Policy Invariance under Reward Transformations for Multi-Objective Reinforcement Learning

Patrick Mannion[a,b,*], Sam Devlin[c], Karl Mason[b], Jim Duggan[b], Enda Howley[b]

[a] *Department of Computer Science & Applied Physics, Galway-Mayo Institute of Technology, Ireland*
[b] *Discipline of Information Technology, National University of Ireland Galway, Ireland*
[c] *Department of Computer Science, University of York, United Kingdom*

Abstract

Reinforcement Learning (RL) is a powerful and well-studied Machine Learning paradigm, where an agent learns to improve its performance in an environment by maximising a reward signal. In multi-objective Reinforcement Learning (MORL) the reward signal is a vector, where each component represents the performance on a different objective. Reward shaping is a well-established family of techniques that have been successfully used to improve the performance and learning speed of RL agents in single-objective problems. The basic premise of reward shaping is to add an additional shaping reward to the reward naturally received from the environment, to incorporate domain knowledge and guide an agent's exploration. Potential-Based Reward Shaping (PBRS) is a specific form of reward shaping that offers additional guarantees. In this paper, we extend the theoretical guarantees of PBRS to MORL problems. Specifically, we provide theoretical proof that PBRS does not alter the true Pareto front in both single- and multi-agent MORL. We also contribute the first published empirical studies of the effect of PBRS in single- and multi-agent MORL problems.

Keywords: Reinforcement Learning, Multi-Objective, Potential-Based, Reward Shaping, Multi-Agent Systems

[*] Corresponding author
Email address: `patrick.mannion@gmit.ie` (Patrick Mannion)

The published version of this article in Neurocomputing is available from Elsevier at: `https://doi.org/10.1016/j.neucom.2017.05.090`

1. Introduction

In Reinforcement Learning (RL), an agent learns to improve its performance with experience by maximizing the return from a reward function. The majority of RL research focuses on optimising systems with respect to a single objective, despite the fact that many real world problems are inherently multi-objective in nature. Single-objective approaches seek to find a single solution to a problem, whereas in reality a system may have multiple conflicting objectives that could be optimised. Examples of multi-objective problems include stock market forecasting [1], traffic signal control [2] and load balancing in smart grids [3].

Multi-objective optimisation (MOO) approaches address the requirement to make a trade-off between competing objectives. Compromises between competing objectives can be defined using the concept of Pareto dominance [4]. The Pareto optimal or non-dominated set consists of solutions that are incomparable, where each solution in the set is not dominated by any of the others on every objective. In multi-objective Reinforcement Learning (MORL) the reward signal is a vector, where each component represents the performance on a different objective.

Reward shaping augments the reward function with additional knowledge provided by the system designer, with the goal of improving learning speed. Potential-Based Reward Shaping [5] (PBRS) is a specific form of reward shaping that provides theoretical guarantees including policy invariance in single-objective single-agent domains [5], and consistent Nash equilibria in single-objective multi-agent domains [6]. We extend the previous guarantees of PBRS with theoretical proof and empirical results showing that the set of Pareto optimal solutions remains consistent when PBRS is used in multi-objective domains, regardless of the quality of the heuristic used. This means that the increased learning speed that is a characteristic of PBRS can be leveraged in multi-objective problem domains, without any risk of altering the intended goals of the problem. As this is the first published application of PBRS to MORL problems, the empirical sections of this paper focus on the use of PBRS with a well-understood and widely-used MORL algorithm, namely scalarised Q-learning.

The contributions of this work are as follows: (1) We extend the theoretical guarantees offered by PBRS, proving that PBRS does not alter the Pareto front in single- and multi-agent MORL domains; (2) We provide empirical evidence that PBRS can be used to improve learning speed in single-agent domains without altering the set of Pareto optimal policies learned; (3) We demonstrate the effect of PBRS in a multi-agent learning problem, showing that PBRS can alter the set of Pareto dominating joint policies learned.

The remainder of this work is structured as follows: in the next section of this paper, we discuss the necessary terminology and relevant literature. We then introduce our formal proof of Pareto front invariance when applying PBRS to single- and multi-agent learning problems. Section 4 describes experimental work in a single-agent problem domain, and Section 5 presents results in a multi-agent problem domain. The final section concludes our paper with a discussion of our findings and possible future extensions to this work.

2. Background

In this section we discuss the necessary background material, including single- and multi-objective Reinforcement Learning, multi-agent learning and reward shaping.

2.1. Reinforcement Learning

Reinforcement Learning (RL) is a powerful Machine Learning paradigm, in which autonomous agents have the capability to learn through experience. An RL agent learns in an unknown environment, usually without any prior knowledge of how to behave. The agent receives a scalar reward signal r based on the outcomes of previously selected actions, which can be either negative or positive. Markov Decision Processes (MDPs) are considered the de facto standard when formalising problems involving a single agent learning sequential decision making [7]. An MDP consists of a reward function R, set of states S, set of actions A, and a transition function T [8], i.e. a tuple $< S, A, T, R >$. When in any state $s \in S$, selecting an action $a \in A$ will result in the environment entering a new state $s' \in S$ with probability $T(s, a, s') \in [0, 1]$, and give a reward $r = R(s, a, s')$.

An agent's behaviour in its environment is determined by its policy π. A policy is a mapping from states to actions that determines which action is chosen by the agent for a given state. The goal of any MDP is to find the best policy (one which gives the highest expected sum of discounted rewards) [7]. The optimal policy for an MDP is denoted $\pi*$. Designing an appropriate reward function for the environment is important, as an RL agent will attempt to maximise the return from this function, which will determine the policy learned. The value function V gives the expected discounted return for following policy π from state s onwards:

$$V^{\pi}(s) = E^{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^{k} r_{t+k+1} \mid s_t = s \right\} \tag{1}$$

where $\gamma \in [0, 1]$ is the discount factor.

RL can be classified into two paradigms: model-based (e.g. Dyna, Rmax) and model-free (e.g. Q-Learning, SARSA). In the case of model-based approaches, agents attempt to learn the transition function T, which can then be used when making action selections. By contrast, in the model-free approach knowledge of T is not a requirement. Model-free learners instead sample the underlying MDP directly in order to gain knowledge about the unknown model, in the form of value function estimates (Q values). These estimates represent the expected reward for each state action pair, which aid the agent in deciding which action is most desirable to select when in a certain state. The agent must strike a balance between exploiting known good actions and exploring the consequences of new actions in order to maximise the reward received during its lifetime. Two algorithms that are commonly used to manage the exploration exploitation trade-off are ϵ-greedy and softmax (Boltzmann) [7].

Q-Learning [9] is one of the most commonly used RL algorithms. It is a model-free learning algorithm that has been shown to converge to the optimum action-values with probability 1, so long as all actions in all states are sampled infinitely often and the action-values are represented discretely [10]. In Q-Learning, the Q values are updated according to the equation below:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \tag{2}$$

where $\alpha \in [0, 1]$ is the learning rate and $\gamma \in [0, 1]$ is the discount factor.

