A Particle Swarm Social Model for Multi-Agent Based Insurgency Warfare Simulation

Xiaohui Cui and Thomas E. Potok

Applied Software Engineering Research
Computational Sciences and Engineering Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6085
cuix, potokte@ornl.gov

Abstract

To better understand insurgent activities and asymmetric warfare, a social adaptive model for modeling multiple insurgent groups attacking multiple military and civilian targets is proposed and investigated. This report presents a pilot study using the particle swarm modeling, a widely used non-linear optimal tool, to model the emergence of insurgency campaign. The objective of this research is to apply the particle swarm metaphor as a model of insurgent social adaptation for the dynamically changing environment and to provide insight and understanding of insurgency warfare. Our results show that unified leadership, strategic planning, and effective communication between insurgent groups are not the necessary requirements for insurgents to efficiently attain their objectives.

1. Introduction

Insurgency warfare is dynamic, adaptive and non-linear warfare. Studies of insurgency warfare can be traced back to the publications of Chinese strategist Sun Tzu in 400BC[1]. In the last century, relatively modern studies about insurgency warfare have provided general insights and practical guidance into the perspective of insurgents and counter-insurgents [2, 3]. The U.S. military, particularly the Army, has a long history of counter insurgent activity. Until the past decade though, this has not been an area of focus for the U.S. military. The events happened in Somalia, Iraq, and Afghanistan make the military re-assess the 21st century insurgency and revisit its strategy, operational concepts, organization, and doctrine. The investigation of insurgency in the U.S. has reached it's height since the end of the Cold War. But insurgency remains a significant challenge for the U.S. and other governments because of its two dominant characteristics: protraction and ambiguity. One possible solution is to represent insurgency and counter insurgency (COIN) warfare as a complex social adaptive system (CAS)[4, 5]. CAS is a non-linear dynamical system of many interacting agents continuously adapting to a changing environment. Agent Based Model and Simulation (ABM) provides an architecture and platform for the implementation of relatively autonomous agents. Social processes can be simulated by using ABM in the computer. Agents represent people or groups of people. Agent interaction represents the process of social interaction. The advantage of using ABM simulate complex social system is in some circumstances, ABM can be used to carry out experiments on artificial social systems that would be quite impossible or unethical to perform on human populations. This greatly contributed to the establishment of the agent-based CAS simulation. Numerous empirical-based multi-agent simulators[6] have been constructed over recent years to model complex dynamic systems in numerous disciplines. Currently, most ongoing work in the insurgency warfare research area is typically concerned with enhancing existing military capabilities for COIN rather than building scientific understanding of the insurgency. In terms of modeling, there does not appear to be any mature or widely used methodology addressing insurgency warfare.

In this paper, we present a modified particle swarm model to simulate the insurgent groups' social interaction and adaptation in a complex insurgency warfare system. Our simulation results indicate that even without a centrally controlled leadership to coordinate the action of each insurgent member, the particle swarm modeled insurgent swarm can emerge a highly coordinated behavior that optimizes their attacking results. This paper is organized as follows:

0-7695-2867-8/07 $25.00 © 2007 IEEE
DOI 10.1109/SERA.2007.47

IEEE computer society

Section 2 incorporates an introduction to the canonical particle swarm optimization algorithm. Section 3 provides related work on the agent based insurgency warfare simulation; Section 4 describes the particle swarm social adaptive model for our agent based insurgency warfare simulation. Section 5 contains the experimental setups and results. Result discussion and conclusion are presented in Section 6.

2. Particle Swarm Optimization Algorithm

Particle Swarm Optimization (PSO) is an important part of swarm intelligence. It was originally developed by Eberhart and Kennedy in 1995[7], inspired by the social behavior of the bird flock. In the PSO algorithm, birds in a flock are symbolically represented as particles. These particles can be considered as simple agent swarm "flying" through a problem space. A problem space in PSO may have as many dimensions as needed to model the real problem space. Each particle has a location X-vector and a velocity V-vector. A particle's location in the multi-dimensional problem space represents one solution for the problem. When a particle moves to a new location, a different problem solution is generated. This solution is evaluated by a fitness function that provides a quantitative fitness value of the solution's utility.

