International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:5, 2007

Representing Collective Unconsciousness Using Neural Networks

Pierre Abou-Haila, Richard Hall, Mark Dawes

***Abstract*—** Instead of representing individual cognition only, population cognition is represented using artificial neural networks whilst maintaining individuality. This population network trains continuously, simulating adaptation. An implementation of two coexisting populations is compared to the Lotka-Volterra model of predator-prey interaction. Applications include multi-agent systems such as artificial life or computer games.

***Keywords*—** collective unconsciousness, neural networks, adaptation, predator-prey simulation

I. Introduction

Artificial Neural Networks (ANNs) attempt to replicate the connectivity and functionality of biological neural networks in order to simulate learning [1], [2], [3]. They have been successfully applied in a number of domains including robotics, computer vision, pattern recognition, speech recognition, and image processing. These typical applications assume that a neural network represents a single individual.

In order to represent populations of individuals accurately, it could be argued that each individual should have their cognition represented by a dedicated ANN, regardless of how cognition is defined. However, each ANN would need to be trained prior to simulating population interactions otherwise each individual would have no knowledge whatsoever on their instantiation (eg. birth), which might place them at a significant disadvantage with respect to their environment or other more highly-trained individuals. In addition, if all individuals need to be trained before birth, it means that the total number of births needs to be known before run-time; impossible for emergent dynamic simulations such as artificial life. Finally, even if training time for a large population were acceptable, it may be difficult to construct meaningful and reasonable variations of training data that are supposed to engender individuality.

However, it may be unnecessary for each individual to have a dedicated ANN. The Jungian theory of collective unconsciousness states that there is part of the unconscious mind that is shared by a society, a people, or all humankind which contains concepts of science, religion and morality and is a product of ancestral experience [4], [5], [6].

This psychology theory can be loosely interpreted (in engineering terms) as a global common mechanism that represents the cognition of an entire population, assuming that the requirements that individuals take turns in cognitive processing is acceptable. Using a typical ANN for this global mechanism is computationally feasible, because it has exactly the same requirements as a single ANN.

Unfortunately, a side effect of this naive implementation is that the concept of individuality is lost because the cognitive representation for each agent is identical. Consequently agents are just an army of deterministic clones; in exactly the same situation, all agents will necessarily behave in exactly the same way. While it might be possible to avoid placing different agents in the same situation, such an implementation fails to accurately represent the theory of collective unconsciousness, because the theory requires individuality in addition to universality.

In order to avoid cloning, each individual must be allowed to affect the global mechanism in some manner. Since Jung fails to specify precisely how such interaction occurs, the precise details of this interaction can satisfy the whim of any implementation criteria, as long as it agrees with the theory. In this instance, we would like to preserve the ability of the mechanism to simulate learning when it is made global. The advantage of such preservation is that adaptation can occur both at the individual and population levels, which is in accordance with Jung's theory.

The global mechanism must also affect each individual. On instantiation, individuals must be given enough knowledge from the global mechanism to allow them to begin interacting in their environment immediately. In addition, to simulate generation in nature, the way a child agent interacts with the global mechanism could resemble the way its parents interact with the global mechanism. Both these ideas are designed into what we call a population artificial neural network (*PN*).

In comparison to dedicated neural networks, the benefit of using *PN* is that a single neural network is trained once prior to simulation, rather than needing to train as many neural networks as there are individuals. In addition, instead of

This work was supported in part by
Department of Computer Science and Computer Engineering,
La Trobe University, Bundoora, Victoria, Australia.

P. Abou-Haila is with Department of Computer Science and Computer Engineering, La Trobe University, Bundoora, Victoria, Australia. (e-mail: P.Abou-Haila@latrobe,edu,au).

R. Hall, is with Department of Computer Science and Computer Engineering, La Trobe University, Bundoora, Victoria, Australia. (e-mail: R.Hall@latrobe,edu,au).