2.2. Multi-Agent Reinforcement Learning

In a Multi-Agent System (MAS), multiple autonomous agents act independently in the same environment. Agents in a cooperative MAS are designed to work together to achieve a system-level goal [11]. Numerous complex, real world systems have been successfully optimised using the MAS framework, including air traffic control [12], traffic signal control [2], data routing in networks [13], electricity generator scheduling [14, 15], RoboCup soccer [16] and water resource management [17].

The single-agent MDP framework becomes inadequate when we consider multiple autonomous learners acting in the same environment. Instead, the more general Stochastic Game (SG) may be used in the case of a MAS [18]. An SG is defined as a tuple $< S, A_{1...n}, T, R_{1...n} >$, where n is the number of agents, S is the set of states, A_i is the set of actions for agent i (and A is the joint action set), T is the transition function, and R_i is the reward function for agent i.

The SG looks very similar to the MDP framework, apart from the addition of multiple agents. In fact, for the case of $n = 1$ an SG then becomes an MDP. The next environment state and the rewards received by each agent depend on the joint action of all of the agents in the SG. Note also that each agent may receive a different reward for a state transition, as each agent has its own separate reward function. In an SG, the agents may all have the same goal (collaborative SG), totally opposing goals (competitive SG), or there may be elements of collaboration and competition between agents (mixed SG).

Two typical reward functions for credit assignment in Multi-Agent Reinforcement Learning (MARL) exist: local rewards unique to each agent and global rewards representative of the group's performance.

A **local reward** (L_i) is based on the utility of the part of a system that is most directly associated with agent i. Individual agents are self-interested, and each will selfishly seek to maximise its own local reward signal, often at the expense of global system performance when locally beneficial actions are in conflict with the optimal joint policy.

A **global reward** (G) provides a signal to the agents which is based on the utility of the entire system. Rewards of this form encourage all agents to act in the system's interest, with the caveat that an individual agent's contribution to the system performance is not clearly defined. All agents receive the same reward signal, regardless of whether their actions actually improved the system performance.

One of two different approaches is typically used when RL is applied to MAS: multiple individual learners or joint action learners. In the former case multiple agents deployed into an environment each use a single-agent RL algorithm, whereas joint action learners use multi-agent specific algorithms which take account of the presence of other agents. When multiple self-interested agents learn and act together in the same environment, it is generally not possible for all agents to receive the maximum possible reward. Therefore, MAS are typically designed to converge to a Nash Equilibrium [19]. While it is possible for multiple individual learners to converge to a point of equilibrium, there is no theoretical guarantee that the agents will converge to a Pareto optimal joint policy.

2.3. Reward Shaping

RL agents typically learn how to act in their environment guided by the reward signal alone. An often overlooked point when designing RL agents is that the system designer typically has some heuristic knowledge which may be beneficial to the agent during the learning process [20]. Reward shaping provides a mechanism to guide an agent's exploration of its environment, via the addition of a shaping signal to the reward signal naturally received from the environment. Heuristic knowledge provided by the system designer can be encoded into the shaping reward. The goal of this approach is to increase learning speed and/or improve the final policy learned. Generally, the reward function is modified as follows:

$$R' = R + F \tag{3}$$

where R is the original reward function, F is the additional shaping reward, and R' is the modified reward signal given to the agent.

Empirical evidence has shown that reward shaping can be a powerful tool to improve the learning speed of RL agents [21]; however, it can have unintended consequences. A classic example of reward shaping gone wrong is reported by Randløv and Alstrøm [21]. The authors designed an RL agent capable of learning to cycle a bicycle towards a goal, and used reward shaping to speed up the learning process. However, they encountered the issue of positive reward cycles due to a poorly designed shaping function. The agent discovered that it could accumulate a greater reward by cycling in circles continuously to collect the shaping reward encouraging it to stay balanced, than it could by reaching the goal state. As we discussed earlier, an RL agent will attempt to maximise its long-term reward, so the policy learned depends directly on the reward function. Thus, shaping rewards in an arbitrary fashion can modify the optimal policy and cause unintended behaviour.

Ng et al. [5] proposed Potential-Based Reward Shaping (PBRS) to deal with these shortcomings. When implementing PBRS, each possible system state has a certain potential, which allows the system designer to express a preference for an agent to reach certain system states. For example, states closer to the goal state of a problem domain could be assigned higher potentials than those that are further away. Ng et al. defined the additional shaping reward F for an agent receiving PBRS as shown in Eqn. 4 below:

$$F(s, s') = \gamma\Phi(s') - \Phi(s) \tag{4}$$

where $\Phi(s)$ is the potential function which returns the potential for a state s, and γ is the same discount factor used when updating value function estimates. PBRS has been proven not to alter the optimal policy of a single agent acting in infinite-horizon and finite-horizon MDPs [5], and thus does not suffer from the problems of arbitrary reward shaping approaches outlined above. In single agent RL, even with a poorly designed potential function, the worst case is that an agent may learn more slowly than without shaping but the final policy is unaffected.

In MARL, work by Devlin and Kudenko [6] proved that PBRS does not alter the set of Nash equilibria of an SG. Furthermore, Devlin and Kudenko [22] also proved that the potential function can be changed dynamically during learning, while still preserving the guarantees of policy invariance and consistent Nash equilibria. PBRS does not alter the set of Nash equilibria of a MAS, but it can affect the joint policy learned. It has been empirically demonstrated that agents guided by a well-designed potential function can learn at an increased rate and converge to better joint policies, when compared to agents learning without PBRS [16]. However, with an unsuitable potential function, agents learning with PBRS can converge to worse joint policies than those learning without PBRS.

2.4. Multi-Objective Reinforcement Learning

Multi-Objective Reinforcement Learning (MORL) problems may be defined using the MDP or SG framework as appropriate, in a similar manner to single-objective problems. The main difference lies in the definition of the reward function: instead of returning a single scalar value r, the reward function $\mathbf{R}$ in multi-objective domains returns a vector $\mathbf{r}$ consisting of the rewards for each individual objective $c \in C$. Therefore, a regular MDP or SG can be extended to a multi-objective MDP (MOMDP) or multi-objective SG (MOSG) by modifying the return of the reward function. It follows that the value function $\mathbf{V}^{\pi}(s)$ in multi-objective domains returns a vector

$\mathbf{v}$ whose components are the expected discounted returns for each objective when starting in state s and following a policy π:

$$\mathbf{V}^{\pi}(s) = E^{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^{k} \mathbf{r}_{t+k+1} \mid s_t = s \right\} \tag{5}$$

A policy $\pi^* \in \Pi$ (where Π is the set of possible policies) is Pareto optimal if for every $\pi \in \Pi$ either,

$$\forall_{c \in C}[\mathbf{V}^{\pi}_{c}(s_0) = \mathbf{V}^{\pi^*}_{c}(s_0)] \tag{6}$$

or, there is at least one $c \in C$ such that

$$\mathbf{V}^{\pi}_{c}(s_0) < \mathbf{V}^{\pi^*}_{c}(s_0) \tag{7}$$

where $\mathbf{V}^{\pi}_{c}(s_0)$ is the expected discounted return for objective c when starting in state s_0 and following the policy π.

