Genetic Algorithm, Extremal Optimization, and Particle Swarm Optimization Applied to the Discrete Network Configuration Problem

M. Martin, E. Drucker, and W.D. Potter
Artificial Intelligence Center, University of Georgia, Athens, GA, USA

Abstract - *Genetic Algorithm, Extremal Optimization, and Particle Swarm Optimization are applied to the discrete (integer-based) communications network configuration problem. In this preliminary study, each heuristic search scheme is used to determine the optimal (or near optimal) set of communications equipment components needed to satisfy user-specified radio subscriber and wire subscriber requirements. The network configuration problem is based on the U.S. Army's Mobile Subscriber Equipment communications networking system. Although MSE is no longer a major asset in the U.S. Army's communications inventory, it continues to provide an excellent platform for configuration optimization due to its non-permutation yet discrete nature. The comparison of the heuristics includes their overall reliability as well as the number of fitness evaluations needed to converge on the optimal solution. General results indicate that all approaches achieve relatively high reliability using some novel operators and adaptations, but the GA does so using considerably fewer fitness evaluations.*

Keywords: GA, PSO, Extremal Optimization, Network Configuration.

1 Introduction

The need for reliable communications support is of paramount importance, as evidenced by the rapid growth of mobile telephone users and infrastructure all over the world during the past several years. Possessing a reliable means of communication is especially important for military and emergency services operations. In the past, military mission planners would spend many hours developing various scenarios that incorporated all aspects of the mission. The communications aspect of a mission - the development of a battlefield communications network that supports both wire and radio telephone communications requirements - is a particularly difficult task due to the numerous dimensions involved, including the types of components and their quantities.

In the early 1990s U.S. Army communications networks were configured using Mobile Subscriber Equipment, which provided communications support for a typical five-division corps in an area of up to 15,000 square miles. The communications system was characterized as an area-switched system that provided both voice and data communications. An MSE network supported radio subscribers using VHF (Very High Frequency) radio links, and supported wire subscribers using hardwired links, connectivity to other NATO forces, and connectivity to commercial networks.

The backbone of an MSE network was composed of up to 42 Node Centers. Attached to the backbone were various types of Extension Nodes, Control Centers, NATO Interfaces, and Remote Access Units. The differences between the various types of Extension Nodes were directly related to the number of wire subscribers that could be supported and the actual fashion in which they were supported. A Large Extension Node could support nearly 200 wire subscribers. Version 1 of a Small Extension Node could support just under 30 wire subscribers and a Version 2 could support just over 40 wire subscribers. Radio subscribers were supported by Remote Access Units with a capacity of up to 50. Rounding out the completion of a network configuration were the other various components such as NATO interface units and System Control Centers.

Configuring an MSE network amounted to determining which components to use for a specified mission, and how many of each component to use [15]. In the simplest case, the mission requirements identified only the number of wire subscribers and mobile subscribers to support using the network. Other dimensions of a mission requiring consideration included the logistics of getting the equipment to the operating theatre and equipment placement within the operating theatre. We do not address these additional issues here but rather focus our attention on the "shopping list" of components needed for the mission. The shopping list is a sequence of integers corresponding to the number of components needed for the mission, with each position in the list corresponding to a different component. Consequently, we have a discrete integer sequence problem representation which can be used directly by each of our heuristic search schemes.

The heuristic search algorithms that we investigate include the genetic algorithm (GA), the extremal optimization (EO) algorithm, and the particle swarm optimization (PSO) algorithm [3, 7, 9, 11]. The GA is quite likely the most well-

known of these approaches and has been used on a large number of difficult optimization problems, including both experimental test case problems and real-world applications. PSO has become popular over the past several years due to its ease of use, speed of convergence, and relatively high reliability. Extremal optimization may be considered to be the "new kid on the block". It uses a somewhat counter-intuitive approach to reach convergence, but nonetheless has enjoyed some success in its application to various difficult problems [4, 3].

There are some issues related to the heuristics and the representation used in our configuration problem that should to be mentioned so that the reader has a feel for this application. For example, PSO was not designed for discrete problems but rather for continuous problems. However, discrete versions of PSO have shown promise [7, 11]. It is common for problems using a binary solution representation (or even a Boolean representation) to be identified as discrete in the PSO literature. However, integer sequence representations also fall within the discrete category. Another issue is the underlying nature of how each heuristic uses the fitness function to explore the search space. With EO, each component of a solution influences the convergence process individually whereas with the GA and PSO, the entire individual influences the exploration. More details of each of these heuristics are presented in the following sections.

