# How Communication Can Improve the Performance of Multi-Agent Systems

Kam-Chuen Jim[*]
NEC Research Institute, Inc.
4 Independence Way
Princeton, NJ, 08540
kamjim@research.nj.nec.com

C. Lee Giles[†]
School of Information Sciences & Technology
and Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16801
giles@ist.psu.edu

## ABSTRACT

We analyze a general model of multi-agent communication in which all agents learn to communicate simultaneously to a message board. We show that the communicating multi-agent system is equivalent to a Mealy finite state machine whose states are determined by the agents' usage of the learned language. Increasing the language size increases the number of possible states in the Mealy machine, and can improve the performance of the multi-agent system. We introduce the term *semantic density* to describe the average number of meanings assigned to each word of a language. Using semantic density, a simple rule is presented that provides a pessimistic estimate of the minimum language size that should be used for any multi-agent problem in which the agents communicate simultaneously. Simulations on a version of the predator-prey pursuit problem, a simplified version of problems seen in warfare scenarios, validate these predictions. The communicating predators evolved using a genetic algorithm perform significantly better than all previous work on similar preys.

## Keywords

multi-agent communication/collaboration, agent communication languages, multi-agent simulation, evolution of agents

## 1. INTRODUCTION

Allowing agents to communicate and to learn what to communicate can significantly improve the flexibility and adaptiveness of a multi-agent system. This paper studies an ideal case where each agent has access to a small set of local information and through experience learns to communicate only the additional information that is important. While many researchers have shown the emergence of beneficial communication in multi-agent systems, very few have looked into how communication affects the behavior or representational power of the multi-agent system. The results of this paper contribute to this area by looking at the relationship between the communication behavior of a multi-agent system and the finite state machine that completely describes this behavior. With this knowledge we can better understand how communication increases the representational power of a multi-agent system.

### 1.1 Previous Work

The role of communication in multi-agent systems remains one of the most important open issues in multi-agent system design. Previous work has shown that beneficial communication can emerge in a multi-agent system. Ackley and Littman [1] show that agents can evolve to communicate altruistically even when doing so provides no immediate benefit to the individual. MacLennan and Burghardt [11] use genetic algorithms to evolve finite state machines that cooperate by communicating in a simple abstract world. Walker and Wooldridge [16] study the emergence of conventions in multi-agent systems as a function of various hard-coded strategy update functions, including update functions where agents communicate to exchange memories of observed strategies by other agents. Luc Steels [13] show that a set of agents can create their own vocabulary in a random manner, yet self-organization occurs because the agents are coupled in the sense that they must conform to a common vocabulary in order to cooperate through communication. Saunders and Pollack [12] allow agents to communicate real-valued signals through continuous communication channels and show it was possible to evolve agents that communicate the presence of food in a food trail-following task. Balch and Arkin [2] assigned robot agents to 3 tasks (foraging, consuming, and grazing) and showed that communication significantly improves performance on tasks with little environmental communication, and that more complex communication strategies provide little or no benefit over low-level communication.

While many researchers have shown the emergence of beneficial communication, very few have analyzed the nature of the communication and how communication effects the be-

[*]Also with Physiome Sciences, Inc., 307 College Road East, Princeton, NJ, 08540

[†]Also with NEC Research Institute, Inc., Princeton, NJ 08540

havior or representational power of the multi-agent system. Gmytrasiewicz and Durfee developed a "Recursive Modeling Method" to represent an agent's state of knowledge about the world and the other agents in the world [4]. Furthermore, Gmytrasiewicz, Durfee, and Rosenchein used this framework to compute the expected utility of various speech acts by looking at the transformation the speech act induces on the agents' state of knowledge. Hasida et al. [6] show that with certain assumptions, communication can be treated as an n-person game, and the optimal encoding of content by messages is obtained as an equilibrium maximizing the sum of the receiver's and speaker's expected utilities.

## 2. THE PREDATOR PREY PROBLEM

The predator-prey pursuit problem is used in this paper because it is a general and well-studied multi-agent problem that still has not been solved, and it is a simplied version of problems seen in numerous applications such as warfare scenarios and computer games. The predator-prey pursuit problem was introduced by Benda et al. [3] and comprised four predator agents whose goal is to capture a prey agent by surrounding it on four sides in a grid-world. Luke and Spector [10] and Haynes and Sen [7] used genetic programming to evolve predator strategies, and Haynes and Sen [7] showed that a linear prey (pick a random direction and continue in that direction for the rest of the trial) was impossible to capture reliably in their experiments because the linear prey avoids locality of movement. Korf [9] studied a version of the predator prey problem in which the predators were allowed to move diagonally as well as orthogonally and the prey moved randomly. Tan [15] used reinforcement learning and showed that cooperating agents that share sensations and learned policies amongst each other significantly outperforms non-cooperating agents in a version of the predator-prey problem. Stephens and Merx [14] study a simple non-communicating predator strategy in which predators move to the closest capture position, and show that this strategy is not very successful because predators can block each other by trying to move to the same capture position. Stephens and Merx also present another strategy in which 3 predators transmit all their sensory information to one central predator agent who decides where all predators should move. This central single-agent strategy succeeds for 30 test cases, but perhaps the success rate would be much lower if the agents were to move simultaneously instead of taking turns.