Each particle also has memory to record the "best location" that it has experienced in the problem space so far, and the knowledge of the best location found so far by all the particles of the swarm. The "best location" means the problem solution generated on this location has the best fitness value. Particles of a swarm communicate the best location with each other and adjust their own location and velocity based on this best location. It is the particle's personal experience combined with its peers' experience that influences the movement of each particle through a problem space. For every generation, the particle's new location is computed by adding the particle's current velocity V-vector to its location X-vector. Mathematically, given a multi-dimensional problem space, the ***i***th particle changes its velocity and location according to the following equations[8, 9]:

$$v_{id} = w*(v_{id} + c_1 * rand_1 * (p_{id} - x_{id}) + c_2 * rand_2 * (p_{gd} - x_{id})) \quad (1a)$$

$$x_{id} = x_{id} + v_{id} \quad (1b)$$

where $\boldsymbol{V_{id}}$ indicates the speed of the particle moving along the dimensions in a problem space; $\boldsymbol{x_{id}}$ is the particle's current location; p_{id} *(personal best)* is the location of the particle experienced its personal best fitness value; p_{gd} *(global best)* is the location that the particle experienced the highest best fitness value in the entire population; d is the number of dimensions of the problem space; $rand_1$, and $rand_2$ are random values in the range of (0,1). c_1 and c_2 are two positive acceleration constants; $\boldsymbol{w}$ is the constriction coefficient[9] and is computed according to Equation 2a:

$$w = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \quad (2a)$$

$$\varphi = c_1 + c_2, \varphi > 4 \quad (2b)$$

Equation 1a requires each particle to record its current coordinate $\boldsymbol{x_{id}}$, its velocity $\boldsymbol{V_{id}}$, its personal best fitness value location vector P_{id}, and the whole population's best fitness value location vector P_{gd}. The best fitness value X_i is updated at each generation based on Equation 3, where the symbol *f()* denotes the fitness function; X_i *()* denotes the best fitness values; and t denotes the generation step.

$$X_i(t+1) = \begin{cases} X_i(t) & f(P_d(t+1)) \le X_i(t) \\ f(P_d(t+1)) & f(P_d(t+1)) > X_i(t) \end{cases} \quad (3)$$

The P_{id}, P_{gd} and their coordinate fitness values $f(P_{id})$ and $f(P_{gd})$ can be considered as each individual particle's experience or knowledge and Equation 3 is the particle's knowledge updating and learning mechanism. In PSO, the knowledge of each particle will not be updated until the particle encounters a new vector location with a higher fitness value than the currently stored value in its memory.

3. Related Work

A relevant multi-agent based model has been built and studied by Epstein[5]. Epstein reported a simple cellular automata (CA) model for simulating civil violence. In this idealized spatial model, a central authority seeks to use police officers to arrest (remove) actively insurgency from the society for a specified jail term to suppress a decentralized rebellion. This model contains three types of agents: the general citizen, the insurgency, and the police officer. All agents possess local vision and can randomly move to a new unoccupied site within its limited vision over a two-dimensional lattice. By using this simple CA simulation, Epstein showed how the complex dynamics resulting from simple assumptions can generate empirically interesting macroscopic regularities that are difficult to analyze using standard modeling approaches. The MANA model [6], an extension of Epstein's model, introduced specific movement strategies that are aimed at correcting the

purely random movement of agents. However, the agent interaction and cognition in both simulations are too rigid and simplistic to be psychologically plausible. The behavior of the software agent is strictly rule-based: If a particular condition appears in the agent's environment, the agent can only respond with a particular preprogrammed action. There is no direct communication between agents. Each agent does not have any capability of learning from its previous experience. This is clearly a very unrealistic representation of the social world.

