Vol 6. No. 2, June 2013

African Journal of Computing & ICT

© 2013 Afr J Comp & ICT – All Rights Reserved - ISSN 2006-1781
www.ajocict.net

An Intelligent Particle Swarm Optimization Model based on Multi-Agent System

[1]N. V. Blamah
Department of Computer Science,
University of Jos, Jos, Nigeria
blamahn@yahoo.com

A. O. Adewumi
School of Computer Science,
University of KwaZulu-Natal,
Westville, Durban, SA;
adewumia@ukzn.ac.za

G. M. Wajiga
Department of Computer Science,
Federal University of Technology,
Yola, Nigeria;
gwajiga@gmail.com

B. Y. Baha
Department of Mathematical Sciences,
Taraba State University, Nigeria;
bybaha@yahoo.com

[1]Corresponding author

ABSTRACT

Particle swarm optimization techniques are typically made up of a population of simple agents interacting locally with one another and with their environment, with the goal of locating the optima within the operational environment. In this paper, a robust and intelligent particle swarm optimization framework based on multi-agent system is presented, where learning capabilities are incorporated into the particle agents to dynamically adjust their optimality behaviours. Autonomy is achieved by the use of communicators that separate an agent's individual operation from that of the swarm, thereby making the system more robust.

Keywords: Particle Swarm Optimization, Multi-Agent System, Intelligence, Autonomy

African Journal of Computing & ICT Reference Format:
N. V. Blamah, A. O. Adewumi, G. M. Wajiga & B. Y. Baha (2013). An Intelligent Particle Swarm Optimization Model based on Multi Agent System. Afr J. of Comp & ICTs. Vol 6, No. 3. Pp1-8

1. INTRODUCTION

Particle Swarm Optimization (PSO) belongs to the field of swarm intelligence, which is a collection of techniques based around the study of collective behavior in decentralized, self-organized systems. PSO is inspired by the social foraging nature of animals such as the flocking behaviors of birds moving towards an optimal goal [4].
The technique was developed by Kennedy and Eberhart [8] as a stochastic global optimization method for continuous functions, with the idea that the particles, situated within an *environment*, move towards fitter members of the swarm and generally bias their movements towards historically good areas of the environment.

The particles try to achieve the optimal goal by cooperating with their neighbours in addition to taking independent decisions and *actions*. Although there is normally no centralized control structure dictating how individual particles should behave, local interactions among the agents often lead to the emergence of global behavior. Since the development of the early model in Kennedy and Eberhart [8], there have been numerous advances on the model [7, 15, 17] emphasizing on different aspects with the view to improving the overall performance. These have viewed the problem from the traditional sense of implementation, where the particles in the system lack the basic qualities that typical agents possess.

Implementing the PSO scheme from conventional *supervised* approach has a number of setbacks. To begin with, it lacks the autonomy a system of such nature deserves; the particles achieve their goals by executing a fully *elaborate program*, and they are limited by the highly cohesive implementation since they have a *uniform algorithm* for execution that prohibits self-sufficiency and intelligence. But PSO innately fits a system where the agents are delegated goals in some high level way, and then the agents decide for themselves how best to accomplish their goals – the agents here have the ability to decide how best to act so as to accomplish their delegated goals.

The scalability that is desired in such a system is also lacking since a static population is usually assumed during optimization process. There are variants of PSOs where membership of the total swarm population grows or shrinks dynamically [2, 6, 11, 12], depending on the strengths and behaviors of members of the system. Therefore scalability in the form of population growth or reduction is desirable, which is quite difficult to do in a monolithic and highly cohesive system. An ideal model will therefore not assume fixed neighbourhood of particle agents; agents should be capable of moving from one neighbourhood to another, which may have different sizes. This feature is inherently available in Multi Agent System (MAS), since each agent is treated separately.

Complex communication patterns arise among the particles within a typical PSO system, and if not properly implemented, communication can be inherently synchronous. This drastically degrades the overall systems performance especially if the population size is too high. With the MAS approach, there is a natural asynchronous communication, because concurrency is natural, as agents execute independently. Implementing a population-based algorithm without regards for isolated treatment of each candidate of the population makes the entire process a complex one, especially with a large population. Since each particle has a separate behaviour from other particles within the system, it is highly desirable to model the system as such.

