Taxonomies of Multirobot Task and Reward

Tucker Balch
Carnegie Mellon University

Abstract

In most research involving the use of reinforcement learning techniques for training robots, task and reward are tightly coupled: the task for the learning agent is to maximize its reward. There are many cases however, especially in multirobot systems, where the task and reward should be treated separately. Autonomous agents embedded in their environment are not always able to accurately access their performance, and overall performance may also depend on other agents over which the learner has no direct control. The taxonomies of task and reward presented here provide a framework for investigating the impact of differences in the performance metric and reward on multirobot system performance.

Background and Related Work

Robotic agents programmed using reinforcement learning (RL) techniques strive to maximize the reward signal they receive (Kaelbling, Littman, & Moore 1996). Because of this, performance is likely to be maximized when the reward closely parallels the metric used to evaluate them. However, in many embedded autonomous agent applications, especially in multirobot systems, it may not be possible or desirable for agents to be rewarded strictly according to system performance. Some examples include:

- The robot's sensors do not provide enough information for it to accurately evaluate its performance.
- The delay in receiving a reward is too great; learning a sequential task is too difficult and/or takes too long.
- Performance depends on the actions of other agents over which the learning agent has limited knowledge or control.

As a result, the performance metric (task) and reward function are often quite different and should be treated separately. What impact will a disparity between task and reward have on the performance of learning teams of robots? Before we can address this issue we must first identify the various ways tasks and rewards might differ; that is the goal of this work.

In related work, Balch has shown that, depending on the task, similar reward functions can lead to significantly different levels of performance and learning rates for multirobot teams (he investigated robot soccer, multirobot foraging and cooperative movement) (Balch 1998). In this type of work task characterization helps answer questions regarding how the same type of reinforcement can lead to different levels of performance and and learning rates in different tasks. Without answering "how are soccer and foraging different?" for instance, we can't answer "why do foraging agents learn more quickly than soccer agents?"

Balch's work also shows that performance and diversity in a learning team depend on the form of reinforcement used to train the robots. A taxonomy of multirobot reward enables us to distinguish between the various reward functions and identify which features enhance or hinder performance. There are tradeoffs to consider in the selection of a reward function. For instance some functions provide quicker learning but slightly poorer performance. Without a principled characterization of the differences, however, it is impossible for a robot systems designer to consider such tradeoffs intelligently.

In other work closely related to this research, Dudek *et al* introduce a taxonomy of multiagent systems (Dudek, Jenkin, & Wilkes 1993). They developed a set of axes for discriminating between the many types of multiagent robot systems. The axes include: size of the robot team, range of communication, communication topology, communication bandwidth, processing capability of the agents and the composition (heterogeneous or homogeneous) of the team. This work utilizes a similar approach by identifying relevant axes in each taxonomy, but with a different focus. Instead of describing features of the robot team, we are interested in features of the task the team must accomplish.

Ali has developed a taxonomy of of robotic movement tasks in his investigation of human control of multirobot teams (Ali 1997). His taxonomy enables us to distinguish between various movement tasks, but it does not address other task domains. In some tasks, for instance the movement of objects rather than robots might be important. Ali's terms describing the various types of movement for robotic teams are incorporated in the task taxonomy presented below.

The author is not aware of any comprehensive taxonomy of task or reward for multirobot teams. This work is an effort to address the need for these descriptive systems. The following sections outline the proposed taxonomies and illustrate their use in categorizing several example systems.

Multirobot Tasks

The RL literature considers the task and the performance metric as one and the same. That view is adopted here; a robot team's mission is to maximize performance over time. But there are other components of a task as well. In addition to the performance metric, a task is further defined by the environment and robotic platform. Because this work focuses on multirobot systems we are especially interested in a classification that will distinguish between different types of multiagent tasks.

The approach is to examine the performance metric, task environment and robotic platform for constraints and features that distinguish various tasks from one another. In a manner similar to Dudek's taxonomy of multirobot systems (Dudek, Jenkin, & Wilkes 1993), a set of descriptive tags that identify salient features of a task are proposed. Although similar in style to Dudek's formulation, this classification focuses on *task* rather than robot.

