Emotional Context in Imitation-Based Learning in Multi-Agent Societies

Goran Trajkovski[1], and Benjamin Sibley[2]

[1]United States University, 830 Bay Blvd., Chula Vista, CA 91911, USA
gtrajkovski@usuniversity.edu
[2]University of Wisconsin – Milwaukee, Department of Computer Science, 3200 N Cramer St, Milwaukee, WI 53211, USA
bjsibley@uwm.edu

Abstract

In this paper we explain how IETAL agents learn their environment, and how they build their intrinsic, internal representation of it, which they then use to build their expectations when on quest to satisfy its active drives. As environments change (with or without other agents present in them), the agents learn to new and "forget" irrelevant, "old" associations made. We discuss the concept of emotional context of associations, and show a gallery of simulations of behaviors in small multiagent societies.

```
PROCEDURE:   GENERATE_INTRINSIC_REPRESENTATION
GLOBAL:      G: ONTOLOGY, ξ: INBORN_SCHEME
ARGUMENT:    Table: CONTINGENCY_TABLE

BEGIN_PROCEDURE
            INITIALIZE_EMPTY_CONTINGENCY_TABLE
            INITIALIZE_POSITION_IN_G (G; Position)
            TRY_TO_EXECUTE_SCHEME (Position, ξ; ($B_1$, $S_1$))
            ADD_ROW_TO_CONTINGENCY_TABLE ([(λ, λ), ($B_1$,$S_1$)]; $R_\Delta$);

            WHILE (Active_Drive_Not_Satisfied)
                TRY_TO_EXECUTE_SCHEME Position, ξ; ($B_2$, $S_2$));
               ADD_ROW_TO_CONTINGENCY_ TABLE([($B_1$, $S_1$), ($B_2$, $S_2$)]; $R_\Delta$);
               ($B_1$, $S_1$):=($B_2$, $S_2$);
            END_WHILE
            PROPAGATE_CONTEXT (($B_1$, $S_1$), drive; Table).
END_PROCEDURE


PROCEDURE:            TRY_TO_EXECUTE_SCHEME
ARGUMENTS:              Position: POSITION_IN_ONTOLOGY,
                             (B, S): PERCEPTS_ACTIONS_PAIRS

BEGIN_PROCEDURE
            S:=EMPTY_STRING;
            TRY_TO_EXECUTE_ACTION_AT (Position, ξ;
                   (ADD_TO (S, Current_Percept), B));
            REPEAT
                   TRY_TO_EXECUTE_ACTION_AT (Position, B;
                        (Add (S, Current_Percept), B)
            UNTIL NOT ENABLED (B)
END_PROCEDURE


PROCEDURE:            PROPAGATE_CONTEXT
ARGUMENTS:              d: DRIVE; Table: CONTINGENCY_TABLE

BEGIN_PROCEDURE
            N:=0;
            WHILE (B1, S1) ∈ Projection [d] (Table) DO
            BEGIN_WHILE
                     Projection [d] (Table) := CONTEXT_EVALUATION_FUNCTION (N);
                     N++
            END_WHILE
END_PROCEDURE
```

Figure 1. The IETAL learning procedure in an autonomous agent.

Building the Intrinsic Representation

The intrinsic representation is a simple table (that we refer to as contingency table) where sequences of action and percept pairs are paired together in a fashion not unlike that one of a finite automaton. The contingency table is not, however, a finite automaton. Not only that the number of the states may grow or decline in time but rows of the contingency table also contain information on the emotional context of each of the rows (associations) with respect to each of the drives. As it will be discussed later, the emotional context is a numerical measure that denotes how useful an association has been (in past experiences) in satisfying the particular drive of the agent.

The learning procedure, the procedure that an agent follows in building its intrinsic representation, and uses past experiences in its mission to satisfy a drive is schematically given in Fig. 1. The agent's exploration of the environment is guided by the main function in Fig 1., GENERATE_INTRINSIC_REPRESENTATION. Initially, in the Piagetian sense, the agent is a tabula rasa in the environment, and starts trying to execute the actions in its inborn scheme, in the order in which they appear in the sequence. Please note that the parameter G, the ontology is visible to the designer only and is a global parameter in these procedures, as we, the designers, are the ones guiding the experiment. The agent uses its contingency table contents for operative purposes in the environment.

