Contact author prior to further distribution

A Genetic Algorithm Model for Mission Planning and Dynamic Resource Allocation of Airborne Sensors

10-March-1999

Stephen W. Soliday
Raytheon Systems Company
Sensors and Electronic Systems
6620 Chase Oaks Blvd. M/S 8518
Plano, Texas 75023 USA
soliday@raytheon.com

ABSTRACT

Genetic Algorithms (GA) have been very successful in combinatorially–explosive problems such as; Job Shop Scheduling, Non-linear Transportation Model, and the Traveling Salesman Problem. The problem of mission planning for large numbers of airborne platforms contains elements of all three of these classic GA problems. This paper will outline the application of genetic algorithms to mission planning and dynamic allocation of airborne sensors. An experiment will consist of a mission simulation containing (n) airborne sensors in different locations and states of readiness, and (m) requests for imagery with varying mission priorities. The GA will match sensors with requests in order to minimize the cost and time–line and maximize the execution of high priority requests. Results of simulation will be discussed.

1 Introduction

The task of scheduling multiple collections of sensors under diverse constraints is a combinatorial explosive task. Most calculus based optimization routines are not suitable for such tasks. This paper describes the intermediate results of an IDEA program.[1] The program purposed the use of Genetic Algorithms for various mission planning tasks. Mentioned in particular was the task of scheduling Unmanned Airborne Vehicles (UAVs). Section 2 will describe this problem in specific detail. The general task is to match UAVs with requests in order to minimize the total time line and to maximize the number of high priority targets. The system must be able to schedule the UAVs and then reschedule them in the event that some are lost or new higher priority requests arrive after the mission begins.

2 Problem

The problem being studied in this paper is that of allocating multiple collection sensors. Specifically it is, the problem of determining the paths of multiple UAVs in order to complete a number of imagery requests in a constrained environment.

The number of requests was much greater than the number of UAVs. A maximum time was given to complete the mission. This time could be necessary for a variety of reasons. There could be a time set for a major engagement. The time could also reflect the maximum duration of fuel for the UAVs. The imagery could be requested for *Time Critical Mobile Targets.* Each mission objective had an associated priority, or mission value. The positions of the UAVs and the mission objectives were maintained in a two–dimensional map.

In order to define the scope of this study some assumptions were made. It was assumed that the UAVs were homogeneous, each one flew at the same speed, had the same duration, and the same operating costs. Further it was assumed that while the individual mission objectives had differing priorities, time to complete an objective was only governed by final mission time. See Section 5.1 for future enhancements to this study.

There were multiple factors that governed the question: "*What makes a good path or schedule?*" A Measure Of Effectiveness (MOE) was defined to compare sample solutions in order to determine which ones were better. If there were more mission objectives than time permitted in a given path, then the schedules containing paths with more high priority targets were judged better. If all mission objectives were met, then the schedule with the shortest completion time was judged to be the better schedule. Finally if the overall completion times were equal between two schedules then the schedule with the smallest variance in its path lengths was judged to be better.

3 Technical Approach

The problem described in Section 2 involved scheduling under a complex combination of constraints. The solution to such a problem is NP-Complete. This means that the number of trial solutions that must be examined in an exhaustive search grows as an n^{th} degree polynomial with

[1] IDEA Programs are set up in Raytheon to give engineers limited individual funding to pursue ideas in order to improve existing processes

respect to the order [DJS89]. Such problems are difficult at best and usually intractable for traditional calculus based optimization algorithms. For this reason it was decided to apply a genetic algorithm to the problem.

A Hybrid Genetic Algorithm was designed for this effort. The simple GA is not sufficient for the application of scheduling. The term hybrid refers to the fact that both the representation and one or more of the operators have been modified to make the GA application specific. The simple GA or binary GA is a good tool for studying the theory of GA, but more and more modern applications of GA are hybrids.

3.1 Genetic Algorithm

Genetic Algorithm (GA) is a search algorithm that borrows its operators from the biologic model of natural selection and evolution [Gol89]. GA have been very successful in combinatorially–explosive problems such as; Job Shop Scheduling [LHM87, SD85, WS89, CS89], Non-linear Transportation Model [MJxx, MJB90, MVHxx], and the Traveling Salesman Problem [GR85, OSH87, GGRV85] [PSD89, LHRP90, Lid91, WSD89]. The problem of mission planning for large numbers of airborne platforms contains elements of all three of these classic GA problems.