This paper is organized in the following manner. The next section provides a brief discussion of the discrete network configuration problem as well as some general networking issues. Then, we present the various heuristic schemes used (GA, EO, PSO). This is followed by discussions of the experimental setup and results. Finally, we present some observations and conclusions regarding this work, and possible future directions of research.

2 Background

During mission planning, a planner attempts to configure a battlefield communications network using a set of available MSE components (such as Node Centers, various capacity Extension Nodes, Control Centers, NATO interface nodes, and Radio Access Units). The network must function properly, support the current mission, and satisfy other configuration constraints (such as constraints imposed by U.S. Army doctrine, the terrain, and certain limitations of the components, such as proper connectivity). The planner is typically someone with extensive experience with mission planning and network configuration [8, 10, 15].

Configuring an optimal network from various differing components and component quantities is the type of problem that is affected by combinatorial explosion; as the number of network components increases, the number of possible network configurations increases drastically. Using a familiar example, N network components may be configured into $N!$ networks (having N components in each network). This situation worsens if we allow varying network sizes.

Because of the combinatorial explosion in the number of possible networks, configuration algorithms that are guaranteed to find the optimal network structure via an exhaustive or near exhaustive search are of little value when dealing with larger networks because of their extensive runtimes. Since a typical MSE five-division corps contains over 40 Node Centers (not to mention the other types of components), it is clear that we need a good heuristic approach to solve the MSE configuration problem.

3 Early Work

In the development of a good heuristic approach, two knowledge-based system approaches are available to us. These are the rule-based approach using typical if-then rules, and the functional approach based on knowledge about the structure and behavior of the network and its components [13, 5]. Our method could be considered a combination of the rule-based and functional paradigms. That is, although we do not have a typical collection of if-then rules that are manipulated in some fashion, expert rules-of-thumb are incorporated into each heuristic's fitness function in order to evaluate the quality of a particular network configuration. Network configuration means incorporating a given set of network components and a set of constraints associated with the components (and network) into a network that satisfies the mission goals and constraints.

Probably the most famous system to be developed for design/configuration applications is R1 (XCON) which is used to configure computer systems from customer specifications [12, 1]. An early example of a network design system using heuristics is DESIGNET, developed by Bolt, Beranek, and Newman in the early 1980's. DESIGNET is a rule based design aid that focuses on an iterative user interface approach to network configuration based on the process a designer goes through during the design process [6].

Early examples of design aids for network configuration include MAPCon, ELAND, and a system for packet radio network design developed by Ruston [14, 16]. MAPCon, developed at the Industrial Technology Institute in Ann Arbor, Michigan, configures communications networks that use the Manufacturing Automation Protocol. However, MAPCon focuses on accurate component parameter settings rather than components, and quantities. ELAND, developed in Europe, configures distributed information systems operating on a local area network (ELAND stands for Expert Local Area Network Designer). ELAND is slightly different from the other network design systems mentioned here in that it focuses

on the specific type of LAN to install, the computer systems needed, and the software needed to support a user's set of networking requirements. The design system developed by Ruston focuses on the design of high-performance packet radio networks. It proposes changes to the network's parameters that, hopefully, will improve the performance of the network.

4 The Genetic Algorithm

The Genetic Algorithm models the process of evolution by natural selection using techniques inspired by biological processes. The driving force of natural evolution is selection pressure, which is applied by competition between organisms for passing on their genetic material. In the GA, potential solutions are treated as individuals in such a population, and pressure is exerted through the use of selection schemes which choose individuals for mating. For this paper we chose *k*-ary tournament selection, where *k* individuals are selected at random and the competitor with the highest fitness goes on to mate. When two individuals are selected, they exchange genetic material through a process known as crossover. In this paper we use the most common crossover operation, point crossover, where a random point within the representation is selected to divide genetic material from the parents. Crossover combines elements of separate individuals to exploit good areas of the fitness landscape. The newly created solutions then undergo mutation, which introduces an element of randomness. For this application, a mutation operator was devised to modify the values of each number in the representation based on the time the algorithm had been running. When an element of an individual undergoes mutation, a random number is generated within the range for that individual and then divided by the "age" of the individual (the number of times its genetic material has been crossed over). This allows the algorithm to explore the search space more freely early on, but avoid disrupting good solutions later in the run.

