Rough Set adaptive in the Model Based of Cellular Automata and Multi-Agents

Yasser F. Hassan
Department of Mathematics & Computer Science,
Faculty of Science, Alexandria University, Egypt,
y.fouad@sci.alex.edu.eg

ABSTRACT

The need for intelligent systems has grown in the past decade because of the increasing demand on humans and machines for better performance. The researchers of AI have responded to these needs with the development of intelligent hybrid systems. This paper describes the modeling language for interacting hybrid systems in which we will build a new hybrid model of cellular automata, multiagent technology and rough set theory. Therefore, in our approach, cellular automata form a useful framework for the muliagent simulation model response it in simulated cars in traffic system which lies in adapting the local behavior of individual agent using rough sets to provide an appropriate system-level behavior in grid of interacting organisms. The modeled development process in this paper involves simulated processes of evolution, learning and self-organization. The main value of the model is that it provides an illustration of how simple learning processes may lead to the formation of the state machine behavior, which can give an emergent to the model.

Keywords: *Cellular Automata, Rough Set, Multiagents, Emergence, Traffic System.*

1. INTRODUCTION

AI has made great strides in computational problem solving using explicitly represented knowledge extracted from the task. If we continue to use explicitly represented knowledge exclusively for computational problem solving, we may never computationally accomplish a level of problem solving performance equal to humans. From this idea, the paper describes the development of rough set theory that can be used to support the assessment of design performance in the cellular automata model using multiagents technique. Agents represent objects or people with their own behavior, think using rough set model and take the structure of cellular automata lattice.

A Cellular Automaton (CA) [3, 4], as the term is used in this paper, is a discrete state system consisting of a countable network of identical cells that interact with their neighbors. This network can take on any number of dimensions, starting from a one-dimensional string of cells. The cellular automata model is perhaps the simplest, general model available. It is simple in that the basic units are small, local, finite state machines (cells). It is general since: cellular automata model is support universal computation, and the rule represents a general form of local interaction. The two-dimensional cellular automata model is a grid of squares, each square having surrounded adjacent neighbors. A cell occupying a square is born, lives, or dies based on the number of living neighbors it has.

In next section, we start with the general model. This section has subsections: in first and second subsections, an introduction to the basic notions and language used in fields that contributed to this study, namely, rough set theory and cellular automata model are presented. Followed that, we present a general view of the combination system of rough set theory and cellular automata model, in section 3. We present the model characteristics that display several behaviors including: reproduction, growth, and mobility in section 4 and some emergent behavior of the new model will be mentioned. The application of our model to traffic system design is presented in section 5 with a discussion of our results. The section has three subsections; first subsection gives the definition of the traffic flow model and road network. The following subsection describes the behavior of at crosscut road and how we can use rough set theory in the decision-making at road crosscut. The method proposed undergone experimental verification and results of those experiments are mentioned in third subsection. The paper will be concluded in section 6.

2. Formal Definitions of Proposed Model

In this section, we describe the overall design of our automaton. A system with a collection of communicating agents is constructed in this section. We will consider two-dimensional cellular automata model which consists of array of cells $x(i, j)$. The use of a cell as intelligent agent provides an even greater amount of flexibility to the ability and configuration of the system itself. Each agent contains the same or different rule (knowledge) according to which agent states are updated in a synchronous and local manner. The neighborhoods of the agent $x(i, j)$ are the von-Neumann neighborhood of radius r [6], i.e.

$$N_r(i, j) = \{x(m,n) : |m-i| + |n-j| \le r\} \quad (1)$$

where N_r is the neighborhood relation function of distance r. We assign $r = 1$, i.e. the neighbors of an agent X are the four orthogonal adjacent agents plus the agent X itself. The agent at cell $x(i, j)$ is called grid agent if and only if all neighborhood agents

$x(m,n) \in N_r(i, j)$ exist. Otherwise $x(i, j)$ is called a boundary agent. Assume that any agent in the lattice is labeled by its position $x = (i, j)$ where i and j are the row and column indices. At any given instant, the agent is assumed to be in one state of the set of states S. Then abstractly, each agent state function is

$$f : S \times A \rightarrow S \qquad (2)$$

where A = $\{a_1, a_2, \ldots\}$ is a set of actions. The intuition is that an agent decides what action to perform based on its state and its environment states. I.e. the agent takes its current state and an action and maps them to a set of states S; those that are called result from performing action $a \in A$ in state $s \in S$.