The proposed taxonomy is summarized in Table 1 and explained in detail in the following subsections. Several example tasks are examined and classified using the descriptors at the end of this section.

Time

Evaluations of robot systems are typically conducted in discrete experimental trials. `TIME_LIM` tasks are those where each trial runs over a fixed length of time, while `TIME_MIN` refers to tasks where performance is measured as the time required to complete the task. An example `TIME_LIM` task is "collect as many objects as possible in 10 minutes." An example `TIME_MIN` task would be "collect five attractors as quickly as possible."

Some tasks do not have a time constraint; they are referred to as `TIME_UNLIM`. `TIME_UNLIM` tasks are either carried out indefinitely, or terminate on non-time-related criteria. An example indefinite time task might be "patrol the building for burglars" or "balance the pole."

Unlimited time tasks usually include an optimization criteria that specifies a "window" over which performance should be optimized. Without an optimization criteria, it is impossible to distinguish between the multitude of policies that yield infinite performance over infinite time. One policy, for example, might yield a constant performance of 1.0 at each step, while another provides a performance of 2.0. The two are indistinguishable when summed over an infinite future. Three criteria and their associated descriptors are

- `CRIT_AVG`: Average-reward, where performance is averaged over all steps.
- `CRIT_FINITE`: Finite-horizon, where the performance is summed over a finite number of steps.
- `CRIT_DISC`: Infinite-horizon discounted, where performance in the future is discounted geometrically.

These criteria correspond to models of optimal behavior used in RL approaches (Kaelbling, Littman, & Moore 1996).

Descriptor	Meaning
Time	
`TIME_LIM`	Fixed time task.
`TIME_MIN`	Minimum time task.
`TIME_UNLIM`	Unlimited time task.
`SYNC`	Synchronization required.
Criteria	
`CRIT_FINITE`	Optimize over finite period.
`CRIT_AVG`	Average performance over all future.
`CRIT_DISC`	Discount future performance geometrically.
Subject of action	
`OBJECT_BASED`	Movement/placement of objects is important.
`ROBOT_BASED`	Movement/placement of robots is important.
Resource limits	
`RESOURCE_LIM`	Limited external resources.
`ENERGY_MIN`	Minimum energy task.
`COMP_INT`	Competition between team members for resources.
`COMP_EXT`	Team competes with external agencies.
Group movement	
`CONVERGENCE`	Multiple robots converge.
`COVERAGE`	Multiple agents spread apart.
`MOVEMENT_TO`	Movement to a position.
`MOVEMENT_WHILE`	Movement while maintaining position.
Platform capabilities	
`SINGLE_AGENT`	A single agent can perform task.
`MULTI_AGENT`	Multiple agents are required.
`DISPERSED`	Agents must be dispersed.
`SENSOR_COMPLETE`	Can sense all relevant features.
`SENSOR_LIM`	World is partially observable.
`COMM`	Communication is required.

Table 1: Terms characterizing multirobot tasks.

In addition to elapsed time constraints, some tasks include a synchronization requirement. One example is the task of pushing two buttons simultaneously.

These are labeled SYNC tasks. Communication between agents is one way to accomplish SYNC tasks, but other mechanisms (such as clock synchronization), are also available.

The subject of action

Some tasks involve the placement or movement of an external object (e.g. the ball in soccer) while others concern the positioning of the robotic agent itself (formation). Tasks involving external objects are termed OBJECT_BASED tasks while those involving the agents themselves are ROBOT_BASED tasks.

Limited resources and competition

Many tasks, like foraging, involve limited environmental resources; as an agent performs the task it consumes the resource thus making the task more difficult. In multiagent systems, limited resources may force the agents on a team into competition between themselves or with external agencies. In addition to external environmental resources the robot may also have to monitor *internal* resources, such as fuel.

