Evolutionary Learning of Goal-Oriented Communication Strategies in Multi-Agent Systems

Submitted: 8 April 2015; Accepted 28 May

Alhanoof Althnian, Arvin Agah

DOI: 10.14313/JAMRIS_3-2015/24

Abstract:
Previous studies in multi-agent systems have observed that varying the type of information that agents communicate, such as goals and beliefs, has a significant impact on the performance of the system with respect to different, usually conflicting, performance metrics, such as speed of solution, communication efficiency, and travel distance/cost. Therefore, when designing a communication strategy for a multi-agent system, it is unlikely that one strategy can perform well with respect to all of performance metrics. Yet, it is not clear in advance, which strategy will be the best with respect to each metric. With multi-agent systems being a common paradigm for building distributed systems in different domains, performance goals can vary from one application to the other according to the domain's specifications and requirements. To address this issue, this work proposes a genetic algorithm-based approach for learning a goal-oriented communication strategy. The approach enables learning an effective communication strategy with respect to flexible, user-defined measurable performance goals. The learned strategy will determine what, when, and to whom information should be communicated during the course of task execution in order to improve the performance of the system with respect to the stated goal. Our preliminary evaluation shows that the proposed approach has promising results and the learned strategies have significant usefulness in improving the performance of the system with respect to the goals.

Keywords: *multi-agent system, communication strategy, evolutionary communication, and genetic algorithms*

1. Introduction

A number of research efforts investigated the importance of communication and its impact on the performance of multi-agent systems. Studies are usually conducted by varying communication conditions and testing the performance of the system. We consider the work in [1] and [19], where experiments were carried out to study the effect of communicating different types of information when agents are assigned different tasks. As shown in Figure 1.a, the process starts by manually determining the type of information that agents are allowed to communicate - i.e. none, only goals, only beliefs, or both. Then, the average performance of the system over multiple runs is measured with respect to different metrics such as time to complete, interference, communication efficiency, and duplication of efforts. In their work, whenever agents are allowed to communicate information, they broadcast every value update once obtained. Authors concluded that varying the type of information that agents communicate can significantly affect the performance of the multi-agent system with respect to different metrics, especially if no implicit communication present. Moreover, their results showed that more communication does not always guarantee better performance. The latter conclusion has crucial implication. It indicates that even in applications where communication is free, the system designer should not allow full communication and assume that the system is performing at its best level.

The progress that the work presented above made in understanding the impact of communication on MAS performance, together with the fact that system designers must choose a performance goal [1], motivated us to propose a genetic algorithm-based approach for learning a goal-oriented communication strategy. Therefore, rather than manually creating different communication conditions (as in Figure 1.a), the system designer can start from selecting his performance goal and feed it to the learning system. The system, then, learns a centralized goal-effective strategy that determines *what* information instances should be communicated during task execution, *when*, and to *whom* in order to achieve the best performance of the system with respect to the selected goal (see Figure 1.b). During task execution, agents execute the evolved strategy in a decentralized manner. At each time-step, each agent consults the strategy to decide whether it needs to communicate or not. Therefore, in this work, agents' collective behavior, and hence performance, improves as a result of executing a goal-oriented communication strategy, evolved offline by the GA.

The goal of this work is to propose and preliminary test an evolutionary approach that automatically generates an effective communication strategy with respect to a user-defined performance goal in multi-agent systems. Our ultimate aim is to allow system designers to easily vary the goal and automatically obtain the corresponding communication strategy. Therefore, the system designer does not need to know or analyze the properties of each information instance and its effect on the performance goal of the system. This can eliminate a significant design task in developing a multi-agent system. Moreover, the proposed approach can assist system designers to find

out the potentially best performance that the system can achieve with respect to a specific goal, such as the minimum time or energy that a task takes to be completed. Therefore, a system designer will be able to choose among the performance of the system with multiple communication strategies of varying goals and select the one that has the best fit to the system's needs. A multi-agent version of the Wumpus World [16, 22] is used in this work as a testing domain for our approach, where a team of carriers and fighters cooperate to kill wumpuses and collect gold.

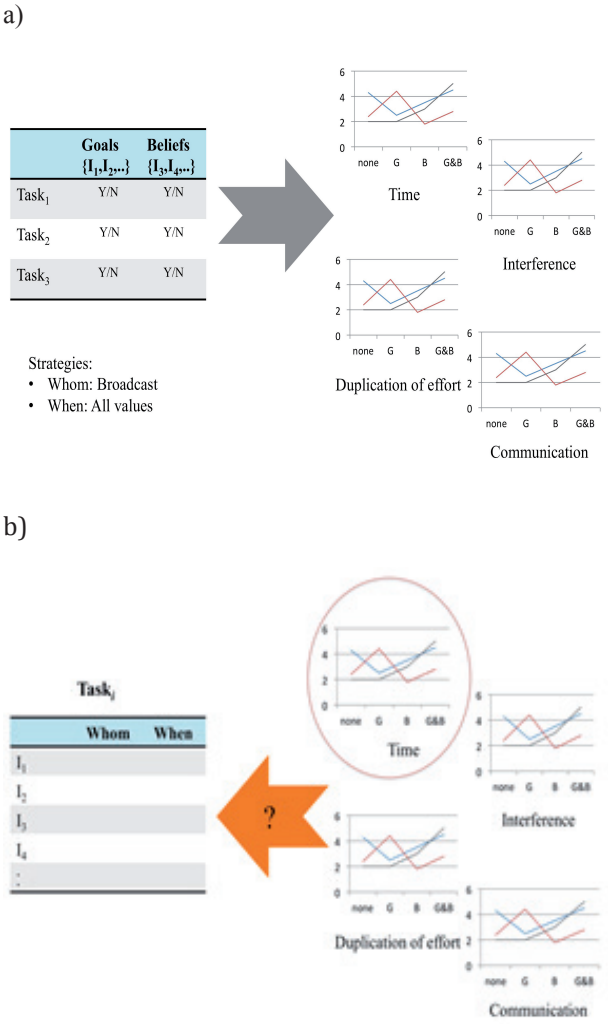

Fig. 1. Reversal of the (a) communication-to-performance investigation process to obtain (b) performance-to-communication learning process

The remainder of this paper is organized as follows. Section 2 briefly overviews related work. In section 3, we analyze the problem of designing a goal-oriented communication strategy for a multi-agent system, and explain why it is a challenge to design one manually. The Wumpus World, our test domain, is explained in section 4. In section 5, we provide detailed information on how we utilize a genetic algorithm to design a learning system that automatically generates an effective goal-oriented communication strategy. We show, in section 6, promising results in preliminary experiments with two performance goals using the Wumpus World domain, and we finally conclude in section 7 and discuss future work.

2. Related Work

Given the impact of using an effective communication on MAS performance, it is not surprising that approaches for learning all aspects of communication, including language, protocols, and strategies, have been proposed in the literature. We focus in this section on existing work related to communication strategies, which help agents decide what, when, and to whom they communicate. The main contribution of this work is not only learning a communication strategy, but also learning communication strategies with respect to flexible goals. We refer to goals as the criteria or metrics used in the communication decision process to determine whether a specific communication act should take place. A rule of thumb is that information should be communicated only if doing so is beneficial to the system's performance, [2].

