1-1-2005

Adaptive battle agents: emergence in artificial life combat models

Thomas J. A. Baker
University of Adelaide

Matthew Botting
University of Adelaide

Matthew J. Berryman
University of Wollongong, mberryma@uow.edu.au

Alex Ryan
Defence Science and Technology Organisation

Anne-Marie Grisogono
Defence Science and Technology Organisation

See next page for additional authors

Adaptive Battle Agents: Emergence in Artificial Life Combat Models

Thomas J.A. Baker[1,2], Matthew Botting[1,2], Matthew J. Berryman[1,2], Alex Ryan[3,4], Anne-Marie Grisogono[4], Derek Abbott[1,2]

[1]Centre for Biomedical Engineering (CBME), The University of Adelaide, SA 5005, Australia.
[2]School of Electrical and Electronic Engineering, The University of Adelaide, SA 5005, Australia.
[3]School of Applied Mathematics, The University of Adelaide, SA 5005, Australia.
[4]Land Operations Division, Defence Science and Technology Organisation (DSTO), Edinburgh, SA 5111, Australia.

ABSTRACT

We explore emergent behavior in an agent-based model of a complex system. The particular complex system we consider is a battlefield simulation. These agents are modeled in the RePast agent-based modeling environment. We will explore how agents of various capabilities and differing task sets affect the outcome of a battle. The capabilities of these agents include, but are not limited to, the ability to maneuver on the battlefield, receive and understand messages, formulate and send messages and attack enemy agents.

Keywords: agent-based modeling, artificial intelligence complex systems, emergent behavior, genetic algorithms

1. INTRODUCTION

The aim of our project is to design an agent-based model to study emergent behavior in a complex system. In this case the system will be a battlefield simulation with virtual military units. This will fit into a defense software system that explores the use of complex adaptive systems (CAS) in evolving military structures to adapt to new and existing threats. Our system provides better artificial intelligence to military simulation systems currently in use.

2. BACKGROUND AND SIGNIFICANCE

2.1 Adaptive agents

We have developed an agent-based model using adaptive agents. These agents are the active elements in the system. The advantages in using an agent-based system over an optimised model are that the agent-based system will more accurately model how an individual unit will interact with other agents and its environment. Also, as the majority of an agent's interactions will be with other agents, it is impossible to predict all the potential situations and problems an agent may encounter. This also leads to it being impossible to optimally specify an agent's behavior in advance.

The agents are capable of evolving a suite of capabilities, within their environment, by adapting their decision-making logic based upon experience from previous decisions made.

2.2 Complex adaptive systems

It is the interaction between large numbers of these adaptive agents that creates our complex adaptive system (CAS). We view CAS as systems composed of many agents described in terms of rules, interacting in different ways. According to John Holland [1], all CAS consist of what he calls the seven basics. These basics are made up of four properties and three mechanisms. The four properties of CAS are aggregation, non-linearity, flows and diversity. The three mechanisms are tagging, internal models and building blocks [1].

Send correspondence to Derek Abbott
Email: dabbott@eleceng.adelaide.edu.au. Telephone: +61 8 8303 5748.

Smart Structures, Devices, and Systems II, edited by Said F. Al-Sarawi,
Proceedings of SPIE Vol. 5649 (SPIE, Bellingham, WA, 2005)
0277-786X/05/$15 · doi: 10.1117/12.582225

2.3 Emergence

A CAS shows *emergent behavior,* defined as the behavior of a system that cannot be applied to its individual agents. Examples of this include (i) particles in a molecule that do not exhibit behaviors of individual elements, and (ii) economies that can be described as being in recession, but individuals cannot [2].

2.4 Genetic algorithms

Not only will the individual agents be required to adapt, but the armies made up of these agents will also evolve through the use of genetic algorithms (GAs). Through the use of GAs we are able to efficiently provide search coverage of the entire parameter space without requiring the computational effort of an exhaustive search. GAs also avoid the danger that an optimum solution may be lost within the step size, when performing an exhaustive search [3].

Attribute	Quantity
Communication	Affects the range that an agent can successfully communicate with an ally over.
Detection	Determines the agent's maximum range of visibility as well as the chances of overlooking a unit and also being able to identify an agents' attributes.
Lethality	Represents an agent's attacking and defensive capability.
Mobility	As RePast is a turn-based system, this decides how many moves agents get per turn.
Processing	This is the agents' intelligence and affects the agents' ability to form plans and process information passed to it via communication.