M. Dawes is with Department of Computer Science and Computer Engineering, La Trobe University, Bundoora, Victoria, Australia. (e-mail: M.Dawes@students.latrobe,edu,au).

individuals learning independently once instantiated, adaptation also occurs at a global level. Consequently, individuals that are instantiated later rather than earlier interact using a *PN* with greater exposure.

We did not know how to properly evaluate such a representation, as a precise mechanistic specification for cognition has yet to emerge from psychology to which such a representation can be compared. Furthermore, an ANN is much simpler than a biological neural network, thus evaluating its biological cognitive accuracy would be difficult for a single individual let alone a population.

However, it is possible to evaluate the feasibility of constructing an interacting population of simple agents, with each population having its own *PN*. We implemented a predator-prey ecosystem because such ecosystems supposedly approximate the Lotka-Volterra model of population numbers [7], [8], [9], [10], [11]. Such numbers are trivial to record during simulation.

The organisation of this paper is as follows. In Section 2, we design the *PN*. In Section 3 we discuss our implementation of the *PN* in the predator-prey ecosystem. In Section 4 we evaluate this ecosystem with respect to the scalability of population size and with respect to the Lotka-Volterra model. Finally, in Section 5 we indicate some new directions for this work that we are currently pursuing.

II. Population Network design

There are many ways in which interactions between the *PN* and individuals could be defined, but no benchmark exists with which these interactions can be compared. Consequently, we establish our own set of design criteria and attempt to meet these. Then, we assess our agents with respect to intelligent agents, since agent designs are usually considered with respect to this field.

It was necessary to establish our own criteria for the manner in which individuals precisely modify the global consciousness, since the theory of collective unconsciousness lacks such criteria. In attempting to keep with the spirit of the theory we established three criteria: universality, individuality, and simplicity.

For *universality*, it was necessary for the *PN* to fully represent the cognition of all individual agents that were connected to it. We assume that members of a population are assumed to have exactly the same characteristics by default, and that this assumption includes cognition. Consequently, all agents use their reasoning mechanism in exactly the same way and modify the same components of the reasoning mechanism. In addition, since the *PN* simulates isolated dedicated neural networks, for universality it is necessary that individuals are unable to affect the ability of the *PN* to represent the cognitive capabilities of the population as a whole.

For *individuality*, it was necessary for individual agents to modify the same components of the *PN* in different ways. Otherwise if agents interact with these components in the same way, their behaviour would still be identical, unless the *PN* changes independently of the agents. In contrast, the theory of collective unconsciousness appears to indicate that the global mechanism is a function of the individual contributions to it, thus such an implementation is outside the scope of this work..

The concept of individuality can be interpreted in strong and weak versions. With the strong version, all agents could be expected to behave significantly differently from each other in every situation. With the weak version, all agents could be expected to behave slightly differently from each other in a single situation. Two agents would engage in similar behaviour when two variables are similar: their input perceptions; and the internal variables of the *PN*. Consequently, internal cognitive state variables need to be represented in agents that are input to the *PN*. Differences in these cognitive state variables can guarantee individual behaviour even by different agents in the same situation with the same *PN*. By instantiating these variables with great similarities or great differences, it is not infeasible that both versions of individuality could be represented.

For *simplicity*, it was necessary for individual agents to modify the *PN* as little as possible, due to the computational expense required by the *PN* and the desired scalability in population numbers. In such a context, even the simple task of fetching the individual contributions from each agent in memory become time consuming with complex *PN* and large numbers of agents, so minimum processing incorporating this contribution into the *PN* makes an appreciable difference depending on the resources of the simulation machine.

Having elaborated our design criteria we now consider interactions between the agents and the *PN*. We firstly consider how agents will affect the *PN* then the way in which the *PN* will, in turn, affect the agents. These proposed interactions are examined with respect to our design criteria.

In order to represent individuality, agents are given private cognitive state variables. We consider two types of variables, world-variables, and *PN*-variables. These two types act as a counterbalance to avoid both extreme versions of individuality. World-variables relate the agent to the environment. Since the environment varies itself, different agent's world variables are in different states, leading to strong versions of individuality. On the other hand, *PN*-variables allow agents to make small contributions to the global mechanism. However, the global mechanism is reasonably similar when different agents use it, leading to weak versions of individuality. The architecture for our *PN* is shown in

Figure 1.

