Title
CORBA-based analysis of multi agent behavior

Journal
Journal of Computer Science and Technology, 20(1)

ISSN
1000-9000

Authors
Bhattacharya, S
Banerjee, A
Bandyopadhyay, S

Publication Date
2005

Peer reviewed

A CORBA BASED ANALYSIS OF MULTI AGENT BEHAVIOR

Swapan Bhattacharya
Dept. of Comp. Sc. & Engg.
Jadavpur University, Kolkata-700032
India.
E-mail: bswapan2000@yahoo.co.in
Anirban Banerjee
Department of Computer Science
University of California, Riverside, CA 92507
USA
E-mail: anirban@cs.ucr.edu
Shibdas Bandyopadhyay
Department of Computer Science
Indian Statistical Institute, Kolkata – 700 018
E-mail: mtc0414@isical.ac.in

Abstract

An agent is a computer software that is capable of taking indepent action on behalf of its user or owner. It is an entity with goals, actions and domain knowledge, situated in an environment. Multiagent systems comprises of multiple autonomous, interacting computer software, or agents. These systems can successfully emulate the entities active in a distributed environment. The analysis of multiagent behavior has been studied in this paper based on a specific board game problem similar to the famous problem of GO. In this paper we have developed a framework to define the states of the multiagent entities and measure the convergence metrics for this problem. An analysis of the changes of states leading to the goal state is also made. We support our study of multiagent behavior by simulations based on a CORBA framework in order to substantiate our findings.

Key Words
Distributed Agents, Convergence metric, State transition, Simulation

1. Introduction

Computer software has been evolving unceasingly over the past few decades in leaps and bounds. One of the most flexible and effective connotation of this evolution is the development of Multiagent based systems. An agent is a migrating piece of computer software that is capable of independent action on behalf of its user or owner [1], [2]. It is an entity with goals, actions and domain knowledge, situated in an environment. Analysis of “behavior” exhibited by Multiagents is a most exciting and novel field of research being rapidly explored by major research organizations and firms. Multiagent Systems (MAS), the emerging subfield of Distributed Artificial Intelligence (DAI) aims to provide both principles for construction of complex systems involving multiple agents and mechanisms for coordination of independent agent’s behavior. Interestingly, behavior of different agents in the MAS must necessarily correspond towards the common goal for a reasonable result within time constraints. Therefore, it is of utmost importance in MAS to analyze the behavior of different agents and to determine whether they are in effect converging towards the goal. This is exactly what our research effort aims to achieve; in addition to this we can measure how close the current state of the MAS to the goal state is. A case study of the game of GO [3], [4], [5] is modified to find a suitable convergence metric. The problem to be discussed in this paper is based on the simplified version of the game of "GO" which consists of a number of predator agents who attempt to surround a target agent and capture it, the game is played out on an *nxn* board where *n* is finite number. Predator agents and the target agent can be placed on any cell in the board but may not be placed at the same location. Put simply, the predator agents try to capture the target agent while the latter tries to evade the former. This behavioral analysis of multiagents allows us a deeper insight into the efficacy of areas dealing with modeling of distributed systems based on multiagents and specifically those involved with target acquisition and tracking. This study is imperative for analyzing the behavior of multiagent based sensor networks [6] in a Pervasive computing [7] environment. Large amounts of data gathered by these sensor nodes deployed densely within these distributed networks must be preprocessed on the fly and must be relayed to the respective fusion points for deriving meaningful information out of the data streams. Multiagents provide an efficient way to achieve this goal. Copious amounts of literature have been published with respect to this exciting new area. An overview of multiagent systems including the requisite distributed algorithms and analysis of stabilization constraints have been cogently described in [1] and [2]. Basic distributed communication protocols described, point towards development of a non-blocking communication scheme in order to remove the side effects associated with waiting, such as deadlock. Our attempt has been to primarily develop convergence metrics to analyze the behavior of the entities which are part of the problem. In this paper we have developed convergence metrics for measuring the effectiveness of the algorithms implemented in reaching towards a final goal state leading to capture of the target. Also, [3] deals with intelligent agents or interactive agents with respect to the telecom scenario wherein the paradigm of "***information any time, at any place, in any form***" is now a major factor in the development of any system, these are based on services like service customization and service provisioning. The game of GO which has been described with relation to multiagent systems in [4], [5] and [6] has been used as a test framework to study the behavior of MAS. A comprehensive discussion of sensor network architectures dealing with sensor nodes, fusion points, mobility support have been described in [7]. Pervasive computing which relies heavily on the data accumulated and processed by such sensor systems have been discussed in [8]. One of the application areas for the behavioral analysis of multiagent systems is distributed fault tolerance system using multiagents to detect faults in the system. Such systems have been described in [9]. [10], [12] and [13] describe the relation of JAVA based technologies in the domain of multiagents, from transparent migration of agents to monitoring systems. [11] and [14] describe intelligent multiagent communication systems wherein the messages passed contain information related to the management of the entire network. This can be effectively harnessed for enterprise wide applications and used for CORBA based architectural analysis of latency and other factors over high speed ATM networks. [15] provides an interesting look into the possible implementation areas of the concepts developed in this paper based on the CORBA framework.

