ANT COLONY OPTIMIZATION APPLIED TO AN AUTONOMOUS MULTIAGENT GAME

Rubén Parma[*], Wilmer Pereira[+] y Juan Rada[#]

Departamento de Sistemas, Avda. Las Industrias, Núcleo Obelisco
Universidad Centro Occidental Lisandro Alvarado[*]
Escuela de Ingeniería Informática, Avda. Teheran, Montalbán,
Universidad Católica Andrés Bello[+]
Departamento de Electrónica, Urb. Chucho Briceño, 3era etapa
Universidad Fermín Toro[#]
Venezuela
E-mail: parmaia@gmail.com, wpereira@ucab.edu.ve, jcrada@gmail.com

KEYWORDS
Ant Colony Optimization, Cooperative Games, Distributed Intelligence.

ABSTRACT

This article shows the implementation of an algorithm based on Ant Colony Optimization [Dorigo2001] to produce a multiagent strategy in the game of a*nimats*. This game takes place on a grid of `MxN` squares where *animats* move according to a given set of rules in order to find as many bombs as possible without exceeding the top movements established by the game's difficulty level. It was developed using Delphi® 2006 for Win32 and we are currently working on a Java version using Multi Objective Genetic Algorithms aiming to make a comparison of both techniques for these kind of games.

INTRODUCTION

Artificial intelligence techniques in games may be used to approach reality with entertainment means in mind. Classic methods for solving games are often challenged because of the game's difficulty nature where conditions tends to get very complex.

Strategies depends on the game's nature. These strategies may be: a) *cooperative* when players share a common interest and act pursuing a common goal, b) *competitive,* when players' interests are in conflict and each one acts pursuing a personal interest, or c) *mixed games* when interests are shared and in conflict .

Some classic approaches to produce a strategy in competitive games are *minimax* and *alpha-beta pruning* where decisions are made based on a search tree. These algorithms had been successfully implemented in variations of chess, checkers, othello and many other games in which had proven efficacy and they are considered to be hard to defeat by humans [Russell2004].

All games mentioned previously are games of perfect information, where each player knows all decisions made by others at the time of making a decision. There are also games of imperfect information where a player may or may not have information about decisions made up to that point [Rada2005]. An example of these class of games is *Paper, Rock and Scissors* as well as many card games. This games require uncertainty handling (perhaps using fuzzy logic) and probabilistic theory [Champandard2003].

Furthermore, there are competitive games like *Go* (well-known in Asia) with simple rules but an important growth in possible movements as the match goes on. This situation makes unthinkable to approach a solution using algorithms like *minimax* or *alpha-beta pruning* given its high-resource requirements for such task. This is the main reason why a classic approach is not feasible and we must consider an emergent approach.

For example, in [Lujan2004] neural networks were evolved to produce a strategy based on competitive coevolution [Rosin1997] in order to learn the game of *Go*. In this approach, two populations evolve mutually so a fitness function is not externally established. This algorithm is inspired in [Lubbert2001] and [Richard1997] algorithms which learned to play *Go* and later competitions took place against GNUGo to analyze its efficacy. The modifications made in [Lujan2004] were meant to consider the different phases of the game: Fuseki, Middle Game and Yose.

Complexity arise in these kinds of games where each player must evaluate too many lines of play in order to choose a good strategy. As seen in [Lujan2004] is natural to choose evolutionary algorithms despite these algorithms' lack of explanations about the reasons for achieving its results.

Another approach in solving games is having multiple agents cooperating with each other in order to achieve a common goal. In this area, [Parker1998] shows a community of space ants cooperating to move asteroids. These ants move according to frustration and impatience variables where a higher frustration implies a higher impatience. Exceeding an established threshold of frustration would make the frustrated ant to require either help from other ants or abandon its current activity. These social rules are important when dealing with cooperative games.

In academic projects, cooperative and competitive games produce a high motivation among students in courses like software engineering or artificial intelligence. At Universidad Centroccidental Lisandro Alvarado, Emergent Computing is a course where sophisticated techniques of artificial intelligence are applied to a wide source of games in which students participate actively and enthusiastically. The game proposed in this article is oriented to serve as a platform for applying evolutionary techniques as well as many other techniques in artificial intelligence.
Finally, we propose an algorithm based in Ant Colony Optimization adjusted for multiple colonies of ants cooperating in the game of *animats* to find all bombs set in the grid. Each *animat* is itself an ant colony and each *ant* has different kinds of pheromones in order to make it able to take different decisions at each position in the grid according to the bomb it is looking for.