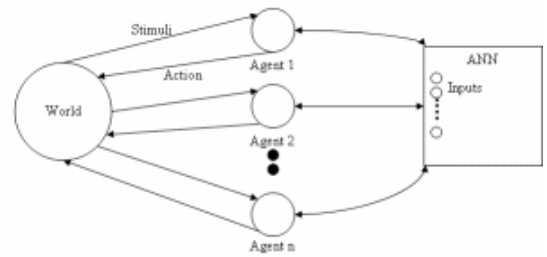

Figure 1: Population Network Architecture

With *world-variables*, agents require some internal state that relates the world to the agent. From a philosophical point of view, since goals exist in individuals, and goals relate individuals to the world around them, we chose to represent simple goals within each agent. Since goals are dynamic, and the rate of their change can be dependent of the situations in which agents find themselves, it is possible that two agents in the same situation will have different goals. Since these goals are input to the *PN* along with perceptual inputs, different goals can produce individual behaviours.

With *PN-variables*, agents require unique and private weights when the reasoning mechanism is a neural network. Thus an individual's weights can be similar, but non-identical, to the weights of every other agent connected to the same *PN*. In every agent, weights must correspond to exactly the same connections, to maintain universality.

It is feasible for agents to submit their weights to the *PN* to be processed in turn, overriding the previous weights in the *PN*. However, such an approach means that agents are using a global mechanism for processing only, rather than contributing to a global consciousness that affects all agents. Consequently, we allow the *PN* to have its own private weights, but view each agent's weights as individual contributions to these global weights. When each agent is connected to the *PN*, its contribution (offset) is summed with the global weights for simplicity.

The *PN* must also affect the weight offsets of individual agents, in order for agents to change with respect to the global mechanism. Since the *PN* is an abstraction of individual cognition, it was felt that whatever modifications were made to the global mechanism after processing should be reflected exactly in each individual. Thus it was necessary to design a single update mechanism only to satisfy the criteria of universality.

The way in which the *PN* changed was thus reflected proportionately in the offsets of each individual agent. The formula used is shown below: w_O is the agent's offset weight which matches a specific neuron, W_1 is the initial total weight for the neuron and W_2 is the weight after the *PN* has completed processing.

$$w_o = w_o + \left[w_o * \left(\frac{W_2 - W_1}{W_2} \right) \right]$$

Although every weight in every agent uses the same formula to preserve universality, the agent offsets ripple-through the *PN* such that each set of W_2 will be unique for every agent. Thus overall individuality is preserved, despite using the same formula for every neuron. The formula was implemented directly, which satisfies the criteria of simplicity in a mathematical sense, having only four arithmetic operations. While this formula could be further simplified to reduce computation, we prioritised individuality over simplicity in this instance, as the computational cost was deemed acceptable given our system resources.

It is necessary to consider how our agents relate to intelligent agents, in order to define the scope of our design. In the field of intelligent agents, interaction between agents and their world is categorised as either reactive or deliberative (although there exists disagreement about the precise meaning of these terms) [12]. We briefly describe these categories, and attempt to position this work within.

Reactive agents interact with the world by attempting to match their perceptions to stimulus-action pairs; if a match is found the predefined action is executed. As such, reactive agents simulate intelligent behaviour without the need for a world model, internal state such as a goal model or memory, or explicit reasoning capability [13].

There are three main differences between our architecture and a classical reactive agent. Firstly, our agents are given simple internal cognitive states that are processed by the *PN* in addition to inputs from the world. Secondly, rather than attempting to match inputs (perception) to a stimulus-response pair (eg. by searching in a lookup table), all inputs are processed simultaneously by the *PN*. Thirdly, the neural network basis does give agents an explicit representation that allows learning, though it is clearly different from memory as agents are unable to explicitly recall past events. Despite these differences, our agents probably fall more strongly into this category than deliberative agents, because agents do attempt to represent a world model that they can reason upon.