Different metrics have been used in the literature to value communication decisions of agents. Examples include performance-based metrics such as minimum communication cost [10, 22], task progress [20], minimum time [8, 9], and avoiding coordination errors [14]. Also, information-based metrics have been used such as timeliness and relevance [22], information redundancy [6], and KL divergence [20]. As stated above, the advantage of this work over existing ones is that it is designed with no assumptions about the desired system's performance. Therefore, system designers have the ability to design their own metric, with which they would like to improve the system's performance.

Communication decisions in cooperative MAS can be either centralized, where a coordinator agent that has a full observation of the global state compute a central strategy, or decentralized, where each agent with local observation compute its own strategy. In the latter case, however, cooperative agents need access to their teammates' states and actions to be able to estimate the overall communication benefits, and hence make communication decisions. Therefore, researchers have designed different approaches to allow agents obtain such information. For example, [10] allowed agents to send feedback about the usefulness of the information they received to senders, [14] allowed agents to take actions based on shared information, and hence know the actions taken by the teammates, and [22] extended agents' observability to enable agents to track team members' mental states, and hence infer what teammates know and when. Based on the domain characteristics, some research efforts, such as [14, 15, 18, 20], proposed approaches based on modeling the team's decision problem using variations of Markov Decision Process (MDP), such as Dec-MDP and Dec-POMDP, to enable agents to estimate the impact of communication, and hence compute their communication policies. The computation, however, is usually based on myopic assumptions, where each agent evaluates the benefit of communication in isolation for 1-step horizon assuming others never communicate. The work in [2, 4] proposed approaches that relax these assumptions, yielding better performance. As mentioned previously, the work presented in this paper addresses this issue by adapt-

ing a centralized learning offline and distributed execution at runtime.

Existing work based on MDP framework have proposed approaches to compute a communication policy online [20], offline [8], or in a hybrid manner [5], as part of the solution to the problem, in a centralized [3], distributed, and round-robin fashion [12]. Other efforts have proposed approaches to learning communication strategies include [10], where authors proposed a distributed probabilistic control system that observes the information entering and leaving an agent and learns the *classes* of information and communication *frequency* that are beneficial to the agent and its *neighbors* using their feedback. Moreover, the work in [7] introduces a Hierarchical Reinforcement Learning (HRL) algorithm to allow agents learn a hierarchical communication and action policy, where the main task is decomposed into cooperative and non-cooperative tasks and a communication task is added under each cooperative task. Using the algorithm, agents can decide *when* to communicate with other agents to obtain their actions. Moreover, reinforcement learning is used in [11] to allow agents learn action policies, which are extended to include communication by adding linguistic state variables in receivers' state space and linguistic action variables to senders' action space. Furthermore, [9] used Genetic Programming (GP) to evolve agents' behavior, represented by parse trees, which include communication actions that enable an agent to request data from another.

In research efforts, some researchers applied heuristics to allow agents to reason on-line about when and what to communicate. Examples include using hill-climbing heuristic to find out what information maximizes the expected reward [15]. Another work in [22] used implicit communication to allow agents reason about teammates' needs and productions, and hence communicate proactively.

Most of the works, presented above, address only part of the communication decisions in MAS. For example, [2, 4, 5, 6, 7, 9, 12, 14, 20] address only when agents communicate, [15] addresses what agents communicate, and [8, 11, 18, 22] address what and when agents communicate. Similar to our work, [10] addresses what, when, and to whom agents communicate. However, authors provided agents with only one timing strategy, which is frequency of communication, equivalent to one of our timing strategy, EveryTimeInterval, which will be explained in a later section.

3. Goal-Oriented Communication Strategy

Results of the work in [19] imply that communicating each information instance that agents obtain during task execution affects the performance of the system with respect to multiple metrics. Therefore, depending on the desired performance goal, it may not be necessary to communicate all available information instances. If it is determined that one should be communicated, determining when (timing) and to whom (recipients) becomes essential as communication incurs cost.

Therefore, it is crucial to consider designing a communication strategy for each information instance. Moreover, if there are multiple types of agents in the system, a separate communication strategy should be designed for each information instance and a recipient type combination. To explain this, consider our test domain, the Wumpus World, where a team of carriers and fighters cooperates to collect gold and kill wumpuses, where carriers are responsible of collecting gold, and fighters kill wumpuses. In this domain, the "wumpus location" information instance means danger to carriers and a spot they should keep themselves away from, while, to fighters, this information means a target and a place that they should run to in order to kill the wumpus and help carriers collect gold in the same location. Moreover, some information instance might be important to only carriers or fighters. For example, "gold location" is only important to carriers. Consequently, the system's designer must keep in mind the properties of each information instance and how it can possibly affect the performance of each agent type and hence the overall performance.

A comprehensive communication strategy guides agents on all their communication needs: *what*, when, and *who*. Consider a domain where agents can obtain k Information inStances (IS).

$$IS = \{IS_i: 1 \leq i \leq k\}$$

In this work, we formulate the communication decisions problem to learning the communication patterns that improve the system performance with respect to flexible, user-defined goals. It is assumed that for each Information instance (IS_i), there are n alternatives for *when* it can be communicated, and m alternatives for *who* it can be communicated to. Therefore, the total number of Communication Strategies (CommSt) for IS_i is computed by:

$$\text{CommSt}\,(IS_i) = (n*m) + 1 \quad (1)$$

The number of alternatives for the *when* component multiplied by the number of alternatives for the *who* component, plus the option of not communicating the information instance. Hence, for k information instances, and c types of agents, the total number of possible communication strategies can be computed as:

$$\text{TotalCommSt} = (n*m+1)^{k*c} \quad (2)$$

The total number of possible communication strategies increases exponentially with the number of information instances k and number of agents' types c. Besides, due to the randomness in multi-agent systems, it is usually not clear upfront which communication strategy will be effective with respect to the task and performance goal [1, 12], and it is difficult and time-consuming to manually try all possible communication strategies. Yet, combinations of different sub-strategies for each information instance may result in unexpected performance. This calls for an automated approach for determining an effective communication strategy with respect to a stated performance goal, which is proposed in this paper. We as-

sume that a communication language already exists, and communication is always reliable.

4. Application Domain

We have designed and built a multi-agent version of the Wumpus World Problem [16, 22] to use as our test domain. This is done using the Repast Symphony plugin for Eclipse [13], which is an open source Java-based agents modeling and simulation toolkit. We designed the world of our domain to contain several rooms that contain gold and wumpuses. This new feature in the domain allows having multiple different beliefs and goals that provide variety of information instances, which adds a key property that makes it effective for evaluating communication strategies. The considered information instances along with their values are listed in Table 1, and will be explained throughout this section.

Our world consists of 12 rooms in a 140x140 grid. One room is the drop-off room, where agents have to drop any gold that they collect, and others either contain gold and/or wumpuses or are empty. Figure 2 shows a screenshot of our world. The long-term goal of the team, carriers and fighters, is to kill wumpuses and collect gold. In our experiments, the world contains 5 carriers, 3 fighters, 5 wumpuses, and 10 pieces of gold, all of which distributed randomly at the beginning of each simulation. Carriers have the information about rooms' locations, but not their contents. Carriers are capable of finding gold and wumpuses, picking up and dropping off gold, while fighters are capable of shooting wumpuses. Similar to [22], the only way that fighters can know about the location of a wumpus is by getting a message from a carrier that observed it (Wumpus location IS_4). When a wumpus is killed, carriers who observed it can determine that the wumpus is dead and the room is safe only by getting a message (Safe room IS_8) from the fighter who killed it.