THE GAME

The game of *Animats* takes place in a grid of `MxN` squares containing obstacles, bombs and *animats*. These squares may be one-way or two-way (except for those containing obstacles) which determine the direction that an *animat* may take.

An *animat* (in this game) is defined as an entity able to choose and follow the directions designated to each square on the grid. Its aim is to find all bombs in the grid. Each *animat* represents a different colony of ants cooperating to find the shortest path for finding all bombs.

Originally, the distribution of bombs were random as well as the distribution of *animats*. However, all tests used both distributions fixed in order to measure parameters of interest among several experiments. Distribution of obstacles is done according to

```
grid(i,j)=obstacle if and only if (i,j mod 2 = 0)
```

Animats may move through the grid either vertically or horizontally depending on each square's assigned direction which may be N, S, E or W. These are assigned according to following conditions:

For each $square_{ij}$

1) if (i mod 4 = 0) then direction := N
    else direction := S;
2) if (j mod 4 = 0) then direction := W
    else direction := E;

Invalid directions must be removed afterwards (directions that lead to an obstacle or out of borders). The following picture shows an extract of the result from applying previous conditions.

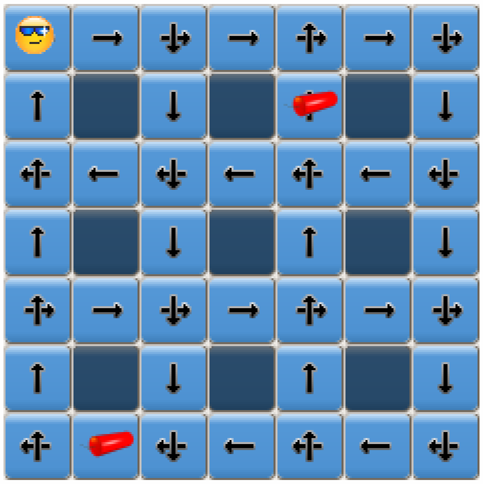

Figure 1: Sample Extract from a Configured Grid.

The difficulty level's for this game are based on the maximum movements available to find all bombs and are defined as follows

- Beginner: less than `2x(MxN)` movements.
- Intermediate: less than `(MxN)` movements.
- Expert: less than `(MxN)/2` movements.

THEORETICAL BACKGROUND

The strategy used for this application was Ant Colony Optimization (ACO) [Dorigo2001] due to its nature for solving problems in multiagent environments where cooperation is involved. Its foundations are based on ant behaviorism when searching for food. ACO's search mechanism is done by pheromones, a chemical substance released to the environment by these ants and it has capabilities to spread over it to serve as guidance to other ants. There are different kinds of pheromone indicating different kinds of messages (food, alerts, among others). Pheromone tends to evaporate if its not reinforced, this is what makes algorithms to find optimal or suboptimal paths due to the ant's recurrence in shorter paths.

These algorithms have been used for modeling problems like TSP [Dorigo1999], network routing[Crichigno2004], among others.

IMPLEMENTATION

In order to solve the problem presented at the game of *Animats,* the grid was modeled as a directed graph where each square represents a vertex and its edges are given by its connections with its adjacent neighbors.

The approach taken can be detailed as follows: *k* ants are set at the initial position of the *animat* and, for each epoch, ants follow the higher amount of pheromone left on the edges. Pheromone is initialized randomly but it takes shape through the course of time diminishing pheromone where paths are not suboptimal. According to [Dorigo1999], the chances to pick a node among the neighbors depends on the distance between the current node and the next one as well as the amount of pheromone; but in *Animats* the distance between nodes its always one (1) so the chances depends entirely on pheromones levels.

Ants will try to reach a bomb once its route has begun. Each ant will spread pheromone over the route taken with a magnitude inversely proportional to the distance of the path. Immediately after, all ants are relocated to the position where the bomb was found and another epoch begin while there still are bombs in the grid or the maximum number of epochs has been reached.

Within each epoch, paths are being reinforced continuously. A path will be reinforced as it is frequently used but a shorter path may have a higher reinforcement making ants to have tendencies deviating to it and in the following epochs it will be more used than others. Pheromone will evaporate eventually on less frequently routes while it will be increased on those most frequently used.

However, diversity in decisions must be taken into account so different ants from different colonies may take different decisions. Consider a case where ants x_A and x_B (from colonies *A* and *B* respectively) coincide at a given square *ij* but are aiming at different bombs so one common decision may not help both ants. This lead us to use different layers of pheromone, defining a layer L_1 for bomb b_1, L_2 for bomb b_2, L_i for bomb b_i. In other words, pheromone is spread over the *i*th layer when seeking the *i*th bomb, this way decisions will not interfere at a given square *ij*.