Tasks that involve limited external resources are termed RESOURCE_LIM. Robotic foraging is typically a RESOURCE_LIM task because there are only a fixed number of objects available for the robots to collect. When agents on a team compete with each other for a limited resource, the task is referred to as COMP_INT (internally competitive). Foraging is internally competitive since one agent's success in collecting an object, reduces the opportunity for others. When the agents compete with external forces, the task is COMP_EXT. Soccer is a COMP_EXT task because the agents on one team compete with the (external) agents on the other team for the opportunity to score goals.

RESOURCE_LIM refers specifically to resources external to the agent (e.g. objects to collect) versus internal (e.g. fuel). Tasks that call for fuel and/or energy conservation are referred to as ENERGY_MIN tasks. "Dock the boat using minimum fuel" for instance, is a TIME_UNLIM ENERGY_MIN task.

Movement

The coordinated movement of several robots is important in search, surveillance, grazing and cleaning tasks. Ali has developed a taxonomy of robotic movement tasks in his investigation of human control techniques for multirobot teams (Ali 1997). His terms, COVERAGE, CONVERGENCE, MOVEMENT_TO and MOVEMENT_WHILE are adopted here for the description of multiagent movement tasks.

Multirobot movement tasks are broken into two primary classifications: COVERAGE and CONVERGENCE. In coverage tasks the robots spread out as much as possible. Examples include search, grazing and cleaning tasks. In convergence tasks robots gather together from dispersed locations. The description of movement tasks is further refined by whether the a robot should move *to* a position or move *while* maintaining a position. In MOVEMENT_TO tasks the robots move from their starting locations to a particular configuration, while in MOVEMENT_WHILE tasks the robots are to maintain a configuration while moving.

Platform dependencies

The capabilities and limitations of the robotic platform can significantly impact the space of potential solutions. A large multi-armed tank robot for example, may be able to individually accomplish many tasks that would otherwise require dozens of smaller robots. Other issues of robot platform, like sensor ranges, can affect the space of possible solutions as well.

Consider the task of transporting a cookie to a homebase. For humans, this is a single-agent problem: pick up the cookie, carry it home. But for ants the task assumes epic multiagent proportions. Tasks in which an individual agent is able to generate positive performance are referred to as SINGLE_AGENT. If it is unlikely that a single agent can perform suitably – that several agents are required – the task is MULTI_AGENT.

Tasks that require agents in several widely separated places at once are DISPERSED. Dispersion might be required because of size, reach, actuator or sensing limitations of individual robots on the team. An example of this type of task is one in which two buttons on opposite sides of a building must be pushed simultaneously (this is also a SYNC task). A task requiring continuous surveillance over areas larger than an individual robot's sensor range require dispersion as well. Soccer is a DISPERSED task since the ball can move rapidly from one part of the field to another; players must be dispersed to ensure effective play. Many COVERAGE tasks, like coordinated search, are DISPERSED because the agents must spread apart to ensure more complete sensor coverage.

In some tasks for robots the agents are provided perfect, or nearly perfect sensing. In other words the agent knows, with complete accuracy, every aspect of the environment's state germane to the task. Many tasks examined in the RL literature share this attribute (e.g. the acrobot (Boone 1997)). These tasks are referred to as SENSOR_COMPLETE. It is often impossible however for an individual agent to sense all aspects of the environment relevant to the task. Even if the robot has a comprehensive sensor suite, the sensors may be degraded in actual use by noise and occlusion. Such tasks are referred to as sensor limited or SENSOR_LIM tasks.

The decision problem for robots in a SENSOR_LIM task is *partially observable* (Littman, Cassandra, & Kaelbling 1995). This means that many distinct situations may be perceived as equivalent by the robot. For instance, as a foraging agent delivers an attractor, it might perceive itself as being in the same state whether it is 5 meters or 4 meters from the delivery area.

Finally, tasks that cannot be accomplished without communication are referred to as COMM tasks. The requirement for communication may be a consequence of sensor limitations. It can be argued for example, that with complete sensing, explicit communication is never required. SENSOR_LIM DISPERSED SYNC tasks however are likely to require communication as a means of achieving synchronization.

The taxonomy described above will now be used to classify two example tasks: multirobot foraging and cooperative movement.