In order to facilitate the achievement of the long-term goal and enhance collaboration, each agent also has a short-term goal. Carriers' and fighters' goals are listed in Table 1. Initially, all carriers have the "ExploreRoom" goal, and have to decide on a goal room. Each carrier can only hold one piece of gold at a time. Therefore, once a carrier picks up gold, its goal will be "DropOffGold" as it must go to the drop-off room and drop the gold there.

Carriers and fighters states, S_c and S_f, respectively, are defined as follows:

$$S_c = \{(s_1, s_2)\}$$

$$S_f = \{s_1\}, \text{ Where: } s_1, s_2 = \{0, 1, \phi\}.$$

The first element s_1 indicates agent's location, whether inside a room (1), in the hallway (0), or disregarded (ϕ). Although agents can recognize what room that they are in, this is abstracted in the state to only three cases to deliver a coherent choice to when an agent can communicate, as we will explain in the timing strategies section. The second element s_2 indicates the carrier's possession of gold, whether it holds gold (1), nothing (0), or disregarded (ϕ).

Fig. 2. Screenshot of the Wumpus World

At each time-step, every agent performs (observe, communicate, act) execution cycle. In observe, agents are able to make the observations {IS_3, IS_4, IS_5, IS_8}, in Table 1, as well as deciding on a goal room or change goal. In the communicate step, agents refer to their communication strategy to see if they should communicate any information at the current time-step. In the act step, both agent types are able to move UP, DOWN, RIGHT, and LEFT. In addition, carriers are able to PICK UP, and DROP OFF gold, and fighters are able to SHOOT a wumpus.

Moreover, each agent maintains a mental model about other agents by communicating goals and goal rooms, if allowed by the assigned communication strategy. The behavior of agents, both carriers and fighters, can be summarized as follows:

- A carrier will choose the closest room that contains gold. If it has no information about gold locations, it will choose the closest room it has never visited to explore.
- When a carrier decides to go to a room, which it knows nothing about the content, the carrier must explore the room.
- A carrier will not explore a room that is currently being explored by another carrier, unless it has no other options.
- If a carrier knows that a room contains gold, it will not need to explore the room; instead the carrier will target the gold and collect it.
- A carrier will avoid empty room.
- A carrier will escape the room once it observes a wumpus.
- A carrier will avoid wumpus rooms unless it receives information that the wumpus is killed, or a carrier tells the agent that it contains gold, which implies that the sender was able to explore the room, and hence the wumpus is killed.
- A carrier will choose a different goal room if it receives information that its goal room is empty/has a wumpus while the agent is moving to it.
- Fighters do not move unless they receive information about a wumpus location.
- If a fighter receives multiple requests, it will choose the closest one.
- A fighter will choose a different wumpus/stay still if it receives information that another fighter has killed the wumpus that the agent is going to kill.

5. Algorithmic Approach

We use a steady state Genetic Algorithm (GA) to evolve goal-oriented communication strategies in a multi-agent system. GA is a model of machine learning that simulates the natural phenomenon of evolution. Starting with a number of random solutions for the problem, GA mimics natural selection by favoring fit solutions in selecting parents for producing offsprings. Once parents are selected, crossover and mutation operations are applied and new offsprings are placed in population to form the next generation. This process continues until a termination condition is met. Although GA is not guaranteed to find optimal solutions, it has been successfully applied to problems in different application domains to find near-optimal solutions.

Information Instance	Possible Values	Producer
Goal (IS_1)	"ExploreRoom", "PickUpGold", "DropOffGold"	Carrier
Goal Room (IS_2)	[1-12]	Carrier
Gold Location (IS_3)	(x,y), 140 > x,y > 0	Carrier
Wumpus Location (IS_4)	(x,y), 140 > x,y > 0	Carrier
Empty Room (IS_5)	[2-12] (1 is drop-off room)	Carrier
Goal (IS_6)	"LookForWumpus", "KillWumpus"	Fighter
Goal Room (IS_7)	[2-12] (1 is drop-off room)	Fighter
Safe Room (IS_8)	[2-12] (1 is drop-off room)	Fighter

Table 1. Information instances of the Wumpus World and their values

Token	Values	Notes
IS_i	[1, *k*]	*k*: number of information instances in the domain.
RT	[1, *c*]	*c:* number of agents' types in the domain.
R	P2P (1), Subset [2, (*n*-1)], Broadcast (*n*)	*n*: total number of agents.
T	EveryUpdate (1), EveryTimeInterval [2, *i*], InState [(*i*+1), (*i*+*s*)]	*i*: maximum time interval. *s*: number of possible agents' states.

Table 2. Possible values for each token in a cell/communication strategy

The key elements required to apply GA to a problem are to seek a suitable coding for the solution individual (chromosome), to design a fitness function to evaluate solution candidates, and to determine a termination condition. Other parameters such as crossover rate, mutation rate, selection approach, and replacement approach need to be tuned to improve the performance of the algorithm.

The proposed evolutionary approach determines a communication strategy for each information instance, whether goal or observation, that could be obtained by any agent during task execution. Hence, the evolved strategy will consist of a set of sub-strategies for the communicated information instances.

5.1. Solution Representation

Each individual in GA population represents a solution candidate, and in our case, a goal-oriented communication strategy. An individual is made of a vector of cells. We define a cell to be a 4-tuple vector: (IS_i, *RT*, *R*, *T*). A cell represents a communication strategy for one information instance. It indicates that an information instance IS_i should be communicated to agents *R* of type *RT* only at time-steps that conform to *T*. The possible values that each token can take are listed in Table 2 and explained below. Individuals in a GA population have different lengths because the number of cells in an individual may differ. This is due to the fact that an individual contains only cells corresponding to communication strategies for information instances that should be communicated. If an information instance should not be communicated, it will not be included in the evolved individual. This flexibility allows learning *what* information instances should be communicated, and hence keeps the communication cost minimum by evolving a strategy that communicates only information instances that contribute to the performance goal. An example of GA individual, i.e. goal-oriented communication strategy is shown in Figure 3.

5.1.1 Recipients Strategies

The recipients type *RT* and recipients *R* in a cell define *who* an information instance *IS* should be sent to. The former defines, as the name indicates, the receiving agents type, which in our case takes two values; either carriers or fighters, and the later determines the number of recipients of type *RT*, which can be (1) Peer-to-Peer, (2) Broadcast, or (3) Subset.