5 Extremal Optimization

EO is a recently developed local search heuristic [3] based on the Bak-Sneppen (BS) model of biological evolution [2]. The BS model demonstrates self-organized criticality, a tendency for systems in statistical physics to organize themselves into non-equilibrium states. Unlike most optimization techniques, EO focuses on removing poor components of solutions instead of favoring the good ones. Also, instead of evolving populations, EO performs updates on a single solution, similar in this fashion to Simulated Annealing.

The EO heuristic is performed on a single solution candidate S as follows:

1) generate an initial value for each component of S, $S_{best} = S$
2) determine a fitness associated with each component of S
3) rank the components by fitness and select the worst to be updated
4) assign the selected component a new value with a random fitness to create S'
5) if the fitness of S' is higher than the previous best, S_{best}, replace S_{best} with S'
6) set S equal to S'
7) repeat steps 2 through 6 a predetermined number of times

Unfortunately, in some domains, step 3 can cause the heuristic to become stuck in local optima. To accommodate for this, Boettcher and Percus introduced a power-law distribution approach for selecting the component to be modified [3, 4]. To select a component using this approach an integer $1 \leq k \leq N$ is determined from the probability function:

$$P(k) \propto k^{-\tau} \tag{1}$$

where N is the number of components in the configuration, and τ is a constant that determines how stochastic or deterministic the selection should be [4]. With higher values of τ, EO is more likely to get stuck in local optima for the same reasons it does without a selection function at all. For lower values of τ, EO sometimes replaces better components of a solution in order to explore a larger part of the search space. If τ is set to zero, EO produces similar results to random search. Because of the amount of randomness involved, EO can jump in and out of near-optimal solutions at any point in time.

6 Particle Swarm Optimization

Particle Swarm Optimization models the natural behavior exhibited by various large groups of animals such as flocks of birds or schools of fish. For birds, the movement of the flock is orchestrated by a combination of influences based on the speed and direction of the lead bird, and the speed and direction of an individual bird's immediate neighbors. Translating the metaphor to solutions within a search space of solutions, a small population of solutions can be "guided" towards the globally optimal solution based on the influence of: (1) the best member ever encountered, and (2) the "path" through the search space of each member of the population. The population "moves" in discrete time steps through the search space based on the following speed and direction characteristics for each member: the difference between the location of the best member and each member's location, ΔGB; the difference between a member's best location and its current location, ΔPB; a social influence:

$$c_1 * Rnd * \Delta PB \tag{2}$$

a cognitive influence:

$$c_2 * Rnd * \Delta GB \tag{3}$$

the new velocity: inertia * old velocity + social influence + cognitive influence, yet limited by V_{min} and V_{max}; and the new location: old location + Int(new velocity). Note that Rnd is a random number on (0,1], and to make this standard PSO scheme a discrete scheme we truncate the new velocity value in order to maintain the integer sequence representation. Note that if a network component's value goes outside of its bounds, it will be reset to the nearest bound.

7 Strength of an Individual

As we have seen, our approaches use various models of natural processes to converge on an optimal solution. As with any heuristic approach, these are not guaranteed to find the optimal solution. However, due to their functional nature, they typically find the optimal or near-optimal solution. The primary driving force behind the GA, EO, and PSO is the mechanism that determines the strength of each individual or particle, called a fitness function.

Our MSE fitness function takes into consideration several important issues in determining the "quality" of an individual configuration. These issues include the mission requirements, the known support capacities of all the various components, the Division and Corps equipment complement, and the known minimum component requirements into consideration while determining the fitness function value. In addition, issues such as the combined total number of components, and connectivity doctrine are included to help distinguish competing individual solutions.

8 Experiment Setup & Result Summary

The GA was run using the age-proportional mutation operator described above, due to early trials showing the use of naïve mutation to be infeasible. Trials indicated that crossover rate did not have a major effect on results, so trials were run on population size (300, 400, 500) and mutation rate (5%, 10%, 15%), with 1,000 runs for each unique setting. Stopping criteria was ten generations with no improvement in the best fitness. The GA ran for an average of 34 generations and slightly less than 14,000 total fitness evaluations. In every case, increasing population size and mutation rate increased reliability, so only results using a mutation rate of 15% are considered here. The most accurate trial was run with a population size of 500 (GA500), giving a reliability of 99.6% (the best solution was found 996 times in 1000 trials), with the average number of fitness evaluations before finding the optimum at 12,000. Decreasing the population size to 400 (GA400), the reliability was barely affected, dipping to 99.1%, and the fitness evaluations for the best solution decreased to 9,200. When run with a population size of 300 (GA300), reliability decreased to 98.1%, with the GA finding the optimal solution after 6200 fitness evaluations.