Classification of example tasks

A foraging task might be expressed as "maximize the number of objects collected and delivered to homebase in 10 minutes." Or more explicitly as

$$\mathrm{delivered}(A_i, t) = \begin{cases} 1 & \text{If object } A_i \text{ is in the deliver zone at time } t. \\ 0 & \text{Otherwise.} \end{cases} \quad (1)$$

$$P = \sum_{i=1}^{N} \mathrm{delivered}(A_i, t_0 + 10\mathrm{min}) \quad (2)$$

where $\mathrm{delivered}(A_i, t)$ indicates whether object A_i is in the delivery area at time t, N is the number of objects available for collection. P is performance, the number of objects in the delivery area 10 minutes after the start of the experiment (t_0). The performance metric includes constraints on time (a 10 minute limit) and on the spatial arrangement of the collected objects (they must be in the delivery area). For this task there are no constraints on the location or number of robots. This foraging task is

- TIME_LIM because performance is measured over a fixed period,
- OBJECT_BASED since performance is based on the location of objects, not agents.,
- RESOURCE_LIM because as agents collect objects, the availability of attractors is reduced.,
- COMP_INT because robots on a team compete for access to attractors among themselves,
- SINGLE_AGENT since an individual agent can perform positively, even though multiple agents may provide improved performance.

Now consider a formation maintenance task: "minimize total position error for four robots in a diamond formation," or mathematically,

$$\mathrm{error}(R_i, t) = |\ \mathrm{position}_i(t) - \mathrm{desired\_position}_i(t)\ | \quad (3)$$

$$P = -\sum_{\tau=t_0}^{t_0+10\mathrm{min}} \{\sum_{i=1}^{4} \mathrm{error}(R_i, \tau)\} \quad (4)$$

where $\mathrm{error}(R_i, t)$ is the formation position error for robot i at time t and P is the sum of errors of the four robots over 10 minutes. This task is distinguished from the foraging task above in that the positions of the agents themselves are important rather than the positions of objects to be manipulated. This task is

- TIME_LIM because the task is carried out over 10 minutes,
- ROBOT_BASED because performance depends on the location of agents, not objects,
- CONVERGENCE since the agents should maintain specific positions close to one another,
- MOVEMENT_WHILE because convergence should be maintained while the robots move,
- MULTI_AGENT since it implicitly requires four agents.

Reinforcement functions

The reinforcement function is usually closely coupled to the performance metric for a task. As outlined in the first section, however, there are a number of situations where the reward function cannot or should not be the same as the performance metric. The following subsections enumerate the various alternative of reward structures that have been developed (by the author and others) in response to these issues. The taxonomy is summarized in Table 2. Each approach has advantages and disadvantages; the challenge for the designer is to select a reward function that most effectively balances these trade-offs.

Source of reward

A reinforcement function is classified as INTERNAL_SOURCE or EXTERNAL_SOURCE depending on whether the learner computes a reward internally based on its own sensors, or the reward is determined externally by another agent. As an example of an externally generated reward, consider a dog being trained by his master to fetch a ball. Reinforcement in this case is provided through verbal cajoling, e.g. "no!" "good dog," etc. Notice that the pet's reward is based on its sensing of the trainer's mood and *not* the location of the object to be fetched. Similarly we might train a robot by equipping it with "reward" and "punish" buttons to be activated by a trainer according to his evaluation of the desirability of the robot's behavior. This is the approach Yanco utilized in training a team of communicating robots (Yanco & Stein 1993).

External rewards might be useful in commercial intelligent systems applications where the manufacturer cannot anticipate how the consumer will use the product. It would be unreasonable for example, to assume that the purchaser of an intelligent water heater would be inclined (or able) to program a reward function for it. A more realistic interface is the provision of a "punish" button to be pressed when the heater fails to turn on and the user receives a a cold shower.

A potential problem with external reward systems however, is the possibility of a mis-match between the agent's sensors and those of the trainer. Suppose the water heater's owner takes showers later in the day on weekends than on weekdays. Unless the heater is equipped with a time of day *and* a day of week clock it will be impossible for it to differentiate between weekends and weekdays. The result being that the heater must remain on for the entire morning to avoid negative reinforcement.

