Potential-Based Difference Rewards for Multiagent Reinforcement Learning

Sam Devlin
University of York, UK
sam.devlin@york.ac.uk

Logan Yliniemi
Oregon State University, USA
logan.yliniemi@engr.orst.edu

Daniel Kudenko
University of York, UK
daniel.kudenko@york.ac.uk

Kagan Tumer
Oregon State University, USA
kagan.tumer@oregonstate.edu

ABSTRACT

Difference rewards and potential-based reward shaping can both significantly improve the joint policy learnt by multiple reinforcement learning agents acting simultaneously in the same environment. Difference rewards capture an agent's contribution to the system's performance. Potential-based reward shaping has been proven to not alter the Nash equilibria of the system but requires domain-specific knowledge. This paper introduces two novel reward functions that combine these methods to leverage the benefits of both.

Using the difference reward's Counterfactual as Potential (CaP) allows the application of potential-based reward shaping to a wide range of multiagent systems without the need for domain specific knowledge whilst still maintaining the theoretical guarantee of consistent Nash equilibria.

Alternatively, Difference Rewards incorporating Potential-Based Reward Shaping ($DRiP$) uses potential-based reward shaping to further shape difference rewards. By exploiting prior knowledge of a problem domain, this paper demonstrates agents using this approach can converge either up to 23.8 times faster than or to joint policies up to 196% better than agents using difference rewards alone.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

General Terms

Algorithms, Performance, Experimentation

Keywords

Reward Shaping; Multiagent Reinforcement Learning

1. INTRODUCTION

Multiagent reinforcement learning solutions benefit from task distribution and adaptive, autonomous behaviour. However, the problem of structural credit assignment is often a limiting factor when deploying such a solution. It is very difficult when multiple agents are acting in the same environment to determine, from the environment's reward function alone, who was responsible for the feedback received.

Difference rewards have been repeatedly demonstrated to help with credit assignment [1, 2, 14, 9, 15, 16] by shaping the global reward to instead reward agents contributing to the system's performance and punish agents that do not. This approach typically improves the resultant behaviour of agents significantly.

Potential-based reward shaping has also recently been used to improve multiagent reinforcement learning solutions [3, 5, 6]. This approach is theoretically guaranteed to learn a joint policy from the same set of possible joint policies agents using the original reward function alone would have learned. However, unlike difference rewards, potential-based reward shaping requires prior knowledge of the problem domain and is typically tailored to each new application.

These two methods have previously only been studied separately but are not mutually exclusive. Therefore, in this work, we use them together to leverage the benefits of both.

One approach to combining these methods is to represent the knowledge captured by difference rewards as a potential function. This approach has the same theoretical guarantees as multiagent potential-based reward shaping and can be applied without the need for domain specific knowledge.

The second approach is to use potential-based reward shaping with a manual heuristic to shape difference rewards. This approach incorporates domain specific knowledge into agents learning by difference rewards and performs significantly better than agents learning by either approach alone.

Specifically, the contributions of this work are:

- **Counterfactuals as Potential (CaP):**
  An automated method of generating a multiagent potential function from the same knowledge represented by difference rewards.
- **Difference Rewards incorporating Potential-Based Reward Shaping ($DRiP$):**
  Shaping difference rewards by potential-based reward shaping to significantly improve the learning behaviour of either alone.

The rest of this paper is organized as follows. The next section covers all relevant background material. Section 3 introduces a unified framework and formally specifies CaP and $DRiP$ within this framework. These approaches are then evaluated in Sections 4 and 5 in two distinct domains.

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*
Copyright © 2014, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

The first is a well studied congestion problem with a known optimal solution chosen for thorough analysis. This study is also the largest scale application of potential-based reward shaping, in terms of number of agents, published to date. The second is a coordination domain with a significantly larger state space. The paper concludes with a section discussing the implications of combining difference rewards and potential-based reward shaping.

2. BACKGROUND

Reinforcement learning allows agents to learn by reward and punishment from interactions with the environment [13]. One common algorithm from this field is Q-learning [17]. After each transition, from state s to s' due to action a, Q-learning updates state-action values $Q(s,a)$ by the formula:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)] \quad (1)$$

where α is the rate of learning, r the reward received from the environment and γ is the discount factor.

From these values a policy mapping states to actions can be generated. To balance the requirements of both exploration and exploitation, one method of action selection is *epsilon*-greedy. This method chooses the highest value action for the current state with probability $1-\epsilon$ and chooses a random action with the remaining probability ϵ [13].

For large domains, storing values for every state can be infeasible. In these instances it is necessary to use function approximation to generalise across states. One common approach is tile coding [13], this method groups states by an exhaustive partition of the state space. By including less tiles than states, agents can learn significantly quicker.

Multiagent systems deploy multiple agents in the same environment and can therefore benefit from the distribution of tasks and redundancy of agents [22].

Unlike single-agent reinforcement learning where the goal is to maximise the individual's reward, when multiple self motivated agents are deployed some compromise must be made. Typically the system is designed aiming to converge to a Nash Equilibrium [4].

Two typical reward functions for multiagent reinforcement learning exist; local rewards unique to each agent or global rewards representative of the group's performance.