So, the agent tries to execute its inborn scheme. It may execute all of its actions, or just some of them. For example, if the agent is in a corridor and in the scheme the next action requires it to turn right, for example, it may not be able to execute this particular action, as it would bump in the wall. It "records" the percepts from the "places" it ends up in after executing an action from the scheme. So, the agent remembers the subscheme it executed, and the percepts perceived while "pushing" this scheme. Then it starts over again, it starts pushing the scheme over again, from the new place in the environment, and registering perceptually the context of the "places" it passes through. These subschema-perceptual (experiential) string pairs are crucial to the building of the contingency table. Its experiences are being kept in the table as associations between two such pairs, telling the agent what experience followed after the first experience.

After the agent finds a drive satisfier (note that all of this is happening while the drive d is active), the emotional context for that drive is being updated. According to the CONTEXT_EVALUATON_FUNCTION, the association that was being built or used when the drive satisfier was found gets the best emotional context value, the association prior to that one, slightly worse, and so on.

In the simulations that we discuss in this paper, we use CONTEXT_EVALUATON_FUNCTION (n) to be exp(-N), and we have also tried other ones, addressed in the gallery portion of this paper.

As the agent explores the environment in quest for food, water, or other items that would satisfy its active drive, it relies on its intrinsic representation and emotional contexts to get to food in an optimal way. Due to the perceptual aliasing problems, that might not always happen. When what is expected and is excepted in the contingency table does not happen in reality is being recorded as a surprise to the agent. The bumping of the agent in a wall/obstacle in the environment is counted as a unit of pain.

People forget, environments are dynamic, memory is limited. Later on we will discuss how we can account for these phenomena in this model. If we have the emotional contexts decrease in time, new associations and expectations would be dominating in a context of a drive, and associations that used to be helpful in finding, say, food, if the food stand is removed would eventually be forgotten.

Context?

The intrinsic representation of the environment in autonomous agents gives us the possibility to introduce flexible and context-dependent environment representations, which questions the traditional approach to the problem of context modeling (McCarthy, 1993).

Normally, the definitions of context depend on the context of the discourse (simulation of human behavior, making intelligent artifacts, mobile agent navigation etc.). We think that the human agents are extremely complex, and experiments based on language and other highly cognitive tasks depend on a range of known and unknown parameters. That is why our approach in its fundaments is based upon and congruent with Drecher's constructivist artificial intelligence (Drecher, 1992), that constructs simple agents and concentrates on researching their internal representation of the environment. As all researchers basically agree that context (whatever it might be or however it might be defined) influence the internal representation of the environment in agents, it is those representations that need defined. We believe that the process of construction of these representations depends on the context, i.e. the form of the representations themselves depends on the contextual knowledge. IETAL agents are simple, and only have inborn goals which are genetically specified. Those goals are defined by the inner drives of the agent, and the way the agent sees the environment at a given time.

So, the agent conceptualizes the environment via its contingency table. Depending on the system of active drives at a given time, the way the agent views the environment might be very different. Contextual information is built in the inner representation of the agent. Therefore, these representations are flexible and contain the inner context of the agent.

MASim

In this section we overview our simulation environment MASim (Multiagent Systems Simulations) that we developed with the intention of studying behaviors in smaller societies of agents. We will give a gallery of selected recorded behaviors and brief comments on each. In order to give agents the ability to make decisions, each agent shall start with an inborn movement schema, which is a series of movements used to form an agent's default path or directional pattern. In addition, for the purposes of creating an atmosphere where the agents learn and adapt to their environment, all agents are randomly placed within the environment.

To represent an agent's decision-making ability, each agent shall utilize two types of memory: exploratory and associative memory. An agent's exploratory memory can be thought of as a basic system of sensors used to map the agent's immediate surroundings, while an agent's associative memory can be compared to a set of unique environmental snapshots ascertained through the agent's sensory system. An agent's associative memory is its decision-making apparatus. It creates images of the environment and places significance on those images in an attempt to aid the agent's efforts in finding food. An agent's exploratory memory deals more with an agent's relative positioning, steering agents clear of cycles and traps. An agent shall utilize its exploratory memory until food is found, at which point its exploratory memory is ported to the agent's associative memory.

Each agent will navigate through a randomly-generated environment consisting of colored squares, obstacle squares, food squares, and other agents. The colored squares serve as "fuzzy" map elements for each agent, meaning the agent will see the colors of the squares as pieces of a whole, rather than storing specific paths.

Square colors and agent's direction are stored in an agent's associative memory, once food is found, and referred to and executed on a recognition-scale basis, meaning the higher the agent's recognition of the square type the more chance that agent will attempt to move onto that square type. For example, if an agent has several nodes in its associative memory where move is defined as "north," the agent will always choose the move that offers the highest or most recognition. This is what is defined as the agent's "emotional context."

