limit on step size prevents a particle from moving too fast from one region of the problem space to another, which may result in overshooting good regions of the search space. The value of V_{max} should be carefully chosen according to the structure of the problem [36].

A fuzzy-logic based particle swarm optimization algorithm was proposed in [25] for the topology design problem of local area networks, which used the same objectives, assumptions, and test cases considered in this paper. Therefore, it is logical to compare the performance of PSO with MEO. Table 4 provides the obtained values for each objective for MEO and PSO, along with the percentage improvements achieved by MEO compared to the PSO algorithm. It can be observed from the table that, overall, MEO showed inferior performance than PSO for all objectives. There are, however, some exceptions where MEO performed better than PSO, such as the test case N15, where MEO showed better performance with cost, latency, and hops objectives (all three of which were statistically significant), while for the reliability objective, the result were statistically insignificant, suggesting that both MEO and PSO achieved reliability of the same quality level. Another exception was the test case N40 with regard to the latency objective where MEO showed statistically better results than PSO.

The superior performance of PSO can be attributed to the fact that PSO is a population based algorithm, and therefore it has stronger search capability due to many particles exploring the search space. Furthermore, PSO has both exploration and exploitation capabilities, which enables it to search for localized as well as far away search areas. In contrast, MEO is single-solution based algorithm and has only a localized view of the search space, thus preventing it from efficient exploration of the search space.

TABLE IV. RESULTS FOR MODIFIED EO (MEO) AND PARTICLE SWARM OPTIMIZATION (PSO) FOR THE FIVE TEST CASES. % IMP. SHOWS THE PERCENTAGE IMPROVEMENT ACHIEVED BY MEO COMPARED TO PSO. STATISTICALLY SIGNIFICANT IMPROVEMENTS ARE IN BOLDFACE.

TEST CASE	COST (US $)		LATENCY (MILLI SECONDS)		HOPS		RELIABILITY		% IMP COST	% IMP LATENCY	% IMP HOPS	% IMP RELIABILITY
	MEO	PSO	MEO	PSO	MEO	PSO	MEO	PSO				
N15	10070	12191	3.00	4.76	6	11	0.284	0.273	**21.05**	**58.41**	**83.33**	- 4.01
N25	16956	14443	4.65	4.42	9	9	0.143	0.082	**- 14.82**	- 4.97	0.00	**- 42.30**
N33	31279	22138	6.19	5.58	11	10	0.065	0.060	**- 29.22**	**- 9.87**	**- 9.09**	**- 7.34**
N40	34610	28197	5.52	6.36	10	9	0.051	0.037	**- 18.53**	**15.20**	**- 10.00**	**- 27.67**
N50	52372	44604	7.09	5.36	11	9	0.041	0.025	**- 14.83**	**- 24.45**	**- 18.18**	**- 39.32**

7. CONCLUSION

This paper proposed and analyzed two variants of the Extremal Optimization (EO) algorithm engineered to solve the topology design problem of enterprise network. The problem is a discrete optimization problem with multiple objectives. Goal programming method was incorporated in the search process to handle the multi-objective nature of the problem. Preliminary results and analysis showed that the second variant, termed as the modified extremal optimization (MEO), produced results of better quality as compared to the basic EO. However, a comparison with particle swarm optimization algorithm revealed the deficiencies of the MEO algorithm in efficient exploration of the solution space.

The current and future research work is directed towards performing more rigorous analysis of the EO algorithm for the EN topology design problem, as well as on population-based EO incorporating some advanced features borrowed from other population based algorithms such as genetic algorithms, ant colony optimization, and particle swarm optimization. A population-based MEO will hopefully enhance the search capabilities of the algorithm, thus resulting in better quality solutions than what were currently obtained.

ACKNOWLEDGMENT

The author acknowledges the support of Deanship of Scientific Research at University of Bahrain for conducting this research under Project Grant 2013/5.

REFERENCES

[1] S. Boettcher. Extremal Optimization: Heuristics via Co-Evolutionary Avalanches. Computing in Science and Engineering, Vol 2, no. 6, pp. 75-82, 2000.