This paper uses an implementation which is probably more difficult for the predators than in all previous work:

1. In our configuration, all agents are allowed to move in only four orthogonal directions. The predators cannot take shortcuts by moving diagonally to the prey, as they do in [9].

2. All agents have the same speed. The predators do not move faster than the prey, nor do they move more often than the prey, as they do in [7].

3. All agents move simultaneously. Because the agents do not take turns moving (e.g. [14]) there is some uncertainty in anticipating the result of each move. In addition, moving the agents concurrently introduces many potential conflicts, e.g. two or more agents may try to move to the same square.

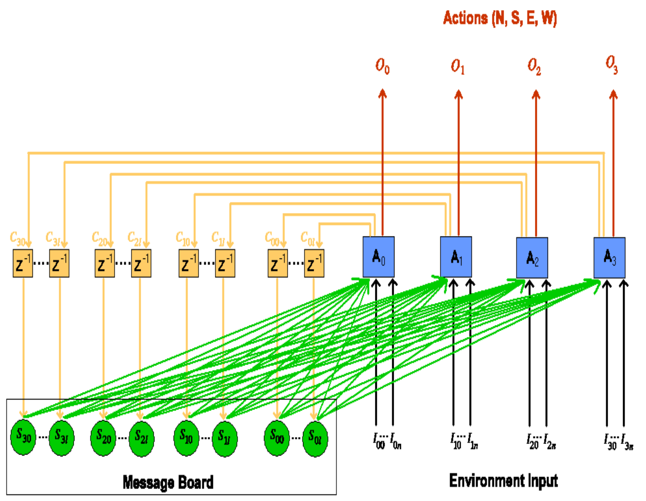


**Figure 1: Multi-Agent Communication as a single Finite State Machine. $l$ is the length of the communication strings.**

4. The predators cannot see each other and do not know each other's location. If this information is essential then the predators will have to evolve a language that can represent such information.

The world is a two dimensional torus discretized into a 30x30 grid. If an agent runs off the left edge of the grid it would reappear on the right edge of the grid, and a similar behavior would be observed vertically. No two agents are allowed to occupy the same cell at the same time, and agents cannot move through each other. If two or more agents try to move to the same square they are blocked and remain in their current positions. At the beginning of each scenario the agents are randomly placed on different squares. Each scenario continues until either the prey is captured, or until 5000 time steps have occurred without a capture.

Two prey strategies are used in the simulations. The Random Prey chooses it's next action at each time step from the set N, S, E, W using a uniform random distribution. The Linear Prey picks a random direction at the beginning of a trial and continues in that direction for the duration of the scenario. It has been shown that the Linear Prey can be a difficult prey to capture [13], [7] because it does not stay localized in an area. In our simulations this is an even more difficult prey to capture because the prey and predators move at the same speed.

## 3. COMMUNICATION

This paper studies a simple framework in which all predator agents communicate simultaneously to a message board. See Figure 1. At every iteration, each predator agent speaks a string of symbols from a binary alphabet $\{0, 1\}$. The communicated symbols are placed on the message board. Each agent then reads all the strings communicated by all the predator agents and determines the next move and what to say next. The strings are restricted to have equal length $l$. We vary the length $l$ of the strings and study the effect on performance.

## 3.1 Equivalence to a Finite State Machine

This type of communication may be represented as shown in Figure 1, where $\{A_m\}$ is the set of homogenous predator agents, $\{O_m\}$ are the actions of the predators, and $\{I_{mn}\}$ is the set of environmental inputs, where $n$ is the number of inputs and $m$ is the number of communicating agents. The message board can be interpreted as a set of *state nodes*.

The entire set of agents can be viewed as one finite state machine (FSM) with the set of possible states specified by the state nodes $\{S_{ml}\}$. The whole multi-agent system is equivalent to a finite state automaton with output, otherwise known as a finite state transducer. One type of finite state transducer is the Mealy finite state machine, in which the output depends on both the state of the machine and its inputs. A Mealy machine can be characterized by a quintuple $M = (\Sigma, Q, Z, \delta, \lambda)$, where $\Sigma$ is a finite non-empty set of input symbols, $Q$ is a finite non-empty set of states, $Z$ is a finite non-empty set of output symbols, $\delta$ is a "next-state" function which maps $Q \times \Sigma \rightarrow Q$, and $\lambda$ is an output function which maps $Q \times \Sigma \rightarrow Z$.