A number of other characteristics of PSO [18] that make it suitable for MAS exist; Natural algorithm: it is based on the behavior of real birds/fish which are real agents; Parallel and distributed algorithm: the swarm is a population of agents move simultaneously, independently and without a supervisor; Cooperative particles: each agent chooses a new point partly based on the information received from other agents. To address these issues, literatures on MAS-based PSOs exist [1, 10, 16]. However, the emphases in these literatures are based on modeling, implementations, and load balancing/fault-tolerance. In this paper, we present a robust and intelligent PSO framework based on MAS where learning capabilities are incorporated into the particle agents to adjust their optimality behaviours.

2. BRIEF BACKGROUND OF INTELLIGENT PARTICLE SWARM OPTIMIZATION SYSTEM

Intelligence is connected with the way *reasoning* is carried out in order to arrive at a conclusion. It refers to the ability to come to correct conclusions about what is true or real, and about how to solve problems [5]. Reasoning in a general sense is a broad subject matter that refers to the capacity to make sense of things, to establish and verify facts, and to change or justify practices and *beliefs* [9]. We use a practical reasoning model to agency [14, 20] to represent intelligent actions. Practical reasoning is the capacity for resolving, through reflection, the question of what is to be done. Deliberation of this kind is practical in the subject matter, insofar as it is concerned with action. But it is also practical in its consequences or its issue, insofar as reflection about action itself directly moves an agent to act [19].

The notion of practical reasoning agent is modeled by looking at an agent as having a set of *beliefs* which are the perceptions of the agent's operating environment, set of *desires* which are the various options at the agent's disposal, and set of *intentions* which are selections made from the desires by filtering the best options. Practical reasoning is comprised of two major components [20]: Deliberation (what state of affairs to achieve, which becomes the agent's *intention*) and Means-Ends Reasoning (how to achieve the intentions, which yields a *plan*). The intentions here are the future directed intentions, which simply represent the state of mind of the agent, with no actions taken.

Deliberation is modeled as option generation and filtering processes [20], which are thus described as follows:

- Option generation function takes the current beliefs and current intentions in order to yield the agent's *desire* set.

Thus,

$option: 2^{Bel} \times 2^{Int} \rightarrow 2^{Des}$ ………….. (1)

- Then the *intentions* to be committed to are obtained by filtering and selecting the best options using the following function:
-

$filter: 2^{Bel} \times 2^{Des} \times 2^{Int} \rightarrow 2^{Int}$……(2)

The agent updates its belief through a belief review function defined as:

$brf: 2^{Bel} \times Per \rightarrow 2^{Bel}$ ……………… (3)

where Per is a set of percepts of the operating environment.

Practically, a particle agent within the swarm will arrive at an alternative by first deliberating on the available options, and then make decisions by acting on the best alternatives. A Particle agent starts by having a particular set of beliefs which are stored in a belief database, and then commits to intentions for actions based on the initial beliefs and desires. As time progresses, the beliefs about the real world may be refined, and the agent's desires and intentions may also be redefined to reflect the changes in the belief database. After an agent deliberates and produces intentions to which it is committed, the agent needs to plan how to accomplish the intentions based on the current state of the environment (agent's belief) and the actions that are available to it. This is means-ends reasoning.

So a particle agent perceives its environment and then adjusts its belief database appropriately, upon which it derives its intentions, and then uses practical reasoning to take an action that alters the real world, which advances the system towards a potential solution.

3. PARTICLE SWARM OPTIMIZATION AS MULTI- AGENT SYSTEM

The properties of the conventional Particle Swarm Optimization (PSO) model make it a suitable candidate for Multi-Agent System (MAS) implementation. In the MAS-based PSO, we model into MAS the appropriate qualities of the conventional PSO and then introduce concepts that improve on the overall systems performance as an optimization technique.

Agents within a MAS consider problems by weighing conflicting considerations for and against competing options, where the relevant considerations are provided by what the agent desires or values and what the agent believes [3, 20]. An agent takes action by first deliberating on *what* state of affairs to achieve from the available options, which represents its *Intentions* that alter its state of mind. Then the agent reasons on *how* to achieve the chosen state of affairs, which results in a plan of how best to achieve the option. By so doing, intelligence is built into the system.