The set of agents in the model needs to interact so that the agents use communication protocol to process their interactions. The communication is held between each agent and a set of neighbor agents. Communication protocols are typically specified at several levels [1]. The lowest level of the protocol specifies the method of interconnection; the middle level specifies the format, or syntax, of the information being transferred; the top level specifies the meaning of the information.

Once the automaton was embedded in the grid, the agent began to follow the rule that stored in it. A single agent cannot do much without interacting with other agents, and it has no concept of the whole. Yet, in combination it can play its part in producing complex results as emergence from local interactions. Briefly, each agent from the grid able to:

1. exercise a degree of freedom in its operations. It takes initiative and exercises a non-trivial degree of control over its own actions.
2. collaborate and exchange information with other agents in the grid (environment) to assist other agents in improving their quality of decision making as well as its own.
3. learn its optimal action by evolving its local rule.
4. change its state and the states of its immediate neighbors.
5. copy its rule into a neighboring died agent (travail self-reproducing). Contention occurs if more than one neighbor attempts to copy itself into the same agent. Such a situation is resolved randomly, i.e. one of the neighbors ‘ wins’ and copies its rule into the cell.
6. neither read nor write directly to other agents in the grid except its immediate neighbors.

3. Rough Sets in the Proposed Model

By embedding the method of rough set decision model within ongoing local control of cellular automata agent and by allowing of agent/environment interaction to take place within the system, we can obtain adaptation of agent behavior, which is reliably good [5, 8, 10].
In our approach, the decision tables define the behavior of the controller in each agent state and are responsible for state changes. Each agent has an associated decision table that defines the combinational actions in this state. In general cases, the agent followed the cellular automata rules for interaction with the neighbor agents, and when it needs to take decision at some spatial situations, it followed the rules that are induced from the decision table.

It is necessary for a specification method based on decision tables to be integrated with agent descriptions [6]. The minimal decision algorithm mentioned above can be described easily by rules such as “***IF*** conditional part ***THEN*** conclusive part (decision),” and used by each agent. In our approach, the decision tables define the behavior of the agent in each state and are responsible for state changes. Consequently, each agent has an associated decision table that defines the combinational actions in the behavior of this agent. This leads to dynamic knowledge bases, in which the active sets of rules are determined by the state of the agent. The decision table of the agent takes the form $(U, A \cup \{d\})$, where U is the set of success time steps, A is the condition attributes of the agent, and d is the decision attribute which describes the behavior of the agent.
In our implementation, the rough agent consists of the following subsystems:

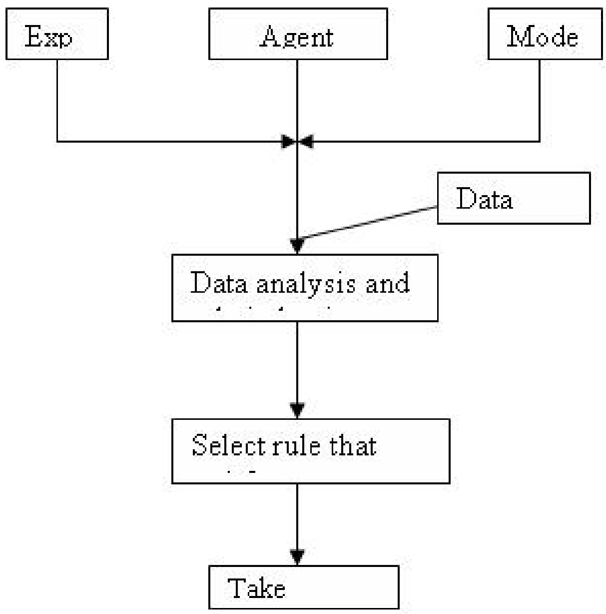

Figure 1: integrated rough set into cellular automata model.