(2,2,3,11) (5,2,1,11) (3,1,2,17) (4,2,3,6) (2,1,2,10)

Figure 3: An example of a GA individual (goal-oriented communication strategy)

The first strategy, Peer-to-Peer (P2P), allows sending the information to only the closest agent of type RT to the sender, and Broadcast allows sending the information to all agents of type RT. However, Subset allows sending the information to the closest *m* agents of type RT, where *m* is the number of recipients, and whose value is evolved by GA (see Figure 4). As explained in a previous section, it is important to determine whether an information instance should be communicated to both carriers and fighters, or only one type. If one information instance (IS_i) is communicated to both types, there will be two cells for IS_i in an individual, each corresponding to one type of recipients. So with this solution representation, GA will be able to evolve a communication strategy that allows communicating information instances to only

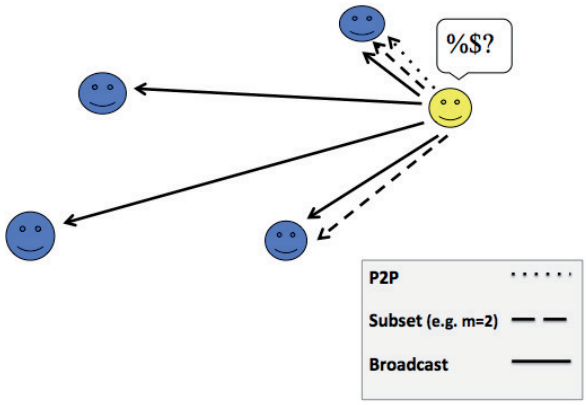

Figure 4: Recipients strategies

those agents that make use of it. Individuals that contain two cells for the same information instance and same recipients' type need to go through correction for duplication elimination, which will be explained in details in a later section.

5.1.2 Timing Strategies

The last token in a GA individual cell is *T*, which corresponds to time or when an information instance should be communicated. It determines the time-steps at which an agent is allowed to communicate the information instance. In this work, we consider three strategies for the '*when*' component. Agents are able to communicate an information instance (1) every update, (2) every time interval, or (3) in a state. Graphical illustration of all timing strategies is shown in Figure 5, which we will refer to frequently in this section. The gray arrow represents the time line of one agent, and blue and red dots represent the time-steps at which communication is allowed or not, respectively.

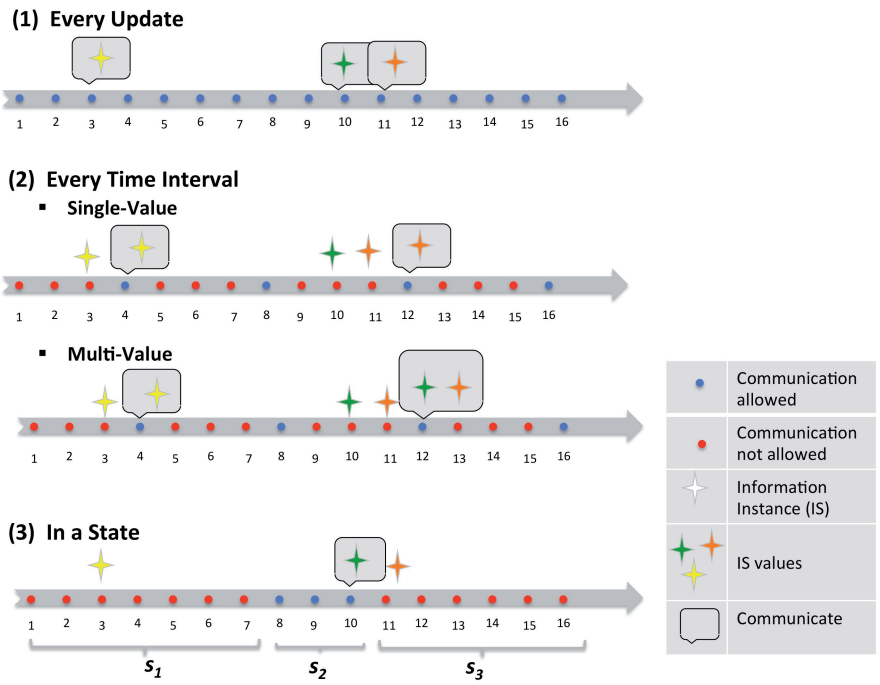

Figure 5: Timing strategies

Before we explain these time strategies, we need to distinguish between two types of information instances; (1) *single-value* and (2) *multi-value* information instances. The single-value instances are those that, at any time-step, an agent is allowed to have only one value of them, while the multi-value, an agent can possess multiple values of the same information instance.

Examples of the single-value are 'goal' and 'goal room', since an agent can have only one goal and one goal room at any point of time. While "wumpus location" and "gold location" are considered multi-value since an agent may have a list of all locations where it observed wumpuses or gold.

As shown in Figure 5, with the first timing strategy, every update, agents are allowed to communicate at all time-steps (all blue dots). Therefore, agents communicate value updates at the same time-step that they obtain them, and hence communicate every value update. If the information instance is single-value, an agent will communicate the information instance every time-step the agent updates it with a different value. If the information instance is multi-value, then whenever a new value is added to the values of the information instance, the agent will communicate only this new value. This strategy is idle for information instances that need immediate response or reaction from others or when lateness is not tolerated.

With the second strategy, every time interval, agents have to wait a specific period of time before they are allowed to communicate a value of the information instance. For this time strategy, GA evolves how often agents are allowed to communicate. For instance, if T=3, agents are allowed to communicate this information every 3 time-steps. If the information instance is single-value, an agent will check every 3 time-steps to see if the information instance's value has been updated since the last time the agent communicated it. If an agent updates its value more than once since the last communication only the last update will be communicated, as shown in Figure 5, green value is not communicated but orange value is communicated. However, if the information instance is multi-value, an agent will check every 3 time-steps to see if it has obtained and added any new value to the values list. All new added values for this information instance will be communicated, and hence both green and orange values are communicated in Figure 5. This strategy is more energy efficient than the previous one because it allows agents to combine multiple values in one message. The possible

time interval values are determined by the system designer. In this work, we allow time intervals in the range [2, 10].

In the last time strategy, in a state, agents are allowed to communicate only values of an information instance that are obtained when the agent is in a specific state *s,* whose value is evolved by GA. For instance, if an agent is only allowed to communicate values obtained in state s_2, *i.e.* in the 8th, 9th, and 10th time-steps, as shown in Figure 5, only the green value is communicated because it is obtained in the 10th time-step. This strategy allows agents to communicate only specific values of the information instance. For example, if carriers are allowed to communicate the 'goal' information instance only if they are in state S_c=(ϕ, 0), which means an agent is not holding gold, and location is disregarded, then agents will only communicate their goal values 'ExploreRoom' and 'PickUpGold', because agents will not have the goal 'DropOffGold' if they hold no gold. This strategy is idle when agents are interested/can make use of only some values of the information instance. For instance, in work by [1], a robot performing "forage" or "graze" tasks is only interested to know if another robot is in "acquire" or "graze" state, respectively. Robots in these states have found useful work, and hence if other robots follow it, they will potentially find useful work too. For such information, it may be beneficial that every agent shares its state only when it is in "acquire" or "graze" state.

5.2. Fitness Function

Since the fitness function measures the overall performance of the system with respect to a specific goal, given a communication strategy, the fitness function will be the average performance of multiple runs of the system using the current strategy. The performance goal can vary according to the system's designer preference. Examples are time to complete, energy cost, and number of moves. Technically, any performance metric that can be objectively measured can be considered as the fitness function. In addition, the fitness function can be the combination of two or more performance metrics, weighted to reflect the designer's interest or priorities. For instance, if the designer is interested in minimizing both the number of messages sent and time to complete, then the fitness function is:

$$MsgTimeFit=(\alpha*time)+(\beta*MsgSent)$$

Where,
α: is weight for time metric.
β: is weight for number of messages metric, where $\beta=1-\alpha$.