Table 1. Agent attributes.

2.5 Applications

The ABM we have developed will be used as part of a larger defense software system. This system is capable of having various types of models inserted into it in a modular fashion. Our ABM enables these other models to behave realistically in a battlefield situation, individually making decisions based on their immediate environment, experience and standing orders from superiors. Our model is thus a much more whole, realistic, and integrated software module compared to existing systems [4-6].

3. SOFTWARE SYSTEM DESIGN

The system starts with two teams of agents, one blue and one red. The chromosomes of our genetic algorithms are generated from these two teams. These chromosomes represent armies selected from the two teams. These armies consist of either random or defined agents. The armies are randomly paired off and go into battle together. After the battle is completed the armies of each team evolve through crossover and mutation depending on the parameters of the system, to evolve the next generation of armies. This new generation of armies is then fed back into the system, and the process repeats. Fig. 1 shows a block diagram of the system.

A prototype version of the software was developed in the NetLogo language [7] and the full version was developed using the RePast platform for Java [8]. RePast is a system for building ABMs that has a lot of built in support for GAs, recording simulation data, and provides a solid foundation for implementing all of the features described herein.

3. 1 Agent design

There is a set of 5 attributes that represents the capabilities each agent possesses. Points are allocated to these attributes from a limited pool of fifteen points. This limit was placed on the agents to ensure that all agents possessed the same

overall ‘strength’. When an agent is created these points are either randomly distributed between the attributes or are specified by the evolutionary process. These attribute scores remain constant over the course of a battle and are used to determine how skilled an agent is at performing actions from its rule set. Table 1 itemizes the attributes possessed by the agents.

At the beginning of every round of battle, each living agent is given one turn by the RePast scheduler. The simulator gives an agent its turn by executing the agent’s step method. Each agent contains two lists. In the first list the agent stores information on the units that it has identified, and the second list contains unidentified the agent’s unidentified neighbors. Once the step method is called up, this second list is cleared.

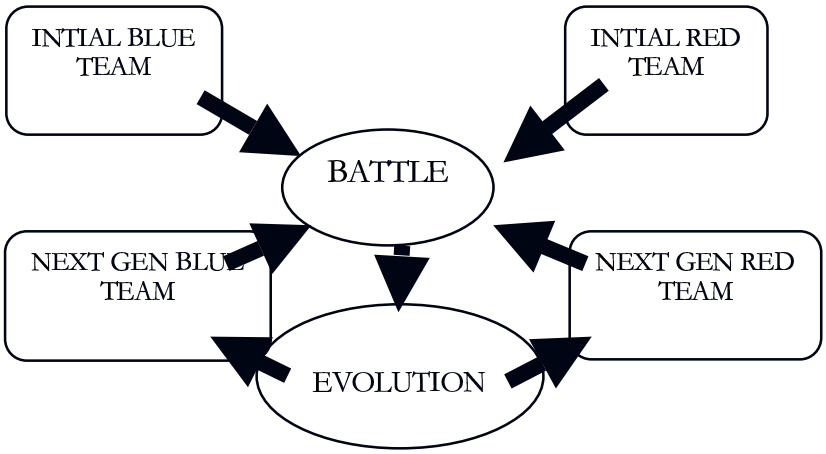

Fig 1. System design block diagram.

The number of moves that an agent can make per turn is calculated as

$$\text{moves} = \left\lfloor \frac{\text{mobility}}{2} \right\rfloor + 1.$$

This gives each agent between one and eight moves per turn. When performing these moves an agent may choose to move, communicate, identify an unknown enemy, or attack an enemy. Agents are also under the restriction that they may attack only once per turn.

At the beginning of a move, an agent will perform a passive identification to gain information on its immediate environment. Passive identification is performed in the agent’s Moore neighborhood, where the Moore neighborhood is the square set of surrounding cells on the battlefield. When agents perform this passive identification, they update the two lists of agents with new information. This passive identification has a limited range. This range is calculated from an agent’s detection capability.