This method can be considered as semi-unsupervised algorithm. The basic procedure of an unsupervised learning method involves grouping inputs together according to similarity or indicernibility. This process discovers cases in which a set of input values typically co-occur and thus tends to nth-order associations between variable values (where n is the number of values making up a complete input). The method can be visualized as an information system building operation which alternates between a statistical exploitation phase in which new competitive objects are produced and a relational exploitation phase in which new variables are produced. This learning method is fully incremental. Objects and rules are added to the information of each agent until the end of time iterations.

VOL. 2, NO. 9, September 2011 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

4. Agents Communication

A closer look on emergence proves that communication is the main reason for achieving global structure by local interaction, dynamic organization by simple (communication) rules, and intelligent behavior of the agent system by dealing with communication protocols. Thus, as communication is one of the main methodologies for achieving emergent behavior, an adequate selection and configuration of communication protocols is required [9].

A general and conceptual description of the communication (agent communication protocol) has to integrate mutual knowledge about its topics (domain), the general process of the dialog, and its current state. Thus, the structure of communication protocols may be divided into a domain dependent (content, problem, and topic) and a domain independent part consisting of process knowledge (methods of communication, address of partners, dialog structure). We assume that an agent has knowledge like transition rule and a mechanism for operating on or drawing inferences from its knowledge using rough sets.

Figure 2 presents two agents communication technique. A lifeline is any combination of instances. First, the agent send initial message to start the communication with another agent. The receiver answers the initial message by sending message depending on the understanding of the communication language. If the receiver understand the language, so it sends proposed message to ask about the neighborhood relation. The sender will use neighbor's relation function to accept or reject the proposed message. If the sender accepts the proposed, the receiver answers by sending its knowledge using inform message.

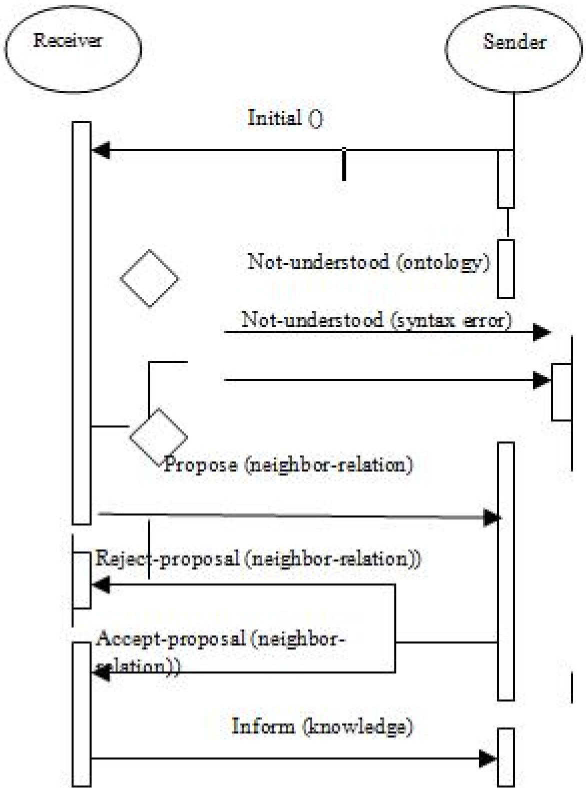

Figure 2: The protocol framework of the agents.

5. APPLICATION TO TRAFFIC SYSTEM CONSTRUCTION

In modern societies, the demand for mobility is increasing daily. Hence, one challenge to researchers dealing with traffic and transportation is to find efficient ways to model and predict traffic flow, even if the behavior of people in traffic is not a trivial problem. Thus, the modeling and prediction of traffic flow is one of science's future challenges. To be effective, such models have to make assumptions about the travel demand, and hence about travel choices and traffic behavior [2, 7].