A new insurgency warfare model that can provide a better understanding of the insurgent communication and learning activity is needed. The individual particles in the PSO model are capable of both individual learning and social learning through the interactions between particles. We propose a particle swarm social adaptive model to simulate insurgency warfare. This report presents a pilot study of an integration of particle swarm social knowledge adaptation and multi-agent approaches for modeling the collaboration of insurgent groups while attacking multiple military and civilian targets.

4. Particle Swarm Social Adaptive Model for Insurgency Warfare Simulation

In this agent based insurgency warfare simulation study, we extend the use of PSO on human social model to simulate the interaction behavior between insurgent agents. Although the PSO algorithm has been widely used as a function optimization tool since it was first published in 1995, the initial research target of the PSO was to develop a human social model and the algorithm itself represents an abstract model of human knowledge social adaptation behavior[10, 11]. Researchers from Europe have applied the PSO model to the simulation of the social behavior in animals [12, 13] and the strategic adaptation in organizations [14]. The research of applying PSO models to the evolution of human society behavior and social cognitive modeling is still unavailable in scientific publications. In this study, we use PSO to model the social adaptive behavior in insurgency warfare. In the following section, we will present a detailed description of each component of the PSO model in our agent based insurgency warfare simulation.

4.1 Simulation Model

The simulation model can be either highly idealized or detail descriptive. The most fundamental decision an ABM researcher makes is how detailed their model will be. Most ABMs are intimately tied to a specific domain because they include a considerable amount of detail derived from real world datasets and their goal is to answer a specific real-world question. The downside of detailed models is that they may predict too many possible outcomes if they have many parameters that are insufficiently constrained. Some other researchers choose to create highly idealized models that distill a collective phenomenon to its functional essence. Researchers pursuing idealized models are typically motivated to describe domain-general mechanisms with a wide sphere of application. Idealized models have widespread application to many real-world domains, and generate comprehensible explanatory accounts by focusing on only a few crucial causal elements. However, the highly idealized model might not map onto any actual case study, and they can oversimplify to the point of leaving out crucial details. The strategy for designing an insurgent simulation starts firstly from a very simple model, which is easy to specify and implement. When we understand this simple model and its dynamics, we will extend the model to encompass more features and more complexity.

4.2 Agents

In an agent based simulation, the most important object is the “agent”. Agents can represent people or groups of people. Agent interaction represents processes of social interaction. Two types of agents are specified in this particle swarm social adaptive model – the insurgent and the authority. There can be multiple insurgent groups. The insurgent agent can be affiliated with different groups. All agents behave, act, and react in accordance with the environment they have detected.

4.3 Simulation Scenario

Different groups of insurgent agents seek efficient attacking strategies to strike the authority. The insurgent agents do not have any prior-knowledge about the strategies. The objective of each insurgent agent is to find out the action strategy that can generate greatest profit for insurgent group as well as highest damage to the authority. In contrast, the authority seeks to reduce the insurgent profits of some particular strategies when the strategies are adopted by the insurgent agents. The insurgent agent that attacks the authority will receive a feedback on the results of the current and historic attack strategy. The strategies’ profit randomly changed in each time step. After a strategy is adopted by many insurgents, the profit of this strategy each insurgent agent gained will gradually reduce. The more insurgent agents adopt the strategy, the lower the profit each insurgent agent can gain.

4.4 Insurgent Information Exchange Rule

In the real world, the information exchange between insurgent groups is not as efficient as that

within the same insurgent group. Because of the reason of competition, some insurgent group may not want to share their newest attacking strategy to other insurgent group. The security reason also force insurgent groups to avoid too many inter-group communications within insurgent groups. Other insurgent groups have to acquire the intelligence information through other method. For instance, they can learn other groups' new attacking strategy through News media, Internet and other public broadcast method. The newest and the most efficient strategy learned from other groups is usually non-accurate or delayed. In this simulation, insurgents belonging to the same group can exchange information without any restriction. But the information exchanged between different groups will be delayed for a pre-defined number of time-steps and some noise will be added to the value of the information to reduce the information's accuracy.