The goal of the MASim project is to determine whether the use of fuzzy logic benefits the agents in their efforts to coordinate and adapt to the random environment they are placed in. Therefore, in terms of applying the above statement to the actual simulation, the purpose behind the simulation is to determine what parameters or settings, applied through the agents and the environment, work best in demonstrating an upward trend in agent learning ability as it pertains to agent motivation, which in this case is finding food. Thus, the ultimate measure of a successful simulation shall be determined by agent confidence, which reflects the amount of "correct moves toward food" made using associative memory. For example, if an agent moves north onto a red square and that move exists in its associative memory, the agent will execute the next move as determined by its associative memory, which is, let's say for this example, west onto a yellow square. If the agent moves west onto a yellow square its confidence will increase, else confidence decreases, meaning when the agent moved west it did not encounter a yellow square.

Agents

All agents will be objects spawned from an agent class. Each agent will be identical in composition; therefore, one agent class will spawn all four agent objects. Each agent object shall capture the following statistics: confidence, pain, surprises, square types, and emotional context of each move.

All agents will have a pre-born movement schema determined by the user at the start of the simulation. Movement schemas shall range from one to five incremental moves per schema, each incremental move being one of five directions: north, east, south, west, or stay (no movement).

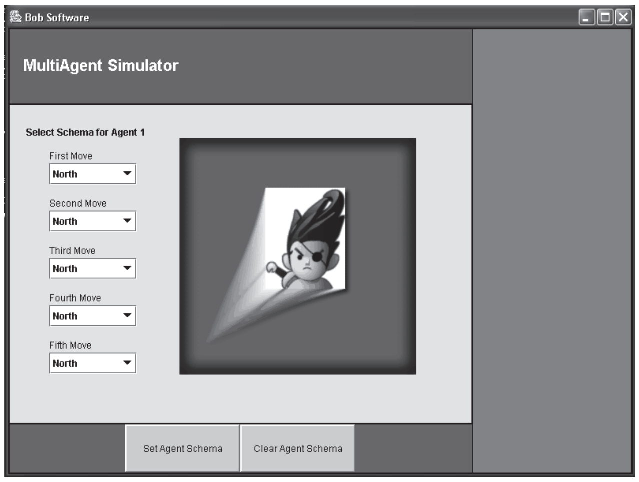
At the start of the simulation, the user is prompted to create each agent's inborn schema, as displayed in Figure 2.

An agent's inborn schema is the default path an agent shall follow. For example, if an agent's inborn schema is set to five moves in the direction of north, that agent shall move north until either colliding into an obstacle (in which a random move is generated in order to evade the obstacle) or the agent invokes its associative memory. An agent's inborn schema can be set between one to five moves heading north, south, east, west, or stay.

To fully understand the concept of an agent's inborn schema and how it works, take for example the following schema: North, East, North, East. On the start of the agent's turn, it moves as indicated by its inborn schema: North. After a successful move north, the agent tries to move east. In this example, moving east results in a collision, meaning the square is either occupied by an obstacle or another agent. Since a collision occurred, the agent will try its next move in the schema, which is north, and so on. If an agent cannot execute any moves within its schema it is trapped and a random move will be generated.

As an agent moves through the environment, it will capture the following data in its exploratory memory in the form of a structure for each incremental move: Direction of move; Type / Color of square; and Relative Position.

Each structure, which represents a move, is then placed into an exploratory linked list node. For example, if an agent moves north, the direction of the move, the color of the square the agent has just moved on, and the square's relative position from the agent's starting point are all placed into a move node that is appended to the exploratory linked list, as displayed in Figure 3.



Select Schema for Agent 1

First Move: North

Second Move: North

Third Move: North

Fourth Move: North

Fifth Move: North

Figure 2. "Select Schema for Agent" screen.

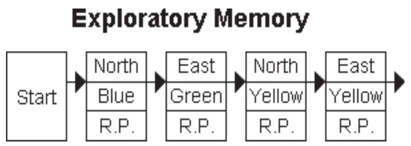


Figure 3: Displays an example of an agent's Exploratory Memory, which consists of incremental moves devised from inborn schema, random and associative moves. R.P. stands for "Relative Position."