The present work constructs a traffic model based on the hybrid system of cellular automata and multiagent system. Our new model of traffic system deals with the basic question of decision-making under such amount of inconsistent information. We will focus on the decision made by an individual driver as well as the consideration of the interaction caused by such a decision on the system as a whole.

We start by proposing a road traffic model suitable for an urban environment. North, east, south and west car displacements and road crossings are

VOL. 2, NO. 9, September 2011 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

possible. *k*-states multi-speed (k finite called maximum velocity v_{max}) car motion is found to be a crucial ingredient to describe highway traffic and phenomena. The basic idea is to consider the grid of cells as two types: first type forms a group to represent buildings (type agent-A) and second agent type constructs groups of cars representing (type agent-B). Each site of agent-B type can be in one of the v_{max} states: it may be empty (state = zero or die), or it may be having an integer velocity between one and v_{max}. This integer number of the velocity is the number of sites each vehicle advances during one iteration.

We will use which is called relative speed. A relative speed measure is to pit the absolute agent (car) speed against each other in a single elimination tournament. Let the relative speed of agent or car in position y to the agent x be:

$$w(x, y) = v(y) - v(x) \qquad (3)$$

It is convenient to apply the motion rule to the lattice with its boundary conditions. We can consider the velocity v of the agent x is changed according to the rule:

$$v_{t+1}(x) = v_t(x) + A \times Dist(x, y) + B \times w_t(x, y) + \sigma \qquad (4)$$

where $v_{t+1}(x)$ is the velocity of agent x at time step (t+1), v_t(x) is the velocity of agent x at time step t, y is the agent to which the car will move to, $w_t(x, y)$ is the relative speed of the car y to the car at cell x at time step t, and *Dist(x,y)* is the number of cells between cars x and y. According to the previous rule, the velocity of each agent is increasing or decreasing according to the state of the road. The formulas A, B are determined by:

$$A = Sign[-Sign(Dist(x, y) - v_t(x)) - 1]$$

$$B = Sign[-Sign(w_t(x, y) - v_t(x)) - 1] \times A \times w_t \qquad (5)$$

where function *Sign(z)* determines the sing of the variable z such that,

$$Sign(z) = \begin{cases} 1, & if \ \ z > 0, \\ -1, & if \ \ z < 0 \\ 0 & if \ \ z = 0. \end{cases} \qquad (6)$$

In real traffic-flow systems, the traffic jam is frequently induced by crossings when streets cross with each other (crosscut road). The crosscut road prevents cars from crossing their road. As soon as crosscut road begins to be congested, a traffic jam spreads from the crosscut road throughout space. The crosscut road exists in the model where a finite number of cars can wait approaching the crossing from each of the four directions. Each car has a desired direction, i.e. to the right, to the left or straight on, which has to be chosen according to a certain rule.

The behavior of cars to reach its destination in this model is actually fairly easy to handcraft, while it is hard to learn for many algorithms. Traffic simulation here is made more realistic by given individual drivers intentions, i.e. an idea of where they want to go and choose an optimal trip. Let us

Each of the N cars in the system starts at a randomly selected site; its desired direction (left, straight on, right) will be determined according to some rule (discussed below). Each car is assigned a destination site on the lattice. Once a car reached its destination it will be assigned a new randomly chosen destination. In the model of traffic system, the goal of the car is the strategy: *Drive as fast as you can and slow down if you have to until arrive the destination site!* The goal of the system is always to get a stable flow or high global average speed for all cars. The driver is free to change its mind if an intersection is momentarily locked. As a result, the load on each road segment is well balanced and, as long as there is a hole in the network, a motion will occur. The traffic is not distributed uniformly and some road segments are much more loaded than the others. The load distribution changes with time, as a result of the microscopic fluctuations.