Finally, the scope of our research effort is to develop and compare various convergence strategies towards the final goal state which would lead to a certifiable conclusion to the problem. We have considered a simplified version of the problem where there are only one target and four predators. Metrics based on the strategies are developed and their performances vis-à-vis each other are compared to obtain an insight into the efficacy of the nuances related to the system under consideration. Algorithms to simulate the behavior of the various entities involved are developed and analyzed.

2. Functional Specification

The board consists of an *n x n* square matrix on which one target agent and four predator agents are placed .They can move only according to certain conditions. The aim of the predator is to capture the target. Position of a particular agent in the square matrix is defined with the help of two parameters row and column e.g. if T is the target currently situated in the i^{th} row and j^{th} column then we denote it by T_{ij}. We define 'capture' to be the situation where the predators in the four surrounding squares except the diagonal ones surround the target or the target and the predator are in the same square. The target and the predator agents are placed at random on the board at the beginning of the problem, however ensuring that the target and the predator agents are not assigned the same cell and that no two predator agents are assigned to the same cell either.

2.1 Valid Moves

There are five types of moves predator and target can take:
Shift-left: $X_{ij} \text{->} X_{ij-1}$ $0<j<n$
Shift-right: $X_{ij} \text{->} Xi_{j+1}$ $0<j<n$
Shift-down: $X_{ij} \text{->} X_{i+1j}$ $0<i<n$
Shift-up: $X_{ij} \text{->} X_{i-1j}$ $0<i<n$
No move: $X_{ij} \text{->} X_{ij}$

When the values of *i* and *j* are extreme then shift moves result in 'No move'.

Distance: Distance between the two cells is the minimum number of cells required to be traversed to reach the destination cell from the source cell. There may be more than one unique way to reach the destination cell from the source cell traversing the same least number of cells.

2.2 State Modeling

Definition of state: Here we consider the states used to model the system and the state transitions are synchronous with time. So, in unit time interval all predator and target agents will update their positions to move to a next state. Formally, the state can be expressed as:
$S_i = f(P_1, P_2, P_3, P_4, T)$
Where P_1, P_2, P_3, P_4 are the positions of the four predators at that instant of time and T is the position of the target. The transition from the state S_i to S_{i+1} occurs when there is a change in the position of the predator or target in a given time interval.

Prior Knowledge: For simplicity, we consider that every agent knows the position of every other agent in the square matrix but one takes his move independently of the others decision to move to a particular cell.

Goal State: Here the goal state is reached when the 'capture' occurs. Mathematically, the goal state *Sg* can be defined as follows

$$Sg = f(P_1, P_2, P_3, P_4, T)$$

Where the positions of the predator and target agents satisfy either of the following two conditions.