Deliberative agents use an internal model of the world and a representation of memory to reason about the effects of its actions in order to select actions that it predicts will achieve its goals. Its perceptions can be assessed in terms of a number of explicitly represented and interacting components such as memory, goals, beliefs, desires, intentions [14], [15]. Agents use these components to construct plans that they continuously evaluate and suspend or discontinue where necessary.

The major difference between our architecture and a classical deliberative agent is that our agents make no plans nor have any of the reasoning components for supporting planning. The absence of these components means that one of the primary advantages of reactive architectures can be maintained: comparatively faster processing requirements [16]. Where planning is critical to achieve optimal agent behaviour, the current specification of our agents might be unsuitable. However, it is outside the scope of this work to consider planning since the theory of collective unconsciousness does not mention it. At the least, the current

specification of our model appears to potentially allow the incorporation of planning capability in our agents.

In this section we described the design of our agent architecture using a *PN* that satisfied our criteria. Our agents can be categorised mainly as reactive, since they have no internal world model, memory, or explicit reasoning capability, and they map inputs directly to actions. However, agents do have two sets of internal state variables: offset weights to the neurons; and goal states, which are considered alongside perceptual input. These internal state variables, in conjunction with their offset modification (simulating adaptation) mean that our agents belong to the deliberative agent class to a small degree. We now discuss the implementation of our agents in a predator-prey ecosystem.

III. Implementation

A simulation of a predator-prey ecosystem requires a number of components to be represented. We briefly list these components then justify their inclusion. Object-oriented modelling was performed on these components, so that the resulting modular implementation could be easily extended. Seven components were required:

The world with which agents interact

The group attributes of each population (X2)

The individual attributes of each agent (X2)

The characteristics of features common to individuals in both populations

The population network (*PN*)

The world with which agents interact was designed to represent three features: time, space, and population tracking. With time, it was necessary to schedule agents because only one agent can interact with the *PN* at a time. In terms of the populations, we decided that predator agents would all be processed before prey agents, although it makes little difference since all agents have access to the *PN* in one turn. Within populations, we use round robin scheduling for simplicity.

With space, it is necessary for agent perceptions to be localised for two reasons. Firstly, it is undesirable for all agents to be able to perceive the entire state of the world simultaneously, because then it means that all agents have exactly the same inputs perceptions (global omniscience), thus unnecessarily placing the responsibility for representing individuality on the representation of internal state. More importantly, the processing capability required for the entire world-state, depending upon its complexity, could be well beyond the resources of a machine. We used a homogenous tile engine to graphically represent different terrain types, where the various types of terrain related to agent goals.

With population tracking, it is necessary to periodically count and store population numbers, in order for changes in population numbers to be compared. Population tracking served only one purpose in the simulation, for evaluating the accuracy of the predator-prey interactions. Population numbers alone, while simple, are often critical in ecological modelling.

The *group attributes of each population* were designed to represent the relationship between populations with the world and population relationships. The relationship between populations with the world was interpreted in terms of territory. With territory, populations would wander roughly within a region of tiles; predators and prey meet conveniently due to territory overlap.

The population relationships were interpreted in two ways: each population had one dominant male and one dominant female, and each population had scheduled mating seasons. Dominant animals were given choice priority in various collective behaviours. For example, where a group of lions met a group of zebras, the unfortunate zebra that had been selected by the dominant lion was also selected as a target by all other lions.

With scheduled mating season, populations would, for a certain number of turns, prioritise the goal of mating over other goals. In the tile world, pairs of agents would move to adjacent tiles and stay there for a time. At the end of the season, agents would re-prioritise their goals to their original states.

The *individual attributes of each population* represented relationships between the individual agent and a number of other agents including mate, offspring, food, and foes. With the predator agents, food included the prey agents; with the prey agents, its foes were the predator. We named the predator population lions, and the prey population zebras, because the agents and population relationships we had constructed simulated a small number of features with these complex real-life predators and prey.