Descriptor	Meaning
Source of reward	
INTERNAL_SOURCE	Reward is internal based on sensor values.
EXTERNAL_SOURCE	Reward is generated by external agent.
COMB_SOURCE	Combined internal and external reward.
Relation to performance	
PERFORMANCE	Reward is tied directly to performance.
HEURISTIC	Reward based on intuition of state value.
Time	
IMMEDIATE	Immediate rewards are provided.
DELAYED	Reward is delayed.
Continuity	
DISCRETE	Reward takes on several discrete values.
CONTINUOUS	Reward drawn from continuous interval.
Locality	
LOCAL	Individual agents receive unique rewards.
GLOBAL	All agents receive identical reward signal.
COMB_LOCALITY	Combination of local and global.

Table 2: Multirobot reward taxonomy.

In addition to the potential for sensor mis-match, external rewards complicate the learning problem for planning systems. If the reward is not provided to the planner as a function of sensor state and robot action, the robot must additionally learn a model of the external rewarding agent.

There are some situations where it might be appropriate for an agent to take part of its reward from an external source and derive part of it internally. As an example, a dog may need to satisfy its hunger by eating (internal), but also wish to please its owner (external). A third descriptor `COMBINED_SOURCE` is used for reward structures that combine internal and external rewards.

Relation to performance

A reward function is defined as `PERFORMANCE` if and only if maximum reward implies optimal performance. As an example of a `PERFORMANCE`-based reinforcement function consider the task of docking a boat in minimum time. Performance for this task is $P = -\text{elapsed\_time}$. The function $R(t) = -1$ is an appropriate `PERFORMANCE`-based reward for the task because reward is maximized if and only if performance is maximized. If T is the number of steps required to complete the task, $\sum_{t=1}^{T} -1$ is maximized (least negative) when the task is completed quickly.

In many tasks a `PERFORMANCE`-based reward is *delayed*: it may take many steps for an agent to reach a non-negative reward. In the forage task for instance, measured performance is negative or zero until an attractor is delivered. This delay makes it more difficult for an agent to assign credit or blame to actions taken in the past than if immediate rewards are provided. Heuristic rewards can address this problem by providing more immediate feedback. `HEURISTIC` rewards are based on the programmer's intuition of the value of an agent's actions in particular states. Mataricproposed a `HEURISTIC` approach called *shaped reinforcement* to address this issue in a team of foraging robots (Matarić 1997).

As an example of a heuristic reward consider reinforcement in a robot soccer task. A `PERFORMANCE`-based function rewards or punishes the agents only when a goal is scored. An alternative `HEURISTIC` approach might reinforce the agents with a reward from -1 to 1 depending on whether the ball is close to their own team's goal (-1), the center of the field (0), or the opponent's goal (1).

While heuristic reward functions usually provide for quicker learning, problems include the possibility of local maxima and global maxima that do not correspond with optimal performance. The learning agents may maximize their reward without necessarily optimizing performance.

Time and continuity

As mentioned above, rewards are often `DELAYED`. This means it may require many (perhaps hundreds) of movement steps before an agent receives a positive reward. Reward functions that provide immediate feedback as to the utility of an action in a particular state are called `IMMEDIATE`. `HEURISTIC` rewards often seek to improve performance by by providing more immediate feedback than their `PERFORMANCE`-based counterparts.

There are some tasks that provide `IMMEDIATE` `PERFORMANCE`-based rewards as well. As an example, consider the task of maintaining a specific water level by opening and closing a valve. The difference between the current water level and the desired level (the error) can be utilized as an `IMMEDIATE PERFORMANCE` reward.

The error signal in water level is a `CONTINUOUS` reward signal because it is drawn from a continuous interval. Other reward functions provide `DISCRETE` values. As an example of `DISCRETE` reinforcement consider the reward for a foraging robot. A discrete function might provide a reward of -1 at every step except +1 when the robot delivers an attractor.