A **local reward** (L_i) is the reward based on the part of the system that an agent i can directly observe. Using this reward signal often encourages "selfish" behaviour, in which the agent may act at cross-purposes with other agents while blindly attempting to increase its own reward, causing poor system performance.

The **global reward** (G) is the system performance used as a learning signal. This encourages the agent to act in the system's interest, but includes a substantial amount of noise from other agents acting simultaneously. An agent's own contribution to the global reward may be dwarfed by the contribution of hundreds of other agents, resulting in a low "signal to noise ratio" [2].

The immediate reward from the environment in both these cases (L_i or G) is often infrequent and ambiguous. This raises issues of both temporal and structural credit assignment [1]; the problems of which action and which agent were responsible for the reward received. The idea of *reward shaping* is to provide an additional, more informative reward to simplify learning [10, 12]. Two popular methods of reward shaping, previously studied separately, are difference rewards and potential-based reward shaping. Each will be reviewed in depth in the following sections.

2.1 Difference Rewards

To use reinforcement learning in a multiagent system, it is important to reward an agent based on its contribution to the system. This is difficult due to the other agents acting in the environment, obscuring the agent's individual contribution to the system objective.

The **difference reward** (D_i) is a shaped reward signal that helps an agent learn the consequences of its actions on the system objective by removing a large amount of the noise created by the actions of other agents active in the system [1, 2]. It is defined as

$$D_i(z) = G(z) - G(z_{-i}) \quad (2)$$

where z is a general term representative of either states or state-action pairs depending on the application, $G(z)$ is the global system performance, and $G(z_{-i})$ is $G(z)$ for a theoretical system without the contribution of agent i.

Any action taken to increase D_i simultaneously increases G, while agent i's impact on its own reward is much higher than its relative impact on G [2]. These two properties allow for the difference reward to boost learning performance (over training with another reward function) in a multiagent system by a significant degree.

The difference reward can generally be estimated, even in extremely complex domains like air traffic in the US airspace [14]. However, because of the nature of the domains used in this work, D_i is directly calculable.

Using difference rewards directly or estimating them has proven to be an effective technique for credit assignment in many applications, including rover coordination, urban road traffic management, data routing, distributed product marketing and satellite coordination [2, 9, 16, 15, 21, 20].

2.2 Potential-Based Reward Shaping

If misused, reward shaping can introduce undesirable behaviours. For example, when trying to make an agent learn to ride a bicycle from point A to point B, Randløv and Alstrom gave an additional positive reward every time step the bike stayed balanced. This advice seems intuitive, however, the agent learnt to exploit it and circled continuously instead of moving towards its intended goal at point B [12].

Potential-based reward shaping $(PBRS)$ was proposed to ensure such problems do not occur [10]. The additional reward given in this approach is the difference of a potential function Φ defined over a source state s and a destination state s':

$$PBRS = r + \gamma\Phi(s') - \Phi(s) \quad (3)$$

where r is the original reward from the environment and γ is the same discount factor as used in the agent's update rule (see Equation 1).

This formulation of reward shaping has been proven to not alter the optimal policy of a single agent in both infinite- and finite- state MDPs [10].

Wiewiora [18] later proved that an agent learning with potential-based reward shaping and no knowledge-based Q-table initialisation will behave identically to an agent without reward shaping when the latter agent's value function is initialised with the same potential function.

More recent work on potential-based reward shaping has removed the assumptions of a single agent acting alone and of a static potential function from the original proof. In multiagent systems, it has been proven that potential-based reward shaping can change the joint policy learnt but does not change the Nash equilibria of the underlying game [6].

With a dynamic potential function, it has been proven that the existing single agent and multiagent guarantees are maintained provided the potential of a state is evaluated at the time the state is entered and used in both the potential calculation on entering and exiting the state. Furthermore, potential-based reward shaping with a dynamic potential function is not equivalent to Q-table initialisation [7].

To ensure these theoretical guarantees in episodic domains it is necessary that the potential of the final state of every episode has a potential of zero. Our preferred method of doing this is to transition all agents at the end of an episode to an absorbing state regardless of the final state or action chosen. This transition receives zero reward from the environment, but still receives the additional reward from potential-based reward shaping. Given that this occurs in all circumstances, is not dependent on state or action and gives no additional environmental reward it can be applied to agents learning by any reward function without altering their learning behaviour.

To apply potential-based reward shaping some prior knowledge of the domain is required. The potential function $\Phi(s)$ should return a value representative of the believed desirability for the agent to be in state s. This has been implemented previously using heuristic knowledge [5, 10, 19] or a STRIPS plan [8]. Until now, no multiagent applications have automated this assignment of potential to states.

3. UNIFIED FRAMEWORK

The primary purpose of this work is to bring together two well studied reward shaping methodologies; difference rewards and potential-based reward shaping. The purpose of this framework is to make a modular system in which different parts of each of the reward shaping methodologies can be used to their greatest advantage. This allows us to leverage the benefits offered by both, while reducing the impact of their respective weaknesses.

We represent this framework through a generic reward-shaping representation, in which the shaped reward is formulated as:

$$r_{shaped} = r(s,a,s') + F(s,s') \tag{4}$$

where r is the original reward received from the environment after the transition from state s to state s' upon taking action a, and $F(s,s')$ is the additional shaping reward.