The *attributes of features common to individuals in both populations* include: a numerical id; food level, injury status, speed, perceptual range, and gender. These attributes affect the way in which the agents interact with the world, thus changing the state of the world that the agents perceive. Some of these attributes are directly input to the *PN*, under the assumption that agents perceive their own internal state as they perceive external state that can also be perceived by other agents.

The *PN* used a Kohonen neural network, which was chosen for two reasons. Firstly, a KNN is supposedly similar in architecture to biological neural networks [3], [2]. Secondly, this type of network learns in an unsupervised manner, placing the responsibility of constructing relationships upon the mechanism instead of on the designers. The *PN* had 70 Kohonen-layer neurons.; it took an hour to train on a dataset consisting of 500 elements processed in 4000 iterations.

The *PN* has 7 input neurons; four were goal-state variables and three were perceptual boolean variables. The four goal-state variables were: hungry, thirsty, injury_level, and ready_to_mate.. The three perceptual variables were: predator-prey detected, mate detected and water detected. The *PN* has four output variables; all output variables were directional: go-to-water, go-to-food, go-to-mate and roam. Once agents had moved on top of tiles that satisfied their

goals, they automatically proceeded to satisfy these goals. Note that a lion does not instantaneously kill a zebra; they inflicted damage, then as the zebra moves, the lion continues to move towards the new tiles containing the zebra, either until the kill was made, another zebra distracted the lion, or the lion became too tired to continue.

An object oriented analysis of our predator-prey simulation requirements lead to the construction of seven objects: World, Kohonen, Pride, Herd, Lion, Zebra and Animal. For each of these objects, the simulation requirements were interpreted in terms of attributes and methods. The complete implementation class diagram for the simulation is shown below in Figure 2. Note the similarities and also the differences between attributes and methods to reflect the predator-prey relationship.

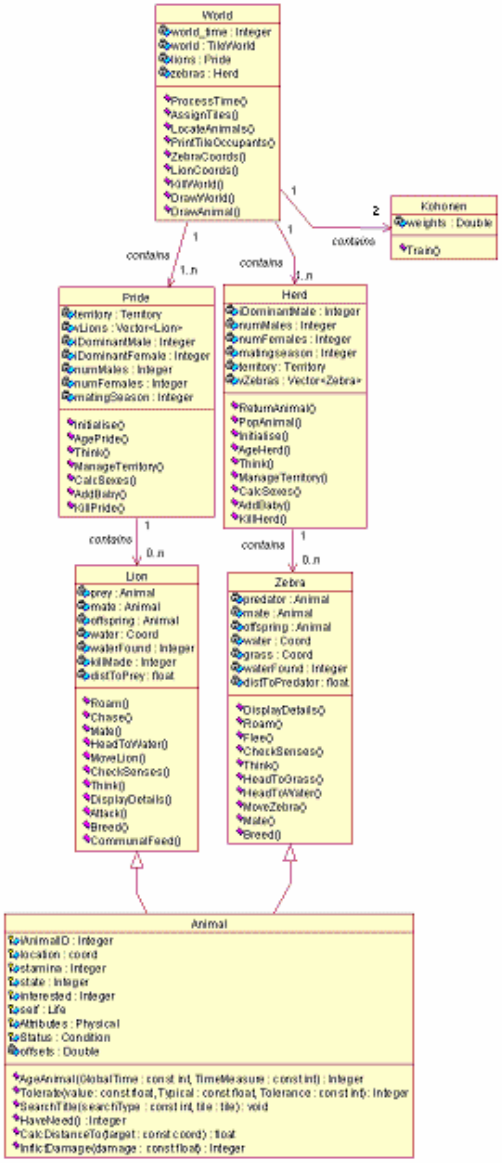

Figure 2: Implementation Class Diagram

An example of interactions between the predators and prey is shown below in Figure 3, where each picture shows a subset of the tile-world (in this instance, all tiles are the same type). In the top left diagram, two lions (on the far edges of the screen), have detected a zebra, and the zebra has seen the lion to its right. In the top right diagram, the zebra turns to flee from the first lion, but is confronted by another lion. In the bottom left diagram, the zebra has turned back again (one emergent property of the simulation was confusion). In the bottom right diagram, the zebra is now in the process of being eaten by one lion who is about to be joined in their meal by the other.