It is easy to show that the multi-agent system is a Mealy machine by describing the multi-agent system in terms of the quintuple $M$. The input set $\Sigma$ is obtained from the set $\{I_{00}I_{01}...I_{0n}I_{10}I_{11}...I_{mn}\}$ of all possible concatenated sensor readings for the predator agents (for all possible values of $I$). A description of the sensor readings is provided later in this paper. The states $Q$ are represented by concatenation of all symbols in the message board. Since the communication strings comprise binary symbols $\{0, 1\}$, the maximum number of states $N_{states}$ in the Mealy machine is therefore determined by the number of communicating agents $m$ and by the length $l$ of the communication strings: $N_{states} = 2^{lm}$. The output set $Z$ is obtained from the set $\{O_{00}O_{01}..O_{0p}O_{10}O_{11}...O_{mp}\}$ of all possible concatenated actions for all the communicating agents, where $p$ is the number of bits required to encode the possible actions for each agent (for all possible values of $O$). In the general case where the actions do not have to be encoded as binary bits, the output set is simply the set $\{O_0O_1...O_m\}$ of all possible concatenated actions for the $m$ communicating agents. The next state function $\delta$ and output function $\lambda$ are determined by the agents' action and communication policies. The policies themselves may be FSMs or something with even more representational power, in such a case the multi-agent FSM is a hierarchical FSM.

## 3.2 Communication Can Help in Partially Observable Environments

When an agent's next optimal action depends on information that is hidden from an agent's sensors it suffers from the *hidden state problem*. Figure 2 shows an example of a typical hidden state problem that is very common in our predator-prey simulations. In this figure, predator 1 sees the same sensory information for two different scenarios due to the fact that predators cannot sense each other directly. In scenario $a$, predator 1 attempts to move South but is blocked by predator 0 in its path, while in scenario $b$ predator 1 is attempting to move South and is not blocked.

Communication allows agents to "tell" each other environmental information that may have been observable only to a subset of the agents. Obviously, communication will be of little use in this respect in the limit when the same set of information is observable to all agents, but this is probably

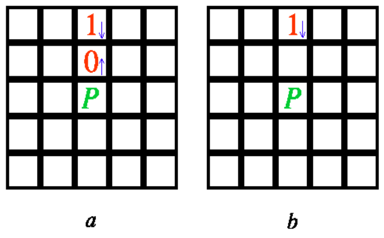


**Figure 2: An example hidden state problem. Predator 1 sees the same sensory information for both scenarios $a$ and $b$ because the predators cannot sense each others' locations.**

not the usual case; e.g. an individual agent will usually have its own internal state that is not observable by other agents. If an agent's state helps determine its behavior, communication may be instrumental in allowing the agents to converge on an optimal plan of action.

## 4. EXPERIMENTAL SETUP

A genetic algorithm is used to evolve predators that communicate. A set of experiments is performed with communication strings of varying length $l$. As the length $l$ increases, the number of strings that are available for communicative acts increases exponentially.

In the sections that follow, GA predators are labeled as "GaPredator($l$)", where $l$ is the length of the communication strings. A communication string of length zero means the predators are not communicating. The performance of *grown* predators (see Section 4.2 below) is also compared. These predators are labeled as "GaPredator($l_0 \rightarrow l_1$)", where $l_0$ is the string length before the agent is grown, and $l_1$ is the length it was grown to.

Separate populations of GaPredator(0), GaPredator(1), GaPredator(2), GaPredator($0 \rightarrow 1$), and GaPredator($1 \rightarrow 2$) predators are matched against the Random and Linear preys. The initial GaPredator($0 \rightarrow 1$) population is grown from the GaPredator(0) population with the best average fitness, and similarly the initial GaPredator($1 \rightarrow 2$) population is grown from the GaPredator($0 \rightarrow 1$) population with the best average fitness.

## 4.1 Encoding Predator Strategies

The behavior of each evolved predator is represented by a binary chromosome string. The length $c$ of the chromosome string is a function of the number of possible states $N_{states}$ observable by the predator based on its sensory information, and the number of actions $b_{actions}$.

The sensory information available to the predators comprise the range and bearing of the prey, and the contents of the message board. The range and bearing are discretized into $N_{range} = 4$ and $N_{bearing} = 8$ sectors. The predators can detect when the prey is 0, 1, 2, and 3+ cells away, measured in terms of Manhattan distance. Note that ranges of 3 or more cells away are lumped under the same sector. The bearing of the prey from the predator is discretized into 8 equal sectors similar to the slices of a pizza pie. The number of symbols on the message board is $ml$, where $m$ is the number of predator agents. The message board can have $N_{messages} = 2^{ml}$ possible messages. The total number of states that can be sensed by a predator is therefore $N_{states} = N_{range} N_{bearing} N_{messages}$. The actions comprise the moves $\{N, S, E, W\}$, and speaking a string of length $l$ at each iteration.. The number of binary bits required to represent the 4 moves are $b_{moves} = 2$. Thus, the total number of action bits is $b_{actions} = b_{moves} + l$ . We arrive at the following equation for the chromosome length $c_{ml}$ of a GA predator that communicates with strings of length $l$ in a team of $m$ predators:

$$\begin{aligned} c_{ml} &= b_{actions} N_{states} \\ c_{ml} &= (b_{moves} + l) N_{range} N_{bearing} 2^{ml} \qquad (1) \end{aligned}$$

so the chromosome length increases exponentially with communication string length $l$ and number of agents $m$.