The shaped reward r_{shaped} then replaces r in the agent's update rule (see Equation 1).

Within this framework, typical applications of potential-based reward shaping take the form:

$$\begin{aligned} r(s,a,s') &= G(s,a,s') \\ F(s,s') &= \gamma\Phi(s') - \Phi(s) \end{aligned} \tag{5}$$

where $\Phi(s)$ is implemented manually with domain specific knowledge.

Alternatively, for a traditional difference reward, this framework takes the form:

$$\begin{aligned} r(s,a,s') &= G(s,a,s') \\ F(s,s') &= -G(s'_{-i}) \end{aligned} \tag{6}$$

This framework can leverage the benefits offered by any particular reward function and shaping signal. If a local reward function represents a fairly robust policy where agents are all independently productive, it may serve well to set $r(s,a,s') = L_i(s,a,s')$. Likewise the reward shaping signal $F(s,s')$ can be adjusted in the same manner. Furthermore, $F(s,s')$ could be replaced with the more general $F(s,a,s',a')$ to include reward shaping based on actions as well as states [19]. However, such an extension was not needed for the methods we have studied here.

The next two sections formally define, within the format of this framework, our two novel combinations of difference rewards and potential-based reward shaping.

3.1 Counterfactual as Potential (CaP)

Drawing inspiration from the difference reward, one useful method for multiagent potential-based reward shaping is to automatically assign potentials to states by using the counterfactual term $G(z_{-i})$. Formally, in the uniform framework CaP is defined as:

$$\begin{aligned} r(s,a,s') &= G(s,a,s') \\ F(s,s') &= \gamma\Phi(s') - \Phi(s) \\ \Phi(s) &= G(s_{-i}) \end{aligned} \tag{7}$$

This rewards agents for high-performing global evaluations (first term), while also encouraging them to be in states (second term) where the other agents in the system are performing well (third term). This reduces the likelihood that they will choose to work at cross-purposes and drive down system performance.

Given that $\Phi(s)$ is based entirely on the state of other agents, CaP will give different potentials for the same local state. This makes CaP an instance of dynamic potential-based reward shaping [7]. Therefore, CaP is guaranteed to learn a policy from the same set of policies as would have been learnt by G alone but is not equivalent to Q-table initialisation. Furthermore, this reward signal is aligned with the global reward, which means that any steps taken to increase it will simultaneously increase $G(s,a,s')$; a benefit commonly attributed to difference rewards [15].

As CaP is guaranteed to have the same Nash equilibria as G alone or with any other potential-based reward shaping, its merit will be judged in comparison to these reward functions.

3.2 Difference Rewards incorporating Potential-Based Reward Shaping ($DRiP$)

Alternatively, we can further shape the difference reward signal by simultaneously applying potential-based reward shaping. Formally, in the unified framework this approach is defined by:

$$\begin{aligned} r(s,a,s') &= G(s,a,s') \\ F(s,s') &= -G(s'_{-i}) + \gamma\Phi(s') - \Phi(s) \end{aligned} \tag{8}$$

This approach directly leverages the benefits of difference rewards, and also allows prior knowledge specified by $\Phi(s)$ to be provided to the agents.

This method is guaranteed to learn a policy from the same set of policies that could have been learnt by difference rewards alone. This can be proven using the same reasoning as the original proof for multiagent potential-based reward shaping [6].

The merit of *DRiP* will be judged by whether it can direct exploration sufficiently to either increase the rate of learning and/or the quality of the final joint policy compared to difference rewards alone.

4. BEACH PROBLEM DOMAIN (BPD)

We present these techniques first in the BPD, a congestion game with multiple states [11]. This domain presents a congestion problem in which the agents must coordinate to maximize a system level goal.

We have also completed the same experiments in the classical version of the El Farol Bar problem and obtained very similar results. However, the BPD allows us to demonstrate the ability of *CaP* and *DRiP* to perform well in problem instances with multiple states and timesteps.

4.1 Description and Algorithm

In the BPD, each individual agent must choose at which section of a beach they will spend their day. Agents start initially near a hotel on one section of the beach. At each time step, an agent knows which section of the beach it is currently on and must take an action to move to either an adjacent section (left or right) or stay still. Once all agents have moved, they are rewarded. The highest rewards are received for sections of the beach where the number of agents present is equal to the optimal capacity ψ. If more than ψ agents are present, the beach section is overcrowded and, therefore, undesirable. Beach sections with less than ψ agents are also undesirable. However, there are many more agents than $\psi * sections$ and, therefore, the BPD is a congestion problem. This process is detailed further in Algorithm 1.

Formally, the enjoyment of the agents on each section of beach is modeled as:

$$L(s,t) = x_t e^{\frac{-x_t}{\psi}} \tag{9}$$

where $L(s,t)$ is the local reward for the beach section s, x_t is the number of agents on that beach section at timestep t, and ψ is the optimal capacity of that section. The global reward or system utility, then, is simply a summation over all sections of the beach:

$$G(t) = \sum_{s \in \mathbf{B}} L(s,t) \tag{10}$$

where $\mathbf{B}$ is the set of sections in the beach.

