Using Computational Swarm Intelligence for Real-time Asset Allocation

Joshua Reynolds, Lauren Christopher
Purdue School of Engineering and Technology
IUPUI
Indianapolis, IN, USA
lauchris@iupui.edu

Russ Eberhart
Phoenix Data Corporation
Indianapolis, IN, USA

Patrick Shaffer
Spectrum Warfare Systems Department
Naval Surface Warfare Center, Crane Division
Crane, IN, USA

Abstract **Particle Swarm Optimization (PSO) is especially useful for rapid optimization of problems involving multiple objectives and constraints in dynamic environments. It regularly and substantially outperforms other algorithms in benchmark tests. This paper describes research leading to the application of PSO to the autonomous asset management problem in electronic warfare. The PSO speed provides fast optimization of frequency allocations for receivers and jammers in highly complex and dynamic environments. The key contribution is the simultaneous optimization of the frequency allocations, signal priority, signal strength, and the spatial locations of the assets. The fitness function takes into account the assets' locations in 2 and 3 dimensions maximizing their spatial distribution while maintaining allocations based on signal priority and power. The fast speed of the optimization enables rapid responses to changing conditions in these complex signal environments, which can have real-time battlefield impact. Initial results optimizing receiver frequencies and locations in 2 dimensions have been successful. Current run-times are between 300 (3 receivers, 30 transmitters) and 1000 (7 receivers, 30 transmitters) milliseconds on a single-threaded x86 based PC. Statistical and qualitative tests indicate the swarm has viable solutions, and finds the global optimum 99% of the time on a test case. The results of the research on the PSO parameters and fitness function for this problem is demonstrated.**

Keywords ***Particle Swarm Optimization, Electronic Warfare, Asset Allocation***

I. Introduction

Particle Swarm Optimization (PSO) is an exciting computational tool for optimization applications in scheduling and logistics, hardware development, artificial neural networks, and many other areas. Examples include: optimization of mission planning, optimization of allocation of electronic warfare resources, medical diagnosis, electric utility system load stabilization, and product mix optimization. PSO is exciting because of the ease and speed which applications can be developed (often weeks or months instead of years) and the performance of these solutions, which is often better and orders of magnitude faster than traditional solutions for complex or computationally-intensive problems.

PSO is an evolutionary computation technique developed in 1995 by James Kennedy and Russell Eberhart [1] [2]; with a text on the subject by Eberhart, Simpson, and Dobbins in 1996 [3] and by Eberhart and Shi in 1998 [4]. PSO methods were included in a formal textbook by Eberhart in 2007 [5]. At the time of the writing of this paper, PSO has been around for two decades; it is being researched and utilized in over 30 countries.

PSO has already been applied to some problems in real-time allocation. For weapons allocation for defensive purposes as seen in [6], PSO was shown successful for small-scale problems. In this thesis, the application of PSO to real-time asset allocation in the area of electronic warfare (EW) is explored. This is follow-on work to a project done for the Expeditionary Electronic Warfare Division, Spectrum Warfare Systems Department, at the Naval Surface Warfare Center (NSWC) Crane [7]. PSO was used in that project to allocate electronic warfare resources in the frequency spectrum in a rapidly changing environment on a near-real-time basis.

In an operational scenario involving the allocation of multiple receiver resources against a suite of dozens of signals with varying powers and priorities in the past required a state-of-the-art system. This system took nearly two hours to calculate the optimal receiver center frequencies. Of course, this is clearly not useful in an operational environment. The reported PSO solution optimally allocates resources in one second or less.

The contribution of this research extends the previous work in [7] optimizing the resources simultaneously in 2D space and across the frequency spectrum. The previous work assumed that all of the assets were co-located at a single point in space. The changes made to the optimizer are described in Section II. Section III shows the graphical user interface (GUI) tools developed in our research. Analysis of the results is found in Section IV. Finally, Section V concludes the research and discusses the next research steps.