The idea behind the neighbourhood approach in PSO [13] was the reduction of the global information exchange scheme to a local one, where information is diffused only in small parts of the swarm at each iteration. Each particle assumes a set of other particles to be its neighbours and, at each iteration, communicates its best position *only* to these particles, instead of to the whole swarm. Thus, information regarding the overall best position is initially communicated only to the neighborhood of the best particle, and successively to the rest through their neighbours.

With this approach, a particle is tied to a fixed neighbourhood for interaction with each iteration without *planning* ahead and envisaging better fitness values with other neighbourhoods within the same iteration; so when an agent belongs to a neighbourhood, it does not share information with agents outside the neighbourhood within a single iteration to see if such interaction will yield better fitness values. In our agent-based approach to PSO, cohesion and diversity within the search space is increased in order to avoid *blind commitment* that easily gets agents trapped in specific local minima.

The agents use the Believe-Desire-Intension (BDI)-like reasoning and dynamically alternate neighbours (whatever neighbourhood topology is used) in the search process. In each iteration, an agent computes several fitness values in parallel (based on neighbourhood bests of the main neighbourhood and other neighbourhoods), and keeps the history for future reference. Depending on the best results obtained with time, the agent sticks to neighbours that yield better fitness values. So the best global behaviour emerges as the agents interact.

In the conventional PSO, an agent's decision, and hence its action, is influenced by both personal cognitive component and social component. These innate attributes of PSO place it as good candidate for MAS design, where the cognitive component is modeled as part of the individual agent's execution, and the social component as part of the multi-agents' execution while the agents interact. This is accomplished by the use of communicators, a concept that is defined later. Learning is highly desirable within a complex system like PSO. A belief database will be designed, which is well suited for keeping the agents' learned experiences overtime as the agents keep refining their beliefs. The qualities described here make the PSO particle agents more autonomous and intelligent.

4. THE MATHEMATICAL MODEL

Let us consider the original Particle Swarm Optimization (PSO) model represented as:

$$V_i^{t+1} = \omega V_i^t + C_1 R_1 (pBest_i^t - X_i^t) + C_2 R_2 (lBest_i^t - X_i^t) \text{......} \quad (4)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \text{.....................................} \quad (5)$$

where t denotes the iteration counter, R_1 and R_2 are random numbers distributed within [0,1], C_1 and C_2 are weighting factors representing the cognitive and social parameters respectively, $pBest_i^t$ is the best position known so far by the particle i, $lBest_i^t$ is the neighbourhood best, V_i^{t+1} is the velocity of particle i in iteration t+1, and X_i^{t+1} is the particle's position in iteration t+1.

Since variants of the PSO [11, 12] show that population can grow or shrink, it means within each iteration, the particle agents in a swarm can communicate with any number of agents ranging from 1 to n-1, n being the total swarm population. So in the agent-based PSO, an agent does not limit its communication to a fixed number of agents, and also to a fixed neighbourhood all the time, for the reason that follows.

Each agent is assigned few other prospective neighbourhoods besides the main neighbourhood it belongs to. In each iteration, the $lBest_i^t$ values of these prospective neighbourhoods are communicated to the agent alongside the $lBest_i^t$ value from the main neighbourhood, and the agent computes both prospective and main fitness values, velocities and positions in parallel. After the computations, the agent stores all the prospective values in a belief database through a vector (hereafter called prospective neighbourhood vector), but uses the main fitness value to keep membership of the main neighbourhood by moving in its direction. After a certain time-stamp equivalent to the size of the belief database, agents take appraisal of their execution history and compare them with the values in the belief database.
The neighbourhoods of the system is then reformulated based on more promising fitness values of agents up to the time of appraisal, and the agents will move in the directions of best fitness values. The time stamp is reinitialized and the process is repeated.

The agent choses more promising neighbours based on its previous knowledge and experience. The desires of the agent increases towards more promising neighbours and it updates its beliefs, and subsequently gets attracted towards more promising regions.

Definition 1: We define the state of environment in which the agent's search space may be as follows:

$$E = \{P_1, P_2, \ldots P_N\} \text{.................................} \quad (6)$$

where $P_i = (P_{i1}, P_{i2}, \ldots P_{in})^T$ is the best positions ever visited by each particle agent, representing the present state ($i = 1, 2, \ldots, N$, N being the population size and n the current iteration counter).