4.5 Insurgent Agent Strategic Searching Rule

The PSO algorithm is used to control the insurgent strategic searching in the virtual strategy space. Under the particle swarm metaphor, each insurgent particle is assumed to move through an attack strategic virtual searching space to search for a functional optimum. Each insurgent particle has two associated properties, a current position ***x*** in the virtual strategy space and a velocity ***v***. Each particle has a memory of its best strategy location (***pbest***) where the strategy will cause a biggest lose to the authority when it is adopt by the insurgent agent. Each particle also knows the global best location (***gbest***) found by all other neighbor particles that belong to the same insurgent group. The ***gbest*** of different groups will be exchanged between different groups. However, these exchanges are always noisy, delayed and some time even in error. When the delayed and noisy ***gbest*** value from other groups arrives, the ***gbest*** value from other groups will replace the ***gbest*** value within the group, if the ***gbest*** value from other groups is greater than the current ***gbest*** value. At each step of the algorithm, an insurgent particle moves from its current position to a new location based on a velocity vector. The velocity vector is influenced by the particle's previous velocity, its current location, and its ***pbest*** and ***gbest*** value. Therefore, at each step, the size and direction of each particle's movement is a function of its own history (experience) and the social influence of its peers.

4.6 Profitable Strategy Moving Rule

The attacking strategies that can bring profit to insurgent are dynamically move in the virtual strategy space. The movement of the profitable strategies includes two moving rules. (1) Randomly wandering. The authority agent will randomly change its security and protection strategy and eventually make the insurgent's profitable strategy randomly move. (2) Adaptive changing. When one attacking strategy is highly profitable, this strategy will be gradually adopted by all insurgent agents. However, when the number of agents that adopt the same attacking strategy is increased, the authority will notice the phenomena and intentionally decrease the profit that the insurgent agents can earn from this strategy.

4.7 Insurgent Agent Memory Update Rule

Based on the strategy moving rule, profitable strategy can randomly move in the environment and the fitness value of each strategy in the environment may change over time following insurgent attacks. The strategy with the highest fitness value found by a specific insurgent particle will not have the highest fitness value after the strategy being adopted by insurgent agents for several time-steps. The dynamic change of the highest strategy fitness value location requires the particle to renew its memory whenever the environmental status does not match its memorized knowledge. However, the traditional PSO lacks an update mechanism to renew the particles' memory when the environment changes. If insurgent particle uses the traditional PSO to direct its attack strategy searching, the PSO searching algorithm can cause the particle to continue using the obsolete knowledge to direct its search, which inhibits the particle from following the path of the current optimal solution. As a result, the particle can be easily trapped in the region of the former optimal solution. Therefore, a memory update mechanism[15] is used to renew insurgent particles' memory when it is necessary. In this mechanism, a new notion, an evaporation constant ***T***, is introduced. ***T*** has a value between 0 and 1. The personal fitness value and global fitness value stored in each particle's memory will gradually evaporate (decrease) at the rate of the evaporation constant over time. The update process is formulated as:

$$X_i(t+1) = \begin{cases} X_i(t) & f(P_d(t+1)) \le X_i(t)*T \\ f(P_d(t+1)) & f(P_d(t+1)) > X_i(t)*T \end{cases} \quad (4)$$

In Equation 4, after the value of fitness evaporates for a period, the fitness value, ***X-fitness***, of the current location may be higher than the evaporated fitness values and will replace the old fitness value. Although all particles have the same evaporation constant T, each particle's updating frequency may not be same. Depending on the particle's current stored best fitness value ***f(P)*** and the current fitness value ***f(X)*** the particle acquired, the particle will update its best fitness value more frequently by using its current fitness value when

the ***f(P)*** is lower and the ***f(X)*** is higher. However, when the ***f(P)*** is higher and the ***f(X)*** is lower in a changing environment indicates the particle's current location is farther away from the current optimal solution compared to the distance between the optimal solution and the best fitness value's position stored in the particle's memory. In this situation, the best fitness value will be kept in the particle's memory till the best fitness value has become too obsolete after several generations. The fitness value update equation enables each particle to self-adapt to the changing environment.