Another issue arise when using multiple *animats* so one *animat*'s route won't get mixed up with other *animat*'s route. Ignoring this would cause all *animats* to join eventually the same path. Given that one animat is an ant colony itself, a different kind of pheromone must be used for each animat, therefore one animat's ant won't misuse another animat's pheromone.

Finally, there is a communication mechanism among colonies by removing a bomb once a colony has reached it so no other colony would find the same bomb. This way, each colony will find its own set of bombs avoiding the possibility of shared bombs between colonies.

RESULTS

A set of 1200 experiments took place to test the algorithm, 600 for each grid varying sizes (11x11, 15x15). In each grid size, 200 experiments were made with 3, 4 and 5 animats.
Figure 2 shows one example of initial configuration for the grid including 3 animats and 10% of bombs chosen randomly.

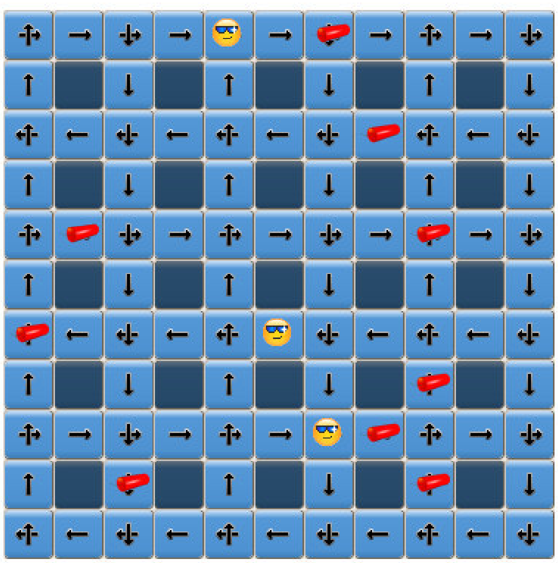

Figure 2: Initial Setup

Figure 3 describes the routes found by each animat to find all bombs in the grid. This result took 19 moves for each animat.

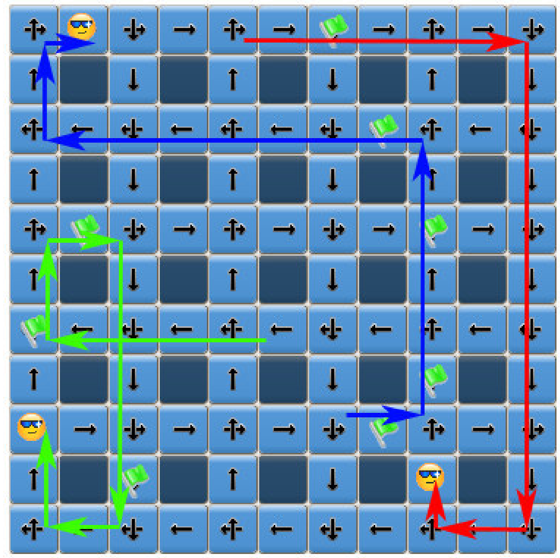

Figure 3: Solution Picked Randomly

Results reflect that this algorithm is capable of producing expert strategies for the game of animats according to the principles that rules the difficulty level as shown in the description of the game, these results can be appreciated in Figure 4 and Table 1.

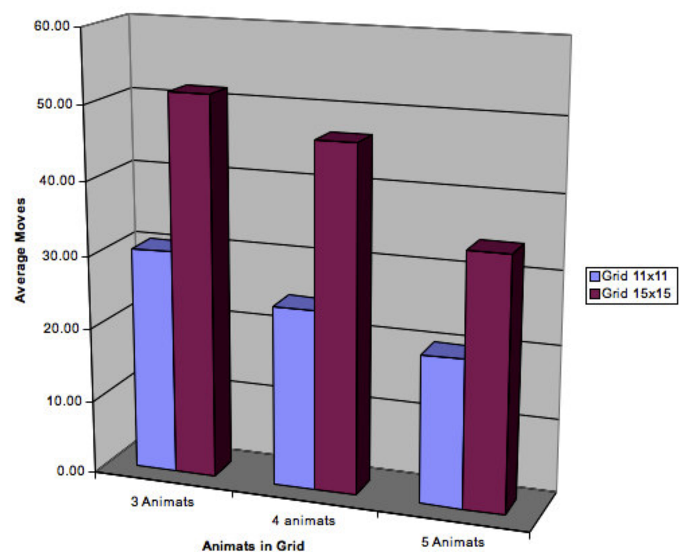

Figure 4: Graph Results

Average Moves			
	3 Animats	4 Animats	5 Animats
Grid 11x11	30.42	24.54	20.11
Grid 15x15	51.67	46.78	32.33

Table 1: Table Results

In following papers we will compare this approach with Multi Objective Genetic Algorithms in order to draw conclusions about virtues and flaws among these emergent techniques applied to the game of *Animats*.