## 4.2 Growing GA Predators - Coarse to fine search

To improve efficiency, it would be useful to *grow* the predators. Growing means taking a population of predators that have already evolved a language from a set of possible strings, and evolving them further after increasing the set of possible strings they are allowed to communicate. This re-uses the knowledge acquired by predators that were limited to a smaller language. This is effectively a coarse-to-fine search; as we increase the search space by increasing the number of possible strings, the agents can refine the language and communicate other useful, but possibly less critical, information.

By growing the language in these experiments we are making it adaptive. Luc Steels [13] defines an adaptive language as one that "expands or changes in order to cope with new meanings that have to be expressed."

When a population of GA predators with chromosome length $c_{ml}$ is grown to a length of $c_{m(l+1)}$, each new chromosome is encoded such that the behavior of the new predator is initially identical to that of the chromosome it was grown from. The portions of the larger chromosome that are new are not visited initially because the predator is making exactly the same decisions as before and will therefore see the same set of sensory states. During the evolutionary process new sensory states will be visited and the agent will evolve accordingly.

In addition, the population size of the grown $c_{m(l+1)}$ predators is always twice the population size of the $c_{ml}$ predators they were grown from. Half of the population of $c_{m(l+1)}$ predators are grown from the $c_{ml}$ predators, the other half are generated randomly. In this manner the grown predators don't rely solely on mutation for introducing new genetic material to the genes that were copied from the predators with chromosome length $c_{ml}$. They can obtain new genetic material through crossover with the randomly generated individuals.

## 4.3 Evaluating the Fitness of Evolved Predators

The fitness of each evolved strategy is determined by testing it on 100 randomly generated scenarios with different starting locations for the predator and prey agents. The maximum number of cycles per scenario is 5000, after which the predators are considered to have failed. Since the initial population is randomly generated, it is very unlikely that the first few generations will be able to capture the prey. We attempt to speed up the evolution of fit strategies by rewarding those strategies that at least stay near the prey and are able to block the prey's path. The fitness $f_i$ of individual $i$ is computed at the end of each generation as follows, where $N_{\max} = 5000$ is the maximum number of cycles per scenario, $T = 100$ is the total number of scenarios for each individual, and $n_c$ is the number of captures:

- If $n_c = 0$, $f_i = \frac{0.4}{d_{avg} + 0.6 \frac{n_b}{N_{\max} T}}$ where $d_{avg}$ is the average distance of the all 4 predators from the prey during the scenarios, and $n_b$ is the cumulative number of cycles that the prey's movement was blocked by an adjacent predator during $T$ scenarios. The fitness of non-capture strategies can never be greater than 1.
- If $0 < n_c < T$, $f_i = n_c$.
- If $n_c = T$, $f_i = T + \frac{10000T}{\sum_{j=0}^{T} t_j}$, where $t_j$ is the number of cycles required to capture the prey at scenario $j$.

| Predator | Population Size | Mutation Rate |
|---|---|---|
| GaPredator(0) | 100 | 0.01 |
| GaPredator(0 $\rightarrow$ 1) | 200 | 0.001 |
| GaPredator(1) | 200 | 0.001 |
| GaPredator(1 $\rightarrow$ 2) | 800 | 0.0005 |
| GaPredator(2) | 800 | 0.0005 |

**Table 1: Population Size and Mutation Rate GA parameters used in the simulations.**

## 4.4 GA Setup

The following GA parameters were found experimentally to be most effective. We use 2-point crossover with a crossover probability of 0.4. The idea behind multi-point crossover is that parts of the chromosome that contribute to the fit behavior of an individual may not be in adjacent substrings. Also, the disruptive nature of multi-point crossover may result in a more robust search by encouraging exploration of the search space rather than early convergence to highly fit individuals. For a discussion of 2-point crossover and generalized multi-point crossover schemes see [8]. A Tournament selection scheme [5] with a tournament size $Tour$ of 5 is used to select the parents at each generation. In Tournament selection, $Tour$ individuals are chosen randomly from the population and the best individual from this group is selected as a parent. This is repeated until enough parents have been chosen to produce the required number of offsprings for the next generation. The larger the tournament size, the greater the selection pressure, which is the probability of the best individual being selected compared to the average probability of selection of all individuals. The population size $p$ and mutation rate depends on the length of the communication string because the search space increases exponentially with the communication string length. The larger search space translates into longer chromosome lengths. As a general rule, longer chromosome lengths warrant a larger population size and smaller mutation rate. The population sizes and mutation rates used in the experiments are listed in Table 1.

10 trials are performed, with the population initialized randomly at the beginning of each trial. The following is a brief description of the algorithm:

1. Repeat the following for 10 trials on selected prey:
   (a) Randomly generate a population of $p$ individuals.
   (b) Repeat until there is no improvement after 200 generations:
      i. Simulate each predator strategy on 100 scenarios and evaluate its fitness based on the performance on those scenarios.
      ii. Select $p$ individuals from the current population using Tournament selection, pair them up, and create a new population by using 2-point crossover with mutation.
   (c) The best strategy found over all generations is used as the solution of this trial. The fitness of this strategy is then recomputed by testing on 1000 new randomly generated scenarios.
2. The strategy that performed best over all 10 trials is used as the solution.