We will consider an alternate update of each direction of motion, where there are synchronized traffic lights at each road crossing site allowing horizontal street motion at two-time steps and vertical street motion at next two time steps. We need four traffic lights: first time step is turn light to allow the vertical motion, therefore cars *a* and *b* can move along and also can turn right. Next time step, the traffic light allows cars *a* and *b* to turn left. Third time step is to the opposite case, i.e. horizontal motion is allowed where cars *c* and *d* can go ahead or turn right. Fourth traffic light is made for cars *c* and *d* to turn left, and again these four time steps are repeated.

consider how we might develop the implementation as a rough set technique. In our approach, the decision tables define the behavior of the vehicle in each state and are responsible for state changes. Each car has an associated decision table that defines the combinational actions in the behavior of this car. This leads to dynamic knowledge bases, in which the active sets of rules are determined by the state of the agent.

Table 1: Condition and decision attributes for the car in the system.

	Attributes	Description	Values
(1)	Relative_Speed1	The velocity of car at that moment with respect to the	{1,….,k}

VOL. 2, NO. 9, September 2011 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

	Attributes	Description	Values
		next car in front of it	
(2)	Relative_Speed2	The velocity of car at that moment with respect to the next car in left road	{1,….k}
(3)	Relative_speed3	The velocity of car at that moment with respect to the next car in right road	{1,……k}
(4)	Move	This car move at that moment or not	{Yes, No}
(5)	Jam	The car exist in jam road or not	{Yes, No}
(6)	Empty_next	The next site has car or not	{Yes, No}
(7)	Shortest_road	Which road has the shortest distance to the target	{left_road, right_road, front_road}
(8)	Longest_road	Which road has the longest distance	{left_road, right_road, front_road}
(9)	Pass	The car pass this site during this period	{Yes, No}
(10)	Time-late	Travel-time/expected-time	{0, 1, 2, 3}
(11)	D	The decision attribute	{go_straight, turn_left, turn_right}

The decision table of the car takes the form $(U, A \cup \{d\})$, where U is the set of success time steps, A is the condition attributes of the car (**Table 1**), and d is the decision attribute which takes values of go straight, turn right or turn left. Basically, road users want to get as fast as possible to their destination, but not at all costs. They are also interested in convenient transportation, which is taken into account by several other criteria. There exist dynamic and static road costs taken as attributes in the decision table of each car. Dynamic road cost is the *travel time*, which is determined on every road online for each car. Static road cost is *road length* where it is note that the shortest path is not always the fastest. Generally, it is difficult to determine an optimal road since there is a high degree of subjectivity involved. For example roads with low traffic density are usually perceived to be faster than other ones, even if they are not, as the mere feeling of getting somewhere is satisfying driver. The conditional attribute "jam" in the **Table 1** is determined by measure the velocity of the car, if this velocity is a minimum value ($v = 1$) for more than four time steps (because four time steps is the maximum traffic light at crossing), then the car lies in a jam. The condition attribute "time-late" can be calculated using what called the "expected time". The expected time is defined as:

$$\text{Expected time} = \frac{expected\_distance}{expected\_velocity} + noise$$

i.e.

$$\text{Expected time} = \frac{|target - current|}{\hat{v}} + noise$$

so

$$\text{Expected time} = \frac{|d_i - i| + |d_j - j|}{\hat{v}} + noise \text{ ------ ------------- (7)}$$

and

$$\hat{v} = \frac{\sum_{i=1}^{t} v_i}{t} \text{ -----------------(8)}$$

Where d_i and d_j are the destination row and column indices, i and j are the row and column indices of the car site, v_i is the velocity at time $t = i$, $i > 0$, and *noise* is a parameter. The possible values for condition attribute "time-late" are:

"0", which means early, If $travel\_time / expected\_time \le 0.8$,

"1", which means in the range, If $0.8 < travel\_time / expected\_time \le 1.2$,

"2", which means some late, If $1.2 < travel\_time / expected\_time \le 2$,

and "3", which means it late, If $travel\_time / expected\_time > 2$.