All exploratory moves will first be checked for cyclic activity (cyclical verification) before being appended to the exploratory linked list. To do this, once an agent enters a square (a "To" move), the agent will verify whether it has entered into a new square by comparing the square's relative position with all relative positions already captured in that agent's exploratory memory. If any of the stored relative positions match the agent's current relative position a cycle flag is set to "true," meaning if the agent returns to that relative position again the cycle flag, being true, will activate a random move. A random move will be generated until the agent's next relative position is unique to all relative positions stored in the agent's exploratory memory. Therefore, an agent's relative position is responsible for tracking already-identified squares as potential cycle or trap points. An example of a trap point would be when an agent is surrounded by three obstacles. Agents will not move in an opposite direction to the previous move unless as a last resort, meaning if an agent moves north, its next move can not be south unless it is trapped. So if an agent is surrounded by three obstacles, its exploratory memory shall determine the relative position of each obstacle, ultimately notifying the agent that it is trapped and must move to where it came from. Thereby, an agent's relative position can be compared to a basic sensory system capable of detecting its immediate surroundings.

Upon an agent's discovery of food, the agent's exploratory memory will be ported into the agent's associative memory. An agent's associative memory will store each move in a sequence of "From" and "To." Each associative memory move will consist of two incremental moves, where the agent came from and the destination it moved to. Therefore, an associative memory move cannot span more or less than two adjacent squares.

Figure 4 displays the "From" and "To" concept of an agent's associative memory.

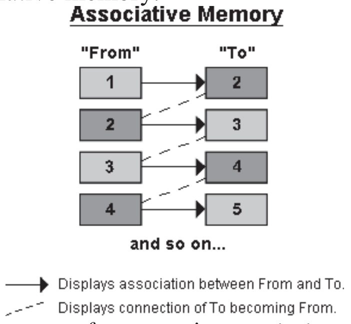

Associative Memory
"From"
"To"
1
2
2
3
3
4
4
5
and so on...
Displays association between From and To.
Displays connection of To becoming From.

Figure 4. Structure of an agent's associative memory.

A vector will be used to store all of an agent's associative memory moves. A vector offers the benefit of dynamic expansion, or, in other words, a vector can more than double its size "on the fly." Therefore, if a path being ported to associative memory is 50 nodes in size and the agent's associative memory is only 25 nodes in size, the vector will automatically expand to 150 nodes, allowing the inclusion of the 50 node path.

Each new "From" and "To" move will be added to the vector through an associative memory move node, as displayed above. Each move node in an agent's associative memory will contain the following data: Current Direction; Current Square Type; Previous Direction; Previous Square Type; and Emotional Context - updated when agent finds food.

At the start of an agent's movement process, the first node of the associative memory vector will not capture the initial "From" stats, which are attributed to the square the agent starts from.

An agent's emotional context is a quantitative value stored with each move in an agent's associative memory; it is based on the move's location in relation to food. The closer a move is to a food square, the higher the emotional context. For example, if an agent moves to a square containing food, the highest emotional context of the path being added to associative memory shall be applied to the previous move.

An agent's emotional context shall correspond to the number of moves within the "steps to food." For example, Figure 5 depicts the emotional context for each move at the time an agent finds food.

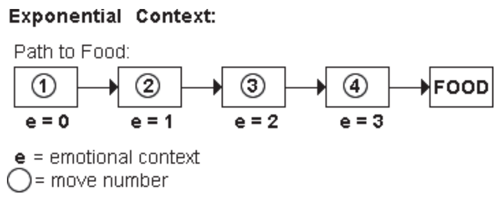


Figure 5. Emotional context for each move; this occurs at the time an agent finds food.

If an agent finds food, only the unique exploratory memory nodes (defined nodes not already existing in associative memory) shall be added to the agent's associative memory. If a node defined in exploratory memory matches a node within the agent's associative memory only the emotional context is updated. To update the emotional context of a duplicate node, the emotional context of the exploratory node is added to the emotional context of the associative node.

Once the emotional context is calculated and the path to food nodes added to the agent's associative memory, the associative memory is then sorted by emotional context. The sorting mechanism utilized shall be the QuickSort algorithm. As agent's move through the environment and gather food, emotional context will be used by the agent to select the best available moves to make from its associative memory. For example, if an agent makes a move that is recognized or exists within its associative memory, the agent will select the next move based on the prior move by matching the previous move's "To" move with the next move's "From" move that offers the highest emotional context.

When an agent finds food, that agent's confused / confident level shall be incremented by 1 and the agent shall then be re-spawned in a randomly generated location within the environment. Obstacles, agent-occupied squares, and food-occupied squares will be exempt from the agent's placement. Therefore, an agent can only be placed in an empty or free square.

If an agent is traversing a path and recognizes a location (meaning there has been a positive comparison in that agent's associative memory), as stated earlier, that agent shall follow the direction defined within its associative memory. Once the associative memory node is identified and executed, the agent then expects to move to an already identified square. If the square the agent moves to is not identical to the associative memory node executed, the agent's confused / confident level is decremented by 1, which is known as a "surprise."