[2] S. Boettcher. Extremal optimization for graph partitioning. Physical Review E, Vol. 64, pp. 02611401 - 02611413, 2001.

[3] P. Bak, C. Tang, and K.Wiesenfeld. Self-Organized Criticality. Physics Review Letters, Vol. 59, pp. 381 – 384, 1987.

[4] S. Kirkpatrick, C. Gelatt Jr., and M.Vecchi. Optimization by Simulated Annealing. Science, Vol. 220, no. 4598, pp. 671-680, 1983.

[5] J. H. Holland. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, 1975.

[6] K. Miettinen. Some Methods for Nonlinear Multi-objective Optimization. 1st International Conference on Evolutionary Multi-Criterion Optimization, LNCS, Springer, pp. 1 - 20, 2001.

[7] P. G´omez-Meneses and M. Randall. Extremal optimisation with a penalty approach for the multidimensional knapsack problem. 7th International Conference on Simulated Evolution and Learning, pp. 229 - 238, 2008.

[8] P. G´omez-Meneses and M. Randall. A hybrid extremal optimisation approach for the bin packing problem. 4th Australian Conference on Artificial Life: Borrowing from Biology, pp. 242-251, 2009.

[9] P. G´omez-Meneses and M. Randall. A hybrid multi-objective extremal optimisation approach for multi-objective combinatorial optimisation problems. IEEE Congress on Evolutionary Computation, pp. 292-299, 2010.

[10] P. G´omez-Meneses and M. Randall. A multi-objective extremal optimisation approach applied to RFID antenna design. O. Schutze, C. Coello, A. Tantar, E. Tantar, P. Bouvry, P. Del Moral, and P. Legrand, Eds., Advances in Intelligent Systems and Computing, Springer, Vol. 175, pp. 431- 446, 2012.

[11] S. Boettcher and A. Percus. Nature's way of optimizing. Artificial Intelligence, Vol. 119, no. 1-2, pp. 275-286, 2000.

[12] S. Boettcher and A. Percus. Extremal optimization at the phase transition of the 3-coloring problem. Physical Review E, Vol. 69, page 066703, 2004.

[13] Y. Lu, M. Chen, and Y. Chen. Studies on extremal optimization and its applications in solving realworld optimization problems. IEEE Symposium on Foundations of Computational Intelligence, pp. 162 - 168, 2007.

[14] F. Luis de Sousa, V. Vlassov, and F. Ramos. Generalized extremal optimization for solving complex optimal design problems. Gentic and Evolutionaty Computation Conference (GECCO), pp. 375 - 376, 2003.

[15] C. Ersoy and S. Panwar. Topological Design of Interconnected LAN/MAN Networks. IEEE Journal on Selected Area in Communications, pp. 1172 -1182, 1993.

[16] H. Youssef, S. Sait, and S. Khan. Fuzzy Simulated Evolution Algorithm for Topology Design of Campus Networks. IEEE Congress on Evolutionary Computation, pp. 180 - 187, 2000.

[17] H. Youssef, S. Sait, and S. Khan. Fuzzy Evolutionary Hybrid Metaheuristic for Network Topology Design. IEEE/ACM First International Conference on Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, Springer-Verlag, pp. 400 - 415, 2001.

[18] H. Youssef, S. Sait, and S. Khan. Topology Design of Switched Enterprize Networks using a Fuzzy Simulated Evolution Algorithm. Engineering Applications of Artificial Intelligence, Vol. 15, pp. 327 - 340, 2002.

[19] H. Youssef, S. Sait, and S. Khan. A Fuzzy Evolutionary Algorithm for Topology Design of Campus Networks. Arabian Journal for Science and Engineering, Vol. 29, no. 2b, pp. 195 - 212, 2004.

[20] S. Khan and A. Engelbrecht. A New Fuzzy Operator and its Application to Topology Design of Distributed Local Area Networks. Information Sciences, Vol. 177, no. 12, pp. 2692 - 2711, 2007.

[21] S. Khan and A. Engelbrecht. Fuzzy Hybrid Simulated Annealing Algorithms for Topology Design of Switched Local Area Networks. Soft Computing, Vol. 3, no. 1, pp. 45 - 61, 2009.