5. Experimental Settings and Results

Simulations are carried out in the Netlogo[16] agent modeling environment. The insurgent particles use Equation 4 to update their best fitness value. The evaporation constant ***T*** is set as ***exp (-1)***. There are 300 insurgent particles randomly distributed in an environment that consists of a ***100x100*** rectangular grid. An example of the initial environment is shown in Fig. 1. In the initial environment, all 300 insurgents are equally belonging to twenty insurgent groups. Eeach group has 15 insurgent members. Each dot represents an insurgent and different colors are used to help to identify different insurgent groups.

To simulate the movement of the strategies and the dynamic change of the profit (fitness value) of the strategies, a test function generator, DF1, proposed by Morrison and De Jong[17], is used to construct the dynamic environment. The generator is capable of generating a given number of peaks (optimal solutions) in a given number of dimensions. For a two dimensional problem, the static evaluation function in DF1 is defined as following:

$$f(X,Y) = MAX[H_i - R_i * \sqrt{(X-x_i)^2 + (Y-y_i)^2}] \quad \text{(i=1,...N)} \quad (5)$$

where ***N*** denotes the number of peaks in the environment. The (***x_i***, ***y_i***) represents each cone's location. R_i and H_i represent the cone's height and slope.

Different movement functions generate different types of dynamic changing environments. In this research, the environment changing rate is controlled through the following logic function[17] :

$$Y_i = A * Y_{i-1} * (1 - Y_{i-1}) \quad (6)$$

where ***A*** is a constant and ***Y_i*** is the value at the iteration. The ***Y*** value produced on each iteration will be used to control the changing step sizes of the dynamic environment. In this research, the dynamic environment is simulated by the movement of the cone's location (***x_i***, ***y_i***). The ***Y*** value represents the moving velocity of the cone location.

The two dimensional visual result from previous dynamic environment simulation model is shown in figure 1. Eight white circuits represent different insurgent attacking strategies and the fitness values of the strategies. The brighter the white color of the circuit, the higher the fitness value or profit an insurgent agent can earn.

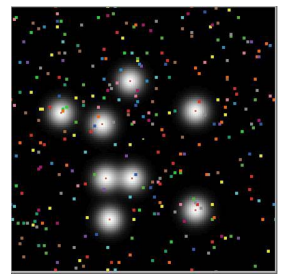
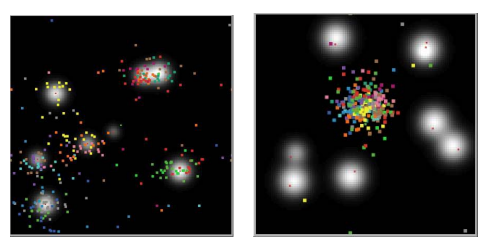


Fig 1: The initial environment

The movement of each insurgent particle follows Equation 2a and Equation 2b, in which ***c_1*** and ***c_2*** are set to 1.49 and ***V_{max}*** is set to 5. The ***w*** value is set to 0.72. The delayed time-steps for information exchange between insurgent groups is 20 time-steps. There is a 20 percent possibility that the information, including the location of the best fitness value and the fitness value itself, is incorrect. Different numbers of insurgent groups are simulated in this study. To evaluate the performance of the insurgent groups in the simulation, average loss and total loss caused by insurgent attacks at each time step are recorded separately. The simulation results are presented in Fig. 2 - 6. The visual result of insurgent particles with updated memory rule (Fig. 2a) is compared with these without updated memory rule (Fig. 2b) and the results are shown in Fig. 3.