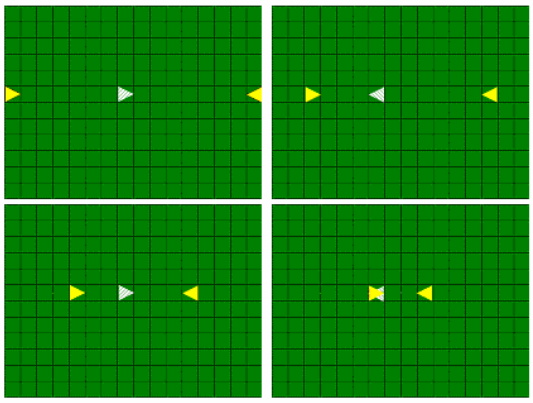

Figure 3: Predator's closing

In this section we realised the design of our agent architecture using a *PN* within the context of a predator-prey ecosystem simulation. Agents are able to reason, taking into consideration the external world and their internal states without requiring an explicit world model. Agents are able to learn without having an explicit learning mechanism. Once the initial training of the *PN* is complete, the global cognitive mechanism continues to learn without explicit representation in the world. Based on our observation of behaviour, it does appear that agents do interact with the world in different ways some of the time, such that the requirement of individuality is satisfied. We now discuss the evaluation of our predator-prey ecosystem.

IV. Evaluation

We evaluated two things: the scalability of representing increasingly large populations using the *PN*; and the accuracy of interactions in a predator-prey ecosystem. It is important to investigate *scalability* because there are always upper bounds that constrain how many members can be represented simultaneously in a population. It is important to investigate the accuracy of interactions because no model of cognition exists to which the *PN* could be compared. Thus cognitive models can only be investigated indirectly- via the interactions of large numbers of agents using the *PN*.

The scalability of the *PN* was evaluated by measuring the length of time it took for all of the agents to access the *PN* once (all predators round-robin, then all prey). The machine

on which this with 512M RAM. The predator-prey ecosystem was implemented in C++, running under the Gentoo Linux operating system.

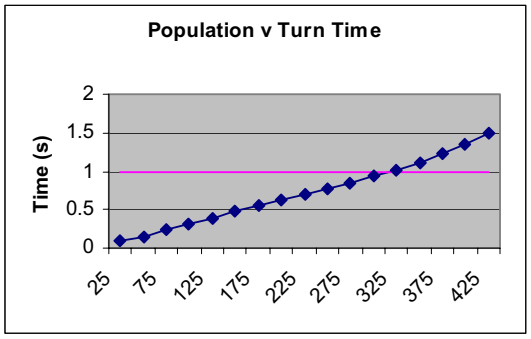

Figure 4: Single turn time for X agents

As shown above in

Figure 4, population numbers were increased in intervals of 25. Note that the population numbers were split 1:1 – for X=100 there were 50 lions and 50 zebras. The graph shown is reasonably linear, fitting the formula $t = X - 325$. We only considered the length of processing time for one turn and did not attempt to represent realistic interactions with such large numbers, even though the tile-world could be flexibly extended and modelled, due to a lack of information to which these interactions could be compared and limited viewing area. These experiments, at least, demonstrated that it was possible to quickly compute reasonably large populations of agents using the *PN* (in the order of a few hundred).

While no psychological mechanism exists to which our *PN* can be compared, the *interactions of the predator-prey ecosystem* (whose individuals use this *PN*) can be assessed. Unfortunately, such an evaluation fails to evaluate the *PN* directly; using another global mechanism instead of a *PN* might be equally effective. The strongest claim that such an evaluation allows us to make is that representing collective unconsciousness using a neural network is similar to a simple mathematical model, and thus such a representation may potentially be useful in simulating artificial life.