This is the author's manuscript of the article published in final edited form as:Reynolds, J., Christopher, L., Eberhart, R., & Shaffer, P. (2015). Using computational swarm intelligence for real-time asset allocation. In 2015 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA) (pp. 1–5). http://doi.org/10.1109/CISDA.2015.7208619

II. PSO AND FITNESS FUNCTION

A. Application Discussion

The application problem is to optimize a scarce asset, such as radio frequency (RF) receivers, simultaneously in 3D space and across the frequency spectrum. Each RF receiver has a certain programmable bandwidth and maximum allowable input power. The receivers must be allocated to a number of transmitters, where each transmitter has a priority and power. In addition, the transmitters are placed in a defined area, simulating the electronic warfare (EW) battlefield. It is desirable to receive signals with the highest priority and power while not overloading the RF front end of any receiver. Furthermore, it is advantageous for the receivers to be spread out in 3D space. This spatial dispersion is useful for optimization of problems like battlefield resource distribution of mobile assets, cell phone tower locations, distribution hubs for order fulfillment, etc. Therefore, an ideal solution will give the best compromise between the spatial spread of the receivers as well as the received power and priorities. The fitness function is designed as the weighted sum of these four components: priority, power, spread, and distance. The following sections describe the fitness function in detail for these four areas.

B. Priority

Priorities can be an input by an operator. In our test cases, priority is randomly assigned during initialization using a uniform distribution. Each transmitter signal in the spectrum is given a priority from the set {1, 3, 5}, where a higher number represents a higher priority. The fitness function calculates the priority fitness component as the sum of the all of the priorities of the received signals.

$$Fitness\ Priority\ Component = \sum_{i \in Received\ Signals} Priority\ of\ Signal_{(i)}$$

As in the previous work [4], it is possible for two receivers to overlap in frequency such that they are both receiving the same signal. In this case, the fitness function only counts the priority once.

C. Power

Likewise, the power component is found by summing the powers of the received signals. When summing the signal powers, the fitness function must account for the distance between the receiver and signal source so that the free space path loss of the signal is calculated according to:

$$Loss\ in\ dB = 20 * log10(d) + 20 * log10(f) + 32.45$$

where d is in kilometers and f is in MHz.

Thus, the power of each signal is calculated from the perspective of each asset. The power of each received signal is then summed in magnitude form. As with the priority component, the fitness function does not count twice any signal that is received by two or more receivers. The total sum of the received power is converted to dB scale and used in the fitness component. A problem arises when negative dB values are encountered. If the conversion to dB scale results in a negative value, the returned fitness component would subtract from the overall fitness even though it may be beneficial to receive the signals. To overcome this, we add an appropriate offset to the final dB value such that the returned value is guaranteed to be a positive value.

$$Fitness\ Power\ Component = 10 \cdot \log_{10}\left[\sum_{i \in Received\ Signals} Power\ of\ Signal_{(i)}\right] + Offset$$

D. Spread

One of the main requirements of this research is to ensure that the optimizer produces a solution where the assets are spatially dispersed (spread) as much as possible. The spread component of the fitness function can be calculated in several ways. The simplest method takes the sum of the Euclidean distances between all of the receivers. Calculating the spread fitness this way produced some undesirable side-effects in initial testing. By design, the spread component and power component of the fitness will fight each other. It is not possible to maximize both at the same time, since a high-spread fitness solution will place the receivers far away from the signals and thus cause the power fitness component to have a lower score.

Initial tests with three receivers showed that one or two of the receivers ended up very near the signals, giving a very high power score. At the same time, the remaining receivers were pushed out far from the signals, giving a very high spread score. Thus the PSO solution found the best was to "sacrifice" the power score for one of the receivers in hopes of gaining a higher spread score. Through testing, we found that it was possible to counter this behavior by calculating the spread component as the distance between the two closest assets. Calculating the spread component in this manner forced the optimizer to spread the assets more evenly around the solution space.

A challenge arises from the fact that RF loss is input to the system in dB, and follows a log function as distance increases. On the other hand, the spread component is linearly proportional to distance. Two fitness functions need to balance each other for proper operation, so a log of the distance between receivers is the better choice both theoretically and experimentally. The calculation of the fitness spread component is according to the following equation, in which $Distance_{(ij)}$ represents the Euclidean distance between $receiver_{(i)}$ and $receiver_{(j)}$.

$$Fitness\ Spread\ Component = \log_{10}\left[min\left(Distance_{(ij)},\ i \neq j\right)\right]$$