## 5. RESULTS

Figure 3 shows the best average capture times (over 1000 randomly generated scenarios) and the cumulative number of evolutionary generations that were needed to achieve such capture times. If $G(l)$ is the number of generations that a GaPredator($l$) population was evolved, and $G(l_0 \to l_1)$ is the number of generations that a GaPredator($l_0 \to l_1$) population was further evolved after it was grown from $l_0$ to $l_1$, then the cumulative generations for the best GaPredator($0 \to 1$) and GaPredator($1 \to 2$) populations are computed as follows:

$$G_{cumulative}(0 \to 1) = G(0) + G(0 \to 1)$$
$$G_{cumulative}(1 \to 2) = G_{cumulative}(0 \to 1) + G(1 \to 2)$$

Below is a summary of the performance and convergence results:

- As the length of the communication string increases, the capture time decreases. However, the best capture performance of GaPredator(1) against the Random prey is comparable to the best performance of GaPredator(2) and GaPredator($1 \to 2$), which indicates that a communication string of length 1 was sufficient against the Random prey.
- The evolutionary generations required increases with the length of the communication string.
- The capture performance of grown predators is comparable to the performance of the equivalent non-grown predators, but requires significantly less evolution time. Thus, incrementally increasing the language size is an effective coarse-to-fine method which reduces the search time.
- The evolved communicating predators perform better than all previously published work to our knowledge. A previous work whose experimental setup is most similar to our work is perhaps Haynes and Sen [7], although their setup makes the predators' job easier because they are allowed to move more frequently than the prey. Haynes and Sen and other previous work [9] on similar preys report results as a percentage of trials that lead to capture, whereas the results reported here show 100% capture rate when the predators are allowed to communicate.

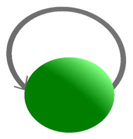


**Figure 4: Finite State Machine of non-communicating GA predators. When the predators don't communicate they act like a FSM with only one state.**

### 5.1 Evolved Mealy Machines

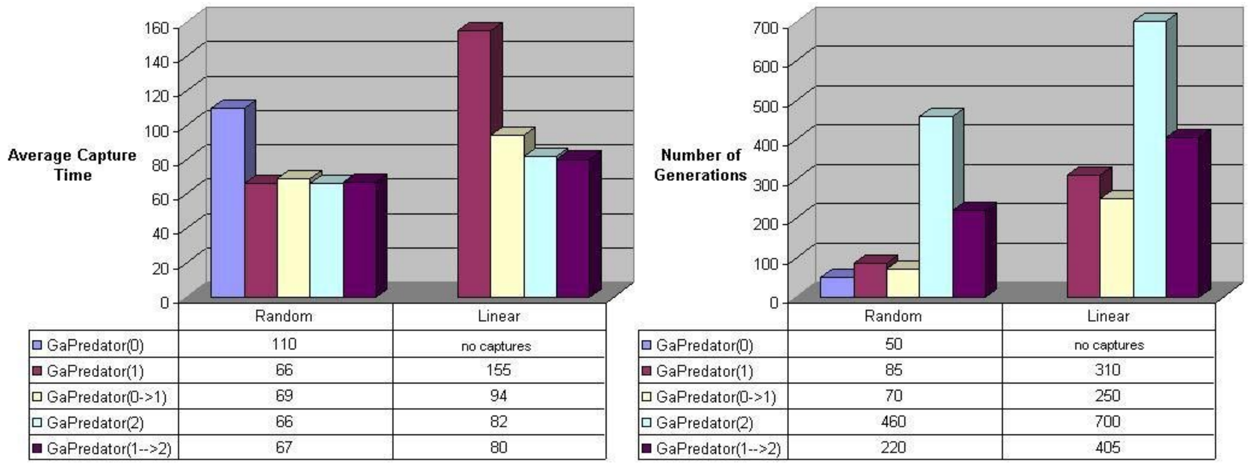
The Mealy machines were obtained by "listening" to the predators talking during actual trials, as opposed to analyzing the predators' GA string to determine what they would say for each possible sensory permutation. This way we only account for states and links on the multi-agent Mealy machine that are ever visited, and ignore states and links that do not contribute to the behavior of the predators because they are never visited anyway.

After obtaining the communication activity of the predators, the states of the Mealy machine are constructed by concatenating the words spoken by all predators on the message board. A different multi-agent state is associated with each unique concatenation. The links represent transitions between multi-agent states (i.e. transitions in the content of the message board) at each time step as a result of the inputs sensed from the environment.

Figures 4, 5, 6 and 7 show the best evolved Mealy machines for non-communicating and communicating predators that were evolved against the Linear prey. The Mealy machines are depicted using what we call Scaled Finite State Diagrams (SFSD). SFSDs provide more information than standard finite state diagrams by representing the relative importance of links and nodes in a visual manner. A Scaled Finite State Diagram is described as follows:

- Links are combined to *meta-links*. A meta-link is an aggregate of all links that connect the same two nodes together, irrespective of their input/output pairs. This simplifies the figures because otherwise the individual links are so numerous that they would completely fill all the space.
- The thickness of a meta-link indicates the number of individual links that were combined to form the meta-link.
- The size of a node indicates its attractiveness and significance. This is measured by the number of incoming links that are connected to that node. A large state node indicates that many environmental input combinations from various states would move the multi-agent system to this state.