Definition 2: Each particle agent has a range of actions at its disposal, which are the consequences of agent's invocation. If we generally define the set of actions as $Ac = \{\alpha, \alpha', \ldots\}$, then specifically, the ith particle agent in the system has these actions:

$$Ac_i = \{V, X, N^*, \sigma, C\} \text{................................} \quad (7)$$

where V and X are the velocity and position functions respectively of the agent within the main neighbourhood, N^* is the prospective neighbourhood vector described earlier, which is an action that computes the values for the belief database and updates same, and C is a communicator which the agent uses to communicate with other agents, thereby separating the social activities of the Multi-Agent System (MAS) from the individual agent's activities. σ is a recap function that permits an agent to appraise its history from the last time stamp in order to decide whether or not to change neighbourhood.

Particle agents in the search space have single-minded commitments, because an agent continues to maintain an intension of improving the fitness values within a particular neighbourhood until it believes either that the intension has been achieved, or else it is no longer a more feasible option to remain in that neighbourhood, in which case it is rational for the agent to move away to a more promising neighbourhood. We assume that the size of neighbourhoods can vary because there may be increase or decrease in population, agents can move from one neighbourhood to another, or any other unforeseen factor.

Definition 3: A run, r, of an agent in an environment is a sequence of interleaved environment states and actions. If we let R be the set of all such runs, then we have:

$$R = \{r, r', \ldots\} \text{..} \quad (8)$$

Let R^{Ac} be the subset of these that end with an action and R^E be the subset of these that end with an environment state.

Definition 4: When a particle agent invokes an action on an environment, it transforms the environment state, and this effect is modeled by the state transformer function defined as:

$\tau: R^{Ac} \rightarrow 2^{E}$ .... (9)

where 2^{E} is the power set of E.

This means that from runs which end with actions taken by a particle agent, the system will always end up in a particular environment state; taking an action by a particle agent on a previous environment state moves the environment to another state.

Definition 5: We define an environment as a tuple:
$\xi = \langle E, e_0, \tau, T, \eta \rangle$ .... (10)

where E is the set of environment states, $e_0 \epsilon E$ is an initial state, τ is a state transformer function, T is the active topology in the environment, and η is a set of neighbourhoods defined as $\eta = \{n_1, n_2, \ldots, n_k\}$, where k is the total number of neighbourhoods.

Definition 6: In a MAS, the agents drive the system. The state of the environment emerges as a result of the agents' actions – based on the behaviours and interactions among the agents. Since actions are produced by agents when they execute in the system, we model agents as function of execution, which yield an action (whose effect is the state transformer function).

Thus, a particle agent is defined as:

$A : R^{E} \rightarrow Ac$ .... (11)

So if an action, say the position update function $X \epsilon Ac$, is desired of a particle agent A', the agent produces this action by executing on an existing run ending with environment state, say r', which is its current position, as follows:

$$X = A'(r')$$

This leaves the run to end with an action. The effect of taking this action, which is modeled by the state transformer function, τ, is to produce a new environment state.

Definition 7: We define Swarm S to be the set of all agents, as follows:
$S = \{A_1, A_2, \ldots, A_n\}$ .... (12)

and the Swarm System is thus defined as $R(S, \xi)$, where R is the set of all runs.

5. ALGORITHM AND DISCUSSIONS

Having established the definitions and the relationships above, we now describe the algorithms that aid practical reasoning particle agents to execute within the swarm. Particle agents need to *plan* ahead and envisage better fitness values with other neighbourhoods within the same iteration by sharing information with agents outside the main neighbourhood. The particle agents thus maintain single-minded commitments of achieving better fitness values by making practical reasoning and dynamically alternating neighbours in the search process.

Within each iteration, an agent computes several fitness values in parallel and keeps the history for future reference, and as time progresses, the agent sticks to neighbours that yield better fitness values. So the best global behaviour emerges as the agents interact. As earlier explained, Practical reasoning = deliberation + means-ends reasoning

Deliberation = option generation function and Filtering function

Means-end reasoning = planning

These components of the practical reasoning are obtained below.

The environment state as initially perceived by particle agents, denoted by equation (6), represents the initial belief of the agents. So $B = B_o = \{P_1, P_2, \ldots P_N\}$
The agents will initially look at the state of affairs to achieve as their initial intentions. The desired state of affair at the beginning is to apply the velocity function of equation (4) and then take a move using the position function of equation (5).