The passive identification is calculated using rings based upon the detection capability as shown in Fig. 2. The visibility range of an agent is calculated as

$$2 \times \text{detection} + 1.$$

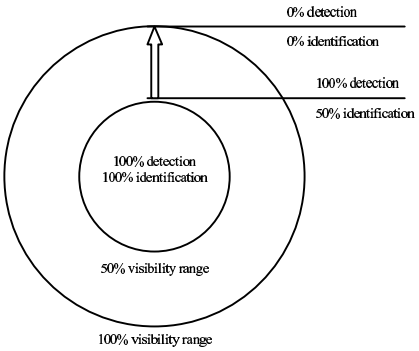

Fig 2. Passive detection ranges.

Using this information about its neighbors, an agent updates a bit string that represents its environment. Rules from an agent's rule set are "activated" based upon this bit string.

If the conditions for a rule's activation are met, the rule is added to a set of activated rules, from which one rule is selected to be performed.

Each action has a corresponding weight that is used to determine what rule will be executed, when multiple rules have been activated. These weights are initially set to values based upon the agent's relevant capabilities. Agents also dynamically update these some of these weights depending on whether the performing the action has a desirable outcome or not. The weights may also be changed through mutation in the evolution process.

There are seventeen rules that agents may perform. This rule set includes rules that allow the agents to maneuver on the battlefield, receive and understand messages, formulate and send messages and attack enemy agents.

3.2 Evolution design

As in Nature, GAs are used to allow for survival of the fittest. Here we are considering armies as organisms to be evolved. The two major steps involved in this model are selecting breeding pairs and then evolving the next generation from them.

3.2.1 Selection of breeding pairs

Our approach to the design of the genetic algorithm for evolution was in treating an army of agents as a chromosome, with the individual agents being the genes. These armies are then randomly paired off against opposition armies and then sent into battle. Each army's performance in the battles is evaluated against using a fitness function. The armies of each team are then ranked by this performance metric. Agents are then selected from armies that make up the current generation. These agents are swapped between armies of the same team to create new armies. When evolving the next generation of armies we give a higher probability to selecting agents from the armies with higher fitness. Fig 3 shows a small example of how this algorithm works.

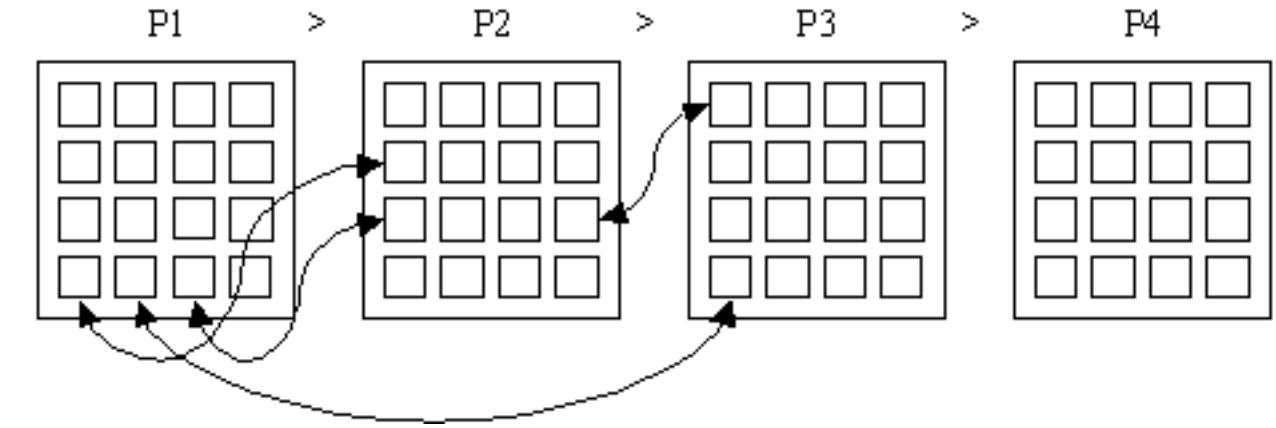

Fig 3. Example of crossover implementation. with the small squares representing individual agents, larger squares representing an army of agents and the *Pn* values representing the fitness of each army.

3.2.2 Performance metric

The performance metric used to measure the fitness of an army is a ratio of enemy agents killed to agents lost. To avoid the problem of dividing by zero, we add one to the number of agents lost.

To perform crossover we also need a previous generation of armies. The first generation of armies is created randomly. As this generation is not a product of evolution, we refer to it as generation zero.