Each node is labeled by a number, which is computed by concatenating all the communicated words on the message board and using the language size as the base power. The



| | Random | Linear |
|---|---|---|
| GaPredator(0) | 110 | no captures |
| GaPredator(1) | 66 | 155 |
| GaPredator(0->1) | 69 | 94 |
| GaPredator(2) | 66 | 82 |
| GaPredator(1-->2) | 67 | 80 |

| | Random | Linear |
|---|---|---|
| GaPredator(0) | 50 | no captures |
| GaPredator(1) | 85 | 310 |
| GaPredator(0->1) | 70 | 250 |
| GaPredator(2) | 460 | 700 |
| GaPredator(1-->2) | 220 | 405 |

**Figure 3: Best capture times and the corresponding number of evolutionary generations required to evolve the communicating predators against Random and Linear preys, at communication string lengths 0, 1, 2.**

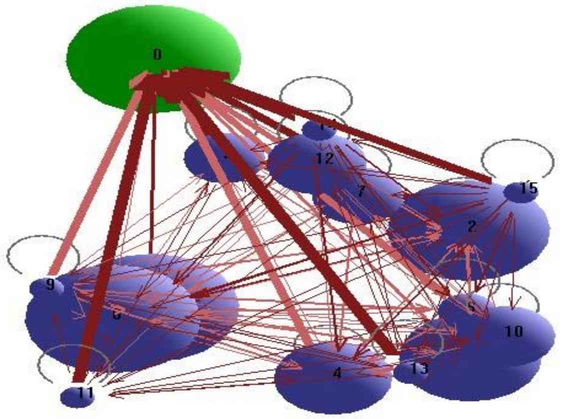


**Figure 5: Finite State Machine of best communicating GAPredator(1) evolved against the Linear prey. All 16 possible states are used. States 0, 2, 4, 5, 6, and 8 are more significant than the other states.**

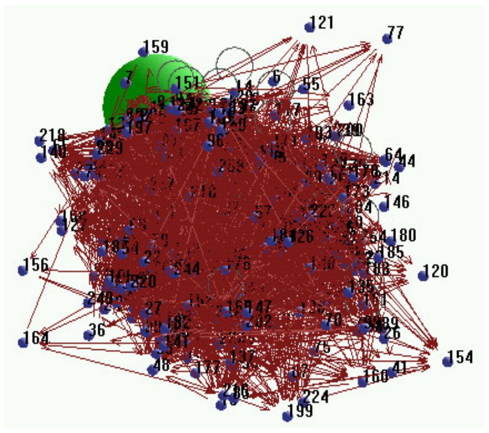


**Figure 6: Finite State Machine of best communicating GAPredator(2) evolved against the Linear prey. All 256 possible states are used.**

start node is labeled "0" because at the start of each scenario the message board is initialized to all zeroes.

Observation of the evolved Mealy machines indicate the following:

- The start state is always very significant in the evolved Mealy machines.

- Growing a language results in a Mealy machine with fewer states than an evolved language that was not grown. For example, the average number of states in the evolved GaPredator(2) machines was 252, while for the grown predators GaPredator($1 \rightarrow 2$) the average was only 87.

- The size of the Mealy machine appears to increase with the difficulty of the problem. See Table 2. For example, the Mealy machines evolved against the Random prey are smaller than the Mealy machines evolved against the more difficult Linear prey. Also, note that the Mealy machine for GaPredator($1 \rightarrow 2$) (see Figure 7) only uses 87 out of 256 possible states, which indicates that increasing the language size (and thus the number of possible states) would not improve results.

## 5.2 Evolved Languages

Table 3 shows an excerpt of the language evolved by the best GaPredator($0 \rightarrow 1$) agents. This excerpt was obtained by clustering the observed communication activity using the *Minimal Spanning Tree* algorithm and displaying some of the larger clusters. As an example, the first line is interpreted as follows: "If the prey is to the far north of me (range of 2, bearing 2) and the message board consists of the symbols (0,0,0,0), speak the symbol "0" and move North."

An important observation from the evolved languages is that it is very difficult, if not impossible, to explain the evolved languages. Looking at Table 3, one would be hard-pressed to say, for example, what the symbol "0" means to the predators since there does not appear to be a pattern to its usage. However, the evolved languages are obviously