The values of the two parameters can be adjusted according to the importance of the associated metrics. For instance, if the two metrics are of equal priority, α can be assigned the value of 0.5. However, if time is more important, then more weight can be added to α such as (α, β)=(0.6, 0.4) or (α, β)=(0.7, 0.3) based on the designer's goals.

5.3. Genetic Operators

5.3.1. Crossover

Since each solution candidate consists of multiple cells corresponding to sub-strategies for different information instances, a special crossover operator that swaps a subset of cells between two parents is needed. Allowing crossover point to only take place between cells and never divide a cell can enforce this (see Figure 6). This special crossover operator has been used previously in another work [21], where a cells-like GA individual representation has been used. This has the advantage of producing valid offsprings, as well as preserving GA's property of maintaining and using successful cells, a.k.a., sub-solutions or sub-strategies, found in previous generations as a building block to discover new valid solutions. In this work, we use one-point crossover, where the crossover point is randomly chosen for each parent. Figure 6 shows an example of two parents going through crossover.

5.3.2. Mutation

When two offsprings are produced after crossover, they go through mutation. While crossover helps GA exploit the good solutions found so far, mutation makes sure that GA explores new solutions. In this work, mutation rate is the probability that one element of a cell will be changed.

5.3.3. Selection

Selection in GA corresponds to the criteria to select two parents from the population to produce new offsprings. In this paper, *roulette wheel selection* is adopted as our selection method. *It is a fitness proportionate selection method, in which fitted individuals are more likely to be chosen to produce new offsprings, since they have more potential to produce highly fitted individuals.*

5.3.4. Replacement

When new offsprings are produced, a good replacement strategy must be used to choose which individuals in the population are eliminated to make spots for the new individuals. A trivial strategy could be eliminating the least fit individuals in the population. However, this strategy will quickly expel diversity out of the population.

Parents Before Crossover:
Parent 1:
(6,2,2,7) (6,1,1,10) (4,1,3,7) (2,2,3,10) (5,1,1,12)
Parent 2:
(3,2,2,15) (5,1,2,1) (4,2,3,8)
Random Crossover Point for Parent 1 is 3
Random Crossover Point for Parent 2 is 2
Parent 1:
(6,2,2,7) (6,1,1,10) (4,1,3,7) | (2,2,3,10) (5,1,1,12)
Parent 2:
(3,2,2,15) (5,1,2,1) | (4,2,3,8)
Result Off-springs:
Offspring 1:
(6,2,2,7) (6,1,1,10) (4,1,3,7) | (4,2,3,8)
Offspring 2:
(3,2,2,15) **(5,1,2,1)** | (2,2,3,10) **(5,1,1,12)**
After Correction (randomly select a redundant cell for deletion):
Offspring 1:
(6,2,2,7) (6,1,1,10) (4,1,3,7) (4,2,3,8)
Offspring 2:
(3,2,2,15) (5,1,2,1) (2,2,3,10)

Figure 6: Crossover and correction operations

In this work, we follow [17] in assuming that parents are the most similar individuals to the new offsprings; and therefore, the four individuals, two parents and two offsprings, compete for insertion in the population. The best two out of the four individuals are inserted in the population in order to form the new generation.

5.3.5. Correction

When subsets of cells are exchanged during crossover to form new offsprings, it is likely that the new offspring could contain duplicated cells. The duplication does not necessarily mean that the two cells are exactly the same, but two cells define communication strategies for the same information instance and to the same recipients type, i.e., the first two elements of the cells are identical, such as; (2,1,5,3) and (2,1,3,1). If this occurs after crossover and mutation, one of the duplicated cells will be chosen randomly and deleted (see Figure 6).

6. Preliminary Experiments

In this section, we report results of preliminary testing of the proposed approach with simple goals. We evolved communication strategies with respect to two different performance goals/fitness functions: (1) total energy consumed, and (2) time to complete.

In both fitness functions, individuals with lower fitness value are better, i.e. considered to be fitted. Carriers consume energy when they move, pick up gold, drop off gold, and communicate. Similarly, fighters consume energy when they move, shoot wumpus, and communicate. Depending on the application domain, each action can consume different amount of energy. In this work, we seek unbiased learning in which no preference is made between action and communication. Therefore, all actions including communication consume the same amount of energy. The fitness function of first performance goal is the total energy consumed by all agents, i.e., carriers and fighters, to finish the task. The second fitness function is the total number of time-steps required to complete the task. We chose the time to complete fitness function because we want to compare the best time performance that GA converges to when no communication restriction applied, i.e. communication is free as fitness function includes only time, with what we believe to be the best time performance of the system with the maximum (or full) communication. In addition, we would like to compare the performance of the system with the two learned strategies with respect to different performance metrics, and most importantly time and energy, to understand how the two metrics are related.

GA Parameter	Value
Population Size	30
Selection method	Roulette Wheel Selection
Crossover type	Random one-point
Crossover rate	0.95
Mutation rate	0.015

Table 3. GA parameters

A number of GA parameters were assigned values (Table 3), which were empirically found to be effective. Our test domain was set to run until the agents finish the task, i.e., kill wumpuses and collect all the gold present in the environment, or when 2000 time-steps pass by, whichever occurs first. We observed that if the agents could not finish the task in 1400 time-steps, then they would not be able to finish it, and hence 2000 was set up as a maximum time.

Minimum amount of communication is required to enable agents to finish the task. Since the only way that fighters can kill wumpuses is to get information from carriers about the wumpuses' locations, a complete communication strategy has to include a communication strategy for the 'Wumpus Location' information instance with fighters as the recipients' type. No matter how good a communication strategy is with respect to the performance goal, without enabling agents to finish the task, the system will run for a long time and the strategy will receive a bad fitness score.

Therefore, in efforts to speed up GA convergence, we decided to provide carriers with a default communication strategy for communicating the "Wumpus Location" information instance to fighters. Hence, during learning, if the communication strategy being evaluated does not include a sub-strategy for communicating the "Wumpus Location" to fighters (referred to incomplete strategies), carriers will use their default strategy to enforce communicating this information instance, along with strategy being evaluated, in order to allow agents finish the task. We emphasize that strategies in GA population will not be altered or modified; rather their fitness score reflect their complete counterpart. Consequently, GA will be able to make use of good but incomplete strategies. During learning, if the communication strategy being evaluated is complete, that is it includes a sub-strategy for communicating the "Wumpus Location" to fighters, carriers will use the provided strategy for this information instance, rather than the default one. After multiple experiments, we determined that the communication strategy for the 'Wumpus Location' information instance that has minimum number of messages sent is 'communicate *every update* of the information instance to *fighters* as *P2P'*. Although the used *when* strategy, i.e. every update, sends each wumpus observation in a separate message, we found out that delaying communication will make more carriers exposed to the wumpuses, and hence more messages for the same wumpus will be sent to fighters.

Figure 7 shows the evolution of time-minimal and energy-minimal communication strategies for the wumpus world domain. The GA termination condition is when a predetermined number of generations, *g=100,* pass by without improvement in the average population fitness. Our intuition is that as long as GA is able to improve the average fitness of the population by performing crossover and mutation, it is likely that the so-far-best solution will be improved in a future generation.