The EO algorithm was set up to evaluate each component of the MSE component list. Each component was assigned a penalty according to each goal of the objective function. Since some components could be affected by more goals than others, the penalties were normalized before the components were ranked. Once ranked, a power-law distribution with τ equal to 4 was used to select the component to be modified, and the selected component was assigned a number between 0 and the maximum possible value. An initial trial using this as the only evaluation criteria resulted in a reliability of less than 10% on 1,000 runs of 10,000 EO iterations each. The results of the original experiment demonstrated a need for a less drastic change in the numbers of components, because the heuristic would sometimes become stuck. To improve the component evaluation function, the fitness of each node was evaluated further by running the solution fitness function twice; once after increasing the component number by one, and once after decreasing it. The resulting value was then assigned a weight, and each node was assigned a direction in which it should be modified. If conflicting directions were assigned to a component, or if no direction was assigned, a new random value was assigned to the component if it was selected. Using directions, an innovative adaptation of the EO approach, reliability was drastically improved to 92% on 1,000 runs.

The EO algorithm was then run 1,000 times for each of 3,000 parameter settings. The parameters were τ (0 to 10 in increments of 0.01) and number of iterations (1,000, 5,000, 10,000). The best value of τ was found to be in the range of 4.94 to 5.03. Using τ equal to 5, the EO algorithm was run on a range of set iterations (100 to 20,000 in increments of 100), and each set number of iterations was run 1,000 times. The maximum reliability reached was 100% for 19,700 iterations (EO19,700). On average, EO found the best solution after 4,000 iterations, and 57,300 fitness evaluations. Limiting the number of iterations to 10,000 (EO10,000), cutting run-time in half, produced a reliability of 92.2% with the best solution found after an average of 3,200 iterations and 45,700 fitness evaluations.

After several trial runs to narrow down reliable values for the social term (C_1) and cognitive term (C_2), the PSO algorithm was run a thousand times for each of 8 different parameter settings. The parameters included: swarm size (100, 300); C_1 (1.5) and C_2 (4.0); inertia (0.4 to 0.9 with 0.1 steps); maximum allowable generations set to 300; a stopping criteria of no improvement for 20 consecutive generations; and V_{min} set to (-6), and V_{max} set to (6). A reliability of 99.85% was achieved with a swarm size of 300 individuals (PSO300), C_1 set to 1.5, C_2 set to 4.0, and inertia varying from 0.4 to 0.9 (0.1 step size). For six different inertia

settings per run and each run repeated 1,000 times, PSO found the optimal solution 5,991 out of 6,000 times (note that the second best solution was found three times and the third best solution was found twice). In the PSO300 trials, the average number of fitness evaluations was 31,500. Reducing the swarm size to 100 (PSO100) resulted in an expected lower reliability of 98.63%. It was expected that the reliability would decrease, but surprisingly it only dipped one percentage point. This was achieved with C_1 set to 1.5, C_2 set to 4.0, and inertia varying from 0.4 to 0.9 for a total of 6,000 runs, of which we found the best solution in 5,918 runs (the second best solution was found 13 times and the third best solution was found only once). For this experiment, we found that PSO required an average of 11,800 fitness evaluations per run to find the optimal solution.

Table 1
Comparison of heuristic results

Heuristic	Reliability	Fitness Evaluations	Ratio
GA500	99.60%	12,000	8.3000×10^{-3}
GA400	99.10%	9,200	1.0772×10^{-2}
GA300	98.10%	6,200	1.5823×10^{-2}
EO19,700	100.00%	57,300	1.7450×10^{-3}
EO10,000	92.20%	45,700	2.0180×10^{-3}
PSO300	99.85%	31,500	3.1700×10^{-3}
PSO100	98.63%	11,800	8.3580×10^{-3}

Table 1 shows the reliability and fitness evaluations for finding the optimum for the ideal heuristic settings described above. Also shown is the ratio of the reliability percentage to the average fitness evaluations required to achieve it. As can be seen, though PSO achieved higher reliabilities with lower population sizes compared to the GA, it required more fitness evaluations to find the optimum. While EO required drastically larger amounts of fitness evaluations due to running two fitness calculations for each element of a solution in each iteration, it was the only search method to achieve 100% reliability on any parameter setting.