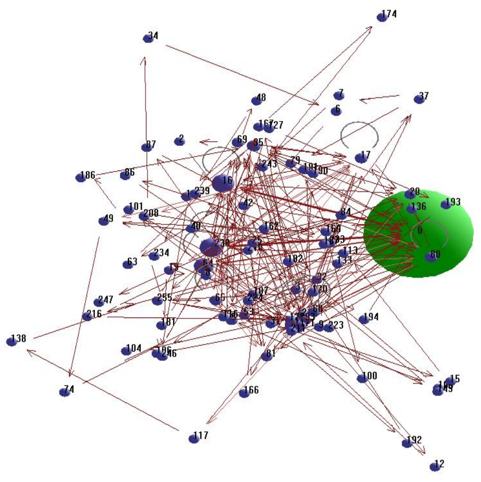


**Figure 7: Finite State Machine of best communicating GAPredator$(1 \to 2)$ evolved against the Linear prey. Only 87 out of 256 possible states are used, but the start state (state 0) is much more significant.**

| **Predator** | **PREY** | |
|---|---|---|
| | Random | Linear |
| GaPredator(0) | 0 | 0 |
| GaPredator$(0 \to 1)$ | 8 | 16 |
| GaPredator(1) | 12 | 16 |
| GaPredator$(1 \to 2)$ | 8 | 87 |
| GaPredator(2) | 12 | 252 |

**Table 2: Average number of states in best predators' multi-agent Mealy machine over ten trials as a function of prey and communication size.**

very suitable because it allows the predators to outperform all previous work on similar preys. We thus conclude that allowing the agents to evolve their own communication language is very useful, since it would have been very difficult for a human designer to construct a similar language that can perform as well.

Also, the evolved languages are tightly coupled with the learning problem and cannot be re-used on a different problem. The languages are integrated with the strategies and available actions of the agents in their environment. Therefore, the portability of the evolved languages is dependent on the portability of the evolved multi-agent strategies.

### 5.2.1 *Semantic Density*

Let us define the *semantic density* of a language as the average number of meanings assigned to each word of the language. The semantic density $\delta$ can be computed as

$$\delta = \frac{\gamma}{\kappa},$$

where $\gamma$ is the total number of meanings represented by the language, and $\kappa$ is the number of words in the language.

We can compute an upper bound $\gamma_U$ on the number of possible useful meanings that the predator agents can communicate by assuming that the space of useful meanings that a predator can possibly communicate includes only the agent's sensory information and its next move. This assumption is justified in our simulations because the agents do not have any internal state information that need to be communicated, and the agents' plan of action applies only to the current time step. Accounting for the environmental information observable for each agent and the 4 actions (N,S,E,W) that an agent can take, we get the following equation for the upper bound on the number of useful meanings:

$$\gamma_U = 4N_{range}N_{bearing} = 128.$$

$\gamma_U$ represents the maximum number of unique meanings that a predator agent can possibly communicate regarding its sensory information and its next action.

Assuming that the agents use all the words available to them, an upper bound on the semantic density of the evolved languages in our simulations is simply

$$\delta_U = \frac{\gamma_U}{\kappa} = \frac{128}{2^l} = 2^{(7-l)},$$

where $l$ is the length of the binary communication string. Effectively, $\delta_U$ is the maximum average number of meanings that need to be assigned to each word to allow for an optimal multi-agent strategy that has access to all available local information.

However, a tighter bound can be obtained by observing traces of the sensory input and movements of all the predators during actual runs. We observed that in all runs the number of words used is still $\kappa = 2^l$, however the number of possible meanings $\gamma_U$ is less than the limit $4N_{range}N_{bearing}$ because not all combinations of sensory input and actions are experienced by the agents. In other words, the observed upper bound on the density $\delta_U^*$ appears to be much less than the theoretical upper bound $\delta_U$. This is illustrated in Table 4, which shows the theoretical upper bound density $\delta_U$ and the average observed $\delta_U^*$ for the best predators at each communication string length. The interpretation of $\delta_U^*$ is slightly different from the interpretation of $\delta_U$: whereas $\delta_U$ is an upper bound that allows for an optimal strategy using all available local information, $\delta_U^*$ is an upper bound that allows for the *best evolved strategy* observed, which may or may not be the optimal strategy.

Our simulations verify that there is indeed heavy re-use of symbols (or words) in the evolved languages. A symbol is used differently depending on the state of the message board. For example, the symbol "1" is used differently when the state of the message board is 1001 versus 0111 in Table 3. Thus the evolved languages are compact and are able to represent more concepts than the $2^l$ possible symbols available to each agent. This re-use of words is also observed in natural languages, where the same word can have different meanings depending on the *context* in which it is used.

| **Input(Range/Bearing)** | **MessageBoard** | **Say** | **Move** |
|---|---|---|---|
| 2 / 2 | 0 0 0 0 | 0 | North |
| 2 / 3 | 0 0 1 0 | 0 | West |
| 2 / 1 | 0 0 0 1 | 0 | East |
| 1 / 6 | 1 0 0 1 | 1 | South |
| 1 / 1 | 1 1 1 1 | 1 | South |
| 1 / 6 | 0 1 1 1 | 1 | West |

**Table 3: Excerpt of the language evolved by best GaPredator$(0 \to 1)$ agents**

| **Predator** | $\delta_U$ | $\delta_U^{*Linear}$ | $\delta_U^{*Random}$ | $N^*$ |
|---|---|---|---|---|
| GaPredator (0) | 128 | | | 1 |
| GaPredator ($0 \to 1$) | 64 | 38 | 10 | 16 |
| GaPredator (1) | 64 | 38.5 | 16 | 16 |
| GaPredator ($1 \to 2$) | 32 | 19.75 | 10 | 87 |
| GaPredator (2) | 32 | 20 | 20 | 252 |