3.2.3 Mutation

Each time we perform crossover and create new agents there is also a probability that the agent will mutate. When mutation occurs an agent has its attributes and decision weights randomly reallocated. To make sure that mutating agents do not violate the limits on their capabilities, we perform mutation by reallocating a portion of these points. We must also ensure that all of the values for the agent's attributes and weights remain positive.

4. TESTING

The first tests that we ran on our model, were chosen in order to demonstrate that the simulation was working correctly. These tests would also allow us to investigate the evolution of the armies within the simulation against various forms of enemies.

The evolution tests we ran involve running an evolving blue team vs. random red team for 100 generations, an evolving blue team vs. static red teams for 100 generations and an evolving blue team vs. an evolving red team. We ran two different versions of the static red team tests. The static red team was evolved against a random team for 50 generations and a second time it was evolved against a random team for 100 generations.

After these evolution tests, we ran tests that had been designed to measure the effects of various parameters on the evolved agents. We decided to investigate the effects of disabling agent communication, using different rank selection probabilities in the genetic algorithm and also looked at the effect of a larger battlefield.

All of these tests are run with the 20 Chromosomes (armies) made up of 20 individual agents each. Each of the tests were run on 100 × 100 cell battlefield. The rank selection probability was set to 40% and the probability of mutation was 5%. Each of these tests was independently run 10 times in an attempt to gain statistically valid results.

Each battle is also limited to a maximum 1000 turns at which time it ends, unless one army is completely wiped out before this time. When one army is completely killed, the surviving team is labeled as the winner. In the case that both

teams still have units left after 1000 turns the battle is declared a draw. This value of 1000 turns is an arbitrary choice that was chosen in an attempt to keep the test run time reasonable.
The performance metric for armies is based upon the ratio of the number of enemies killed to the number of casualties taken. We added one to the denominator so that we are not dividing by zero when no casualties are taken. This number is then multiplied by 100 and then rounded to get an integer.

5. RESULTS

5.1 Evolving blue army versus random red army

We can see that slope of the trend line for the average fitness value is 12.3. This tells us that overall in every generation the fitness value approximately increases by an average of 12 units. We can see from the graph that the actual improvement in the fitness of generation reduces as the average approaches the optimum value of 2000.

We also investigated the armies and tactics that the system had evolved against a random enemy over the 100 generations. For this investigation we extracted the 400 agents from the final generation of each run, giving us a total of 4000 agents (20 units ×20 chromosomes × 10 runs). Out of these 4000 units, we found that the system had evolved 103 unique agents. This value corresponds to 2.5% of the population. The average values for the agents' capabilities are shown in Table 2.

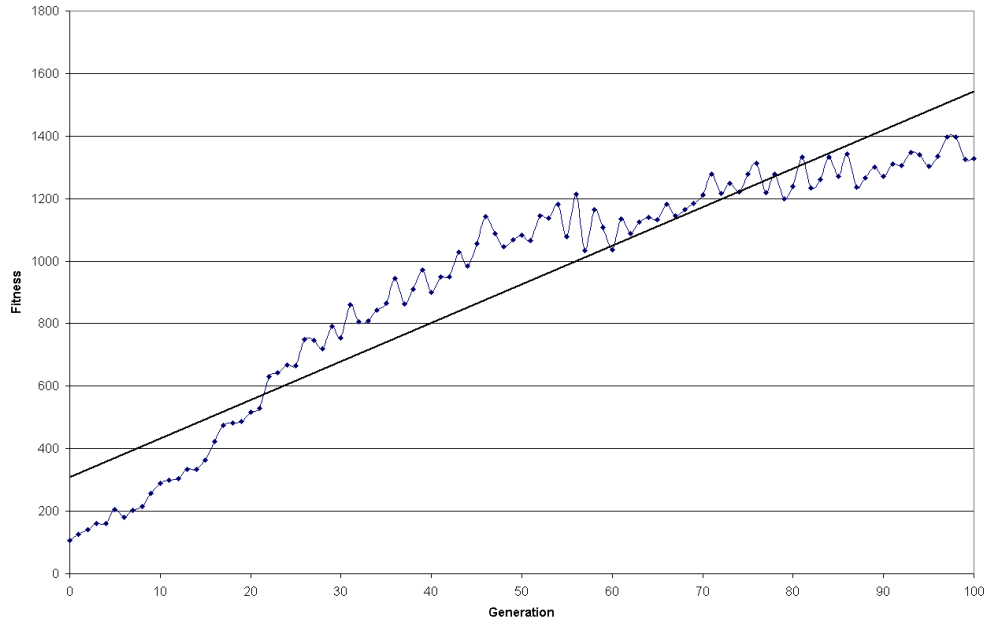

Fig 4. Average blue fitness vs generations.