Locality

An important consideration for learning multirobot teams is whether each agent should be rewarded individually or all agents should receive the same reinforcement. GLOBAL reinforcement refers to the case where a single reinforcement signal is simultaneously delivered to all agents, while LOCAL reinforcement rewards each agent individually. In the case of soccer, a global system would reward all team members when any one of them scores a goal. With local reinforcement only the agent that scored the goal would be rewarded. Global rewards correspond more closely with overall system performance, but they may not be appropriate for all tasks.

In a SENSOR_LIM task, it may not be feasible for each agent to monitor activities of all the others in order to compute the global reward. In this case, implementation of global reinforcement may require expensive communication hardware. Also, there may be a weak correlation between an individual agent's actions and the value of the global signal; an agent may receive a positive reward while executing the "wrong" action. This could occur in a globally-rewarded foraging task if two agents simultaneously deposit attractors – one in the correct spot but the other in the wrong spot. Since all agents are simultaneously rewarded for the first agent's successful delivery, the second agent may incorrectly infer it behaved correctly.

Local reinforcement has the advantage of a stronger correlation between reward and the individual's actions, potentially providing for faster learning. But agents using local rewards for training may not be able to optimize performance of the overall system. Some investigators try to balance these goals by combining or blending local and global rewards (Mataric 1994). These reward systems are referred to as COMBINED_LOCALITY.

Discussion and summary

Even though most work in RL views the task performance metric and reward as one and the same, there are a number of situations, especially in multiagent systems, where different types of reward functions are appropriate. To investigate the merits of the various reward functions in different task domains, principled taxonomies of task and reward are required. To address this issue, we introduce classification tools for multirobot tasks and reward functions. These taxonomies provide a framework for investigating how different types of reward impact performance in different tasks.

Task classification is based on features of the performance metric, environment and robotic platform that differentiate one task from another. The characterization focuses on features of tasks that may correlate with a need for cooperation, communication or diversity. This classification system helps us identify why multirobot performance and diversity might change for different tasks, even when the same type of reinforcement is utilized to train the agents.

The paper also introduces a taxonomy of reinforcement functions. As mentioned above, ideally learning robots are trained using a reward signal that parallels the performance metric. It is not always feasible however to utilize performance-based rewards in a multirobot system. There are tradeoffs to consider in the selection of an appropriate reward function; for instance some functions provide quicker learning but slightly poorer performance (Balch 1998). The taxonomy of reward functions provides a framework within which a robot system designer can make these tradeoffs intelligently.

This work is a starting point. It is the author's hope that others will expand and build upon these classifications by identifying other aspects of multiagent task and reward that further distinguish them from one another. This will extend our understanding of the complexities of learning multiagent robotic systems.

References

Ali, K. 1997. Phd thesis proposal. College of Computing, Georgia Institute of Technology.

Balch, T. 1998. *Behavioral Diversity in Learning Robot Teams.* Ph.D. Dissertation, College of Computing, Georgia Institute of Technology.

Boone, G. N. 1997. Efficient reinforcement learning: Model-based acrobot control. In *IEEE International Conference on Robotics and Automation*, 229–234.

Dudek, G.; Jenkin, E.; and Wilkes, D. 1993. A taxonomy for swarm robots. In *Proc. 1993 IEEE International Conference on Intelligent Robots and Systems*, 441–447.

Kaelbling, L.; Littman, M.; and Moore, A. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4:237–285.

Littman, M.; Cassandra, A.; and Kaelbling, L. 1995. Learning policies for partially observable environments: Scaling up. In *Proceedings of the Twelfth International Conference on Machine Learning.*

Mataric, M. 1994. Learning to behave socially. In *Proceedings of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats 3.*

Matarić, M. 1997. Reinforcement learning in the multi-robot domain. *Autonomous Robots* 4(1):73–83.

Yanco, H., and Stein, L. 1993. An adaptive communication protocol for cooperating mobile robots. In *From Animals to Animats: Proc. 2nd International Conference on the Simulation of Adaptive Behavior*, 478–485. Boston: MIT Press.