The top line in the Figure shows the fitness value of the worst individual in the population of each gen-

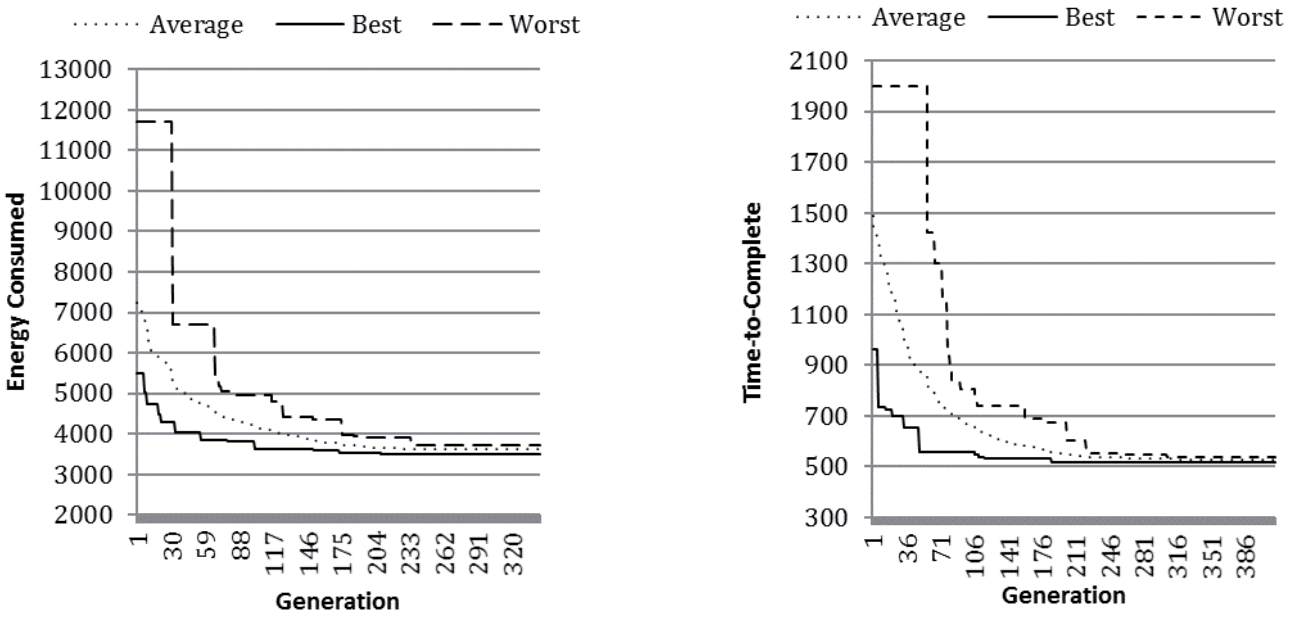

Figure 7. Evolution of (a) Energy-Minimal and (b) Time-Minimal communication strategies

eration. The lines in the middle and bottom show the average fitness value and the best fitness in the population of each generation, respectively.

A somewhat surprising observation in both figures is that although all communication strategies include the minimum required communication for finishing the task, agents were still unable to finish the task using some communication strategies. This can be seen in Figure 7.b as the worst individual has fitness value of 2000 in the first generation, which is the maximum time the simulation can run. After some investigation of such communication strategies, we learned that some combinations of communication sub-strategies could prevent agents from finishing the task.

An example of such strategies is when a carrier broadcasts the location of an observed wumpus to other carriers and communicates it to fighters using any strategy, but concurrently fighters do not communicate the news about killed wumpus. Therefore, the room that contained the wumpus will never be visited by carriers since they all were warned about it but never told about being a safe room.

Table 4 shows the evolved communication strategies. Each row corresponds to the communication strategy of one information instance. The column to the left lists all information instances that we consider, and for each we show the learned timing (*when)* and recipients (*who)* strategies for each recipient type of carriers and fighters, if applicable. Columns to the right correspond to the learned energy-minimal and time-minimal communication strategies.

It is clear that the time-minimal strategy communicates more information than the energy-minimal strategy. This is due to the fact that the fitness function for evolving the time-minimal strategy considers only the number of time-steps that the agents need to complete the task. Therefore, strategies that enable agents to complete the task faster are always favored no matter how much they cost. The time-minimal strategy communicates all information instances that the energy-minimal strategy communicates but to more receivers, as can be seen in IS_2 and IS_4.

The energy-minimal strategy communicates the carriers' goal room as P2P every five time-steps, but does not communicate carriers' goal at all. Since initially all carriers have the 'ExploreRoom' goal and each carrier knows that others have this goal in their mental model, not communicating any update of the goal will make each carrier believe that others always have 'ExploreRoom' goal. A carrier will not explore a room that is currently explored by another carrier, hence, with this communication strategy, a carrier will never explore a room that is currently explored or even visited by another carrier to pick up gold. It is surprising that GA has enforced such behavior that was not originally implemented in carriers. We believe that the reason for evolving such strategy is because the world has 12 rooms but only 10 pieces of gold; hence each room is more likely to have a small number of gold, usually one, or empty.

Bearing this mind, along with the fact that exploring a room takes time and energy, it is probably time and energy efficient to avoid exploring a room that others are exploring or picking gold from. More experiments with larger number of gold will be conducted in the future to observe if such behavior will still evolve.

As expected, the time-minimal communication strategy includes strategies for communicating some useless information instances to the recipients (shaded cells) such as sending locations of gold to fighters. Communicating such information only increases the number of messages sent, without affecting the recipients' behavior or the overall performance with respect to other metrics. Existence of such information instances in the evolved strategy is expected be-

cause it complies with the fitness function that only considers time. Therefore, two strategies that allow completing the task with minimum time will have the same fitness value, even if one can achieve this with less communication.

Figure 8 compares the performances of the multi-agent system with the two evolved communication strategies; MinTime and MinEnergy, as well as the minimum and maximum communication strategies, MinComm and MaxComm, respectively. This should give the big picture and enable us to see how the evolved strategies compare to the two communication extremes. As mentioned previously, minimum communication strategy allows only communicating *every update* of *the wumpus location* to *fighters* as *P2P,* which is the strategy with minimum number of messages that enables agents to complete the task, while maximum communication strategy *broadcasts every update* of *all* information instances to *both* agents' types. The following performance metrics are considered:

- **Energy:** Measures the total energy consumed by all agents to finish the task. This performance metric can be used if the user needs the task to be completed with minimum energy cost.
- **Time:** Measures the number of time-steps needed to complete the task. The task is determined completed when the last gold piece in the environment is picked up. This metric can be used if the user needs the task to be completed as fast as possible.
- **Moves:** Measures the number of moves of all agents until the task is complete. A move is defined as changing position. This metric can be used if the user needs the task to be completed with minimum travel cost/distance.
- **Messages:** Measures the number of messages exchanged until the task is completed.
- **Energy Variance:** Measures the variance of *n* numbers; where *n* is the number of agents in the world. Each number represents the amount of energy consumed by one agent. This performance metric can be used if the user needs the task to be completed with minimum energy variance. The lower the variance the closer the amount of energy consumed across all agents.
- **Load Variance:** Load is the amount of work each agent has completed. In Multi-agent systems, it is important to distribute the load evenly among agents with no overloaded/idle agents. This performance metric can be used if the user needs the task to be completed with maximum load balance/minimum load variance. The lower the variance, the more balanced the load is across the agents.
  - **Carriers Load Variance:** Measures the variance of *n* numbers; where *n* is the number of carriers in the domain. Each number represents the amount of gold picked up by one carrier.
  - **Fighters Load Variance:** Measures the variance of *m* numbers, where *m* is the number of fighters in the domain. Each number represents the number of wumpuses killed by one fighter.
- **Work Duplication:** Work duplication occurs when two agents target the same goal at the same time.