For each agent, the difference reward can be directly calculated by applying Equation 2; every section not currently attended by the agent at a particular timestep cancels out, as the agent has no impact on their evaluations, leaving:

$$D_i(t) = L(s,t) - (x_t - 1)e^{\frac{-x_t - 1}{\psi}} \tag{11}$$

where x_t is the number of agents attending the same section of the beach s as agent i at timestep t. This evaluation is precisely equivalent to applying Equation 2 to Equation 10 directly.

In the BPD, with a sufficient number of agents, the optimal solution is for all of the agents to converge to a single

Algorithm 1 Beach Problem Domain with *DRiP*

```
1:  initialize $Q$-values: $\forall s, a | Q(s,a) = -1$
2:  initialize static potential $\Phi$ via Equations 12
3:  for $episode = 1 \rightarrow end\_episode$ do
4:      for $timestep = 1 \rightarrow end\_timesteps$ do
5:          for $i = 1 \rightarrow total\_agents$ do
6:              sense current section $s$
7:              set potential $\Phi(s)$ (Equation 12)
8:              choose action $a = \{-1, 0, 1\}$, using $\epsilon$-greedy
9:              move agent to $s' = \{s-1, s, s+1\}$
10:             set potential $\Phi(s')$ (Equation 12)
11:         end for
12:         move agents $s' \notin \mathbf{B}$ to nearest section
13:         for All beach sections $s \in \mathbf{B}$ do
14:             evaluate local rewards (Equation 9)
15:         end for
16:         evaluate global reward (Equation 10)
17:         for $i = 1 \rightarrow total\_agents$ do
18:             evaluate difference rewards (Equation 11)
19:             set $r_{shaped}$ (Equation 8)
20:             update $Q(s,a)$ values (Equation 1)
21:         end for
22:         reduce $\epsilon$ by multiplication with decay rate
23:         reduce $\alpha$ by multiplication with decay rate
24:     end for
25:     for $i = 1 \rightarrow total\_agents$ do
26:         choose action $a$, using $\epsilon$-greedy
27:         move to absorbing state
28:         set $r_{shaped} = 0 - \Phi(s')$
29:         update $Q(s', a)$ values (Equation 1)
30:     end for
31: end for
```

section of the beach, while only ψ agents stay on each of the other sections of the beach. This results in the highest possible G calculation. Therefore, for an instance of this domain with 5 beach sections, the manual, hand-coded heuristic used by *DRiP* encourages this behaviour. Specifically, for all states $\Phi(s) = 0$ except for the following cases:

$$\begin{array}{lll}
\text{if } agent\_id & \in [0, \psi - 1] & \Phi(0) = 10 \\
\text{if } agent\_id & \in [\psi, 2\psi - 1] & \Phi(1) = 10 \\
\text{if } agent\_id & \in [2\psi, X - 2\psi - 1] & \Phi(2) = 10 \\
\text{if } agent\_id & \in [X - 2\psi, X - \psi - 1] & \Phi(3) = 10 \\
\text{if } agent\_id & \in [X - \psi, X - 1] & \Phi(4) = 10
\end{array} \tag{12}$$

where $X = \sum_{i=0}^{size(\mathbf{B})} x_i$ i.e. the number of agents.

While this potential function is not optimal, it does provide a very high amount of information to the agents and aids greatly in their coordination. This heuristic was also used with the global reward to represent a typical solution using potential-based reward shaping.

4.2 Experimental Setup

We present two studies in the BPD; the first is a single-step instance (i.e. $end\_timestep = 1$) with $\gamma = 0.9$ whilst the second is time-extended with $end\_timestep = 5$ and $\gamma = 1$. The experimental parameters were as follows: $\alpha = 0.1$, $alpha\_decay\_rate = 0.9999$, $\epsilon = 0.05$, $epsilon\_decay\_rate = 0.9999$, $end\_episode = 20,000$, $num\_agents = 100$, capacity $\psi = 7$, beach sections $size(\mathbf{B}) = 5$. The first $num\_agents/2$ begin at a hotel on beach section 1, the rest at section 3.

In all cases, regardless of the learning signal, we report on the final system-level performance G and include error bars representative of the standard error of the mean based on 30 statistical runs. Specifically, we calculate the error as $\sigma/\sqrt{n}$ where σ is the standard deviation and n is the number of statistical runs. For some reward functions, in some figures the error bars are smaller than the symbols used to plot the result. However, they are present on all graphs for all reward functions.

4.3 Experimental Results

In the single-step instance of the BPD, presented in Figure 1, we see the typical result of agents learning by L_i performing poorly, those learning by G performing significantly better and those learning by D significantly better still.

The more interesting contribution of this study, however, is to consider the performance of CaP with regard to G alone and $G+ManualPBRS$ and the performance of $DRiP$ with comparison to D alone.

In this instance, as will become a recurring pattern, $DRiP$ learns significantly quicker than any other method and learns a joint policy representative of the highest global performance. More specifically, in this instance of the BPD, agents using $DRiP$ converge 23.8 times faster than agents using difference rewards alone where time of convergence is taken to be the first 10 consecutive episodes to all be within 5% of the average performance of the final 150 episodes.

$G + CaP$ significantly outperforms G alone but is beaten by $G + ManualPBRS$. However, the performance of $G + ManualPBRS$ is highly dependent on the quality of the heuristic given. Furthermore, the potential function given by Equation 12 and presented in Figure 1 is very strong.