1) Let *XPij* denotes the position of any of the four predator and *Tij* is the position of the target and *XPij=Tij* i.e. target and at least one or more predator is in the same square

2) If the four ways to the movement of the target is blocked by the four predator:
Let *XPij, YPij, ZPij , UPij* denotes the position of the four predator and Tij is the position of the target then these four equations are satisfied

$$XPij = Ti{+}1j$$
$$YPij = Ti{-}1j$$
$$ZPij = Tij{-}1$$
$$YPij = Tij{+}1$$

Valid state transitions: The valid state transitions of the entire system are as below:
$S_i \rightarrow S_g$
$\Rightarrow S_i : f(P_{1i}, P_{2i}, P_{3i}, P_{4i}, T_i) \rightarrow S_g : f(P_{1g}, P_{2g}, P_{3g}, P_{4g}, T_g)$ where every agent may move or if they remain static their coordinates may remain invariant. The number of possible state transitions at the very beginning is dependent on

the actual number of changes that can be affected on the coordinates of the individual agent entities. However as certain restrictions have been put in place in order for the predator/predator-agents to capture the target/target-agent, we do need to consider all the possible transitions from a particular given state to another. Thereby allowing us to compute only a finite number of decreasing possible transitions with each iteration of the simulated environment in which the agent entities are interacting with each other. This is indirectly pruning the complete transition graph to obtain a much smaller and more easily traversable graph.

3. Convergence Metric

We define convergence metric as the measure of how close the present state is to the goal state compared with the previous state. This gives a measure of how close are we to reach the goal state. This is a measure of whether the system at any point of time is actually diverging away from the intended behavior.

The most rudimentary option to develop a measure of convergence would be to consider the arithmetic sum of all the least distances, in terms of cells to be traversed, from the different predator agents to the target agent. However one drawback of this simplistic approach would be that, the metric based only on this factor would be greatly influenced by the extreme values of cells to be traversed. A better measure would be to measure convergence not only as a function of the individual minimum distances to the target but to give weight to the proposition suggesting that due credit must be given to the number of predator agents which lie on the escape routes of the target agent and thereby lower the chances for it's successful escape, similarly consideration must be given to the closeness of predator agents to the escape routes of the target agent. So, the three convergence metrics considered can be expressed mathematically as follows:-

Metric 1:- The first metric considers only the sum of the distances of the four predators from the target. Here the distance means the minimum number of cells to be traversed from predator to the target. We denote this convergence metric as M_1. So,

$$M_1=\sum_{a=1}^{4} (|P_{ax} - T_x|+| P_{ay} - T_y|)$$

where P_{ax} is the row in which predator a is situated , T_x is the row in which the target is present, P_{ay} is the column in which predator a is situated and Ty is the row in which the target is present. The formula directly follows from the result that the minimum number of cells to be traversed is the summation of number of cell to be traversed along the row and number of cells to be traversed along the column because the agents can only travel row-wise or column-wise.

Metric 2:- The second metric also takes into account the number of predator present in the escape route of the target. Here the escape routes of the target indicate the rows and columns in the four degrees of freedom of the target. This metric gives equal importance to metric 1 and the number of predator blocking the route. Let us denote this metric as M_2. So,

$$M_2= 0.5NP + 0.5[\sum_{a=1}^{4} (|P_{ax} - T_x|+| P_{ay} - T_y|)]$$

where NP denotes the number of predator blocking the escape route of the target.

Metric 3:- This last metric takes the closeness of the predator from the escape route of the predator into consideration and also gives equal importance to metric 1 and metric 2. We denote this metric as M_3..

So,

$$M_3= 0.33[\sum_{a=1}^{4} (| P_a - T|)] + 0.33NP + 0.33[\sum_{a=1}^{4} (|P_{ax} - T_x|+| P_{ay} - T_y|)]$$

where $|P_a\text{-}T|$ denotes the distance of the predator a from the nearest escape route of the target.

4. Simulation Details

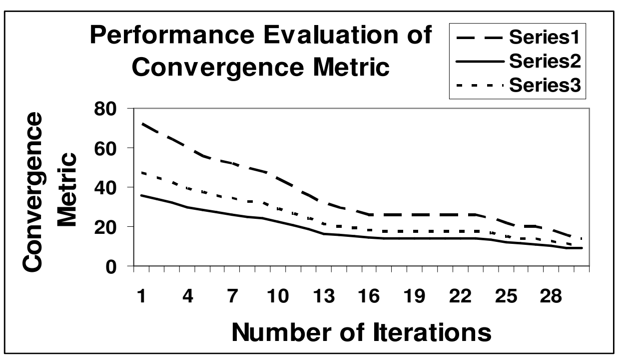
The following results were thus obtained as a result of the simulations that were carried out in order to gain a perspective of the system behavior. The results have been listed in order of simulation as individual cases. In each of the following cases the initial positions of the predator agents according to which each particular simulation has been carried out is mentioned. Standard deviation statistics for each of the convergence schemes has been compared in the following plots. The X-axis of the plots denotes the number of iterations the predator agents must execute in order to catch the target agent. While the Y-axis represents the value of the convergence metrics at each iteration during the simulation.