That is, π^* is Pareto optimal if there exists no feasible policy π which would increase the value of one objective beyond that of π^* without causing a simultaneous decrease in the value of another objective. A policy that does not meet these criteria is dominated by another policy in Π. All policies not dominated by another are part of the non-dominated set (NDS).

The majority of MORL approaches make use of single-policy algorithms in order to learn Pareto optimal solutions. Examples of single-policy algorithms include traditional temporal difference methods such as Q-learning and SARSA. In order to apply single-policy algorithms to MORL problems, scalarisation functions are used to transform a reward vector $\mathbf{r}$ into a scalar reward signal r. An agent learns using the scalarised version of the reward vector, and selects actions as normal by comparing the expected scalarised Q values for actions in a given state (e.g. using ϵ-greedy). Linear scalarisation (+) and hypervolume scalarisation (λ) are commonly used scalarisation functions in MORL literature, and are shown in Eqns. 8 and 9 respectively:

$$r_{+} = \sum_{c \in C} \mathbf{w}_c \mathbf{r}_c \tag{8}$$

$$r_{\lambda} = \prod_{c \in C} \mathbf{r}_c \tag{9}$$

where $\mathbf{w}$ is the objective weight vector, $\mathbf{w}_c$ is the weight for objective c, r_{+} and r_{λ} are scalarised reward signals, $\mathbf{r}_c$ is the component of the reward vector **r** for objective c, and C is the set of objectives. When using linear scalarisation, altering the weights in the weight vector allows the user to express the relative importance of the objectives.

Although widely used in MORL research, linear scalarised single-policy algorithms have the fundamental limitation that they can only learn solutions located in convex regions of the Pareto front [23]. MOO approaches typically seek to produce a set of solutions that approximate the true Pareto front of the problem. In order to produce a set of Pareto optimal solutions using linear scalarised single-policy RL algorithms, researchers typically conduct a number of independent runs that use different weight vectors [24]. Thus a number of policies are learned, and compared with one another to produce an approximation of the Pareto front.

Another approach taken by researchers is to apply multi-policy RL algorithms in order to learn a set of optimal policies in a single run. Examples of such algorithms include Convex Hull Value Iteration [25], Pareto Q-Learning [26] and the work of Shelton [27]. As the focus of this work is on the effect of PBRS on the set of Pareto optimal policies, and not on overcoming the limitations of scalarised MORL, we have employed single-policy algorithms in the empirical sections of this paper. PBRS was originally designed for use with single-policy algorithms, and to the best of our knowledge has not been tested with any multi-policy algorithms to date. We leave the question of developing and testing reward shaping techniques tailored for use with multi-policy algorithms for future work.

For a more complete survey of MORL beyond the brief summary presented here, we refer the interested reader to a recent survey article by Roijers et al. [28].

3. Pareto Front Invariance under Reward Transformations

3.1. Multi-Objective Potential-Based Reward Shaping

In this section, we extend the existing theoretical guarantees of PBRS with proof that the set of Pareto optimal policies is invariant when PBRS is applied to both infinite- and finite-horizon multi-objective domains. Here we consider two ways in which PBRS could be applied to MORL problems: (a) each objective could be shaped separately, or (b) a scalar combination of the objectives could be shaped. We will first formally analyse case (a), before moving on to discuss case (b).

3.2. Proof for Shaping Each Objective Separately (Infinite-Horizon)

If each objective is shaped independently, the general form of shaping is:

$$\mathbf{R}'_c = \mathbf{R}_c + \mathbf{F}_c \tag{10}$$

where $\mathbf{R}_c$ is the reward function component for objective c, $\mathbf{F}_c$ is the shaping applied to objective c, and $\mathbf{R}'_c$ is the shaped reward function component for objective c. If the reward function is modified, it follows that the expected return from the value function is also modified, by adding a term which accounts for all of the expected shaping rewards received during an infinitely long learning trial:

$$\mathbf{V'}_c = \mathbf{V}_c + E^{\pi}\left\{\sum_{t=0}^{\infty} \gamma^t \mathbf{F}_c(s_t, s_{t+1})\right\} \tag{11}$$

where $\mathbf{V}_c$ is the value function component for objective c, and $\mathbf{V}'_c$ is the shaped value function component for objective c. Assuming that we start in state s_0 and follow a given policy π, and that the shaping rewards are of the form in Eqn. 4, the value of objective c when receiving PBRS is:

$$\begin{aligned}
\mathbf{V'}^{\pi}_c(s_0) &= \mathbf{V}^{\pi}_c(s_0) + E^{\pi}\left\{\sum_{t=0}^{\infty} \gamma^t \mathbf{F}_c(s_t, s_{t+1})\right\} \\
\mathbf{V'}^{\pi}_c(s_0) &= \mathbf{V}^{\pi}_c(s_0) + E^{\pi}\left\{\sum_{t=0}^{\infty} \gamma^t (\gamma\Phi_c(s_{t+1}) - \Phi_c(s_t))\right\} \\
\mathbf{V'}^{\pi}_c(s_0) &= \mathbf{V}^{\pi}_c(s_0) + E^{\pi}\left\{\sum_{t=0}^{\infty} \gamma^{t+1} \Phi_c(s_{t+1}) - \sum_{t=0}^{\infty} \gamma^t \Phi_c(s_t)\right\} \\
\mathbf{V'}^{\pi}_c(s_0) &= \mathbf{V}^{\pi}_c(s_0) + E^{\pi}\left\{\sum_{t=1}^{\infty} \gamma^t \Phi_c(s_t) - \sum_{t=0}^{\infty} \gamma^t \Phi_c(s_t)\right\} \\
\mathbf{V'}^{\pi}_c(s_0) &= \mathbf{V}^{\pi}_c(s_0) - \Phi_c(s_0)
\end{aligned} \tag{12}$$

All π are evaluated from the same starting state s_0, and $\mathbf{V}^{\pi}_c(s_0)$ is modified by the same amount $-\Phi_c(s_0)$ for all $\pi \in \Pi$, therefore the Pareto relation between all $\pi \in \Pi$ is invariant, and the NDS remains consistent:

$$\forall_{c \in C}\{[\mathbf{V'}^{\pi}_{c}(s_0) = \mathbf{V'}^{\pi^*}_{c}(s_0)] \iff [\mathbf{V}^{\pi}_{c}(s_0) = \mathbf{V}^{\pi^*}_{c}(s_0)]\} \tag{13}$$

And:

$$[\mathbf{V'}^{\pi}_{c}(s_0) < \mathbf{V'}^{\pi^*}_{c}(s_0)] \iff [\mathbf{V}^{\pi}_{c}(s_0) < \mathbf{V}^{\pi^*}_{c}(s_0)] \tag{14}$$