An agent's confused / confident level will start at zero. The confused / confident level shall be displayed for each agent within the simulation interface.

An agent's pain level is based on collisions made with obstacles. If an agent collides with an obstacle, that agent's pain level will be incremented by 1. An agent's pain level cannot be decremented. Each agent's pain level will start at zero at the beginning of the simulation and shall be displayed within the simulation interface.

When agents collide, each agent shall receive and incorporate the other agent's associative memory. For example, if agent A and agent B collide, agent A shall add agent B's associative memory to its own associative memory and vice versa. Once an agent incorporates another agent's associative memory into its own, the associative memory vector is sorted by emotional context. In addition, during a collision/negotiation, each agent shall reflect the higher confident level of the two agents. For example, if agent A collides with agent B and agent A's confident level is higher than agent B's confident level, agent B's confident level shall equal agent A's confident level and visa versa.

Agents shall utilize a turn-based movement system. Each agent shall move its full move, a maximum of five moves, before the next agent may move. For example, agent B shall sit idle until agent A finishes its full move, and so on.

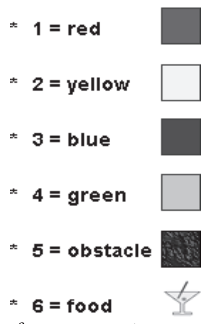


Figure 6. Elements of agent environment.

Environment

The agent environment will consist of a grid of squares each consisting of one of five colors or a food element, where one of the colors is black and considered an obstacle. Figure 6 displays square colors:

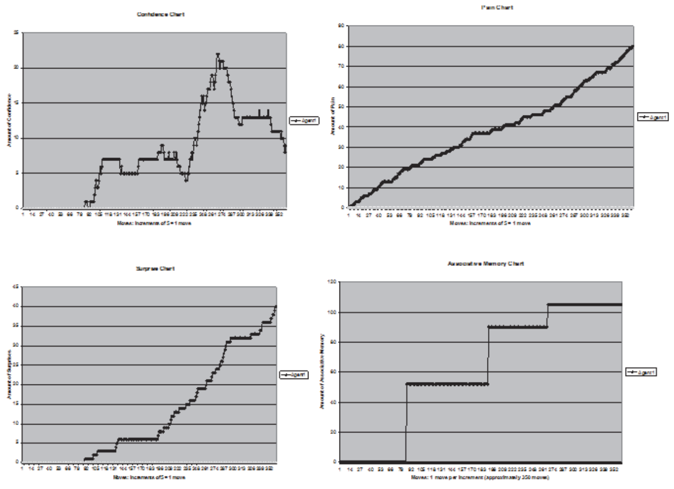
Square types will be randomly generated by the agent environment class at simulation start-up. The environment grid shall be a fixed dimension, 12 x 18 squares in size. After the environmental grid colors have been established, food items will be randomly placed. No more than three food items will be placed on the grid. Once a food item is placed it is fixed to that position and cannot be moved for the duration of the simulation. Food items cannot be placed on obstacles.

Agents Lost, Agents Found

In a situation when the agent's inborn schema is relatively small compared to the size of environment, interesting phenomena can be observed. Perceptual aliasing then becomes a significant problem. The parameters that we observe, although not perfect indicate on these problems. In this section we show a selection of cases produced under such circumstances. Interagent communication generally does not improve the success of the individual agents, as all agents are more or less equally incompetent to the environment, due to their inborn constraints. The importance of this study is to understand the relationship between the parameters measured in the agents – pain, surprise, confidence and number of entries in the associative memory.

The cases in this section will be denoted by N/M, where N denotes the number of agents and M the length of the inborn schema used.

For the 1/1 case simulation (Figure 7), during the course of approximately 350 moves the single agent accumulated a maximum confidence level of 22 and finished with a confidence level of 9. The agent spiked in confidence between (as represented on the above chart) move increments 222 and 274, which represents the agent using its associative memory to following a path already traveled. The single agent steadily increased its level of pain. This is mostly attributed to the random environment generation and, upon finding food, random agent placement. In addition, the agent was isolated without the influence of additional agents. The Surprise Chart displays the agent's level of confusion. The agent's Surprise Level, as displayed be the chart, increases heavily due to the random environment generation and, once food is found, the random agent placement procedures. In addition, the agent is not provided the influence of other agents. The Associative Memory Chart displays the number of time the agent found food (since no other agents were active in this simulation – no agent exchanges of memory). In this case, the agent found food three times, increasing it's associative memory to approximately



105 nodes.