9 Observations and Conclusions

By running trials to find ideal parameter settings, all three search techniques were able to reliably find the optimal network configuration. GA and PSO, the more similar of the three algorithms, both saw a boost in reliability when their population size was increased, but this also caused the number of fitness evaluations necessary to find the optimal configuration to increase. In these cases, the best approach would often be to run the less reliable configuration more times, due to the lower number of fitness evaluations required for this approach. This approach would be particularly important in more complex configuration problems where fitness calculations were more expensive. Further research into more specialized operators and strategies for these heuristics, such as the age-proportional mutation operator used, could allow the population sizes to be reduced further for faster results.

Though the EO procedure only operates on one potential solution, it required a much larger number of fitness evaluations to find the optimum due to its component evaluation procedure. This evaluation procedure is the area of the algorithm that would benefit most from further research and development. Procedures that could evaluate a solution without running the entire fitness function for each component would be far less computationally expensive, and more competitive with other heuristics. EO shows great promise as it was the only algorithm to achieve 100% reliability due to the novel direction-based adaptation, and if its component evaluation procedure could be simplified, its complexity would decrease by an order of magnitude.

EO and PSO, the newer strategies, both performed well in terms of reliability when compared to the baseline GA. Though neither reached the same ratio of reliability to fitness evaluations achieved by the GA, both show great promise, and encourage investigation into new variants and adaptations similar to those used in this paper. All three strategies achieve a sufficiently high reliability for finding the ideal configuration for this problem to be considered for use in other complex discrete problem areas.

10 References

[1] Bachant, J., and J. McDermott (1984). "R1 Revisited: Four Years in the Trenches," in AI Magazine, Vol. 5, No. 3.

[2] Bak, P. and K. Sneppen (1993). "Punctuated Equilibrium and Criticality in a Simple Model of Evolution," in Physical Review Letters, Vol. 71, No. 24, pp. 4083-4086, December.

[3] Boettcher, S. and A.G. Percus (1999). "Extremal Optimization: Methods derived from Co-Evolution," in GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, San Francisco, CA, pp. 825-832.

[4] Boettcher, S. and A.G. Percus (2001). "Extremal Optimization for Graph Partitioning," in Physical Review E, Vol. 64, 026114.

[5] Ceri, S., and L. Tanca (1990). "Expert Design of Local Area Networks," in IEEE Expert, Vol. 5, No. 5, pp. 23-33, October.

[6] Cronk, R.N., P.H. Callahan, and L. Bernstein (1988). "Rule-Based Expert Systems for Network Management and Operations: An Introduction," in IEEE Network Magazine, Vol. 5, No. 2, pp. 7-21, September.

[7] Engelbrecht, A. P. (2005). Fundamentals of Computational Swarm Intelligence, Wiley and Sons, Ltd., England.

[8] Gillis, P.D., and R.W. Pitts (1991). "A C++ Frame System for Decision Aiding and Training for MSE," in AI Exchange, Vol. V, No. 2, pp. 10-12, Office of Artificial Intelligence Analysis and Evaluation.

[9] Goldberg, D.E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Co.

[10] Hiebert, L. (1988). "AI and Network Planning," in AI Expert, Vol. 3, No. 9, pp. 26-33, September.

[11] Kennedy, J., and R. C. Eberhart (2001). Swarm Intelligence, Morgan Kaufmann Publishers.

[12] McDermott, J. (1981). "R1: The Formative Years," in AI Magazine, Vol. 2, No. 2.

[13] Mittal, S., and F. Frayman (1989). "Towards a generic model of configuration tasks," in Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, Vol. 2, pp. 1395-1401, August.

[14] Parunak, H.V.D., J. Kindrick, and T. Toth-Fejel (1990). "MAPCon: An Expert System to Configure Communications Networks," in Proceedings of the Fifth Conference on Artificial Intelligence Applications, pp. 23-28, March.

[15] Potter, W.D., R. Pitts, P. Gillis, J. Young, and J. Caramadre, "IDA-NET: An Intelligent Decision Aid For Battlefield Communications Network Configuration," in the Proceedings of the
Eighth IEEE Conference on Artificial Intelligence Applications (CAIA'92), pp. 247-253, March 6, 1992.

[16] Ruston, L., and P. Sen (1989). "Rule-Based Network Design: Applications to Packet Radio Networks," in IEEE Network Magazine, Vol. 3, No. 4, pp. 31-39, July.