To emphasise how much potential-based reward shaping with manual implementations of Φ depends on the quality of heuristic, we designed 3 other heuristics and applied them in the single-step instance of the BPD. Figure 2 illustrates $G+ManualPBRS$ with each heuristic and Figure 3 illustrates $DRiP$ with each of them. The heuristics used are:

- **Overcrowd One:** The same heuristic used in the previous experiment and defined in Equation 12.
- **Middle:** All agents are invited to a party at beach 2.

$$\text{if } s == 2 \quad \Phi(s) = 10 \tag{13}$$
$$\text{else} \quad \Phi(s) = 0 \tag{14}$$

- **Spread:** Agents are encouraged to spread evenly across the beach.

$$\begin{aligned}
&\text{if } agent\_id && \in [0, (X/size(\mathbf{B})) - 1] && \Phi(0) = 10\\
&\text{if } agent\_id && \in [X/size(B), (2X/size(\mathbf{B})) - 1] && \Phi(1) = 10\\
&\text{if } agent\_id && \in [2X/size(B), (3X/size(\mathbf{B})) - 1] && \Phi(2) = 10\\
&\text{if } agent\_id && \in [3X/size(B), (4X/size(\mathbf{B})) - 1] && \Phi(3) = 10\\
&\text{if } agent\_id && \in [4X/size(B), (5X/size(\mathbf{B})) - 1] && \Phi(4) = 10
\end{aligned} \tag{15}$$

where $X = \sum_{i=0}^{size(\mathbf{B})} x_i$ i.e. the number of agents.

- **Overcrowd All:** Encourages agents to go to slightly overcrowded beaches.

$$\text{if } \psi < x_t < 2\psi \quad \Phi(s) = 10 \tag{16}$$
$$\text{else} \quad \Phi(s) = 0 \tag{17}$$

Figures 2 and 3 show there can be significant differences in learning behaviour when different heuristics are used but $DRiP$ is more resilient to changes in the heuristic shaping it than G. Both figures also show examples of a heuristic (overcrowd all) that negatively affects the final performance of all agents. However, agents learning from $DRiP$ with a bad heuristic learn a joint behaviour of equivalent final performance to those using G with its best heuristic (middle).

Next, we consider the results from the time-extended instance of the BPD presented in Figure 4. In particular we note that a minor change in the problem domain, moving from a single timestep to 5 timesteps, has significantly reduced the performance of $G+ManualPBRS$. $G+CaP$ now significantly outperforms both G alone and $G+ManualPBRS$.

Furthermore, $DRiP$ again significantly outperforms all other reward functions. Its resilience to quality of heuristic is also present in its ability to handle changes in characteristics of the problem domain.

Finally, it is worth noting, that these studies are significantly larger, in terms of number of agents, than any previous published study of potential-based reward shaping. Before this work, the largest application was to a system with 5 agents learning simultaneously [5]. With so few agents in previous studies, the question has been raised as to whether the characteristic effects of multiagent potential-based reward shaping (increased rate of learning and improved final performance) would still occur with a larger number of

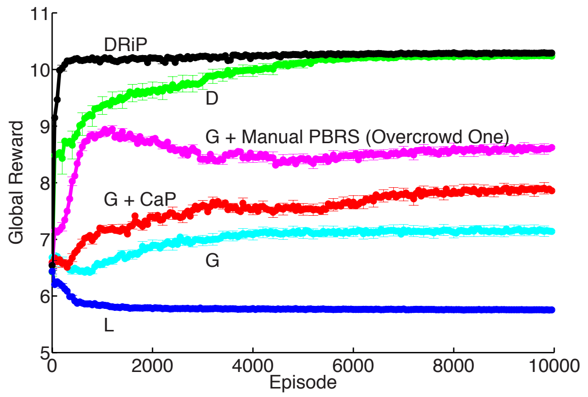

Figure 1: Single-step Beach Domain Results

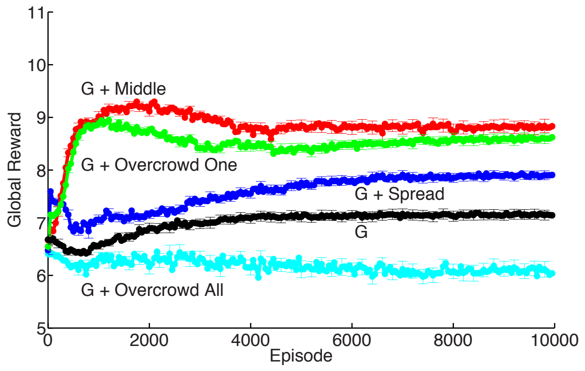

Figure 2: Single-step Beach Domain with $G + ManualPBRS$ using Various Heuristics