Attribute	Average Value	Standard Deviation
Communication	1.05	1.32
Detection	4.94	1.49
Lethality	6.44	1.80
Mobility	1.26	1.00
Processing	1.31	1.54

Table 2. Average attribute capability scores.

From these results we can see that the most desirable attributes are lethality and detection. This result leads us to believe that the most vital capabilities for an agent are to be able to detect the enemy units and then be able to dispose of them. It does appear that all of the other attributes are also useful or they would have values of much closer to zero.

From the studying the average weights of the decision variables, we see that the most successfully executed actions are 'attack an unidentified enemy' followed by 'active identify' and 'attack a weaker enemy.' We expect that the weight of attacking an enemy will increase as this rule will have a greater than 50% probability of success, with this probability increasing as the agents evolve higher lethality scores. The most interesting result is that "attack an unidentified enemy" is by far the most successfully executed action. This leads us to infer that agents are attacking an enemy agent as soon as they encounter it, relying on the fact that their own high lethality will give them a good chance at success.

5.2 Evolving blue versus static red

For our second test we evolve an initially random blue team against static red team. This static red team is the result of 50 generations of evolution against a random enemy. We then repeated this test against a static enemy that had been evolved over 100 generations. Our expectation was that the red team would hold a large initial advantage. The question we were investigating was how quickly the evolving blue team would catch up to this red team.

Fig. 5 shows the average number of blue wins for each generation. We can clearly see that in the early generations the strong static red team dominates the blue team. The graph also shows that the blue army quickly evolves to being on a more level playing field, achieving victory 50% of the time by the 20th generation. And by the 40th generation the blue team has increased its average success rate to greater than 75%.

Fig. 6 shows the same graph, however, this time the blue team is facing an enemy that has been evolved over 100 generations. Here the results are similar but the improvement through evolution is occurring much more slowly. Now after 100 generations the blue team is only winning 6 battles on average instead of 17.