Intuitively, interaction in a predator-prey ecosystem has two components. Firstly, an increase in the number of prey can support a larger population of predators by increasing the food supply. Secondly, when the number of predators reaches a certain size, they eat too many of the prey, thus the food supply dwindles, thus only a small number of predators can be supported because some predators starve to death (assuming no other food supply). It is trivial to assess these interactions. At particular intervals, the numbers of each population are recorded and these numbers are plotted on a graph. Changes in these numbers over time are due to births and deaths.

A number of mathematical models of predator-prey interaction have been proposed; the simplest is the Lotka-Volterra model which was developed independently by Lotka (1925) and Volterra (1926) [7]: The model has been compared to real predator-prey populations such as the arctic lynx and snowshoe hare (whose numbers have been closely monitored over a great length of time) and appears to be a surprisingly accurate approximation.

The LVM equations consist of two interacting differential equations: one for each species. In the equation shown below, X = is the number of prey, Y = the number of predators, a = the rate of prey population increase, b = predation rate coefficient, c = reproduction rate of predators per single prey eaten, and d = predator mortality rate.

Measurement was performed was an Intel Pentium 4 2.4 GHz

$$\frac{dX}{dt} = aX - bXY$$

$$\frac{dY}{dt} = cXY - dY$$

A graph of these equations is shown below in Figure 5, where X = 60 initially, Y = 20 initially, a = 0.142, b = 0.00355, c = 0.00142, and d = 0.1065. These constants gave a clear representation of the interactions in the predator-prey ecosystem. The cyclic nature of the LVM perfectly matches the way numbers should intuitively rise and fall.

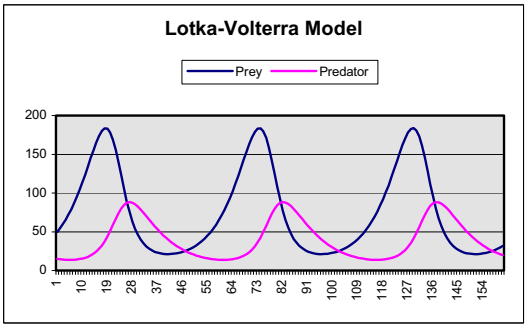

Figure 5: LVM predicted population numbers by steps

In order to compare LVM to agents using our PN, we constructed a predator-prey simulation with the same initial numbers as the LVM graph above, having 20 predators and 60 prey. The world we constructed for our agents consisted of 100*100 tiles. There was four tile rings: at the centre, water, dirt, grass, and then water at the perimeter. Zebras tended to stay close to the edges of the outer ring at the intersection of water and grass. Those zebras that roamed towards the inner edge of the grass occasionally became thirsty and crossed the dirt to the central water source. If they were hungry as well and did not make it back to the grass, the zebras could starve to death. Since there were less zebras at the centre, more lions in the centre also tended to starve more than lions on the perimeter. Population numbers recorded in this simulation were tracked and the resulting numbers are graphed below in Figure 6.

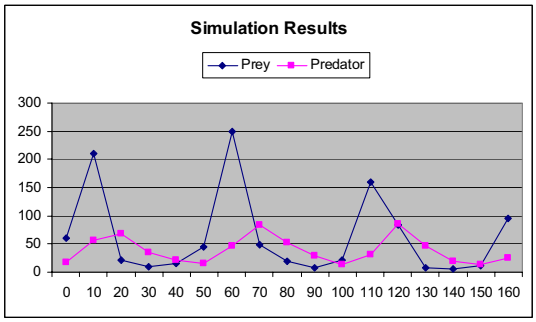

Figure 6: Predator-prey numbers by turns

The similarities in these graphs were unexpected. The cycles of interacting simulation population numbers seemed to follow the LVM reasonably closely. On the other hand, the maximum numbers of each population at their peak were inconstant. The reason for this variance is possibly because the four constants used by the LVM (a,b,c,d), are dynamic in an actual simulation, depending on a number of factors. Unfortunately, no immediate pattern is discernable in these peak changes, and further investigation is required into how these constants are a function of the simulation state. The LVM also makes two other assumptions that could contribute to these differences: it assumes that prey only dies because predators kill them; and that all individuals in each species behave identically

In this section we evaluated the scalability of representing increasingly large populations using the *PN*; and the accuracy of interactions in a predator-prey ecosystem. We found that it was possible to compute a few hundred agents for one turn in around a second. We also found that population numbers fit the cyclic pattern of the Lotka-Volterra model of predator-prey interactions.