GA differ from traditional optimization and search algorithms [Gol89, pp 7]:

Genetic Algorithm	**Traditional Optimization and Search**
work with coding of the parameters	manipulate parameters directly
search from a population of points	use a single point to search
use payoff (objective function) information	use derivatives or auxiliary knowledge
use probabilistic transition rules	use deterministic rules

Similar to simulated annealing, GA uses a population of possible solutions to search the solution space. GA uses biologically inspired operators to improve the search. These operators alter the population and allow information about possible solutions to be shared amongst the individuals. The basic GA has the following operators that affect the population:

- Initialization
- Fitness Evaluation
- Selection
- crossover
- Mutation

Each of the following sections will begin with a statement describing the function of the particular operator as it pertains to genetic algorithms in general. This is followed by a discussion of how the operator has be implemented to handle this particular problem, thus making this tool a hybrid genetic algorithm.

3.1.1 Initialization

Each population member must contain a representation that describes *how* to solve a particular task. Classically GA's used binary strings called alleles to represent all problems. The intention was that the GA should be independent of the task being solved. Many problems existed that led to the development of Hybrid GAs. In a Hybrid GA the representation of the allele is specific to the problem being solved.

Most path planing and scheduling problems rely on either node or edge allele representations, resulting from the use of directed graphs to describe the solution. In this project a novel representation for describing simultaneous open–paths was developed.

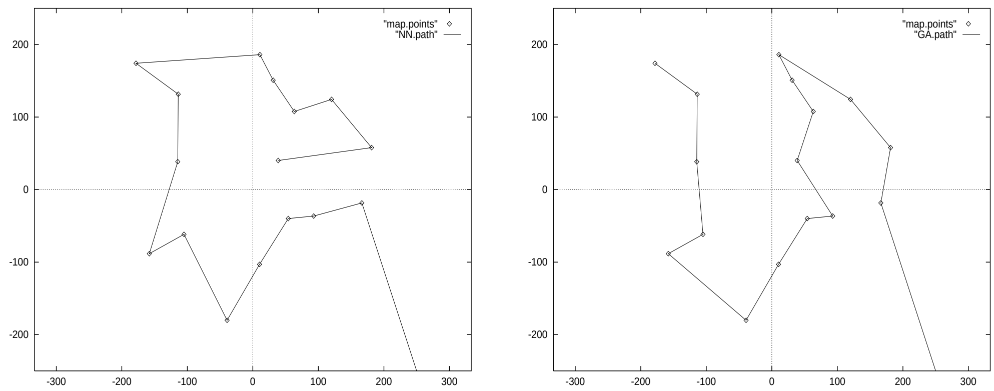


Figure 1: Graphs showing two possible paths through the same set of points.

One naive solution to open–path problems is the nearest neighbor search. The UAV would simply fly out to the nearest target and then fly on to the nearest remaining target and so on until all targets were reached. Figure 1 (left) shows an example of such a path. For this map the path length is 1605.81 units long. The figure on the right shows the shortest path, of 1487.97, though the same points. Both paths began at (250,-250).

In the early stages of this project a GA was constructed using a novel representation based on the nearest neighbor search. Each allele describes an n^th nearest neighbor. The use of this representation allowed simple two point crossover and random mutation to be utilized, see Sections 3.1.4 and 3.1.5 respectively. This GA was tested against low order problems, that were within the practical range of performing exhaustive searches. In each case the GA was able to find the shortest path that was found by the exhaustive search. The GA was then tested against an ordered set GA implemented in a previous project. For medium and high order problems this new representation out–performed the ordered set representation. The new GA was able to produce equal or shorter paths using fewer generations and smaller populations.

The next step was to modify this representation to facilitate simultaneous paths. Two or more paths traverse all of the points. Each path contains a non–overlapping subset of the points.