Case1

INITIAL CONFIG. predator agent is at 0,0 and 0,n-1and n-1,0 and n-1,n-1 (all at corners)

standard deviation

metric 1	metric 2	metric 3
16.66657	7.84913	10.67905

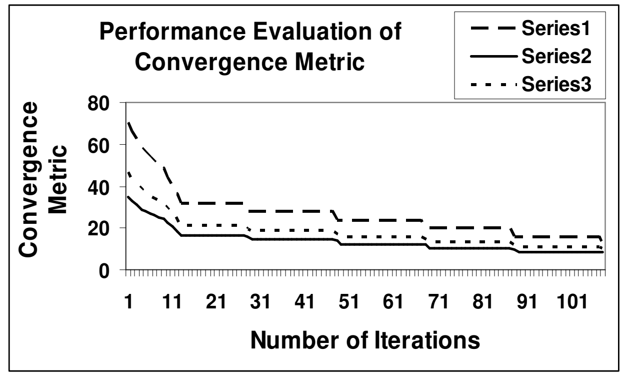


Case2

INITIAL CONFIG. predator agent is at 1,1 and 0,n-1and n-1,0 and n-1,n-1 (3 at corners)

standard deviation

metric 1	metric 2	metric 3
11.4151741	5.56065072	7.4851881

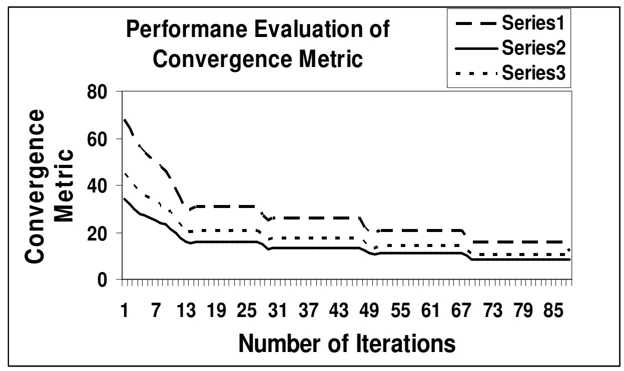


Case 3

INITIAL CONFIG. predator agent is at 1,1 and 1,n-2 and n-1,0 and n-1,n-1 (2 at corners)

standard deviation

metric 1	metric 2	metric 3
12.1419192	5.42527549	7.53453875

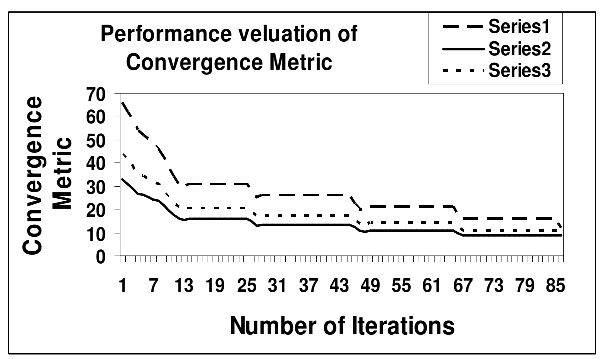


Case 4

INITIAL CONFIG. predator agent is at 1,1 and 1,n-2and n-2,1 and n-1,n-1 (1 at corner)

standard deviation

metric 1	metric 2	metric 3
11.02096	5.36295815	7.15856219



Inference:

It is thus observed from the results of the simulations that metric 2 is the most smoothly converging metric in comparison with metrics 1 & 3, and thus we describe the proof of convergence in the next section based on this metric.

5. Proof of Convergence

It is the intent of all the predator agents to converge on the target agent by blocking the columns and rows along which the target agent can exercise its four degrees of freedom. Let the initial configuration of the predator agent be P_{ij} and the configuration of the target agent be T_{mn} , we discuss the various possibilities in the subsequent subsections.