E. Distance Component

While the fitness spread component successfully disperses the receivers in space, it does not provide any means to distribute the assets near the receivers. It is true that the power fitness component tends to place the assets near the receivers in order to achieve a higher overall power. However, in our testing this sometimes produced unsatisfactory results due to the way in which the spread component and power component tend to fight each other. Prior to adding this fitness component, we observed cases where one asset that had relatively few signals assigned to it would be placed an infinite distance (if the boundaries were removed) from the signals. In these cases, the optimizer sacrificed one of the assets by causing its power contribution to become almost non-existent in order to gain an increase in the fitness spread component. Attempts to counter this behavior by adjusting weights on the fitness components were not very successful. Increasing the weight on the power component or decreasing the weight on the spread component had the effect of causing the assets to congregate too close to the receivers. Thus it was difficult to achieve a good middle ground. The addition of the fitness distance component gave more stability to the solutions obtained. This component is calculated by taking the mean of the distances between each asset and the center of mass of the transmitters that it is receiving as shown in the equation below. In this equation, $D_{(i)}$ represents the distance between receiver$_{(i)}$ and the center of mass of the transmitters that receiver$_{(i)}$ is receiving. This distance, $D_{(i)}$ is subtracted from a constant Max Distance to so that a higher score is given to smaller distances and so that a positive value is always returned.

$$Fitness\ Distance\ Component = \frac{1}{N}\sum_{i=1}^{N} Max\ Distance - D_{(i)}$$

F. *Weights and total fitness*

The overall fitness is calculated by taking the weighted sum of the three fitness components. Weights for the three fitness components were determined experimentally and chosen so that the dynamic range of each component would be similar.

G. *Receiver keep away*

Finally, a keep-away penalty has been added to keep all the assets outside of a spatial boundary, geographically separated from the transmitters. A sharp penalty is added to the overall fitness when any asset enters a pre-defined boundary around the signal sources. The overall fitness is multiplied by 0.5 for each asset inside the boundary. Prior to adding this boundary, at least one of the receivers ended up on top of the transmitter signals in order to achieve a high power score. In addition, the solution space has been limited to keep the receivers in a square of +- 100 Km in the X and Y dimensions. The keep-away boundary is user-selectable between a circular boundary and a straight, linear boundary. Future research will explore the use of a flexible boundary, with the eventual goal of matching it to the battlefield terrain.

H. *PSO settings*

The Particle Swarm Optimization algorithm in [4] was used to converge to the solution. The PSO in this problem was set to 200 particles (population size), uses a neighborhood optimization strategy with a noisy inertia weight. The fitness components were weighted to balance the contribution of the three areas, with a special emphasis placed on the priority assignments. The swarm was run to 1000 generations, although typical convergence was less than 500 generations. These were used as our ***test*** parameters. Experimentation was done with another set of swarm parameters using a population size of 50, a neighborhood size of 1, and a method of terminating the swarm early when convergence is detected. These are our ***performance*** parameters. Repeatability and runtimes of the optimizer using the ***test*** parameters and ***performance*** parameters are examined in the results section.

III. GRAPHICAL USER INTERFACE

The Qt software framework was used to develop an interactive GUI for this research. Qt is an open-source and cross-platform framework for UI development in C++. A current version of the GUI is shown in Figure 1. The Allocation Plot on the top left shows the spatial location of the receivers and transmitters. The receivers are randomly distributed in the center. Color-coding is used to differentiate between the priorities of each transmitter. The keep-away boundary is depicted by a black circle around the transmitters. The PSO will attempt to optimize with highest priority signals (yellow in the spectrum plot) first, and mid priority second (green) and finally low priority (blue). In this test case, the transmitter location, priority and power was set randomly.