3.3. Proof for Shaping Each Objective Separately (Finite-Horizon)

In finite-horizon domains, each learning episode has a certain number of timesteps, referred to as the horizon H. The modified value function component for objective c in this case is:

$$\mathbf{V'}_c = \mathbf{V}_c + E\left\{ \sum_{t=0}^{H-1} \gamma^t \mathbf{F}_c(s_t, s_{t+1}) \right\} \tag{15}$$

Assuming again that we start in state s_0 and follow a given policy π, and that the shaping rewards are of the form in Eqn. 4, the value of objective c when receiving PBRS in a finite-horizon domain is:

$$\begin{aligned}
\mathbf{V'}^{\pi}_{c}(s_0) &= \mathbf{V}^{\pi}_{c}(s_0) + E^{\pi}\left\{ \sum_{t=0}^{H-1} \gamma^t \mathbf{F_c}(s_t, s_{t+1}) \right\} \\
\mathbf{V'}^{\pi}_{c}(s_0) &= \mathbf{V}^{\pi}_{c}(s_0) + E^{\pi}\left\{ \sum_{t=0}^{H-1} \gamma^t (\gamma \Phi_c(s_{t+1}) - \Phi_c(s_t)) \right\} \\
\mathbf{V'}^{\pi}_{c}(s_0) &= \mathbf{V}^{\pi}_{c}(s_0) + E^{\pi}\left\{ \sum_{t=0}^{H-1} \gamma^{t+1} \Phi_c(s_{t+1}) - \sum_{t=0}^{H-1} \gamma^t \Phi_c(s_t) \right\} \\
\mathbf{V'}^{\pi}_{c}(s_0) &= \mathbf{V}^{\pi}_{c}(s_0) + E^{\pi}\left\{ \sum_{t=1}^{H} \gamma^t \Phi_c(s_t) - \sum_{t=0}^{H-1} \gamma^t \Phi_c(s_t) \right\} \\
\mathbf{V'}^{\pi}_{c}(s_0) &= \mathbf{V}^{\pi}_{c}(s_0) + E^{\pi}\left\{ \gamma^H \Phi_c(s_H) \right\} - \Phi_c(s_0) \\
\mathbf{V'}^{\pi}_{c}(s_0) &= \mathbf{V}^{\pi}_{c}(s_0) + \gamma^H \sum_{s \in S} \left\{ Pr^{\pi}(s_H = s) \Phi_c(s) \right\} - \Phi_c(s_0)
\end{aligned} \tag{16}$$

This result is similar to that for the infinite-horizon case, apart from the additional term $+\gamma^H \sum_{s \in S} \left\{ Pr^{\pi}(s_H = s) \Phi_c(s) \right\}$, which represents the expected shaping reward for the final state s_H. The probability of being in a certain state s at time H depends on the policy π being evaluated, which in turn determines what shaping reward will be received. This additional term could potentially cause changes in the Pareto relation between policies.

If $\gamma < 1.0$ and the value of H is sufficiently large, the value of the term $+\gamma^H \sum_{s \in S} \left\{ Pr^{\pi}(s_H = s)\Phi_c(s) \right\}$ would become insignificant. However, this term can be eliminated completely by ensuring that the final potential equals zero[2], i.e. $\Phi_c(s_H) = 0$. This requirement can easily be taken into account when designing potential functions for episodic problems where the terminal states are known, by assigning a zero potential to all terminal states.

In cases where the terminal states are not known beforehand, any state after H timesteps is a terminal state, and policy invariance may be preserved by dynamically changing the potential of the final state experienced to 0. This means that the potential function is time-variant, since it must be different for timestep H compared to other timesteps.

A more robust method to preserve the theoretical guarantees of PBRS in finite-horizon problems is to implement an additional absorbing state s_{abs} with zero potential (i.e. $\Phi(s_{abs}) = 0$). Upon reaching the terminal state, all agents select actions as normal, and are transitioned to the absorbing state. No reward is received from the environment for this transition, but agents do receive the shaping reward as calculated by Eqn. 4. As all possible policies now terminate in the absorbing state, $Pr^{\pi}(s_H = s_{abs}) = 1.0$, meaning that the additional term $+\gamma^H \sum_{s \in S} \left\{ Pr^{\pi}(s_H = s)\Phi_c(s) \right\}$ is guaranteed to sum to zero. Assuming that the final potential is set to zero using any of these methods, the form of $\mathbf{V'}^{\pi}_{c}(s_0)$ is exactly the same as in the infinite horizon case, and therefore the NDS remains consistent when PBRS is applied to finite-horizon MORL domains.

3.4. Discussion

Following the same methods as the proofs above, it can easily be demonstrated that the guarantees of Pareto front invariance also hold when PBRS is used to shape a scalarised combination of objectives, if the objective specific terms $\mathbf{V'}^{\pi}_{c}(s_0)$ and $\mathbf{V}^{\pi}_{c}(s_0)$ are replaced with scalarised versions, and $\mathbf{F_c}$ is replaced by a single shaping F.

[2] While this article was under review, Grześ [29] independently demonstrated that a zero-valued final potential will ensure policy invariance when PBRS is applied to finite-horizon domains.

PBRS has seen almost no applications in MORL domains thus far, and therefore questions remain as to how best to apply it. Previous work by Brys et al. [30] used PBRS to add additional psuedo objectives to a single-objective problem in order to speed up learning. However, their work is not an evaluation of PBRS in a true multi-objective problem. Besides the work of Brys et al. we are unaware of any prior attempts to use PBRS with MORL, apart from the present article. Therefore, the area of reward shaping for MORL merits substantial further study.

One remaining question is whether PBRS should be applied by (a) shaping each objective separately or (b) shaping a scalarised combination. Further investigation is required into this topic, although the efficacy of one option versus another is likely to be domain specific, and will also depend on the nature of the heuristic information that is available. We expect that option (a) may be more useful in domains where there is a low degree of correlation between objectives, whereas option (b) may be more suited to domains with strongly correlated objectives. Shaping scalarised combinations may also be preferable when applying PBRS to complex domains, where designing individual potential functions to shape each objective is impractical or unintuitive. In the empirical sections of this paper, we have used both options (a) and (b) in our single-agent study, and option (b) in our multi-agent study. These studies will examine the effects of PBRS in MORL domains, and provide supporting evidence for the proofs in this section. While we focus on scalarised Q-learning for the empirical sections of this paper, we note that option (a) is also suitable for use in combination with MORL algorithms that do not use scalarisation, as each component of the reward vector is shaped independently.

Following from our proofs and the proof of policy invariance by Ng. et al [5], we expect to demonstrate that agents learning with and without PBRS in single-agent MORL domains will learn the exact same set of Pareto optimal policies. As we have used single-agent domains where the true Pareto front is known, the performance of the agents can easily be judged by comparing how quickly all Pareto optimal policies are learned.