Related to attribute "time-late", If the car arrive the target site and $|expected\_time - travel\_time| \le \varepsilon$, where ε is a parameter, then the car add all path-traveling information to its decision table. The attribute "pass" means that the car lost, or enters in a loop, and then it cannot arrive its destination. So if the car pass on the site more than twice in it's way to this destination, so the attribute takes value "yes". As we will see this attribute affects the decision, so the car does not choose the same decision again.

VOL. 2, NO. 9, September 2011 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

We will define the measure of distance between current position and target one, which we will use to calculate the values of attributes “shortest-road” and “ longest-road”. Let the position of the car is the site (a,b) and the target site is (a',b'), then the distance is defined as:

$$dis = \sqrt{(a-a')^2 + (b-b')^2} \text{ .-------------(9)}$$

When the driver needs to calculate the shortest or longest road, he/she assigns (a,b) to the next site in each road, i.e. next site he/she will go in each road (its road, right road or left road). The driver goes straight when there does not exist any crossing in front of the car. While at road crossing, the driver use the decision table of the car to get set of rules using rough set technique that describe the decision which he/she should take at crossing of going straight or turning. According to the rule satisfies the conditions of the car at that moment, the driver takes decision. If no rule satisfies the car’s conditions, then the decision of going on or turning is taken randomly.

The two-dimensional automata model consists of $L \times L$ sites on rectangular lattice with periodic boundary conditions in both directions. We simulated system of size L = 20, 30, 50, 100 and no striking difference in the behavior of the system was observed. Hence, we exclude finite size effects. The fact that already comparatively small lattices exhibit this independence from their size seems to indicate the existence of only weak spatial correlations between sites separated by a large distance.

We observe some rules for number of cars that we consider as good comparing with human expert knowledge. We will illustrate new rules that differ than in the previous phases as:

R_1→ **IF** (move = no) **AND** (jam = yes) **AND** (longest_road = right_road) **THEN** decision = left.
R_2→ **IF** (relative_speed1 = 1) **AND** (shortest_road = left_road) **THEN** decision = left.
R_3→ **IF** (empty_next = yes) **AND** (longest_road = left_road) **THEN** decision = right.
R_4→ **IF** (short_road = left_road) **AND** (time_late = 1) **THEN** decision = left.
R_5→ **IF** (move = no) **AND** (longest_road = right_road) **AND** (pass = no) **THEN** decision = left.
R_6→ **IF** (move = yes) **AND** (jam = no) **AND** (longest_road = right_road) **THEN** decision = go_straight.
R_7→ **IF** (empty_next = yes) **AND** (jam = yes) **AND** (time_late = 0) **THEN** decision = go_straight.
R_8→ **IF** (move = yes) **AND** (pass = no) **AND** (time_late = 0) **THEN** decision = go_straight.
R_9→ **IF** (pass = no) **AND** (longest_road = left_road) **AND** (time_late = 0) **THEN** decision = go_straight.

Figure 3 shows the changing of average speed of cars in the system with the density of cars. We observe that the average speed of the cars reduced with the increase of the density of cars in the system. **Figure 4** shows the changing of the flow in the system with the density of cars. We observe also that the flow of cars is relatively good in the high density of cars.

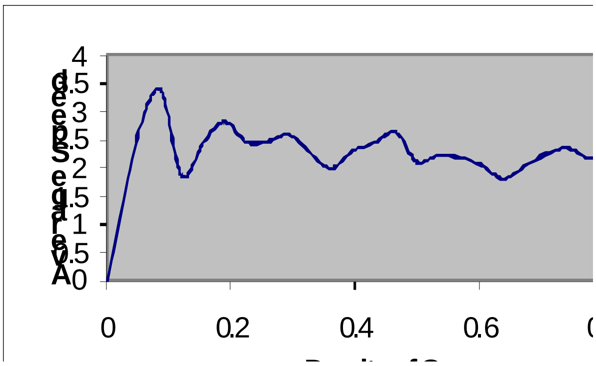

Figure 3: The relation of total average speed and density of cars.