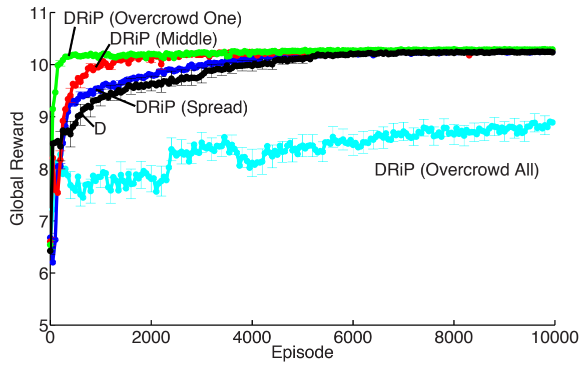

Figure 3: Single-step Beach Domain with *DRiP* using Various Heuristics

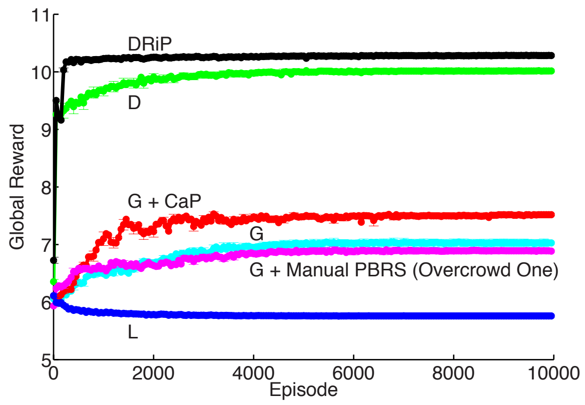

Figure 4: Time-Extended Beach Domain Results

agents. This study of the BPD contributes many positive examples to support the argument that potential-based reward shaping does scale well with the number of agents.

5. GRIDWORLD DOMAIN (GWD)

To exhibit the generality of *CaP* and *DRiP*, we present results from a second domain: a multiagent grid world where agents seek to observe distributed points of interest. Given the settings chosen, this domain presents a coordination problem, as opposed to the congestion problem of the previous domain. Furthermore, in this study we scaled in terms of state-action space as opposed to number of agents to further differentiate the two studies presented.

5.1 Description and Algorithm

In the GWD, multiple agents are deployed within a two dimensional grid. Each of the agents in the set $\mathbf{A}$ must individually choose their actions from $\{up, down, left, right, stay\_still\}$ to navigate toward points of interest (POIs $\in \mathbf{P}$) to observe. The quality of observation increases as the agent gets nearer to the POI, and is summed over all timesteps.

Agents are initially bunched toward the center of the grid world, and POIs are placed near the corners (we performed other trials with other POI configurations and arrived at similar results). Additional agents observing a POI do not increase the system utility; only the highest quality observation is counted. The optimal policy is for agents to spread out and obtain full coverage over the most valuable POIs.

The quality of observation O of each POI is calculated as:

$$O(agent, poi) = \begin{cases} value(poi) & dist(agent, poi) <= 2 \\ \frac{value(poi)}{dist(agent,poi)^2} & 2 < dist(agent, poi) <= 10 \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

where $dist(agent, poi)$ returns the manhattan distance between the agent and the POI and $value(poi)$ returns a numerical value representative of the desirability of the POI.

The total value of the POIs was set to a static value, to ensure all experiments could receive the same range of rewards. However, the value of each individual POI was stochastically determined before each run to limit the quality of the heuristics that could be manually designed.

The agent's local reward is set as the quality of observations that they conducted on all POIs, regardless of any other agents that may be making observations of the same POI:

$$L_i(t) = \sum_{poi \in \mathbf{P}} O(i, poi) \tag{19}$$

The global system utility is set as the quality of the best observation across all POIs:

$$G(t) = \sum_{poi \in \mathbf{P}} \max_{i \in \mathbf{A}} O(i, poi) \tag{20}$$

Difference rewards are calculated as the quality of the best observations attained, not considering agent i:

$$D_i(t) = G(t) - \sum_{poi \in \mathbf{P}} \max_{i \in \mathbf{A}_{-i}} O(i, poi) \tag{21}$$

Thus, the potential function for *CaP* is calculated as:

$$\Phi(s) = \sum_{poi \in \mathbf{P}} \max_{i \in \mathbf{A}_{-i}} O(i, poi) \tag{22}$$

Algorithm 2 Gridworld Domain with $G + CaP$

```
 1: initialize Q-values: $\forall s, a | Q(s, a) = -1$
 2: for $episode = 1 \rightarrow end\_episode$ do
 3:     for $t = 1 \rightarrow end\_timesteps$ do
 4:         for $i = 1 \rightarrow total\_agents$ do
 5:             sense current state $s = \{x, y\}$
 6:             choose action $a$, using $\epsilon$-greedy
 7:             move agent to $s'$ according to $a$
 8:         end for
 9:         evaluate global reward (Equation 20)
10:         for $i = 1 \rightarrow total\_agents$ do
11:             set $\Phi(s')$ (Equation 22)
12:             set $r_{shaped} = G + \gamma\Phi(s') - \Phi(s)$ (Equation 7)
13:             update $Q(s, a)$ values (Equation 1)
14:         end for
15:         reduce $\epsilon$ by multiplication with decay rate
16:         reduce $\alpha$ by multiplication with decay rate
17:     end for
18:     for $i = 1 \rightarrow total\_agents$ do
19:         choose action $a$, using $\epsilon$-greedy
20:         move to absorbing state
21:         set $r_{shaped} = 0 - \Phi(s')$
22:         update $Q(s', a)$ values (Equation 1)
23:     end for
24: end for
```

As the static potential function, used for $DRiP$ and $G + ManualPBRS$, we simply direct the agents to move away from the center point to encourage exploration:

$$\Phi(s) = dist(agent, center) \qquad (23)$$

where *center* is the point at the center of the grid world near which the agents started.