In scheduling problems multiple chromosomes may be used to represent a solution, see Figure 2. One chromosomes would represent each path and the alleles would represent the order

path 0:	735	*is equivalent to*	7126489305 / 0211221010
path 1:	2690		
path 2:	148		

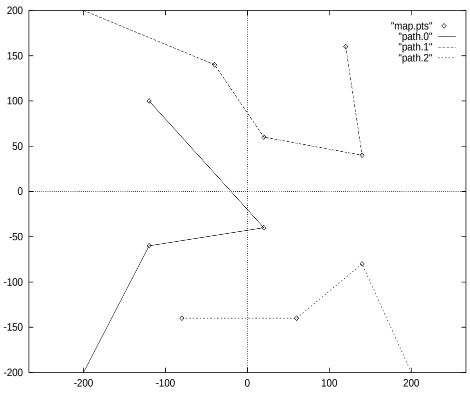
Figure 2: Two equivalent representations using multiple chromosomes (U)

in which the points were visited. An alternative method is that the first chromosome would represent the order in which points were visited and the second chromosome would match points with a path.

Both of these representations require complicated crossover operators that are highly susceptible to destruction under mutation.[2]

The new representation only requires a single chromosome. However, a modification had to be made to describe simultaneous paths. Two constants were defined. *depth* is the maximum number of *next to the nearest neighbors*, and *nuav* is the number of UAVs traversing simultaneous paths. A pair of functions were developed that determined the level of the nearest neighbor and the index of the UAV moving to that point. This allowed the use of a single string of alleles for each population member.

node	x	y
UAV-0	-200	-200
UAV-1	-200	200
UAV-2	200	-200
0	120	160
1	140	-80
2	-40	140
3	20	-40
4	60	-140
5	-120	100
6	20	60
7	-120	-60
8	-80	-140
9	140	40



path 0:	735	*is equivalent to* [1588551480]	
path 1:	2690		
path 2:	148		

Figure 3: The new representation only requires a single chromosome. The table provides the coordinates of the UAVs and Targets

Figure 3 shows how this new representation is equivalent to the ones in Figure 2. The *depth* and *nuav* were defined once for the entire population. Each population member was initially randomized.

[2](U) The probability of destruction under a given GA operator is one of the metrics used to analyze the usefulness of that operator. It is defined as the probability that the resulting individual will have a poorer fitness score than before the operator was applied.

3.1.2 Fitness Evaluation

The second most important consideration in designing a GA to solve a problem is the development of a fitness function. This is a function of the alleles that describes how one possible solution compares to others in the population. The fitness function should be continuous and preferably monotonically increasing or decreasing. The later constraint is not crucial. One of the strengths of GA is its ability to move past local optimum and converge on a global optimum point.

The selection method chosen for this GA, see Section 3.1.3, allowed a more qualitative fitness function to be used than in traditional GA implementations. In order to determine whether one solution was better than another, a multiple step decision process was used instead of generation a numeric score. If there were more mission objectives than time permitted in a given path, then the schedules containing paths with more high priority targets were judged better. If all mission objectives were met, then the schedule with the shortest completion time was judged to be the better schedule. Finally if the overall completion times were equal between two schedules then the schedule with the smallest variance in its path lengths was judged better.

3.1.3 Selection

Selection is the method by which members of a previous population are allowed to interact and either survive or generate a new population. There are many stochastic methods of selection for GA. Highly fit individuals must have an exponentially greater probability of surviving and interacting than lower fit individuals [Hol75]. It is necessary to allow a small percentage of "low" fit individuals to survive. This helps maintain genetic diversity and prevent convergence of the population on local optimal points.

Tournament selection was chosen as the method of selection for this GA. Literature has shown that tournament selection is superior to other stochastic forms of selection [BT95]. The most attractive feature of tournament selection is that it is only necessary to determine qualitatively if one individual solution is better than another. Stochastic selection requires a ranking of all individuals and a fitness function that can tell quantitatively how much better one solution is from another.

3.1.4 Crossover

Crossover is the operator that allows the sharing of information between individuals. One of the principle weaknesses in traditional optimization is the "*local optimal trap*". Hill climbing and other calculus based optimization routines that utilize a single search point can become trapped in a local optimum. GAs utilize a population that is, hopefully distributed widely across the search space. Parameters such as probability of crossover, probability of mutation, population size and tournament size can affect the genetic diversity of the population. A saturated population can also become trapped in large local optimum. crossover when combined with a good selection algorithm allows the transfer of knowledge between individuals in the population.