**Table 4: The theoretical upper bound $\delta_U$ on the meaning density and the average observed upper bound $\delta_U^*$ against the Linear and Random preys. $N^*$ is the average number of states in the evolved multi-agent Mealy machines.**

In the communication framework studied in this paper, the content of the message board, or equivalently the state of the Mealy machine, determines the context for the spoken symbols. Therefore, the maximum number of contexts per word is equivalent to the number of states in the evolved Mealy machine, and this places a structural upper bound on the semantic density that can be represented by the multi-agent system. Table 4 shows that for most cases the evolved Mealy machines can more than accommodate the upper bounds on semantic density because the average number of states in the Mealy machines are greater than the semantic density upper bounds. In fact, the cases where the observed upper bound on semantic density $\delta_U^*$ is greater than the number of states are exactly the cases where a larger language improved performance in our simulations. For example, $\delta_U^*$ against the Linear prey with communication strings of length 1 is greater than the number of possible states, and in our simulations increasing the communication length to 2 improved capture performance.

Thus, one pessimistic estimate for the minimum communication string length $l$ is the following rule:

$$\text{Increase } l \text{ until } N_{states} \geq \delta_U,$$

where $N_{states} = 2^{ml}$ is the number of possible states (semantic contexts) in the Mealy machine that represents the multi-agent strategy, and $m$ is the number of communicating agents. The value of $\delta_U$ will be different for each problem, and indeed it may be difficult to estimate in problems where one does not know the space of local information available to each participating agent or when the agents maintain internal state information.

## 6. CONCLUSIONS

A multi-agent system in which all the agents communicate simultaneously is equivalent to a Mealy machine whose states are determined by the concatenation of the strings in the agents' communication language. Thus, evolving a language for this type of communicating multi-agent system is equivalent to evolving a finite state machine to solve the problem tackled by the multi-agent system. The simulations show that a genetic algorithm can evolve communicating predators that outperform the best evolved non-communicating predators, and that increasing the language size improves performance. A method is introduced for incrementally increasing the language size that results in a coarse-to-fine search that significantly reduces the time required to find a solution. Furthermore, a simple rule is derived for estimating the minimum language size that should be used for any multi-agent problem.

## 7. REFERENCES

[1] David H. Ackley and Michael L. Littman. Altruism in the evolution of communication. In *Proceedings of Artificial Life IV* . MIT Press, 1994.

[2] Tucker Balch and Ronald C. Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52, 1994.

[3] M. Benda, V. Jagannathan, and R. Dodhiawalla. On optimal cooperation of knowledge sources. Technical Report BCS-G2010-28, Boeing AI Center, Boeing Computer Services, Bellevue, WA, August 1985.

[4] Piotr J. Gmytrasiewicz, Edmund H. Durfee, and Jeffrey Rosenschein. Toward rational communicative behavior. In *AAAI Fall Symposium on Embodied Language*. AAAI Press, November 1995.

[5] D.E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms*, pages 69–93. 1991.

[6] Kôiti Hasida, Katashi Nagao, and Takashi Miyata. A game-theoretic account of collaboration in communication. In *Proceedings of the First International Conference on Multi–Agent Systems (ICMAS)*, pages 140–147. MIT Press, 1995.

[7] Thomas Haynes and Sandip Sen. Evolving behavioral strategies in predator and prey. *IJCAI-95 Workshop on Adaptation and Learning in Multiagent Systems*, August 1995.

[8] Kenneth A. De Jong and William M. Spears. A formal analysis of the role of multi-point crossover in genetic algorithms. *Annals of Mathematics and Artificial Intelligence Journal*, 5(1):1–26, 1992.

[9] Richard E. Korf. A simple solution to pursuit games. In *Working Papers of the 11th International Workshop on Distributed Artificial Intelligence*, pages 183–194, February 1992.

[10] S. Luke and L. Spector. Evolving teamwork and coordination with genetic programming. In J.R. Koza, D.E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Proceedings of the First Annual Conference on Genetic Programming (GP-96)*, pages 150–156, Cambridge, MA, 1996. MIT Press.

[11] Bruce J. MacLennan and Gordon M. Burghardt. Synthetic ethology and the evolution of cooperative communication. *Adaptive Behavior*, 2(2):161–188, 1993.

[12] Gregory M. Saunders and Jordan B. Pollack. The evolution of communication schemes over continuous channels. In *From Animals to Animats 4: Proceedings of the 4th International Conference on Simulation of Adaptive Behavior*. MIT Press, 1996.

[13] Luc Steels. Self-organizing vocabularies. In *Proceedings of Alife V*, 1996.

[14] Larry M. Stephens and Matthias B. Merx. The effect of agent control strategy on the performance of a dai pursuit problem. In *Proceedings of the 10th International Workshop on DAI*, 1990.

[15] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proc. of 10th ICML*, pages 330–337, 1993.

[16] Adam Walker and Michael Wooldridge. Understanding the emergence of conventions in multi-agent systems. In *Proceedings of 1st ICMAS*, 1995.