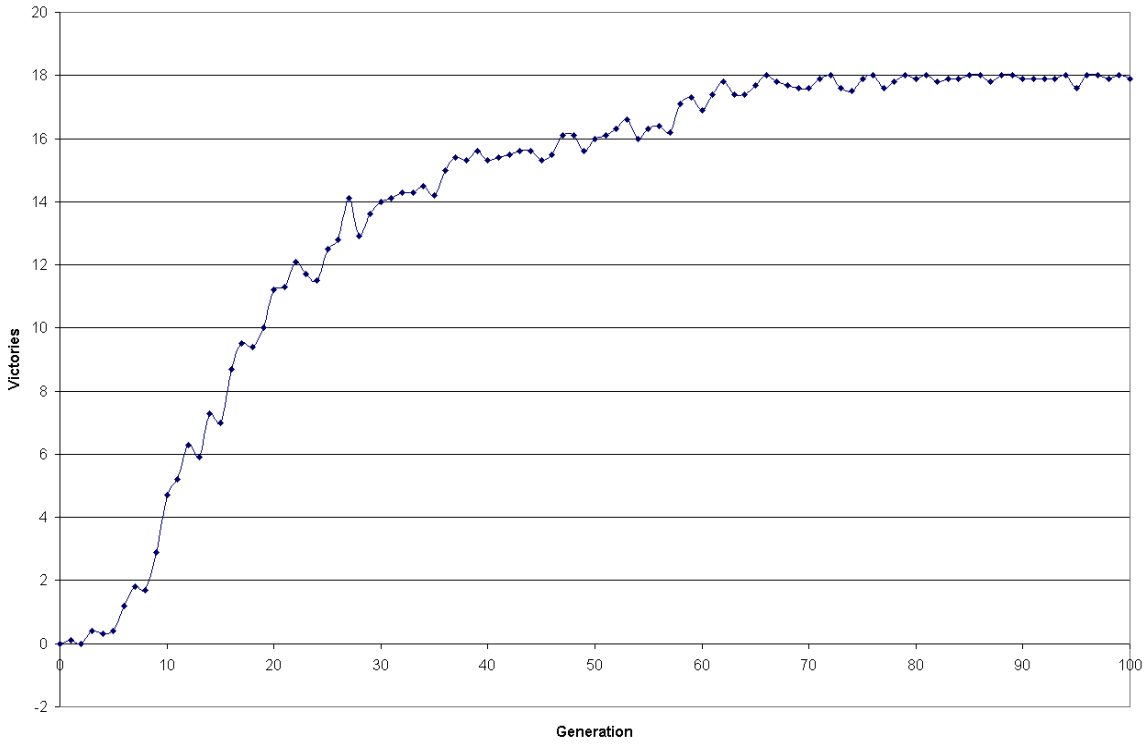

Fig 5. Number of blue victories vs static red enemy evolved over 50 generations.

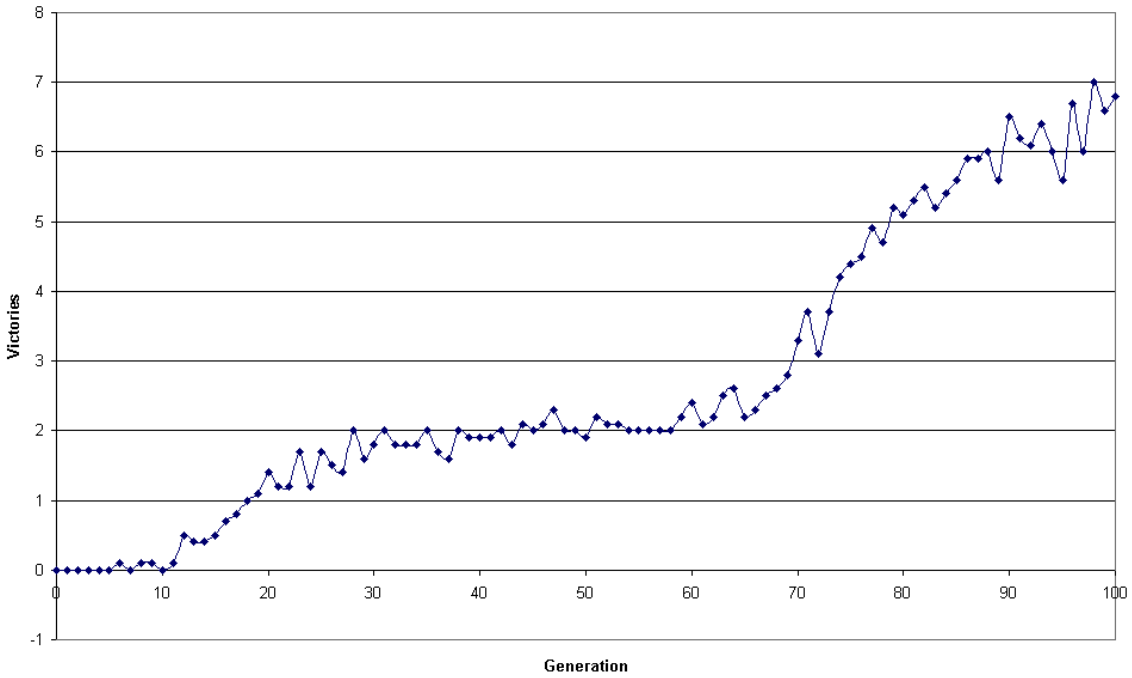

Fig 6. Number of blue victories vs static red enemy evolved over 100 generations

5.3 Coevolution of the blue team and red team

In this test both red and blue teams were initially generated at random. For this test, when battles are completed, both of the teams will undergo evolution. Our goal was to investigate how the two teams will evolve to counteract changes that the opposition makes. We also looked at how often the overall "lead" changes between the teams.

After taking the average number of wins of red and blue across all 10 runs and 100 generations it appears that red came out on top with an average of 14 wins and blue with 6 wins. However, the standard deviations are of this average number of wins and of similar magnitude. This implies the difference is not significant. In fact, this is known as a "red queen" situation—named after a chapter in Lewis Carroll's *Alice Through the Looking Glass*, where both parties go nowhere while running. The average performance metric also corresponds to the win/loss ratio, with the red team having an average of 220 and the blue team only 80.

We were able to calculate the average number of times that the overall lead changes per run based upon the average performance metric. We discovered that over the 100 generations the simulation averages 7 changes in the overall lead. We also investigated when these lead changes occurred during the 100 generations.