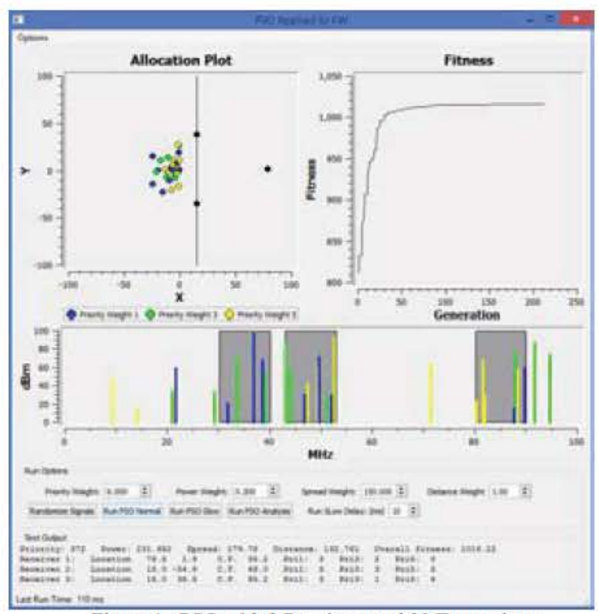

Figure 1. PSO with 3 Receivers and 30 Transmitters.

In Figure 2, by hovering over one of the receivers, the corresponding transmitters are highlighted, and the frequency allocation is highlighted in red. The subplot on the top right shows the fitness value versus the swarm generation. This is helpful to determine how the swarm evolved over generations. Ideally, the swarm should quickly converge to a maximum fitness and then the fitness should remain nearly constant for successive generations. Future work will be done to detect the point at which the swarm reaches a sufficient solution so that the swarm can be terminated.

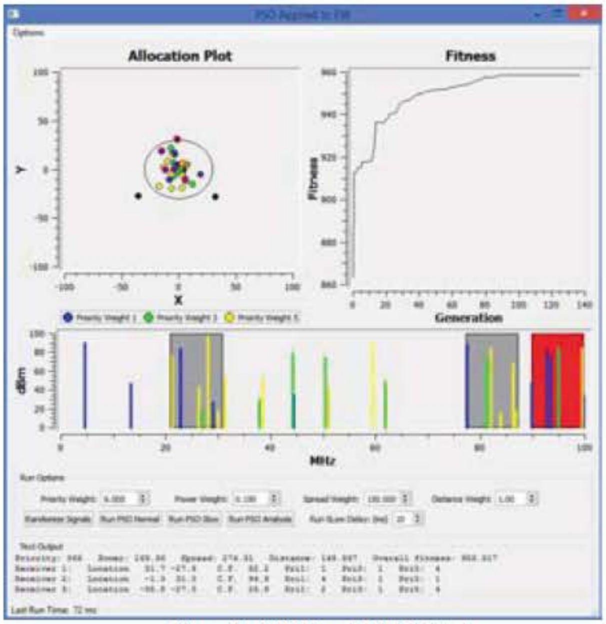

Figure 2. PSO Result Highlighted.

Other settings that can be changed in the GUI are: the number of transmitters and receivers, the radius of the transmitter distribution, the radius of the keep-away boundary, and the fitness function component weights. Options exist to run the swarm at full speed, or to run the swarm slow enough that a human can observe it converge. After each run, additional textual information at the bottom of the GUI is given showing exact locations and frequencies of the receivers.

IV. Results

A. Analyzing Results

To understand the distribution of final fitness values, we ran optimizations on a number of different configurations. We calculated the means and variances of the final fitness values for these configurations, which are summarized in Table 1 and Table 2. It is important to note that the fitness values have no physical meaning by themselves. The values listed here are merely useful for comparison between the different tests.

TABLE 1
Fitness Means and Variances Using Test PSO Parameters

Receivers / Transmitters	Mean	Std. Dev.
3, 30	804	1.18
4, 30	891	5.57
4, 50	1234	11.6

TABLE 2
Fitness Means and Variances Using Performance PSO Parameters

Receivers / Transmitters	Mean	Std. Dev.
3, 30	794	10.7
4, 30	868	14.4
4, 50	1197	25.8

B. Run time Analysis

Several run-time analyses were performed on the optimizer. Run-times were found for varying problem sizes and varying swarm parameters. For each run-time analysis, the PSO was run 50 times and the resulting run-times were averaged. Tests were run with both sets of PSO parameters as described in Tables 2.1 and 2.2. All run-time tests were performed on an Intel Core i7-4710HQ processor. All tests were run using a single thread of execution. Table 3 and Table 4 summarize these run-time tests.