These are initialized in $I = I_o$.

Having obtained the initial beliefs and the initial intentions, an agent can be executed using the algorithm in figure 1, where lines 1 and 2 are the initial beliefs and intentions, respectively.

```
1: B = $B_o$;          // Initial beliefs
2: I = $I_o$; //initial intentions
3: while true do
4:      get next percept $\rho$ of the swarm by making a call to communicator C;
5:      $B \leftarrow brf(B, \rho)$;
6:      $D \leftarrow option(B, I)$;
7:      $I \leftarrow filter(B, D, I)$;
8:      $\pi \leftarrow plan(B, I, Ac)$;
9:      while not (empty($\pi$) or succeeded(I, B) or impossible(I, B)) do
10:          $\alpha \leftarrow head(\pi)$;
11:          execute ($\alpha$);
12:          $\pi \leftarrow tail(\pi)$;
13:          get next percept $\rho$ of the swarm by making a call to communicator C;
14:          $B \leftarrow brf(B, \rho)$;
15:          if reconsider(I, B ) then
16:               $D \leftarrow option(B, I)$;
17:               $I \leftarrow filter(B, D, I)$;
18:          end-if
19:          if not sound($\pi$, I, B ) then
20:               $\pi \leftarrow plan(B, I, Ac)$;
21:          end-if
22:      end-while
23: end-while
```

Figure 1: Agent-based Particle Swarm Optimization

The percept ρ on line 4 is a process where an agent perceives its environment by using its previous belief, and then communicating with other agents through the communicator component C of equation (7) to get the current $lBest$ value. Line 5 is the Belief update process, which is obtained using equation (3). Lines 6 and 7 are the deliberation process that is carried out by option generation and filtering using equations (1) and (2) respectively, to produce the corresponding Desires and Intentions.

Having obtained these values, a particle agent carries the means-ends reasoning by formulating a plan on line 8 using the new values of the belief (line 5), intention (line 7), and the available actions at its disposal (represented by equation (7)). The agent can then execute its stated intentions, which is accomplished by the intention loop [20] from lines 9 through 20.

6. CONCLUSION AND FUTURE WORK

The design presented above provides for natural way of implementing the Particle Swarm Optimization (PSO) algorithm. Particle agent's intentions and actions as influenced by both personal cognitive component and social component are separated into the individual agent's execution and the multi-agents' execution respectively, as the agents interact. This is realized by the use of the communicators. This way, autonomy is achieved within the particle agent system. Also since multi-agent systems are inherently modular, it is easier to add new agents to a multi-agent system than it is to add new entities to a non-multi-agent-based system.
The desired learning capabilities are also incorporated into the design, since agents make parallel computations of fitness values and store them in the belief database for future reference. These qualities make the PSO particle agents intelligent and more autonomous.
In order to have a wider scope of application for the intelligent multi-agent based PSO model, we intend, in the future, to discretize and apply it to the Travelling Salesman Problem.

REFERENCES

[1] Ahmad, R., Lee, Y., Rahimi, S., and Gupta, B. (2007). A Multi-Agent based approach for Particle Swarm Optimization, *In IEEE Publications*, 1-4244-0945-4/07, Paper 17.

[2] Bell, W.J., Roth, L., and Nalepa, C. (2007). *Cockroaches: ecology, behavior, and natural history*, The Johns Hopkins University Press.

[3] Bratman, M.E. (1990). *What is Intention?* In Cohen, P.R., Morgan, J.L., and Pollack, M.E., editors, Intentions in communication, The MIT Press: Cambridge, MA.

[4] Brownlee, J. (2011). *Clever algorithms: nature-inspired programming recipes*. 1st ed., Lulu. http://www.CleverAlgorithms.com

[5] Davidson, H. (1992). *Alfarabi, Avicenna, and Averroes, on Intellect*, Oxford University Press, pp.6.

[6] Halloy, J., Sempo, G., Caprari, G., Rivault, C., Asadpour, M., Tache, F., Said, I., Durier, V., Canonge, S., Amé, J.M., Detrain, C., Correll, N., Martinoli, A., Mondada, F., Siegwart, R., and Deneubourg, J.L. (2007). Social integration of robots into groups of cockroaches to control self-organized choices, *Science*, November, 318(5853), pp.1155–1158.