In the multi-agent case, we expect that agents learning with and without PBRS may converge to different Pareto optimal joint policies. This follows from the proof of consistent Nash Equilibria in SGs by Devlin and Kudenko [6]. While applying PBRS does not alter the true Pareto front of a MOSG, it may alter the Nash equilibrium reached by the agents, and therefore different policies could be learned compared to agents learning without PBRS.

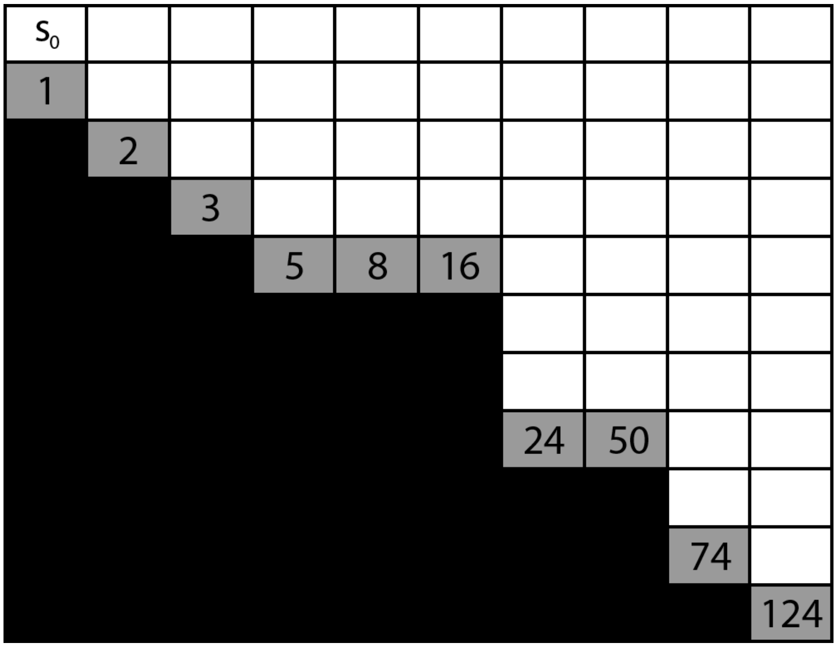

Figure 1: The Deep Sea Treasure environment

However, the set of possible policies that could be learned and their Pareto relation to one another remains consistent when PBRS is applied. Our multi-agent study is conducted in a realistic multi-objective electricity generator scheduling domain, where the true Pareto front is unknown.

4. Deep Sea Treasure

4.1. Problem Description

The Deep Sea Treasure (DST) environment was proposed by Vamplew et al. [24] as a benchmark problem for single-agent MORL algorithms. This is a useful benchmark problem, as the true Pareto front is known. Thus, it will allow us to accurately evaluate the effect of PBRS on the set of Pareto optimal policies that are learned in single-agent MORL domains.

The DST environment, shown in Fig. 1, consists of 10 rows and 11 columns. An agent controls a submarine, which searches for undersea treasures. There are 10 treasure locations in all, and the agent begins each episode in the top left state (labelled s_0 in Fig. 1). An episode ends after 1000 actions, or when the agent reaches a treasure location. The agent's state

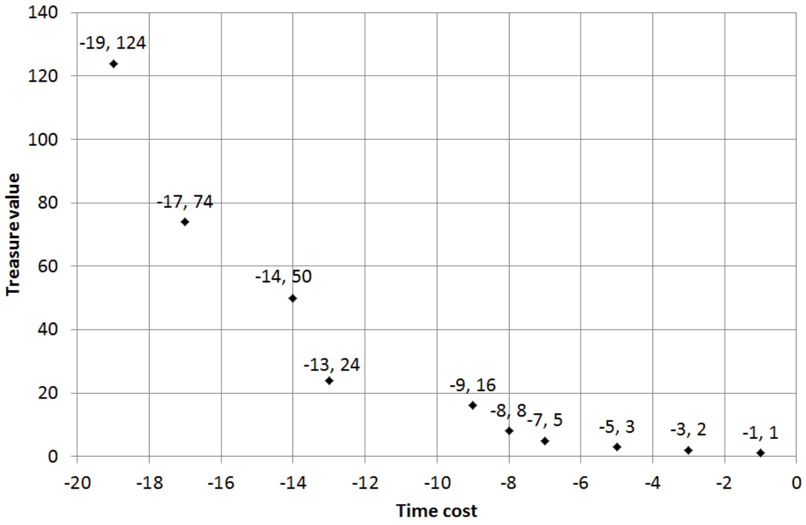

Figure 2: The Pareto front for the Deep Sea Treasure environment

is defined as its current position on the grid, and the actions available correspond to moving one square in one of the four cardinal directions. Actions which would cause the agent to leave the grid leave its position unchanged.

There are two objectives in this domain: to minimise the time taken to reach a treasure, and to maximise the reward received when a treasure is reached. After each action selection, the agent receives a reward vector with two elements. The first element is the time reward, which is -1 for all turns. The second element is the treasure reward, which is the value for the corresponding cell in Fig. 1 if a treasure is reached, and zero for all other turns. The Pareto front for this problem consists of 10 elements, with a non-dominated policy corresponding to each of the 10 treasure locations. The Pareto front is plotted in Fig. 2 and is globally concave, with local concavities at the second, fourth and sixth points from the left [24].

We also use a modified version of the DST domain called the Convex Deep Sea Treasure (CDST) environment. In the CDST, the values for the treasure rewards have been altered to create a Pareto front that is globally convex. This modification means that scalarised Q-learning can learn all of the Pareto optimal policies in the CDST. The CDST and its Pareto front are shown in Figs. 3 and 4 respectively.