Figure 7. The confidence, pain, surprise and size of associate memory statistics of a type 1/1 case simulation in MASim

During the course of approximately 700 moves the single agent in the 1/2/ simulation (Figure 8) accumulated a maximum confidence level of six and finished with a confidence level of one. Most of the simulation the agent had a negative confidence level, with a minimum -15. The chart also displays the agent gaining more confidence by the end of the simulation. During the course of approximately 700 moves the single agent steadily increased its level of pain. This is mostly attributed to the random environment generation and, upon finding food, random agent replacement. In addition, the agent was isolated without the influence of additional agents. Pain ration is approximately one pain per every 5.8 moves. The Surprise Chart displays the agent's level of confusion. The agent's surprise level, as displayed by the chart, increases heavily due

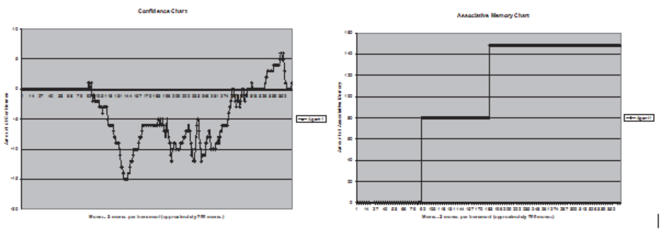


Figure 8. Some of the case 1/2 statistics

to the random environment generation and, once food is found, random agent replacement. In addition, the agent is not provided the influence of other agents. The agent's surprise level ratio is about one surprise per every 8.2 moves. The Associative Memory Chart displays the number of time the agent found food (since no other agents were active in this simulation – no agent exchanges of memory). In this case, the agent found food two times, increasing the agent's associative memory to approximately 148 nodes.

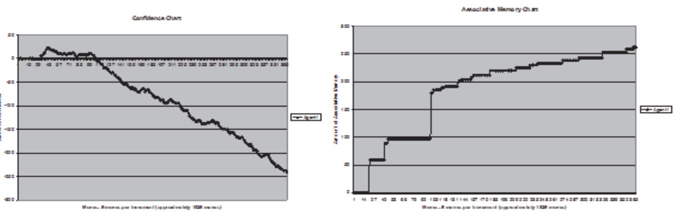


Figure 9. Selected 1/5 statistics: confidence and associative table size charts

In the 1/5 case (Figure 9) during the course of approximately 1825 moves the single agent accumulated a maximum confidence level of about 25 and finished with a confidence level of -245. Most of the simulation the agent had a negative confidence level, with a minimum -245. Pain ration is approximately one pain per every 5.8 moves. The agent's surprise level ratio is about one surprise per every 3.3 moves. The Associative Memory Chart displays the number of time the agent found food. In this case, the agent found food 19 times, increasing the agent's associative memory to approximately 260 nodes.

During the course of approximately 300 moves in the 2/1 simulation (Figure 10) agent 1 accumulated a maximum confidence level of 12 and finished with a confidence level of 12; agent 2 accumulated a maximum confidence level of 24 and finished with a confidence level of 23. Agent 2 spiked in confidence between (as represented on the chart above) move increments 250 and 300, which represents the agent using its associative memory to following a path already traveled. In addition, the chart displays a memory exchange between agent 1 and agent 2 at move increment 220, where agent 1 shared its memory with agent 2. After the memory exchange, agent 1's confidence stayed fairly steady, while agent 2's confidence rose quiet steadily. Both agents steadily increased their level of pain, although agent 1 had experienced a significant less amount of pain than did agent 2. As displayed by the pain chart, agent 1's pain level seems to level off a bit as its confidence level increases (as displayed in the confidence chart). Pain ration for agent 1 is approximately one pain per every 7.3 moves. Pain ration for agent 2 is approximately one pain per every 4.7 moves. The surprise chart displays an agent's level of confusion. Agent 1's surprise level stays fairly low throughout the simulation, as opposed to agent 2's surprise level. Agent 1's low surprise level corresponds with a steadily rising confidence. Agent 2's surprise level increases considerably until the agent's spike in confidence, which is a result of the memory exchange made from agent 1 to agent 2. Agent 1's surprise level ratio is about one surprise per every 30 moves, which is exceptional; this may be due to fortunate agent placement and relocations or the lack of overall associative memory. Agent 2's surprise level ration is about one surprise per every 8.6 moves. The associative memory chart displays the number of time each agent found food or benefited from a memory exchange. In this case, agent 1 found food or experienced a memory exchange two times, increasing its associative memory to approximately 60 nodes. Agent 2 found food or experienced a memory exchange five times, increasing its associative memory to approximately 122 nodes.