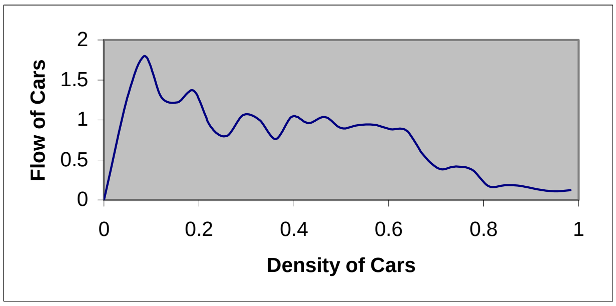

Figure 4: The varying of flow with the modification of density of cars.

We observe that the overall dynamics are quite sensitive to the driver’s behavior at road crossing for choosing its destination. The emergent dynamical property of driving control logic is defined in this model clearly. It is possible to observe the emergent behavior in this application of our new model such that the emergent behavior in the traffic model lies into two aspects one emerges from each other. Firstly, the cars in the system emerge from the cellular automata transition rule, where the interactions between agents or cars produce the movement of cars with different speed.

6. CONCLUSION

The proposed model has provided a direct approach to studying how dynamical systems perform emergent computation; that is, how the interaction of simple components with local information storage and communication gives rise to coordinated global information processing. Whether in real-life situation, the topology of the interconnections that gives meaning to the term immediate neighbor can change frequently. Although every agent participating in the system must be able to communicate with its immediate neighbors, the system itself should not depend on knowledge of the overall system topology. The state transition within each agent could be identical throughout the system or unique to each agent. In practice, the state transitions within the agents can most conveniently be viewed as shared by all agents, but with local adaptations as a function of either static or dynamic local conditions.

REFERENCES

[1] Abul, O., Polat, F., and Alhajj, R., 2000, Multiagent Reinforcement Learning Using Function Approximation, IEEE Transaction on systems, man, and cybernetics-part c: application and reviews, Vol. 30, No. 4, pp. 485-497.

[2] Darbha, S., Rajagopal, K., and Tyagi, V., A review of mathematical models for the flow of traffic and some recent results, Nonlinear analysis 69, 950-970, 2008.

[3] Gotts, N., 2000, Emergent Phenomena in Large Sparse Random Arrays of Conway's 'Game of Life', International Journal of System Science, Vol. 31, No. 7, pp 873-894.

[4] Terrazas, G., Slepmann, P., Kendall, G., and Krasnogor, N., An evolutionary methodology for automated design of cellular automaton-based complex systems, Journal of cellular automata, vol. pp 77-102, 2007.

[5] Yasser Hassan and Eiichiro Tazaki, Emergence Decision using Hybrid Rough Sets/Cellular Automata, International Journal of System and Cybernetics "Kybernetes", vol. 35, no. 6, pp. 797-813, 2006.

[6] Hassan, Y., rough sets for adapting wavelet neural networks as a new classifier system, Springer applied intelligence journal, DOI: 10.1007/s10489-010-0218-3, 2010

[7] Ngueyn, H.S., Skowron, A., and Szczuka, M., Situation Identification by Unmanned Aerial Vehicle, In: Rough Sets and Current Trends in Computing, (Ziarko, W., and Yao, Y. (eds.)), Springer Verlag, pp. 49-56, 2002.

[8] Polkowski, L., Tsumoto, S., and Lin, Y., Rough Set Methods and Applications, Physica Verlag, 2000.

[9] Tavakoli, Y., Javadi, H., and Adabi, S., A cellular automata based algorithm for path planning in multi-agent systems with a common goal, International journal of computer science and network security, Vol. 8, No. 7, 2008.

[10] Thangavel, K. and Pethalakshmi, A., Dimensionality reduction based on rough set theory: a review, Applied soft computing 9m 1-12, 2009.