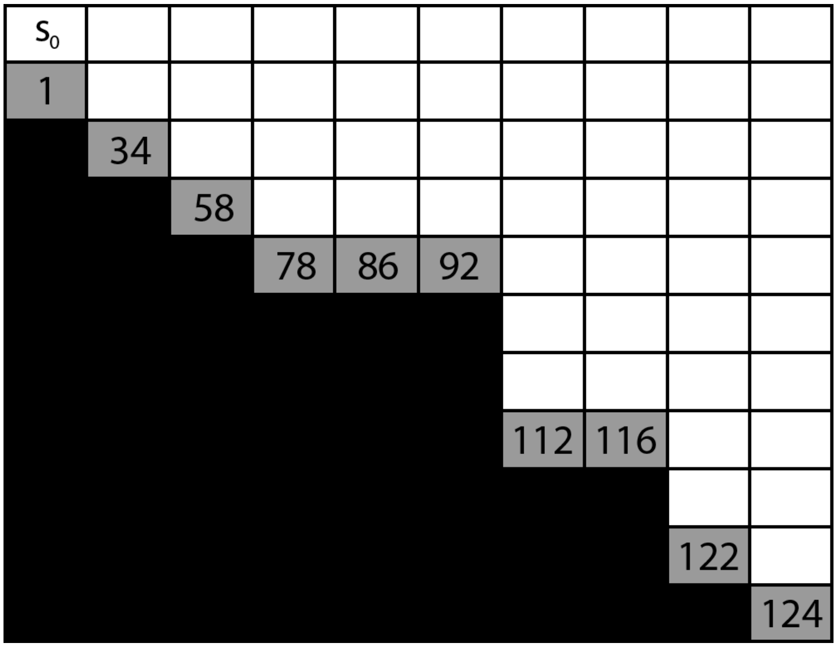

Figure 3: The Convex Deep Sea Treasure environment

4.2. Experimental Procedure

For scalarised Q-learning, the reward vector received at each timestep is converted into a single scalar value using linear scalarisation. As we noted earlier, a shortcoming of scalarised Q-learning is its inability to sample policies in concave regions of the Pareto front. Thus we expect that scalarised Q-learning will not learn all Pareto-dominating policies in the DST environment, but this domain will nevertheless prove useful when evaluating the effects of PBRS. We expect that scalarised Q-learning will learn all policies in the CDST due to the globally convex Pareto front. We ran this algorithm with 100 different weight vectors, where the continuous range $[0, 1]$ is uniformly discretised with step size $\frac{1}{100-1}$, so that a number of different policies would be learned. At the start of each run, the action values were optimistically initialised to $[0, 125]$ scalarised with the appropriate weight vector for non-terminal states, and to 0 for terminal states. The learning parameters used were as follows: $\alpha = 0.1$, $\gamma = 1$. The action value initialisation method, and values for α and γ are the same as were used by Vamplew et al. [24] in their empirical study on the DST domain. The exploration rate, ϵ, was set to 0.998^e, where e is the episode number. Vamplew et al. [24] used a fixed

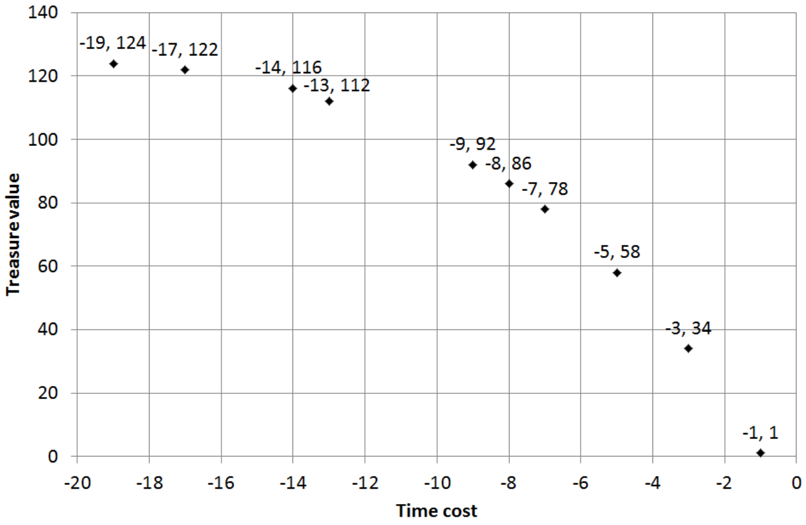

Figure 4: The Pareto front for the Convex Deep Sea Treasure environment

$\epsilon = 0.1$, but instead we chose a decaying exploration rate in order to allow for sufficient exploration in the early stages of learning, and to maximise exploitation of the learned policies towards the end of the training period.

To test the effect of PBRS in this problem domain, we use two different types of potential functions: good and poor. In these potential functions each entry corresponds to a cell in the DST environment. Entries in bold denote treasure locations, and blank entries denote unreachable cells.

The good heuristics for the DST and CDST are shown in Eqns. 17 and 18 respectively. These heuristics were chosen to illustrate the increased learning speed that is possible when good domain knowledge is available, and are used to shape the treasure component of the reward vector that is received. Agents are encouraged to explore along the Pareto front when using the good heuristics, with states close to high valued treasures assigned the highest potentials.

Eqn. 19 is an example of a poorly-designed potential function, and is used to shape the scalarised combination of the reward vector that is received in the DST and the CDST. This potential function encourages an agent towards the upper right corner of the domain, far away from any treasure locations. We expect that this potential function will reduce the learning speed of any agents that receive it.

$$\Phi_{Good}(s) = \begin{bmatrix} 6 & 12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 19 & 25 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & \mathbf{2} & 31 & 37 & 2 & 2 & 2 & 2 & 2 & 2 \\ & & \mathbf{3} & 43 & 50 & 56 & 62 & 3 & 3 & 3 \\ & & & \mathbf{5} & \mathbf{8} & \mathbf{16} & 78 & 4 & 4 & 4 \\ & & & & & & 74 & 5 & 5 & 5 \\ & & & & & & 81 & 87 & 93 & 6 \\ & & & & & & \mathbf{24} & \mathbf{50} & 99 & 7 \\ & & & & & & & & 105 & 112 \\ & & & & & & & & \mathbf{74} & 118 \\ & & & & & & & & & \mathbf{124} \end{bmatrix} \tag{17}$$

$$\Phi_{Good}(s) = \begin{bmatrix} 6 & 12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 19 & 25 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & \mathbf{34} & 31 & 37 & 2 & 2 & 2 & 2 & 2 & 2 \\ & & \mathbf{58} & 43 & 50 & 56 & 62 & 3 & 3 & 3 \\ & & & \mathbf{78} & \mathbf{86} & \mathbf{92} & 78 & 4 & 4 & 4 \\ & & & & & & 74 & 5 & 5 & 5 \\ & & & & & & 81 & 87 & 93 & 6 \\ & & & & & & \mathbf{112} & \mathbf{116} & 99 & 7 \\ & & & & & & & & 105 & 112 \\ & & & & & & & & \mathbf{122} & 118 \\ & & & & & & & & & \mathbf{124} \end{bmatrix} \tag{18}$$

$$\Phi_{Poor}(s) = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \mathbf{0} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ & \mathbf{0} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ & & \mathbf{0} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ & & & \mathbf{0} & \mathbf{0} & \mathbf{0} & 3 & 4 & 5 & 6 \\ & & & & & & 2 & 3 & 4 & 5 \\ & & & & & & 1 & 2 & 3 & 4 \\ & & & & & & \mathbf{0} & \mathbf{0} & 2 & 3 \\ & & & & & & & & 1 & 2 \\ & & & & & & & & \mathbf{0} & 1 \\ & & & & & & & & & \mathbf{0} \end{bmatrix} \tag{19}$$