By using the novel representation described in Section 3.1.1 it was possible to employ a simple two point crossover. Two parent individuals are chosen in the selection process described above. A random number in the range of [0:1] was compared to the individual's probability of crossover.

P1:	0	2	0	1	0	0	1	0	1	1	2	1
P2:	1	1	0	0	0	2	0	1	1	0	1	0
C1:	0	2	0	0	0	2	0	1	1	1	2	1
C2:	1	1	0	1	0	0	1	0	1	0	1	0

Figure 4: Two point crossover

If it is determined that crossover should take place, then two points along the allele string are chosen at random. The middle segments between the cut points are exchanged between the parents in order top form two children. If crossover does not occur then the two parents are cloned to form the two children. This process is repeated until the population from the previous generation is replaced.

3.1.5 Mutation

Mutation introduces random variation into a population. This allows the possibility of introducing a new region of the solution space to the population. When GAs were strictly binary strings and the crossover operator was independent of the problem being solved, mutation was thought to be an unnecessary operator.

Mutation was also simplified by the use of this novel representation. Each allele of each population member was tested for its probability of mutation. A random number in the range of [0:1] was compared to the individual's probability of mutation. If it was judged that mutation should occur then the value of that allele was replaced by a random number in the range of [0:d*u] where (d) represents the maximum level of nearest–neighbor that may be chosen, and (u) represent the number of ready UAVs

4 Results

The novel representation in this GA was tested against low order problems, that were within the practical range of performing exhaustive searches. In each case the GA was able to find the shortest path that was found by the exhaustive search. In a higher order test this representation was tested against a GA that utilized a traditional ordered set representation. The other GA contained a combination of exiting operators and operators that were developed for a previous project. [SHL95] For medium and high order problems the new representation out performed the ordered set representation. The new GA was able to produce equal or shorter paths using fewer generations and smaller populations.

4.1 Mission Scenario One

The first scenario involved allowing the GA to run for a short while and then to change the mission priority of three of the requests. The GA was then allowed to replan the mission. This was compared to the GA running under those mission constraints from the beginning. Several different request combinations were altered. In each run the GA converged on the new mission significantly sooner than the GA that was run from scratch.

4.2 Mission Scenario Two

The second scenario involved allowing the GA to run for a short while and then to eliminate one or more of the UAVs. The GA was then allowed to replan the mission. This was compared to the GA running under those mission constraints from the beginning. Several different combinations of UAVs were disabled. In each run the GA converged on the new mission only slightly sooner than the GA that was run from scratch. This was due in part to the redistribution of the targets amongst the remaining UAVs. This is an area that could stand some investigation from a heuristic operator for rebalancing the entire population when the number of UAVs changes.

5 Conclusion

The GA was able to effectively schedule the UAV missions under the very complex constraints. Moreover the GA was able to dynamically replan the missions due to changes in mission priority or loss of assets.

The purpose of machine intelligent optimization is not necessarily to produce perfect results, but rather to produce the best results under the constraints of time and cost. This hybrid genetic algorithm was successful in producing timely simultaneous paths for multiple UAVs in relatively short time.

5.1 Future Work

It would be interesting to begin with heterogeneous collection of UAVs, then add threats, and incorporate probability of detection. Each UAV could have a different unit cost and mission value. More work should be done on the dynamic nature of this planning scheme. Scenarios could be developed that include changing threats. In addition to planning paths that would minimize mission time, the cost due to the probability of loss could be minimized. After launch replaning has many opportunities to expand the scope of this algorithm.

5.2 Acknowledgments

I would like to acknowledge the Raytheon System Company IDEA Program for their support in this investigation.

References

[BT95] Tobias Bickle and Lothar Thiele. A mathematical analysis of tournoment selection. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference (ICGA95)*, pages 9–16, San Francisco, CA, 1995. Morgan Kaufmann Publishers, Inc. ISBN 1–55860–370–0.