Note that this potential function offers no operational knowledge about the location or value of the POIs, and merely serves to encourage the agents to explore away from the center. This is a contrasting type of potential function to that employed in the BPD, which expressed significant domain knowledge.

5.2 Experimental Setup

We present two studies in the GWD; the first is on a 10x10 grid with deterministic actions whilst the second is on a 100x100 grid approximated by the agents by a single 10x10 tiling with 5% chance of action failure. The first setting ran 2500 episodes of 50 time steps. The second ran 2500 episodes of 250 time steps.

The experimental parameters were as follows: $\alpha = 0.1$, $alpha\_decay\_rate = 0.9999$, $\epsilon = 0.2$, $epsilon\_decay\_rate = 0.9999$, $\gamma = 0.9$, $num\_agents = 4$, and $num\_POIs = 4$. All POIs have a value of 1, except one that is chosen at random at the beginning of a run to have value 5. By assigning the value of POIs randomly, designing a suitable heuristic becomes significantly harder.

In all cases, regardless of the learning signal, we report on the summation of system-level performance G over all time steps and include error bars representative of the standard error of the mean based on 30 statistical runs. Specifically, we calculate the error as $\sigma/\sqrt{n}$ where σ is the standard deviation and n is the number of statistical runs. For some reward functions, in some figures the error bars are smaller than the symbols used to plot the result. However, they are present on all graphs for all reward functions. Please note, given the longer episodes, the agents in the second setting can receive significantly higher rewards.

5.3 Experimental Results

In the first instance of the problem domain, the results presented in Figure 5 show that $G+CaP$ again significantly outperforms G alone and $DRiP$ again significantly outperforms all other solutions.

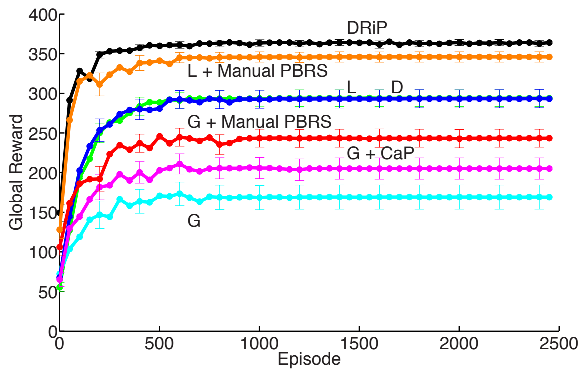

Figure 5: 10x10 GridWorld Domain

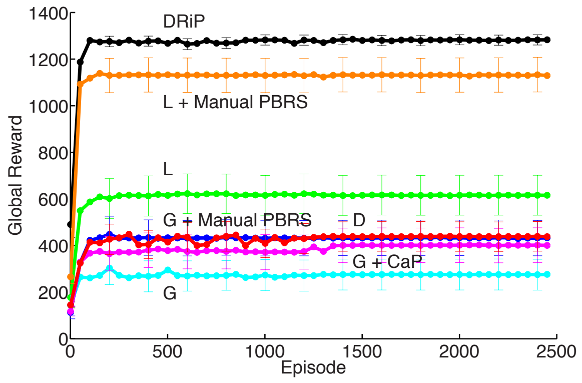

Figure 6: 100x100 Stochastic GridWorld Domain

The largest difference in this result compared to those in the BPD is that agents learning by the local reward can now match those using difference rewards. This occurs because agents acting in their own self interest have a very high signal-to-noise ratio on their local learning signal, allowing them to quickly discern which of their actions increased their reward. In this domain agents are less likely to work at cross-purposes than in the BPD, where the congestion creates a situation where doing so is more likely. This observation further emphasises the difference between the two domains presented in this paper, supporting our argument that CaP and $DRiP$ are generally applicable.

Given the high performance of agents using local rewards, we performed additional experiments with $L+ManualPBRS$. As with global rewards, the manual heuristic used is suitable in this instance of the problem domain and, therefore, $L + ManualPBRS$ significantly outperformed L. However, despite being equal to difference rewards when neither received potential-based reward shaping, $DRiP$ still significantly outperformed $L + ManualPBRS$.

This result becomes even more impressive when we move on to the larger instance of the problem domain, results from which are presented in Figure 6. In this example, agents learning by local rewards alone now significantly outperform those learning with difference rewards alone.

However, agents learning from $DRiP$ are still the best performing of all agents. The effect of adding potential-based reward shaping to difference rewards (a 196% increase in performance) is significantly larger than adding it to local rewards. By exploiting domain specific knowledge, $DRiP$ is able to learn a suitable policy in a problem domain where difference rewards alone cannot.

6. DISCUSSION

To conclude this paper we collate our experiences of using $DRiP$ and CaP in all domains studied. We have seen repeatedly throughout that potential-based reward shaping and difference rewards can be used together to leverage the benefits of both.

$DRiP$ consistently outperforms any other reward function combining L, G, D and $PBRS$. If suitable domain specific knowledge is available, exploiting it by potential-based reward shaping in addition to difference rewards significantly improves agents' learning performance over using difference rewards alone.