5.1 Traversal in relatively similar direction

Target and predator agent make a move in the same direction $T_{mn} \Rightarrow T_{m(n+1)}$ and concurrently $P_{ij} \Rightarrow P_{i(j+1)}$. Say Relative distance R_d along a particular direction is the minimum number of cells to be traversed to reach the column or row along the degree of freedom of the target agent closest to it.

$R_d = |n+x-(j+x)| = |n+1-(j+1)| = |n-j|$

Since the problem board is of finite dimension, $A \ X \ A$, hence in A-J cells it has the choice of maintaining its current choice of direction, however at $J=A$ the target agent can either execute a static move or it can execute a move which would result in movement in opposite direction with respect to the motional direction of the predator agent. If the target executes a static move when it is blocked at board boundary $J=A$ then $J_{t1} < J_{t2}$ and $N_{t1} = N_{t2}$ $\forall$ $t1<t2$ ($t1,t2 \in$ iteration at time $t1,t2$)

$\Rightarrow |JI_{t2} - N_{t2}| < | J_{t1} - N_{t1} |$

and if $R_{t1} = |n-j-f|$ then $R_{t2} = |n-j-f'|$ where $f,f'>0$ and $f'>f \Rightarrow R_{t2} < R_{t1}$

Hence as the relative separation in the number of cells to be traversed is decreasing between the target and the predator agent the system is converging towards a state wherein the predator agent lies on the column or row corresponding to one of the degrees of freedom of the target agent.

5.2 Traversal in relatively opposite direction

Target and predator agent make move in the opposite direction $T_{mn} \Rightarrow T_{m(n-1)}$ and concurrently $P_{ij} \Rightarrow P_{i(j+1)}$ Say Relative distance R_d along a particular direction is the minimum number of Cells to be traversed to reach the column or row along the degree of freedom of the target agent closest to it.

$R_d = | n+x-(j+x)| = |n+1-(j+1)| = |n-j|$

$J_{t1} < J_{t2}$ and $N_{t1} > N_{t2}$ $\forall t1<t2$

($t1,t2 \in$ iteration at time $t1,t2$)

$\Rightarrow | N_{t2} - J_{t2} | < | N_{t1} - J_{t1} |$

and if $R_{t1} = |n-j-f|$ then $R_{t2} = |n-j-f'|$ where $f,f'>0$ and $f'>f \Rightarrow R_{t2} < R_{t1}$

Hence as the relative separation in the number of cells to be traversed is decreasing between the target and the predator agent the system is converging towards a state wherein the predator agent lies on the column or row corresponding to one of the degrees of freedom of the target agent. This concept is extended to the four degrees of freedom mainly the columns and the rows. Now at this particular stage we introduce the concept of horizontal and vertical contraction. Let us assume the positions of the predator agents to be $P_{i1,j1}$, $P_{i2,j2}$, $P_{i3,j3}$, $P_{i4,j4}$ and that of the target agent to be T_{mn}

<u>Vertical Contraction (VC)</u>

when all predator agents lie on the respective degrees of freedom Say, $j1=j3$ $i2=i4$ $n-i1=>3$ and $i3-n=>3$;

Then $P_{ixt,jxt} = P_{ix+3\ (t+3)\ ,jx\ (t+3)}$ | when x=1; where $P_{ixt,jxt}$ is the position of predator agent $P_{ix,jx}$ at iteration t

$P_{ixt,jxt} = P_{ix-3\ (t+3)\ ,jx\ (t+3)}$ | *when x=3;*

$P_{ixt,jxt} = P_{ix\ (t+3)\ ,jx+1\ (t+3)}$ | *when x=2;*

$P_{ixt,jxt} = P_{ix\ (t+3)\ ,jx-1\ (t+3)}$ | *when x=4;*

<u>Horizontal Contraction (HC)</u>

when all predator agents lie on the respective degrees of freedom Say, $j1=j3$ $i2=i4$ $n-j2=>3$ and $j4-n=>3$;