We found that, out of the average 7 lead change, 2.4 of them occurred in the first 20 generations whilst only 1.9 occurred in the last 40 generations. This indicates that the lead does swap around throughout the entire evolutionary process, but it oscillates more frequently at the start of the process. We believe this is due to the initial generations being formed randomly, rather than being evolved from a previous team.

Fig. 7 is a graph of one of the ten runs of this test. It is used as comparison with the averages shown above. This graph appears to fit the average case reasonably well. We can see that the average fitness values oscillate quickly, early on in the evolutionary process, and at a decreased rate as we reach the later generations. Also by plotting the number of wins and the average performance metric on the same graph, we can see that they are very much dependant on each other. In particular between generations 0 and 40 as well as 70 and 90 red has a much higher average metric than blue, which corresponds to it winning almost of the battles over these regions.

Fig 7. Single coevolutionary run

5.4 Disabled communication

For this test we disabled explicit communication between agents to investigate what impact it would have on the system and the tactics evolved. The communication is disabled in the decision making logic so that the communication rule is never activated. As the action is never executed the agent will not adapt the weight of the rule. This weight may, however, be updated by mutation but it will have no effect on the system.

As in first test we performed an analysis on the 4000 agents evolved over 100 generations. This time there were only 78 unique agents or about 2% of the population. This is a decrease from the 2.5% that we obtained in the first test. Table 3 shows a comparison between the capabilities agents evolved in the first test and this one.

Attribute	Communication Activated	Disabled Communication
Communication	1.05	0.39
Detection	4.94	6.17
Lethality	6.44	6.55
Mobility	1.26	1.30
Processing	1.31	0.59

Table 3. Comparison of average capabilities

From Table 3 we can see that the system has realised that the communication and processing attributes are no longer useful for an agent and has almost completely removed them from the system.

The fact that these values have not dropped to zero can be attributed to the rank selection probability of 40%—the system was keeping some agents with a lower fitness and mutation was randomly reassigning points. This maintains diversity in the agents and allows them to adapt to varying conditions more readily—for example, if half way through the simulation communications was suddenly restored, these agents would be able to capitalise on it immediately.

We can also see that the points removed from the communication and processing capabilities were reassigned to detection. This indicates that agents were able to operate with a lower detection capability, because of their ability to communicate with allies and share local information on enemy agents. With communication disabled, agents were forced to rely on their own detection capability when searching for enemy agents.

By removing an agent's communication capability, we have shown the importance of having communications. In our first test it was difficult to say whether the small number of points allocated to communication and processing were actually useful or not. However when we can compare the result with this test we can see that communication is a great asset to the teams and could become even more important if we added more sophisticated communication to the model.

5.5 Varying rank selection probability

The rank selection probability used in crossover determines how we select the breeding pairs. A high probability implies that we are more likely to just use our fittest chromosomes for breeding. While for a lower probability there is more chance for a chromosome to be selected for breedin, even if it is not one of the fittest in the group.

It is important to select a good value for the rank selection probability for the system to perform well. If the probability is too high the system may approach a "good" solution more rapidly, it will however lose its diversity and be unable to adapt to change. If the probability chosen is too low, then the system may maintain its diversity but will converge extremely slowly to a 'good' solution or may it not converge at all. In this test we investigated the effects of changing the probability to 10% and then to 90%. These results can also be compared to our coevolutionary agents, whose rank selection probability is 40%.

We analysed the 4000 blue agents evolved through the 100 generations for each rank selection probability, to investigate how they compared with the results of the coevolution test. The results of this are in Table 4. Surprisingly after 100 generations it does not seem to make much difference as to what the rank selection probability was set at. All of the averaged attributes seem to converge to similar values. The number of unique units does support the theory about diversity, however we expected these values to be spread even further apart.

Rank. Sel. Prob.	Communication	Detection	Lethality	Mobility	Processing	Unique Agents
10%	1.162	6.219	6.254	0.811	0.555	116
40%	0.478	5.682	6.584	1.078	1.179	97
90%	1.633	5.588	5.995	0,926	0.859	83

Table 4. Average agent capabilities and number of unique agents

5.6 Larger battlefield

In previous tests we have used a battlefield of size 100 × 100 units. In this test we investigated what affect using a battlefield 4 times larger (200 × 200 units) would have on the system. We expect to achieve similar battle results to previous tests, but would also expect more battles to result in draws. We also expect the agents to tend to have a higher mobility score.