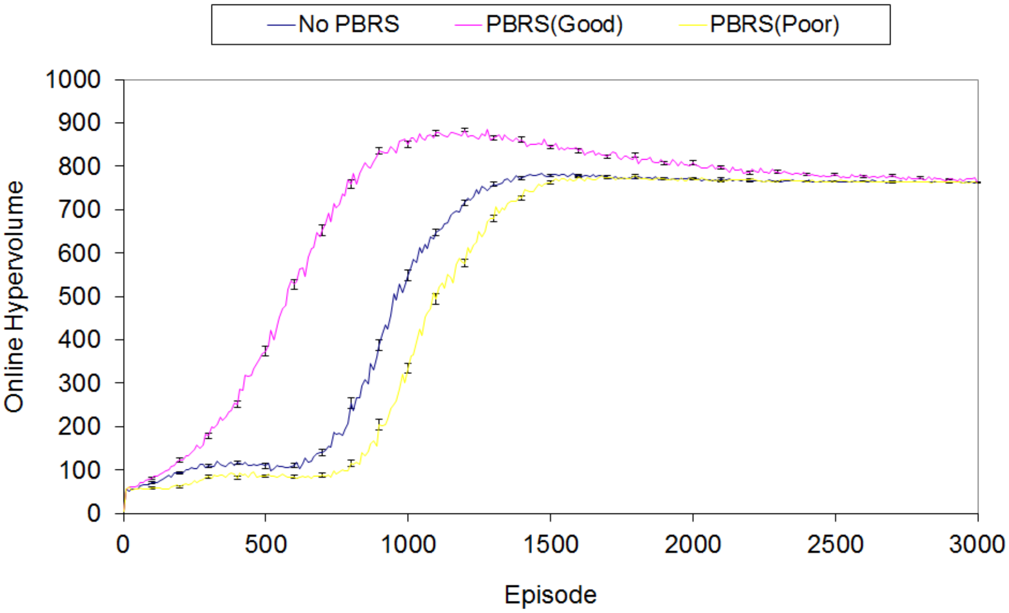

Figure 5: Online hypervolume of non-dominated policies learned in the DST environment

4.3. Results

All plots 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. Error bars are included on all plots at 100 episode intervals. The plots show the average performance across the 30 statistical runs that were conducted at 100 episode intervals.

We will first discuss the results from the DST domain. Figs. 5 and 6 show the online and offline hypervolumes of the non-dominated policies learned by each approach over the training period. The online hypervolume is calculated using the accumulated rewards during learning, while the offline hypervolume is calculated using the accumulated rewards received by greedily evaluating the current policy. In both cases we begin by first removing the Pareto-dominated accumulated reward vectors, and the remaining Pareto optimal vectors are then used to calculate the hypervolumes. All hypervolumes are calculated using a reference point of [-25,0]. The hypervolume of the true Pareto front for the DST domain is 1155 using the reference point specified. A summary of the final hypervolumes and final policies learned in the DST

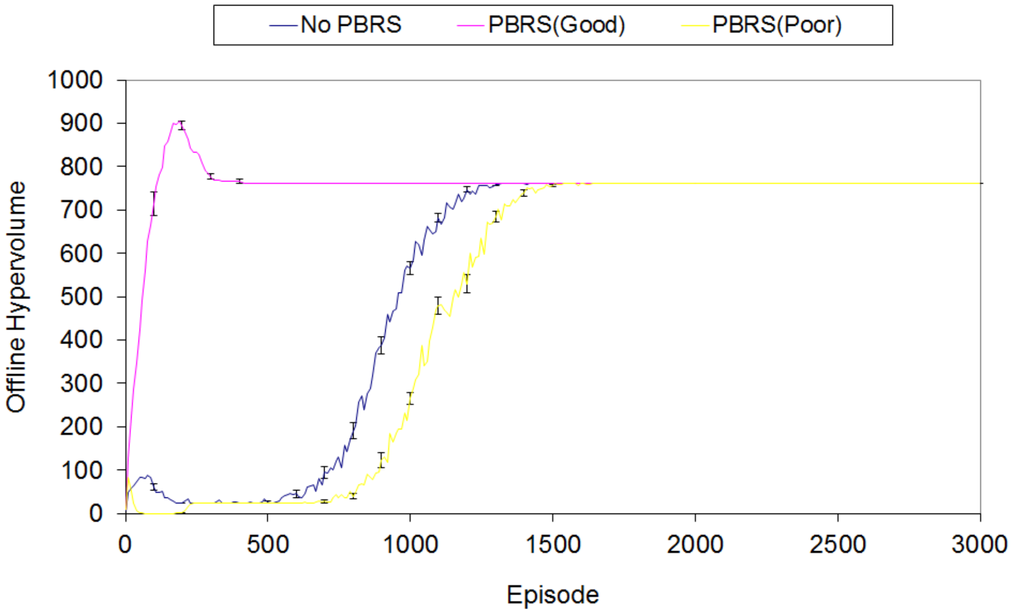

Figure 6: Offline hypervolume of non-dominated policies learned in the DST environment

domain is provided in Table 1.

As expected, scalarised Q-learning falls far short of the maximum possible hypervolume, as it only learns two Pareto optimal policies upon convergence, corresponding to the extreme solutions at either end of the Pareto front, $[-1, 1]$ and $[-19, 124]$. This results in a hypervolume of 762, and our experience matches with that of Vamplew et al. [24] and Van Moffaert and Nowé [26], who also report that scalarised Q-learning can only learn the policies on the convex portion of the Pareto front in this domain.

Figs. 5 and 6 show a considerable improvement in learning speed when a good PBRS heuristic is added to scalarised Q-learning. We note also that the hypervolume actually increases beyond its final value when PBRS is added. This shows that the addition of PBRS causes more of the Pareto optimal policies to be sampled while learning; however, upon convergence it reaches the exact same hypervolume as that reached without PBRS. The increased learning speed that is a characteristic of PBRS is displayed here, but the set of policies learned upon convergence has not been altered. The poor heuristic exhibits reduced learning speed as expected, but it still learns the same final policies as agents learning without PBRS and with a good PBRS heuristic.

Table 1: Pareto dominating policies for DST at the end of the training period

	Hypervolume	Pareto dominating policies
True Pareto front	1155	[-1,1],[-3,2],[-5,3],[-7,5],[-8,8],[-9,16], [-13,24],[-14,50],[-17,74],[-19,124]
No PBRS	762	[-1,1],[-19,124]
PBRS (Good)	762	[-1,1],[-19,124]
PBRS (Poor)	762	[-1,1],[-19,124]

Table 2: Pareto dominating policies for CDST at the end of the training period

	Hypervolume	Pareto dominating policies
True Pareto front	2166	[-1,1],[-3,34],[-5,58],[-7,78],[-8,86],[-9,92], [-13,112],[-14,116],[-17,122],[-19,124]
No PBRS	2166	[-1,1],[-3,34],[-5,58],[-7,78],[-8,86],[-9,92], [-13,112],[-14,116],[-17,122],[-19,124]
PBRS (Good)	2166	[-1,1],[-3,34],[-5,58],[-7,78],[-8,86],[-9,92], [-13,112],[-14,116],[-17,122],[-19,124]
PBRS (Poor)	2166	[-1,1],[-3,34],[-5,58],[-7,78],[-8,86],[-9,92], [-13,112],[-14,116],[-17,122],[-19,124]

The results from the CDST domain are plotted in Figs. 7 and 8, and the final policies learned and final hypervolumes are presented in Table 2. In this domain, the maximum hypervolume is 2166 when all 10 Pareto optimal policies are learned, calculated as before using a reference point of [-25,0]. The basic scalarised Q-Learning agent is capable of learning all 10 Pareto optimal policies in the CDST, as evidenced by the final hypervolume, which reached the maximum value of 2166. When a good PBRS heuristic is added, the maximum hypervolume of 2166 is reached more quickly. Here PBRS has improved the learning speed, without altering the set of Pareto optimal policies learned for the problem. When using a poor PBRS heuristic, the learning speed is reduced, but the agent still converges to the maximum hypervolume, and successfully learns all 10 Pareto optimal policies.