Then $P_{ixt,jxt} = P_{ix+1\ (t+3)\ ,jx\ (t+3)}$ | when $x=1$; where $P_{ixt,jxt}$ is the position of predator agent $P_{ix,jx}$ at iteration t

$P_{ixt,jxt} = P_{ix-1\ (t+3)\ ,jx\ (t+3)}$ | *when x=3*;

$P_{ixt,jxt} = P_{ix\ (t+3)\ ,jx+3\ (t+3)}$ | *when x=2*;

$P_{ixt,jxt} = P_{ix\ (t+3)\ ,jx-3\ (t+3)}$ | *when x=4*;

Application of HC and VC via the algorithm allows for the Relative distance R_{1t} , R_{2t} , R_{3t} , R_{4t} to satisfy

$R_{1t} < R_{1t+3}$;
$R_{2t} < R_{2t+3}$;
$R_{3t} < R_{3t+3}$;
$R_{4t} < R_{4t+3}$;
Hence the algorithm finally converges to a tangible solution for this problem.

5.3 State transitions and convergence

As already discussed in Section 4, the number of transitions from state $S_i \rightarrow S_g$
Implies a modification in the internal values of the tuples
$S_i : f(P_{1i}, P_{2i}, P_{3i}, P_{4i}, T_i) \rightarrow S_g : f(P_{1g}, P_{2g}, P_{3g}, P_{4g}, T_g)$
The numbers of transitions at the initiation of the problem simulation are significantly more than at any time during the simulation for the simple fact that as the simulation progresses the number of possibilities available to a particular entity is actually pruned by the laws governing those transitions, as discussed in Section 4. To give an example of this pruning, we analyze the case used for simulation, discussed in Section 7.
FAT (Feasible Alternative Transitions)
P (Predator) , TA (Target Agent), TO (Total)

FAT (P1)	FAT (P2)	FAT (P3)	FAT (P4)	FAT (TA)	FAT (TO)
2	2	2	2	4	12
1	1	1	1	2	6
1	1	1	2	1	6
1	1	1	1	2	6
1	1	1	0	1	4

6. Algorithms

Algorithm for the movement of the target and the predator is based on the real life situation. We assume that everyone in this world knows the position of everyone. The movement of every person is synchronous with respect to time and so at the start of each iteration every person makes his move knowing the positions of other at the start of the iteration. As there are two types of agents governing the behavior of the target and the predator respectively the algorithm is divided into two parts as follows:- The algorithm is written in a pseudo code format. Comments are written in standard C format.
Movement of the predator:-
Various considerations have to be taken when a predator makes move.

```
Move-predator(Pa)
  /* if the predator is in the escape route forward it towards target */
     if  Pa.x=T.x
          if Pa.y > T.y
              move_left(Pa)
          else
               move_right(Pa)
          end if
      end if
     if Pa.y=T.y
          if Pa.x > T.x
            move_up(Pa)
          else
            move_down(Pa)
          end if
      end if
  find_closest_route(Pa)
   if blocked_route(Pa)=true
           if((Pa.x-T.x)>(Pa.y-T.y))
             if Pa=T.right
                move_left(Pa)
             else
                 move_right(Pa)
           end if
      else
             if Pa= T.top
               move_down(Pa)
             else
                move_up(Pa)
             end if
       end if
    end if
```

Movement of the target :-
/* check the four degrees of freedom to see whether they are blocked*/
if top_blocked(T)=false

```
    move_up()
if down_blocked(T)=false
    move_down()
if left_blocked(T)=true
   move_right()
if right_blocked(T)=true
    move_left()
max_distance=max(all distances)
move_dir(max_distance)
```