CaP provides a method of automatically generating a dynamic potential function for multiagent problem domains. CaP will be useful in future multiagent applications as it removes the need to implement a domain-specific potential function. Furthermore, CaP maintains the theoretical guarantee of consistent Nash equilibria provided by the proofs for dynamic and multiagent potential-based reward shaping.

As CaP captures the same knowledge represented by difference rewards, using CaP with $DRiP$ provides no further advantages. In our experiments this combination typically performed worse than difference rewards alone. This occurs because these two signals represent the same knowledge by different methods. Combining them increases the noise of the signal whilst not increasing how informative it is.

The computational cost of CaP is equivalent to difference rewards alone. However, dependent on the complexity of the potential function used, $DRiP$ may be more computationally expensive. In all examples studied the calculation of D is more complex than the potential functions used and, therefore, the theoretical bound of complexity is unaffected.

Furthermore, it is known that difference rewards work best if the original reward function has a gradient throughout [1]. Therefore, given the close links, so do CaP and $DRiP$. Problem domains with a piecewise reward function may still benefit from CaP and $DRiP$, however, if the reward function can be gradual the positive effect is likely to be larger.

Finally, according to the proof of necessity for potential-based reward shaping [10], there must exist a problem domain for which difference rewards alter the Nash equilibria of the system. Therefore, if an application specifically requires the theoretical guarantees of potential-based reward shaping, we would recommend CaP as it benefits from the theoretical properties of both difference rewards and potential-based reward shaping whilst significantly improving performance over G alone. Alternatively, if theoretical guarantees are not a concern, we recommend $DRiP$ for its significantly better performance.

7. REFERENCES

[1] A. Agogino and K. Tumer. Unifying temporal and structural credit assignment problems. *Proc. of the 3rd Intl. Jt. Conf. on Autonomous Agents and Multiagent Systems-Volume 2*, pages 980–987, 2004.

[2] A. K. Agogino and K. Tumer. Analyzing and visualizing multiagent rewards in dynamic and stochastic environments. *Journal of Autonomous Agents and Multi-Agent Systems*, 17(2):320–338, 2008.

[3] M. Babes, E. de Cote, and M. Littman. Social reward shaping in the prisoner's dilemma. In *Proceedings of The 7th Annual International Conference on Autonomous Agents and Multiagent Systems*, volume 3, pages 1389–1392, 2008.

[4] L. Busoniu, R. Babuska, and B. De Schutter. A Comprehensive Survey of MultiAgent Reinforcement Learning. *IEEE Transactions on Systems Man & Cybernetics Part C Applications and Reviews*, 38(2):156, 2008.

[5] S. Devlin, M. Grześ, and D. Kudenko. An empirical study of potential-based reward shaping and advice in complex, multi-agent systems. *Advances in Complex Systems*, 2011.

[6] S. Devlin and D. Kudenko. Theoretical considerations of potential-based reward shaping for multi-agent systems. In *Proceedings of The 10th Annual International Conference on Autonomous Agents and Multiagent Systems*, 2011.

[7] S. Devlin and D. Kudenko. Dynamic potential-based reward shaping. In *Proceedings of The 11th Annual International Conference on Autonomous Agents and Multiagent Systems*, 2012.

[8] M. Grzes and D. Kudenko. Plan-based reward shaping for reinforcement learning. *4th International IEEE Conference on Intelligent Systems*, 2:10–22, 2008.

[9] M. Knudson and K. Tumer. Coevolution of heterogeneous multi-robot teams. *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, 2010.

[10] A. Y. Ng, D. Harada, and S. J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. *Proceedings of the 16th International Conference on Machine Learning*, pages 278–287, 1999.

[11] S. Proper and K. Tumer. Coordinating actions in congestion problems: Impact of top-down and bottom-up utilities. *Autonomous Agents and MultiAgent Systems*, 27(3):419–443, 2013.

[12] J. Randløv and P. Alstrom. Learning to drive a bicycle using reinforcement learning and shaping. *Proceedings of the 16th International Conference on Machine Learning*, pages 463–471, 1998.

[13] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[14] K. Tumer and A. Agogino. Distributed agent-based air traffic flow management. *Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems*, pages 330–337, 2007.

[15] K. Tumer and A. Agogino. Multiagent learning for black box system reward functions. *Advances in Complex Systems*, 12:493–512, 2009.

[16] M. Vasirani and S. Ossowski. A market-inspired approach to reservation-based urban road traffic management. *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems*, 2009.

[17] C. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

[18] E. Wiewiora. Potential-based shaping and q-value initialization are equivalent. *Journal of Artificial Intelligence Research*, 19(1):205–208, 2003.

[19] E. Wiewiora, G. Cottrell, and C. Elkan. Principled methods for advising reinforcement learning agents. *Proceedings of the 20th International Conference on Machine Learning*, pages 792–799, 2003.

[20] D. H. Wolpert, J. Sill, and K. Tumer. Reinforcement learning in distributed domains: Beyondteam games. In *Proc. of the 17th Int. Jt. Conf. on Artificial Intelligence*, pages 819–824, Seattle, WA, 2001.

[21] D. H. Wolpert and K. Tumer. Collective intelligence, data routing and Braess' paradox. *Journal of Artificial Intelligence Research*, 16:359–387, 2002.

[22] M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley and Sons, 2002.