TABLE 3
Run times Using test PSO Parameters

Receivers / Transmitters	Average Run-time (ms)
3, 30	452
4, 30	667
7, 50	1098

TABLE 4
Run times Using Performance PSO Parameters

Receivers / Transmitters	Average Run-time (ms)
3, 30	24
4, 30	45
4, 50	85

C. Repeatability Analysis

A special test problem was designed with a known global maximum solution. The test problem was designed to contain local maxima in which the PSO might become stuck. A statistical analysis was run using this test setup to determine how well the PSO finds the global best solution without becoming stuck in local maxima. Figure 3 shows the test case where 4 transmitters of alternating priority and equal power are uniformly spaced along the battlefield line and the global best solution for two assets. Any other solution is a local maximum solution. Using the test PSO parameters, it was found that the optimizer found the global best solution with a probability greater than 0.99. Running the swarm in this configuration for 50 configurations gave a mean fitness value of 645.8 and a standard deviation of 0.001. However, when the performance PSO parameters were used, the probability of finding the global best solution dropped to 0.77. The mean fitness value in this case was 628.9 and the standard deviation of fitness was 37.8.

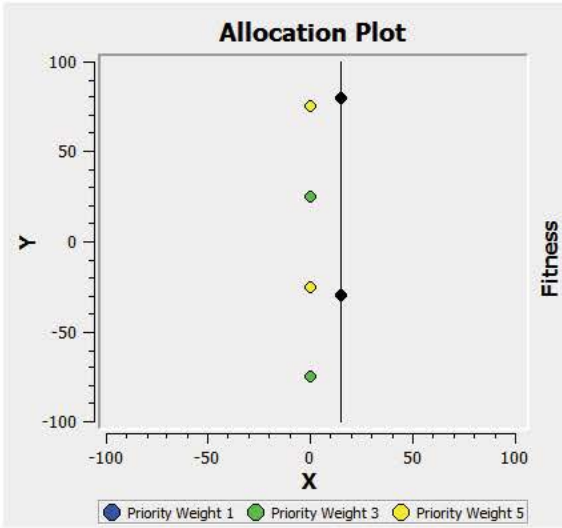

Figure 3. PSO Result Highlighted.

V. Conclusions

The Particle Swarm Optimization has been shown to be useful for the allocation of scarce resources of receivers in Electronic Warfare. This new research has shown that optimization over both frequency and 2D space is feasible with a very rapid run time of one second or less. A trade-off exists between obtaining repeatable solutions and obtaining solutions quickly. Next, continued research on the optimization in all 3 spatial dimensions will be performed.

Future research is planned in the following five areas: expand receivers to multiple layers of assets, include more complex battlefield 3D terrain, restrict the movement of receivers, optimize while the transmitters are in motion, and finally, include humans in the swarm process with respect to the configuration of the exclusion zones.

Acknowledgment

The authors would like to acknowledge the support from the sponsor, the Expeditionary Electronic Warfare Division, Spectrum Warfare Systems Department, at the Naval Surface Warfare Center (NSWC) Crane. This research was sponsored by Crane NSWC through CRNBAA14-002.

References

[1] Kennedy, J.; Eberhart, R., "Particle swarm optimization," *Neural Networks, 1995. Proceedings., IEEE International Conference on* , vol.4, no., pp.1942,1948 vol.4, Nov/Dec 1995

[2] Eberhart, R.; Kennedy, J., "A new optimizer using particle swarm theory," *Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on* , vol., no., pp.39,43, 4 6 Oct 1995

[3] Eberhart, R. C., Simpson, P. K., and Dobbins, R. W., *Computational Intelligence PC tools*, 1st ed. ed. Boston, MA: Academic press professional, 1996.

[4] Yuhui Shi; Eberhart, R., "A modified particle swarm optimizer," *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on* , vol., no., pp.69,73, 4 9 May 1998

[5] R. C. Eberhart and Y. Shi, *Computational Intelligence: Concepts to Implementations*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007.

[6] Johansson, F.; Falkman, G., "Real time allocation of defensive resources to rockets, artillery, and mortars," *Information Fusion (FUSION), 2010 13th Conference on* , vol., no., pp.1,8, 26 29 July 2010

[7] Eberhart, R.C.; Keller, J.; Kraft, J.; Verdon, J., "Plenary speakers and invited tutorial speakers," *Computational Intelligence for Security and Defence Applications (CISDA), 2012 IEEE Symposium on* , vol., no., pp.1,6, 11 13 July 2012