Information (*what*)	Energy-Minimal Communication Strategy		Time-Minimal Communication Strategy	
	Time (*when*)	**Recipients** (*who*)	**Time** (*when*)	**Recipients** (*who*)
Carrier's goal (IS_1)	/	/	Every update	Carriers (Subset of 2)
Carrier's goal room (IS_2)	Every 5 TS	Carriers (P2P)	Every 10 TS	Carriers (Subset of 3)
Gold location (IS_3)	/	/	Agent is inside a room and holds gold.	Carriers (Subset of 3)
			Every 5 TS	Fighters (Broadcast)
Wumpus location (IS_4)	Every update	Fighters (P2P)	Every 10 TS	Fighters (Subset of 2)
Empty room (IS_5)	Agent holds no gold	Carriers (broadcast)	Every 3 TS	Carriers (broadcast)
			Every 7 TS	Fighters (P2P)
Fighter's goal (IS_6)	/	/	Every 5 TS	Carriers (P2P)
			In hallway	Fighters (Subset of 2)
Fighter's goal room (IS_7)	/	/	Every 3 TS	Carriers (Subset of 2)
			Every 4 TS	Fighters (Subset of 2)
Safe room (IS_8)	/	/	Every 7 TS	Carriers (Subset of 4)

Table 4. Evolved goal-oriented communication strategies

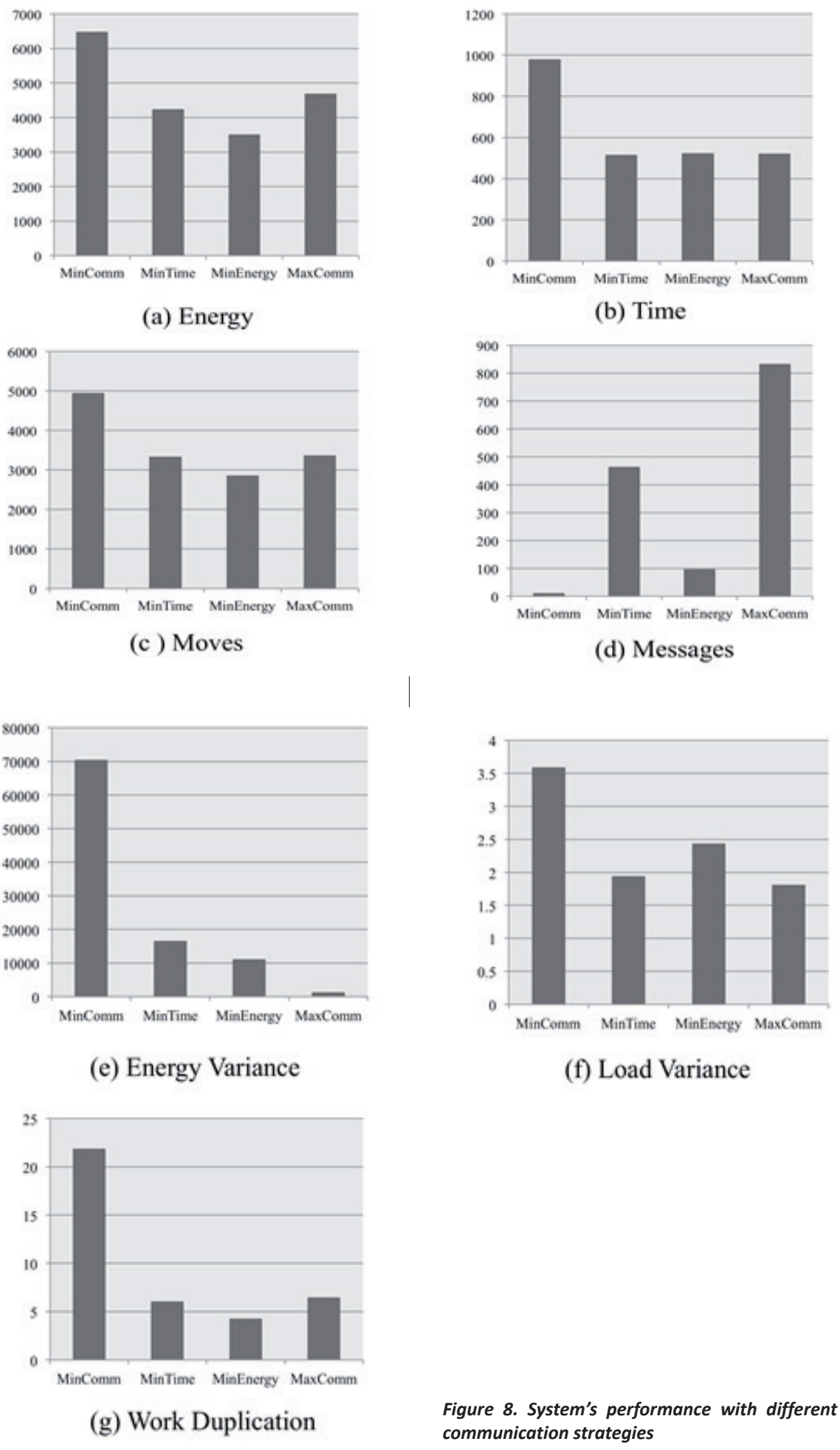

Figure 8. System's performance with different communication strategies

- **Carrier:** Number of times two or more carriers explored a room at the same time and number of times a carrier fails to pick up gold because someone else has already collected it.
- **Fighters:** Number of times two or more fighters target the same wumpus.

With maximum communication, agents have a full observation of the world, which allows them to cooperate maximally and make the best decisions. This behavior is reflected in minimum time, energy variance, and load variance, as shown in Figure 8, comparing to other strategies. However, the system did not perform the best with respect to energy, moves, and work duplication, which confirms the conclusion of previous studies that more communication does not guarantee better performance [19].

Energy-minimal communication strategy consumes the minimum amount of energy, compared to other strategies, followed by time-minimal strategy, and then maximum communication strategy, while minimum communication strategy has the highest energy consumption. With only minimum communication, agents cooperate minimally, and hence each agent depends only on its own view of the world, which results in significantly longer time to complete the task, large number of moves, and high work duplication, energy variance, and load variance as can be seen in Figure 8. Comparing to other strategies, the system has the worst performance with the minimum communication strategy, with respect to all metrics except messages.

It is not surprising that the energy-minimal and time-minimal strategies have the same performance with respect to time (Figure 8.b). When GA evolves for minimum energy, it implicitly evolves for minimum time since more time always means more energy consumption, but the opposite is not always true, thus the time-minimal strategy consumes more energy (Figure 8.a) due to more moves (Figure 8.c) and communication (Figure 8.d).

Both time-minimal and energy-minimal strategies have the same performance with respect to time as the maximum communication strategy, which makes us believe that the GA was able to find the minimum time to complete the task, and hence optimal time-minimal strategy. Moreover, the energy-minimal strategy has the second lowest number of messages after the minimum communication strategy. Yet, it improves the system performance significantly over minimum communication strategy with respect to all other metrics, including energy. In this case, GA was able to evolve a strategy that communicates only the right amount of important information that significantly contributes to improving the performance goal.