(a) (b)

Fig 2: The visual results of insurgent particles (a) with update memory rule, (b) without update memory rule

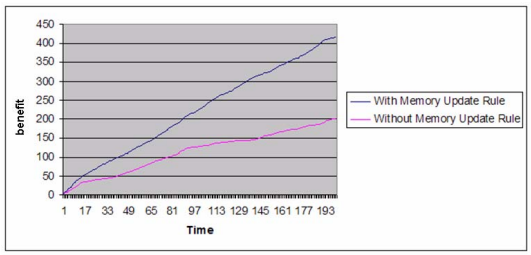
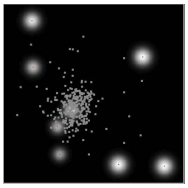
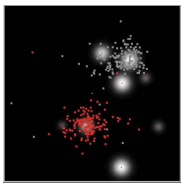


Fig 3: The average loss caused by insurgents with memory update rule and without memory update rule

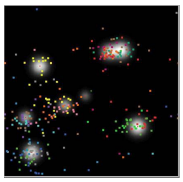
Fig. 3 illustrates that, in the dynamic environment, the insurgent group modeled with PSO model without the memory update rule fails to track the randomly moving optimal solution. As shown in Fig. 2b, all particles are trapped at the center of the environment. However, as shown in Fig. 2a, the insurgent particles that use the memory update rule can, in real time, track the dynamically moving target and surround themselves at the vicinity of the targets. This can help insurgent groups generate higher total loss on the targets.



(a)

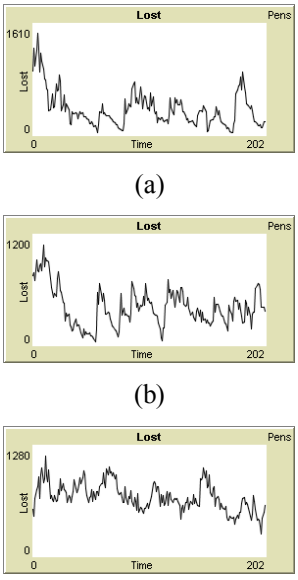


(b)



(c)

Fig 4. The results of insurgent particles for (a) one group, 300 insurgents, (b) two groups, 150 insurgents per group, (c) twenty groups, 15 insurgents per group



(a)

(b)

(c)

Fig 5. Total gained benefit at each time-step for (a) one group, 300 insurgents, (b) two groups, 150 insurgents per group, (c) twenty groups, 15 insurgents per group

Fig 6. The average benefit earned by insurgents with different group numbers

To investigate whether lacking unified leadership, planning, and effective communication among insurgent groups can inhibit or facilitate insurgents obtaining their goals, the performance of the insurgent particles for one insurgent group with 300 insurgents (Fig. 4a), two insurgent groups with 150 insurgents in each group (Fig. 4b), and twenty insurgent groups with 15 insurgent in each group (Fig. 4c) are compared. The results are shown in Fig. 5. The average loss caused by insurgents with different group numbers is shown in Fig. 6. With time increases, the average loss increases in all three situations. However, as shown in Fig. 6, the simulation of twenty insurgent groups can gain more total benefit and cause more damage to the authority than both the one group and the two groups do, although there is no unified leadership, planning or effective communication

among these twenty insurgent groups in the simulation compared to the one group insurgent simulation.

6. Discussion and Conclusion

In 1962, President John Kennedy advised the West Point graduates that they would have to deal with "…another type of war, new in its intensity, ancient in its origins—war by guerrillas, subversives, insurgents, assassins; war by ambush instead of by combat; by infiltration, instead of aggression, seeking victory by eroding and exhausting the enemy instead of engaging him....". After more than 40 years, this kind of warfare is still the most significant military tactic being used against American forces around the world. Discerning how insurgent groups interacting, learning and how emergent behaviors emerging from aggregate interactions in a dynamic environment is crucial for understanding insurgency.