V. Conclusion

In this paper we described the design and implementation of a common mechanism to represent the cognition of groups of agents, based on Jung's theory of collective unconsciousness. We demonstrated that it was computationally possible to process the simulated interactions between hundreds of simple agents. We also demonstrated that the simple predator-prey ecosystem we constructed had similar population number oscillation to a predictive mathematical model for these numbers.

There are three main areas that are currently undergoing investigation. Firstly, we are curious to discover how particular terrain impacts the predator-prey interactions, since terrain differences can be compared to the real world. Secondly, we are experimenting both with different contributions from each agent to the *PN*, and different contributions back from the *PN* to each agent, to evaluate how the *PN* and the agents are affected. Finally, we are developing a component-based approach to constructing the *PN*, so that different agents will use different neural network combinations. It is hoped that such an arrangement will produce class-based behavioural differentiation; agents with the same components will demonstrate visibly similar behaviour and agents with different components will demonstrate visible behavioural differences.

There are several areas in which this representation of collective unconsciousness using neural networks could be applied. Artificial life and ecological modelling typically represent large populations of interacting agents that also interact with the world around them. The need to train many neural networks prior to simulation limited their application in these fields. Many computer games also represent small populations that engage in behaviour that approximates the predator-prey relationship. Specific agent offsets could represent non-playing characters at various levels of experience.

References

[1] J. R. B.Muller, "Neural Networks: An Introduction 2nd Edition," pp. 2-12, 1990.
[2] H. B. D. Martin T. Hagan, Mark H. Beale, *Neural Network Design*, Paperback ed: Martin Hagan, 2002.
[3] R. Rogers, "Object-Oriented Neural Networks in C++," pp. 133-171, 1997.
[4] C. G. Jung, "The Archetypes and the Collective Unconscious," *Collected Works of C.G. Jung*, vol. 9, pp. 87-110, 1980.
[5] B. MacLennan, "Evolutionary Neurotheology and the Varieties of Religious Experience," in *NeuroTheology: Brain, Science, Spirituality, Religious Experience*: University Press, California, 2002.
[6] R. Sun, "Individual Action and Collective Function: from Sociology to Muli-Agent Learning," *Cognitive Systems Research*, vol. 2, 2001.
[7] M. P. a. W. Mende, "The Predator-Prey Model, Do We Live in a Volterra World?," pp. 3-15, 1986.
[8] Y. Takeuchi, *Global Dynamical Properties of Lotka-Volterra Systems*: World Scientific Publishing Company, 1996.
[9] D. H. Z. Guillermo Abramson, "Statistics of extinction and survival in Lotka-Volterra systems," 1998.
[10] P. J. Morin, *Community Ecology*: Blackwell Publishers, 1999.
[11] J. H. E. R. M. Sibly, T. H. Clutton-Brock (Editor), *Wildlife Population Growth Rates*: Cambridge University Press, 2003.
[12] M. Wooldridge and N. Jennings, "Intelligent Agents: Theory and Practice," *Knowledge Engineering Review*, vol. 10, pp. 115-152, 1995.
[13] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, vol. RA-2, pp. 14-23, 1986.
[14] M. Bratman, *Intention, Plans, and Practical Reason*: Harvard University Press, 1987.
[15] M. W. Ian Dickinson, "An Initial Response to the OAS'03 Challenge Problem," presented at Autonomous Agents and Multi-Agent Systems (AAMAS-03), Melbourne, Australia, 2003.
[16] M. Namee and B. Cunningham, "A Proposal for an Agent Architecture for Proactive Persistent Non Player Characters," *TCD-CS-2001-20, Trinity College Dublin*, 2001.