[22] S. Khan and A. Engelbrecht. A fuzzy ant colony optimization algorithm for topology design of distributed local area networks. IEEE Swarm Intelligence Symposium, pp. 1 - 7, 2008.

[23] S. Khan and A. Engelbrecht. Application of ordered weighted averaging and unifed and-or operators to multi-objective particle swarm optimization algorithm. IEEE 6th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 176 - 180, 2009.

[24] S. Khan and A. Engelbrecht. Assessment of the .Evaluation. function in the simulated evolution algorithm. IEEE 7th International Conference on Natural Computation, pp. 1062 - 1066, 2011.

[25] S. Khan and A. Engelbrecht. A fuzzy particle swarm optimization algorithm for computer communication network topology design. Applied Intelligence, Vol. 36, pp. 161 - 177, 2012.

[26] P. Bak and K. Sneppen. Extremal optimization at the phase transition of the 3-coloring problem. Physical Review Letters, Vol. 71, pp. 4083 - 4086, 1993.

[27] J. Wright and H. Loosemore. An Infeasibility Objective for Use in Constrained Pareto Optimization. 1st International Conference on Evolutionary Multi-Criterion Optimization, LNCS, Springer, pages 256 - 268, 2001.

[28] S. Gass and T. Saaty. The Computational Algorithm for the Parametric Objective Function. Naval Research Logistics Quarterly, Vol. 2, pp. 39 - 45, 1955.

[29] L. Zadeh. Fuzzy Sets. Information Contr., Vol. 8, pp. 338 - 353, 1965.

[30] A. Charnes and W. Cooper. Management Models and Industrial Applications of Linear Programming. John Wiley, 1961.

[31] L. Duckstein. Multiobjective Optimization in Structural Design: The Model Choice Problem. North-Holland, Amsterdam, 1984.

[32] C. A. Coello Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. Knowledge and Information Systems, Vol. 1, no. 3, pp. 269 - 308, 1999.

[33] C. Romero. Handbook of critical issues in goal programming. Pergamon Press, 1991.

[34] B. Werners. An interactive fuzzy programming system. Fuzzy Sets and Systems, Vol. 23, pp. 131 - 147, 1987.

[35] J. Kennedy and R. Eberhart. Particle Swarm Optimization. IEEE International Conference on Neural Networks, pp. 1942 - 1948, 1995.

[36] J. Kennedy and R. Eberhart. The Particle Swarm: Social Adaptation in Information Processing Systems. D. Corne, M. Dorigo, F. Glover, Eds., New Ideas in Optimization, McGraw-Hill, pp. 379 - 387, 1999.

[37] Y. Shi and R. Eberhart. Parameter Selection in Particle Swarm Optimization. V. W. Porto, N. Saravanan, D. Waagen, and A. Eiben, Eds., Evolutionary Programming VII, pp. 611 – 616, 1998.

[38] A. P. Engelbrecht. Fundamentals of Computational Swarm Intelligence. John Wiley & Sons, 2005.

[39] R. Eberhart, P. Simpson, and R. Dobbins. Computational Intelligence PC Tools. Academic Press, 1996.

Salman A. Khan received M.S. in Computer Engineering from King Fahd University of Petroleum & Minerals, Saudi Arabia in 2000 and the PhD degree in Computer Science from University of Pretoria, South Africa in 2009. He is currently an Assistant Professor in the Computer Engineering Department at University of Bahrain, and an Adjunct Senior Researcher with Computational Intelligence Research Group, Computer Science Department, University of Pretoria. He has published over 30 research articles in reputed journals and conferences. His research interests include Evolutionary Computation, Swarm Intelligence, Nature-inspired Algorithms, Fuzzy Logic, Single-objective and Multi-objective optimization and decision-making, Computer Networks, and Mobile Communication Systems. He serves as a reviewer for various reputed journals and conferences annually. He is a member of IEEE, IEEE Computational Intelligence Society, IEEE Systems, Man, and Cybernetics Society, and ACM Special Interest Group on Evolutionary Computation.