In this paper, a modified PSO model is developed to simulate the complex interactions between the insurgent groups and targets and to analyze how an un-organized, un-planned insurgent riot can reach their effect-based operation in a highly dynamic environment. Our primary aim is to demonstrate how individual insurgent violence on dominant power targets can produce effect based operations, which usually requires highly organized and professional planning. We construct a novel agent based simulation model to examine the impact of different attack scenarios that the insurgent groups may conduct. The objective of this research is not to develop a tool for optimizing an optimizing attack strategy that can cause highest loss to authority, but to apply the particle swarm metaphor as a model of insurgent social adaptation for the dynamically changing environment and provide an insight and scientific understanding of the insurgency warfare. Results from our simulation have shown that lack of unified leadership, planning, and effective communication are not the necessary requirements for insurgents to attain their objective.

Acknowledgments

Prepared by Oak Ridge National Laboratory, P.O. Box 2008, Oak Ridge, Tennessee 37831-6285, managed by UT-Battelle, LLC, for the U.S. Department of Energy under contract DE-AC05-00OR22725.

Reference

[1] *T. Sun, The Art of War. Oxford: Oxford University Press, 1963.*

[2] *C. Guevara, Guerrilla Warfare. Lincoln: University of Nebraska Press, 1998.*

[3] *T.-t. Mao, On Guerrilla Warfare Champaign. Illinois: University of Illinois Press, 2000.*

[4] *J. Arquilla and D. Ronfeldt, Swarming and the Future of Conflict, vol. DB-311. Santa Monica, CA: RAND, 2000.*

[5] *J. M. Epstein, "Modeling civil violence: An agent-based computational approach," PNAS, vol. 99, pp. 7243-7250, 2002.*

[6] *S. Y. Yiu, A. Gill, and P. Shi, "Investigating Strategies for Managing Civil Violence Using the MANA Agent Based Distillation," presented at Land Warfare Conference, Brisbane, Australia, 2002.*

[7] *R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," presented at Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995.*

[8] *M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," presented at Proceedings of the 1999 Congress on Evolutionary Computation, Washington, DC, USA, 1999.*

[9] *M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," IEEE Transactions on Evolutionary Computation, vol. 6, pp. 58-73, 2002.*

[10] *J. Kennedy, "The particle swarm: social adaptation of knowledge," Indianapolis, IN, USA, 1997.*

[11] *J. F. Kennedy, R. C. Eberhart, Y. Shi, and I. NetLibrary, Swarm intelligence. San Francisco: Morgan Kaufmann Publishers, 2001.*

[12] *C. Cecilia Di, P. Riccardo, and C. Paolo Di, "Modelling Group-Foraging Behaviour with Particle Swarms," Lecture Notes in Computer Science, vol. 4193/2006, pp. 661-670, 2006.*

[13] *B. Anthony, S. Arlindo, S. Tiago Ferra de, O. N. Michael, M. Robin, and C. Ernesto, "A Particle Swarm Model of Organizational Adaptation," presented at Genetic and Evolutionary Computation – GECCO 2004, Seattle,WA, USA, 2004.*

[14] *A. S. Silva, F. Tiago; O'Neill, Michael; Matthews, Robin; Costa, Ernesto "Investigating Strategic Inertia Using OrgSwarm," Informatica, vol. 29, pp. 125-141, 2005.*

[15] *X. Cui, C. T. Hardin, R. K. Ragade, T. E. Potok, and A. S. Elmaghraby, "Tracking non-stationary optimal solution by particle swarm optimizer," presented at Proceedings. 6th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/ Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks, 23-25 May 2005, Towson, MD, USA.*

[16] *S. Tisue, "NetLogo: A Simple Environment for Modeling Complexity," presented at International Conference on Complex Systems, Boston, MA, 2004.*

[17] *R. W. Morrison and K. A. De Jong, "A test problem generator for non-stationary environments," Washington, DC, USA, 1999.*