7. Analysis of algorithm

As the algorithm is divided into two parts viz. movement of the predator and the movement of the target, so we analyze the algorithm for the two parts separately. The final runtime complexity can be obtained by summing up the time complexity for the two parts. We analyze the worst case behavior of the predator and target. In case of predator the worst case will be that situation where after checking all the conditions he does not have a valid move and so he remains static. Let us suppose in general that there are n predators and m targets. Let the number of iterations after which the targets will be captured is t. Then in each iteration the predator first checks whether he is in the escape route of the target and so this involves a constant amount of time. Then he checks whether there are other predators in the blocking route. This involves a loop which has a complexity O(m). After the completion of the loop he moves accordingly or remains static. So, in the worst case the runtime complexity for each predator will be O(m). On the other hand, in each iteration the target also checks position of each predator to determine whether there is any predator in the escape route and this involves a time complexity of O(m) as it has to check the position of each predator. So, as there are m predator and each predator requires O(m) time to decide its move, therefore the total time complexity including O(m) for the target will be $O(m^2)$ in each iterations. If there are t iterations then the time complexity will vary according to the magnitude of t as $O(m^2)$ gets multiplied with t. So, if we suppose t as a constant and small relative to m then the time complexity of the algorithm will be $O(m^2)$.

.

8. Conclusion

This paper has successfully dealt with development of a convergence framework to measure the appositeness for the future development of a framework to measure the behavior of multiagent systems. The results provide a clear indication as to the comparison between the different metrics obtained during the simulation. The ideas developed in this paper may further be applied for better simulation purposes related to this area which would definitely lead to a better understanding of the behavior of the entities in the problems similar to the one examined in detail, and would ultimately lead to the development of better mobile agent management and simulation systems. The problem we have considered is restrictive in many dimensions. Our next approach will be to consider a mapping of the given MAS to a state graph and thereby measuring convergence on that graph. This will lead to a more general development of the convergence metrics and hence will apply to wide variety of MAS.

9. References

[1] S.Ghosh, Cooperating mobile agents and stabilization, LNCS 2194, 2001, 242-251.
[2] S.Ghosh, Agents distributed algorithms and stabilization, LNCS 1858, 2000, 1-18.
[3] T. Magedanz, K. Rothermel, S. Krause, Intelligent Agents: An Emerging Technology for Next Generation Telecommunications ?, *INFOCOM '96,* 1996, 464-472.
[4] Daniel R Kunkel, The game of GO and Multiagent Systems, College of Computing and Information Sciences Rochester Institute of Technology, Rochester, NY,14623, 1992.
[5] Martin Müller,Game Theories and Computer Go, In Proc. of the Go and Computer Science Workshop (GCSW'93), Sophia-Antipolis, 1993.
[6] B. Bouzy, T. Cazenave, Shared concepts between complex domains and the game of Go,

(*Multiagent Systems A Modern Approach to Distributed Artificial Intelligence*. MIT press, Masachussets, 1999) http://citeseer.nj.nec.com/62244.html.
[7] Ian F Akyildiz, Weilian Su, Y.Sankarasubramaniam, Erdal Cayirci, A survey on sensor networks, IEEE Commun. Mag., August, 2002, 40(8): 102-114.
[8] D. Saha , A. Mukherjee , Pervasive Computing: A Paradigm for the 21st Century, IEEE Computer Mag., March, 2003, 36(3): 25-31.
[9] Tony White, Andrzej Bieszczad, Bernard Pagurek, (Distributed Fault Location in Networks Using Mobile Agents, In Proceedings of the Second International Workshop on Agents in Telecommunications Applications (IATA '98), Springer-Verlag, Berlin , July 4th-7th, 1998, pp. 130-141.
[10] Stefan Funfrocken, Transparent migration of JAVA based Multiagents, Department of Computer Science, Darmstadt University of Technology, 1998.
[11] L. Hurst, P. Cunnigham, F. Somers, Mobile Agents - Smart Messages, (Broadcom Éireann Ltd., Dublin, Ireland, 1997)
[12] M. Caprini, R. Nacasch, Z. Qian, (Java Mobile Agent for monitoring task :evaluation report, Atlas DAQ, 1998)
[13] Elizabeth.A.Kendall, P.V.Murali Krishna, Chirag.V.Pathak , C.B.Suresh, A Java Application Framework for Agent Based Systems, Computer Systems Engineering Royal Melbourne Institute Of Technology, Melbourne, Australia, 2000.
[14] Seng Wai Loke, Mobile Agent Technology for Enterprise Distributed Applications: An Overview and an Architectural Perspective, Monash University, Australia, 1999.
[15] Aniruddha S. Gokhale, Douglas C. Schmidt, Evaluating CORBA Latency and Scalability Over High-Speed ATM Networks, *Proc. of ICDCS '97*, May, 1997.