[CS89] Gary A. Cleveland and Stephen F. Smith. Using genetic algorithms to schedule flow shop releases. In Schafer, editor, *Proceedings of the Third International Conference for Genetic Algorithms*, pages 160–169. Morgan Kaufmann Publishers, Inc., 1989.

[DJS89] Kenneth A. De Jong and William M. Spears. Using genetic algorithms to solve np–complete problems. In *Proceedings of the International Conference on Genetic Algorithms*, pages 124–132, 1989.

[GGRV85] J. J. Grefenstette, R. Gopal, R. Rosmaita, and Gucht D. V. Genetic algorithms for the traveling salesman problem. In Grefebstette, editor, *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 154–159. Texas Instrument and U.S. Navy Center for Applied Research and Artificial Intelligence, 1985.

[Gol89] David E. Goldberg. *Genetic Algorithms, in Search, Optimization & Machine Learning.* Addison–Wesley Publishing Company, Inc., Massachusetts, 1989.

[GR85] David E. Goldberg and Lingle R. Alleles, loci and the traveling salesman problem. In Grefebstette, editor, *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms.* Lawrence Erlbaum Associates, 1985.

[Hol75] J. H. Holland. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, 1975.

[LHM87] G. E. Liepins, M. R. Hilliard, and Morrow M. Greedy genetics. In *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 231–235. Lawrence Erlbaum Associates, 1987.

[LHRP90] G. E. Liepins, M. R. Hilliard, J. Richrdson, and M. Palmer. Genetic algorithm applications to set covering and traveling salesman problems. In Brown and White, editors, *Operations Research and Artificial Intelligence: the Integration of Problem Solving Strategies*, pages 29–57. Kluwer Academic Press, 1990.

[Lid91] Mark L. Lid. Traveling salesman problem domain application of a fundamentally new approach to utilizing genetic algorithms. Technical report, Air Force Office of Scientific Research and Office of Naval Research, 1991. Research sponsored by Contract F4920–90–G–0033.

[MJxx] Zbigniew Michalewicz and Cezary Z. Janikow. Genetic algorithms for numerical optimization. *Statistics and Computers*, 1(2), 19xx.

[MJB90] Zbigniew Michalewicz, Cezary Z. Janikow, and Krawczyk Jacek B. A modified genetic algorithm for optimal control problems. *29th CDC*, 1990.

[MVHxx] Zbigniew Michalewicz, G. A. Vignaux, and Matthew Hobbs. A non-standard genetic algorithm for the nonlinear transportation problem. *Accepted for Publication in the ORSA Journal of Computing*, 19xx.

[OSH87] I. M. Oliver, D. J. Smith, and J. R. C. Holland. A study of permutation crossover operators on the traveling salesman problem. In Grefenstette, editor, *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference of Genetic Algorithms*, pages 224–230. Lawerence Erlbaum Associates, 1987.

[PSD89] Jog P., J. Y. Suh, and VanGucht D. The effects of population size, heuristic crossover and local improvement on a genetic algorithm for the traveling salesman problem. In Schaffer J. D., editor, *Proceedings of the Third International Conference for Genetic Algorithms*, pages 110–115. Morgan Kauffmann Publishers, Inc., 1989.

[SD85] D. Smith and L. Davis. Adaptive design for layout synthesis. internal report, Texas Instruments, Dallas, 1985.

[SHL95] Stephen W. Soliday, Abdollah Homaifar, and Gary L. Lebby Lebby. Genetic algorithm approach to the search for golomb rulers. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference (ICGA95)*, pages 528–535, San Francisco, CA, 1995. Morgan Kaufmann Publishers, Inc. ISBN 1–55860–370–0.

[WS89] D. Whitley and T. Starkweather. Genitor ii: A distributed genetic algorithm. *J. Expt. Thoer. Artif. Intell.*, 2:189–214, 1989.

[WSD89] D. Whitley, T. Starkweather, and Fuquay D. Scheduling problems and traveling salesman: the genetic edge re-combination operator. In Schafer, editor, *Proceedings of the Third International Conference for Genetic Algorithms*, pages 133–140. Morgan Kaufmann Publishers, Inc., 1989.