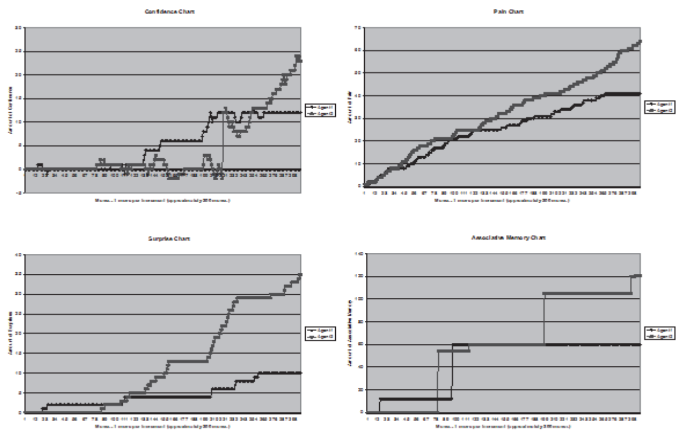


Figure 10. Statistics in the 2/1 case

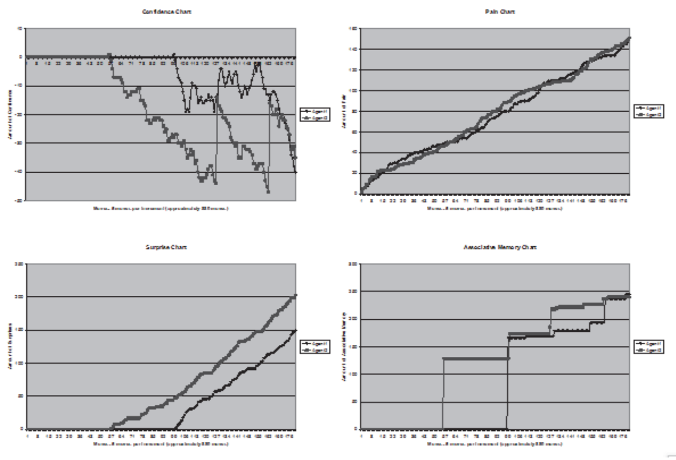


Figure 11. Statistics on the 2/5 case

For the 2/5 case (Figure 11), during the course of approximately 880 moves agent 1 accumulated a maximum confidence level of 1 and finished with a confidence level of -40; agent 2 accumulated a maximum confidence level of 1 and finished with a confidence level of -35. Both agents struggled heavily with accumulating confidence. As depicted by the chart, agent 1 made two major memory exchanges with agent 2 to no avail; agent 2's confidence dropped sharply after both exchanges. Agent 1 was able to raise its confidence, almost entering a positive level, until sharply dropping at the end. This is most likely due to a very challenging environment for the agents. During the course of approximately 880 moves both agents steadily increased their level of pain, both finishing with exact amounts. Pain ration for both agents is approximately one pain per every 5.8 moves. The surprise chart displays an agent's level of confusion. Agent 1's surprise level is more than 50 points less that agent 2's. This should be evident in the confidence chart, as agent 1's confidence level, most of the time, was much higher than agent 2's confidence level. Agent 1's surprise level ratio is about one surprise per every 5.9 moves. Agent 2's surprise level ration is about one surprise per every 4.3 moves. The associative memory chart displays the number of time each agent found food or benefited from a memory exchange. In this case, agent 1 found food or experienced a memory exchange seven times, increasing its associative memory to approximately 245 nodes. Agent 2 found food or experienced a memory exchange nine times, increasing its associative memory to approximately 240 nodes.

In the 4/5 case (Figure 12), during the course of approximately 475 moves agent 1 accumulated a maximum confidence level of 32 and finished with a confidence level of -8; agent 2 accumulated a maximum confidence level of 35 and finished with a confidence level of 1; agent 3 accumulated a maximum confidence level of 35 and finished with a confidence level of 2; agent 4 accumulated a maximum confidence level of 35 and finished with a confidence level of -3. As the chart displays, agents 1, 3 and 4 - who were all in negative territory at the time - greatly benefited from agent 2's high confidence memory exchange