However, the energy-minimal strategy does not offer the best performance with respect to energy variance and load variance, although it has the next best energy variance after maximum communication strategy. We believe that this is due to the fact that these two metrics conflicts with the goal (i.e. energy metric) since they require more messages to be communicated. For example, the energy-minimal strategy does not allow agents to communicate information about the state of the world (i.e. gold and safe rooms).

Agents with both maximum and time-minimal communication strategies share their goal and goal room, which helps carriers not to explore a room currently explored by another carrier, hence lower work duplication than minimum communication. However, agents with energy-minimal strategy share only their goal room, which makes agents not to explore a room currently explored or visited by another carrier to pick up gold. Therefore agents have the lowest work duplication amongst all strategies.

Furthermore, since the time-minimal strategy includes communicating more information about the state of the world (gold and safe rooms) over energy-minimal strategy. This results in the agents sharing the same belief about the state of the world, which provides all the agents the same opportunity to collect gold, as can be seen in the low load variance (almost as low as maximum communication).

7. Conclusion

We proposed a GA-based approach for learning an effective goal-oriented communication strategy in multi-agent systems. The usefulness of the proposed approach lies in the fact that it removes a significant designing load from system designers since they do not need prior knowledge about the connection between communicating an information instance and the system performance. The learned communication strategy is a comprehensive one that determines what, when, and to whom agents communicate.

We ran preliminary experiments with two performance goals; namely, total energy consumed and time to complete. The results obtained are quite promising and satisfactory, and indicate that the proposed approach has great potential. We observed that, on one hand, evolving an energy-minimal communication strategy implicitly minimizes the number of messages sent, time to complete, number of moves, and work duplication, because they all cost energy and any increase in them increases total energy. On the other hand, evolving time-minimal strategy allows communicating more information, which enhances cooperation that is reflected as low moves, load variance, and work duplication as the maximum communication strategy.

There are numerous experiments that we are interested in running with variations of the performance goal and task's parameters. Next, we aim to examine our approach with other fitness functions and varying goals' weights. Furthermore, there are multiple parameters that we wish to tune and study their effects on the evolved communication strategies, and hence the system's performance. Most of these parameters are domain-dependent, as they meant to create variations of the same problem/task. Broadly speaking, we are interested in studying three types of parameters; action and communication costs, agents population, and task complexity.

ACKNOWLEDGEMENT

Althnian would like to acknowledge King Saud University for the scholarship support.

AUTHORS

Alhanoof Althnian * – Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, Kansas, USA.
E-mail: alhanoof@ku.edu
Arvin Agah – Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, Kansas, USA.
E-mail: agah@ku.edu

* Corresponding author

REFERENCES

[1] Balch T., Arkin R. C. "Communication in Reactive Multiagent Robotic Systems", *Autonomous Robots*, Volume 1, 1994, 27-52. DOI: 10.1007/BF00735341.

[2] Becker Raphen, et al., "Analyzing myopic approaches for multi-agent communication", *Computational Intelligence*, vol. 25, no. 1, 2009, 31-50. DOI: 10.1111/j.1467-8640.2008.01329.x.

[3] Boutilier C., "Sequential optimality and coordination in multiagent systems ", *IJCAI*, vol. 99, 1999.

[4] Carlin A., Zilberstein S., "Myopic and non-myopic communication under partial observability", *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, vol. 2, IET, 2009. DOI: 10.1109/WI-IAT.2009.174.

[5] Chakraborty D., Sen S., "Computing effective communication policies in multiagent systems". In: *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, 2007, 36. DOI: 10.1145/1329125.1329168.

[6] Dutta P. S., Goldman C. V., Jennings N.R. "Communicating effectively in resource-constrained multi-agent systems". In: *IJCAI,* 2007, 1269–1274.

[7] Ghavamzadeh M., Mahadevan S., "Learning to communicate and act using hierarchical reinforcement learning". In: *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems,* vol. 3, 2004, 1114–1121.

[8] Goldman C. V., Zilberstein S., "Optimizing information exchange in cooperative multi-agent systems". In: *Proceedings of the second international joint conference on Autonomous agents and multiagent systems,* 2003, 137–144. DOI: 10.1145/860575.860598.

[9] Iba H., Nozoe T., Ueda, K., "Evolving communicating agents based on genetic programming". In: *IEEE International Conference on Evolutionary Computation*, 1997, 297-302. DOI: 10.1109/ICEC.1997.592321.

[10] Kinney M., Tsatsoulis C., "Learning communication strategies in multiagent systems", *Applied intelligence*, vol. 9, no. 1, 1998, 71–91. DOI: 10.1023/A:1008251315338.

[11] Mazurowski Maciej A., Jacek M. Zurada, "Emergence of communication in multi-agent systems using reinforcement learning". In: *IEEE International Conference on Computational Cybernetics*, 2006. DOI: 10.1109/ICCCYB.2006.305696.

[12] Melo F.S., Spaan M.T., "A POMDP-based Model for Optimizing Communication in Multiagent Systems". In: *Proc. 1st Eur. Workshop on Multiagent Systems*, 2011.

[13] North M.J., T.R. Howe, N.T. Collier, Vos R.J., "The Repast Simphony Runtime System". In: *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, 2005.

[14] Roth, M., Simmons, R., & Veloso, M. "Reasoning about joint beliefs for execution-time communication decisions". In *Proceedings of the fourth international joint conference on Autonomous agents and multi-agent systems, 2005,* pp. 786-793. DOI: 10.1145/1082473.1082593.

[15] Roth M., Simmons R., Veloso M., "What to communicate? Execution-time decision in multi-agent POMDPs". In: *Distributed Autonomous Robotic Systems 7*, 2006, 177–186. DOI: 10.1007/4-431-35881-1_18.

[16] Russell S., Norvig P., *Artificial Intelligence: A modern approach*, Prentice-Hall: Egnlewood Cliffs, 1995.

[17] Thierens D., "Selection schemes, elitist recombination, and selection intensity". In: *Proceedings of the 7th International Conference on Genetic Algorithms, 1997,* 152–159.

[18] Tian Le, Jian Luo, Zhili Huang. "Communication based on Interactive Dynamic Influence Diagrams in cooperative multi-agent systems". In: *Computer Science & Education (ICCSE), 2013 8th International Conference on*. IEEE, 2013.

[19] Wei C., Hindricks K., Jonker C. M., "The Role of Communication in Coordination Protocols for Cooperative Robot Teams". In: *International Conference on Agents and Artificial Intelligence*, 2014. DOI: 10.5220/0004758700280039.

[20] Williamson S., Gerding E., Jennings N., *A principled information valuation for communications during multi-agent coordination*, 2008, 137–151.

[21] Wu A. S., Yu H., Jin S., Lin K. C., Schiavone G., "An incremental genetic algorithm approach to multiprocessor scheduling", *IEEE Transactions on Parallel and Distributed Systems,* vol. 15, no. 9, 2004, 824-834. DOI: 10.1109/TPDS.2004.38.

[22] Zhang Y., "Observant and Proactive Communication in Multi-Agent Teamwork". In: *IEEE/WIC/ACM International Conference on Intelligent Agent Technology,* 2006, 460-466. DOI: 10.1109/IAT.2006.97.