[7] Kennedy, J. (1999). Small worlds and mega-minds: Effects of neighborhood topology on Particle Swarm performance. In Proceedings of the 1999 Congress on Evolutionary Computation.

[8] Kennedy, J., and Eberhart, R. (1995). Particle Swarm Optimisation. In: Proceedings of the IEEE Conf. on Neural Networks, Perth.

[9] Kompridis, N. (2000). So we need something else for reason to mean, *International Journal of Philosophical Studies*, 8(3), pp.271-295.

[10] Lorion, Y., Bogon, T., Timm, I.J., and Drobnik, O. (2009). An Agent based Parallel Particle Swarm Optimization – APPSO, *In IEEE Publications*, 978-1-4244-2762-8/09

[11] Obagduwa, I.C. (2012). Cockroaches optimization algorithms, seminar paper presented at the Progress Seminar, March 3rd, UKZN, Durban.

[12] Parpinelli, R.S., and Lopes, H.S. (2011). New inspirations in swarm intelligence: a survey, *International Journal of Bio-Inspired Computation*, 3(1), pp.1–16.

[13] Parsopoulos, K.E., and Vrahatis, M.N. (2010). *Particle Swarm Optimization and Intelligence: advances and applications*, Information Science Reference, Hershey, (NY).

[14] Petrie, H.G. (1971). **Philosophy & Rhetoric** © 1971, Penn State University Press.

[15] Poli, R., Kennedy, J., and Blackwell, T. (2007). Particle Swarm Optimization, an overview. *Swarm Intelligence*, 1, pp.33–57.

[16] Shangxiong, S. (2008). A Particle Swarm Optimization (PSO) algorithm based on multi-agent system, In IEEE International Conference on Intelligent Computation Technology and Automation, 978-0-7695-3357-5/08

[17] Shi, Y., and Eberhart, R.C. (1998). A Modified Particle Swarm Optimizers. In Proceedings of the IEEE International Conference on Evolutionary Computation, pp.69–73.

[18] Sivanandam, S.N., and Deepa, S.N. (2008). *Introduction to Genetic Algorithms*, Springer-Verlag Berlin Heidelberg

[19] Wallace, R.J. (2009). *Practical Reason,* ***The Stanford Encyclopedia of Philosophy*****,** Edward N. Zalta (ed.), URL = http://plato.stanford.edu/archives/sum2009/entries/practical-reason/

[20] Wooldridge, M. (2009), *An introduction to multi-agent systems*. 2nd ed., John Wiley & Sons Ltd, Chichester.

Author's Brief

Nachamada Vachaku Blamah is a Senior Lecturer with the University of Jos, Nigeria. He obtained his Bachelors of Technology, Master of Science, and Doctorate degrees in Computer Science. Dr. Blamah is a member of the IEEE Computational Intelligence Society and the Computer Professionals (Registration Council of Nigeria), and his research interests are mainly in the areas of computational intelligence and multi agent systems.

Gregory Maksha Wajiga is a professor of Computer Science at Modibbo Adama University of Technology, Yola, Nigeria. He has published extensively in combinatorics, optimization and modeling. Prof. Wajiga is actively involved in research, teaching and supervision. Prof. Wajiga is a member of the Nigeria Computer Society, and has at various times been appointed ICT consultant for Adamawa State, National Planning Commission and World Bank supported projects. He is married and has three children.

Benson Yusuf Baha is a Senior Lecturer with Taraba State University (TSU), Jalingo. Presently, he is the Head of Department of Mathematics & Computer Science. He started lecturing with Federal University of Technology, Yola (now MAUTECH) in 2002. In 2008, he joined Afribank Nigeria Plc (now Mainstreet Bank) as Branch Manager and later became the Regional Computer Coordinator North East. He joined TSU in 2012 as Senior Lecturer. Dr. Baha had his B.Tech degree in Computer Scinence from Abubakar Tafawa Balewa University, Bauchi (2000), M.Sc. in Computer Science from Ahmadu Bello University, Zaria (2008) and Ph.D. in Computer Science from MAUTECH, Yola (2012). His M.Sc. and Ph.D. are biased towards ANN and web-database. His present research interest is in soft computing.