Running the battles on the larger battlefield resulted in 3% of them being drawn, in comparison to 1.7% for the smaller battlefield. Table 5 is a comparison of the averaged attributes. Looking at these average values the change in battlefield size has had no impact on capabilities that the agents have evolved.

battlefield size	communication	detection	lethality	mobility	processing	unique units
100x100	0.478	5.682	6.584	1.078	1.179	97
200x200	0.582	5.426	6.667	1.270	1.055	82

Table 5. Average agent capabilities and uniqueness of units

6. CONCLUSION

The first result obtained, from the series of tests we performed, was that the system is exhibiting 'intelligent' behavior in allocating points to an agents' capabilities and in updating of agent decision weights.

The Evolving Blue Team vs Random Red Team test shows that an evolving team has a huge advantage over a randomly generated team. It also showed that under the rules that the battle is currently fought, the most desirable attributes were lethality and detection; however communication, information processing and mobility are still useful attributes to have.

The Evolving Team vs Static team test shows just how robust an evolved team can become by facing generation after generation of random enemy. The red team that was evolved over 100 generations, before being set to static, held a clear advantage over the evolving blue team for many generations. After 100 generations blue had only managed to win half of the battles, on average, when it faced the static red team. There are indications however that given enough generations, the blue team would eventually surpass the red team.

The Coevolution test, in particular, demonstrates the need for additional runs to be made. What this test does demonstrate, however, is how the teams react to each other in a "red queen" scenario. On average, over the 100 generations the lead changes about 7 times.

The Disabled Communication test demonstrates how much the system relies on communication. When communication is not available, the system allocates fewer points to the attribute and instead puts them into detection—as the agents can no longer rely on the other units detecting the enemy and passing the information to them. This test also demonstrates the diversity of the system, even though it may seem a good idea to remove the communication attribute altogether, some units persisted with small communication values.

The goal of the Rank Selection Probability test was to vary the rank selection probability used in crossover and investigate its affect on the system and the evolved agents. This is another test that requires more runs before we can draw conclusive results. From the limited runs, the results indicated that, over 100 generations of evolution, the rank selection probability had little affect on the system. The results do support the hypothesis that the lower the rank selection probability, the more diverse the armies remain.

The Battlefield size tests require more runs and more variation in the battlefield size. The results obtained from the limited testing done, however, suggest that changing the battlefield size does not affect the evolution of the agents.

As indicated above, additional runs of simulations need to be performed in order to conclusively prove or disprove our initial findings. In addition there are many things that can be added to the system, such as new actions for agents as well as adding complexity to the current set of agent actions. It would be interesting to investigate the agents that evolve if the rule set is expanded to include many more team-based rules that really on more than just the individual agent.

The effect of terrain on the battlefield is of interest for future work. At the moment the battlefield is assumed to be perfectly flat, with no movement or visibility penalties. In future work, one may include maps or randomly generate obstacles such as mountains, lakes and forests, as well as neutral agents.

ACKNOWLEDGMENTS

We greatly acknowledge funding from The University of Adelaide.

REFERENCES

1. J.H. Holland, *Emergence: From Chaos to Order*, Oxford University Press, Oxford, 1998.
2. J.H. Holland, *Hidden Order: How Adaptation Builds Complexity*, Perseus Publishing, Cambridge, Mass., USA, 1996.
3. P.H. Winston, *Artificial Intelligence* Third Edition, Addison-Wesley Publishing Company, 1992.
4. F.L. Smith, *An agent based simulation of competitive information diffusion – modeling espionage*, Proc. Santa Fe Institute Complex Systems Summer School, NM, USA, 2003.
5. K.A. Stork, *Toy soldiers: preliminary agent based modeling of networked combat forces*, Proc. Santa Fe Institute Complex Systems Summer School, NM, USA, 2003.
6. E. Matson and S. DeLoach, *An organization-based adaptive information system for battlefield situational analysis*. Proc. Knowledge Intensive Multi-Agent System Conference, Cambridge, MA, USA, 2003.
7. S. Tisue and U. Wilensky, *NetLogo: A Simple Environment for Modeling Complexity*. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL, USA, 2004.
8. N. Collier, *RePast: An RePast: An Extensible Framework for Agent Simulation*, Social Science Research Computing, University of Chicago, 2002.