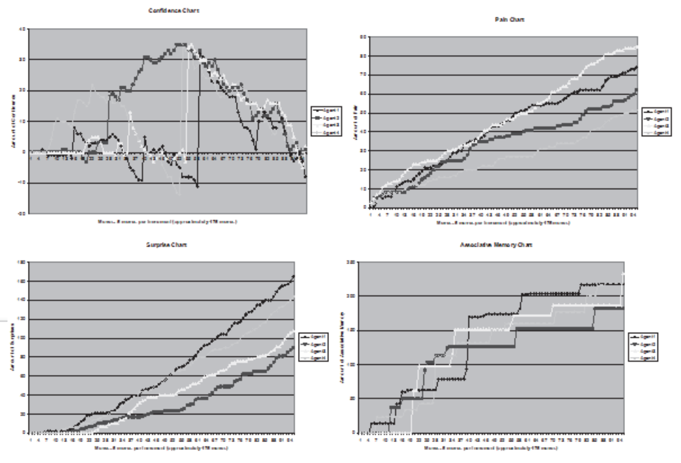


Figure 12. Case 4/5 statistics

halfway through the simulation, although the agents all dropped severely in confidence afterwards.

During the course of approximately 475 moves all agents steadily increased their level of pain, although they ending the simulation all spaced out from one another. Pain ration for agent 1 is approximately one pain per every 6.4 moves. Pain ration for agent 2 is approximately one pain per every 7.7 moves. Pain ration for agent 3 is approximately one pain per every 5.6 moves. Pain ration for agent 4 is approximately one pain per every 3 moves. The surprise chart displays an agent's level of confusion. Again, as with the pain chart, all of the agents end the simulation spaced out from one another. Agent 1's surprise ratio is about one surprise per every 2.9 moves, which is very poor. Agent 1 was constantly confused throughout the simulation. This could be attributed to a challenging environment and/or challenging location and replacement. Agent 2's surprise ration is about one surprise per every 5.3 moves. Agent 3's surprise level ratio is about one surprise per every 4.4 moves. Agent 4's surprise level ration is about one surprise per every 3.3 moves. Since all the agents have fairly poor surprise ratios, one could surmise that they were placed in a fairly challenging environment. The associative memory chart displays the number of times each agent found food or benefited from a memory exchange. In this case, agent 1 found food or experienced a memory exchange 13 times, increasing its associative memory to approximately 219 nodes. Agent 2 found food or experienced a memory exchange nine times, increasing its associative memory to approximately 182 nodes. Agent 3 found food or experienced a memory exchange eight times, increasing its associative memory to approximately 234 nodes. Agent 4 found food or experienced a memory exchange 12 times, increasing its associative memory to approximately 219 nodes.

References

Bazzan A. & Bordini, R. (2001) A Framework for the Simulation of Agents with Emotions: Report on Exepriments with the Iterated Prisoner's Dilemma. *Proc AGENTS'01*, Montreal, Quebec, Canada.

Chaplin D.J., & El Rhabili, A. (2004) IPD for Emotional NPC Societies in Games. *Proc. ACE'04*, June 3-5, Singapore.

Dresher, G. (1992) Made-Up Minds, MIT Press.

Guinchiglia, F. (1992) Contextual Reasoning. *Epistemiologia, Special Issue on I Linguaggi e le Machine*, XVI, 345-364.

Guinchiglia, F., & Bouquet, P. (1996) Introduction to Contextual Reasoning: An AI Perspective. *Course Material at CogSci96*, Sofia, Bulgaria.

Kokinov, B. (1995). A Dynamic Approach to Context Modeling. In Brezillon, P. Abu-Hakima, S. (Eds.) *Working Notes of the IJCAI'95 Workshop on Modelling Context in Knowledge Representation and Reasoning*.

Lee, K. (1999) Integration of various emotion eliciting factors for life-like agents. *Proc ACM Multimedia '99 (Part 2),* Orlando, FL.

McCarthy, J. (1993) History of Circumscription. *Artificial Intelligence*, 59, 23-46.

Ortony A, Clore, G., & Collis, A. (1988) *The Cognitive Structure of Emotions*. Cambridge University Press.

Piaget, J. (1977). Foreword. In Bringuier, J.C. *Conversations libres avec Jean Piaget*. Paris: Editions Laffont.

Siraj-Blatchford, J. (n.d.) Schemes and schemes. Retreived online at: http://k1.ioe.ac.uk/cdl/CHAT/ chatschemes.htm on June 22, 2004.

Stojanov, G., Bozinovski S., & Trajkovski, G (1997a) Interactionist-Expectative View on Agency and Learning. *IMACS Journal for Mathematics and Computers in Simulation*, 44 (3), 295-310.

Stojanov, G., Trajkovski, G., & Bozinovski, S. (1997b) The Status of Representation in Behavior Based Robotic Systems: The Problem and a Solution. *Proc. Systems, Man and Cybernetics*, Orlando, FL, 773-777.