CONCLUSIONS

Emergent computing is a promising approach to take on strategy games due to its low computational time, high efficiency solutions as well as its diversity. This approach takes us a step closer to solve games that present higher challenges requiring attractive solutions in terms of computational time and efficiency where non-classic techniques could show advantages over classic ones.
The approach taken to solve the game of *Animats* extends the use of Ant Colony Optimization to multiple colonies as an innovative alternative to this kind of games. We are aiming to make use of other emergent techniques in order to establish concrete comparisons among them. Expectations are set on these techniques.

Another plus is the remarkable motivation provoked by this game among students in the Master's course of Emergent Computing at Universidad Centroccidental Lisandro Alvarado - Venezuela.

REFERENCES

Champandard A, 2003, *AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors*, New Riders

Cordova C., Pereira W. et al, 2006, Desarrollo de una Estrategia, Inspirada en Teoría de Juegos, para Defensa y Ataque de Robots, *56^{ava} Convención Anual de Avance para la Ciencia* (ASOVAC2006), UDO, Cumana, Venezuela.

Crichigno J. and Barán B., 2004, A Multicast Routing Algorithm Using Multiobjective Optimization, *11th International Conference on Telecommunications* (ICT'04), Ceará, Brasil.

Dorigo M, Di Caro G. 1999, *The Ant Colony Optimization Meta-Heuristic*, McGraw-Hill

Dorigo M. and Stutzle T., 2001, The Ant Colony Optimization Metaheuristic: Algorithms, Applications and Advances, *Metaheuristics Handbook*, International Series in Operations Research and Management Science, Kluwer.

García G., Pereira W. et al, 2005, *Strength By Objective*: Una nueva estrategia de asignación de *fitness* para Algoritmos Evolutivos Multi-Objetivos, *XXXI Conferencia Latinoamericana de Informática* (CLEI2005), Cali, Colombia

Lubbert A. and Miikkulainen R, 2001, Coevolution a Go-Playing Neural Network, *Genetic and Evolutionary Computation Conference*.

Luján A. and Pereira W., 2005, Desarrollo de un Jugador Artificial de GO Basado en Redes Neurales Evolutivas, *55^{ava} Convención Anual de Avance para la Ciencia* (ASOVAC2005), UCV, Caracas, Venezuela.

Murphy R. 2000, *Introduction to AI Robotics*, MIT Press.

Parker L. 1998, ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation, *IEEE Transaction on Robotics and Automation*, Vol. 14, N. 2.

Rada J. 2005, *Robot Inteligente para Jugar Tres en Línea.* Universidad Fermín Toro, Cabudare. Venezuela.

Richard N. Moriarty D. et al, 1997, Evolving Neural Network to Play GO, *7th International Conference on Genetic Algorithms*, East Lansing, MI.

Rosin C., 1997, Coevolution Search Among Adversaries, *Thesis Doctoral*, University of California, San Diego.

Russell S. and Norvig P., 2003, *Artificial Intelligence: A Modern Approach*, Second Edition, Prentice Hall.

BIOGRAPHY

Rubén Parma is a graduated student of Informatic Engineering at Universidad Centrocidental Lisandro Alvarado - Venezuela, where he teaches Laboratory I. He is also an independent Web Developer and he is currently working on his Master Thesis of Computer Science in Artificial Intelligence at Universidad Centroccidental Lisandro Alvarado - Venezuela. One of his major hobbies is building robots using Lego Mindstorms NXT and other kinds of Lego.
http://parmaia.googlepages.com

Wilmer Pereira is a Professor at Universidad Simón Bolívar and Universidad Católica Andrés Bello, in Caracas, Venezuela. Also has a M.Sc. in Computer Networks from l'École Superieure de Électricité and a Ph.D. in Artificial Intelligence from Université de Rennes, both in France. He is author of various international publications in robotics and automatic learning and is the main leader of Grupo de Investigación en Inteligencia Artificial y Robótica - GIIAR.
http://www.ldc.usb.ve/~wpereira

Juan Rada is a graduated student of Computer Engineering at Universidad Fermín Toro - Venezuela, where he teaches Digital Systems Lab. He works as Senior Developer at Sinapsis S.A. in Costa Rica and is currently working on his thesis for Master of Computer Science in Artificial Intelligence at Universidad Centroccidental Lisandro Alvarado - Venezuela.
http://jcrada.googlepages.com