The empirical results in this section demonstrate that the increased learning speed that is characteristic of PBRS can be leveraged in single-agent MORL problem domains, without altering the Pareto optimal policies learned

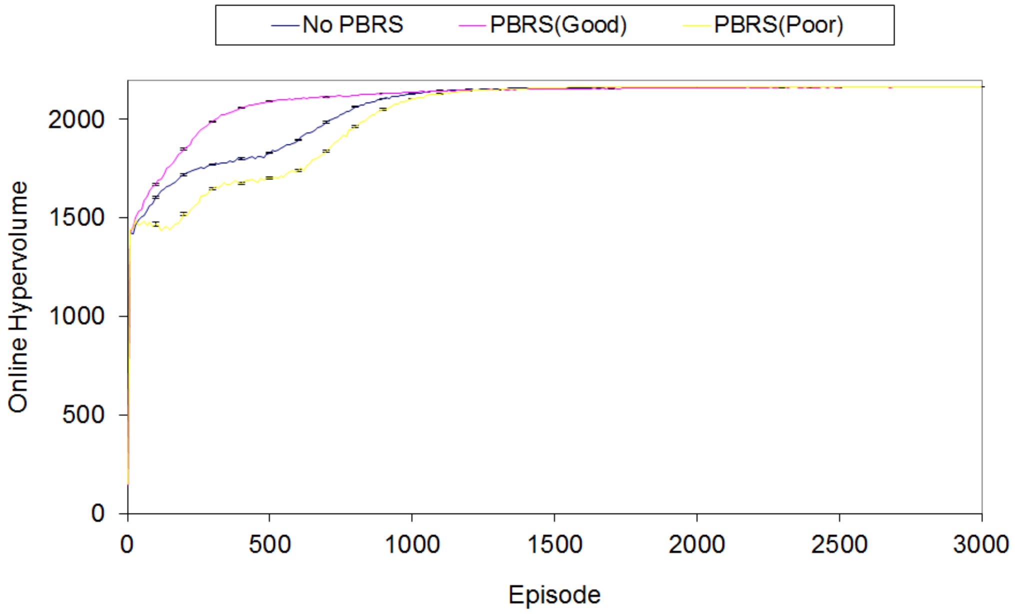

Figure 7: Online hypervolume of non-dominated policies learned in the CDST environment

upon convergence. An interesting finding is the effect of PBRS on an agent's exploration when a suitable heuristic is used. In the early stages of learning in the DST, scalarised Q-learning with a good heuristic achieves a high hypervolume initially, and samples Pareto optimal policies that are in the concave region of the Pareto front. However, upon convergence scalarised Q-learning with a good heuristic reaches the same hypervolume as scalarised Q-learning without PBRS. In both the DST and CDST, it is evident that a well-designed PBRS heuristic can improve learning speed, and conversely that a poor heuristic can be harmful to an agent's initial performance. However, in all cases the set of Pareto optimal policies learned with and without PBRS is consistent in both problem domains, regardless of the quality of the heuristic used. Thus, we have demonstrated that the two main strengths of PBRS for RL in conventional MDPs (i.e. improved learning speed and policy invariance) can also be leveraged successfully with RL in MOMDPs.

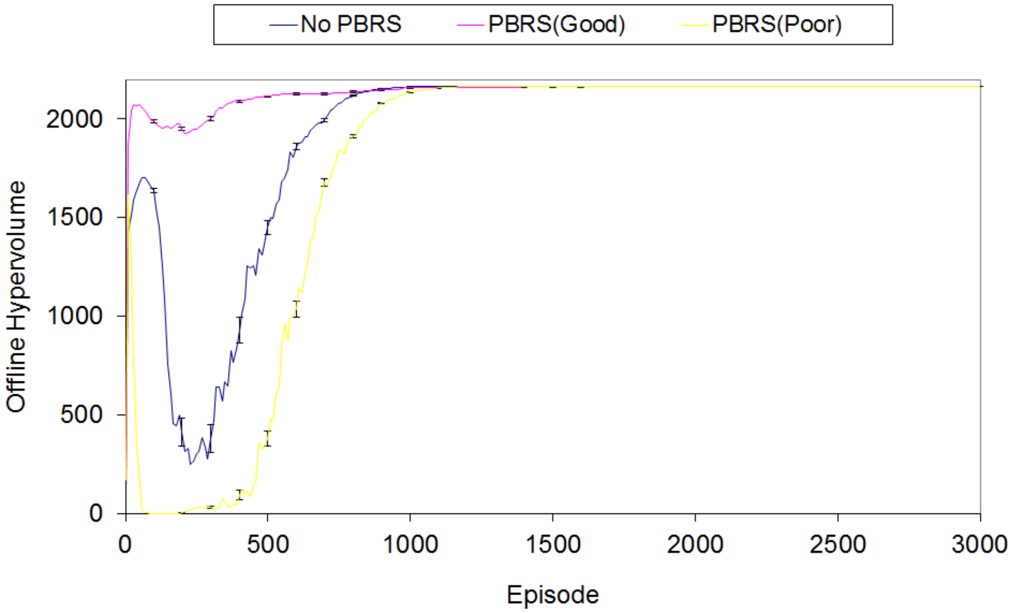

Figure 8: Offline hypervolume of non-dominated policies learned in the CDST environment

5. Dynamic Economic Emissions Dispatch

5.1. Problem Description

In the Dynamic Economic Emissions Dispatch (DEED) problem, a number of electricity generators must be scheduled to meet a specified customer demand over a period of time, while minimising the conflicting objectives of fuel cost and emissions. Generator scheduling is a complex task due to many different factors, including unpredictable fluctuations in demand, power loss within the transmission lines, and varying efficiency levels, power limits and ramp limits among generators in the same plant [31]. Approaches such as Genetic Algorithms [31], Particle Swarm Optimisation [32] and MARL [14, 15] have previously been applied to generator scheduling for the DEED problem. The version of the problem which we analyse here was originally proposed by Basu [31]. Mannion et al. [14, 15] reformulated Basu's version of the DEED problem as a MOSG in order to allow the application of MARL. We will use the DEED MOSG to evaluate the effect of using PBRS in multi-agent MORL problem domains.

In the DEED MOSG, each agent $i \in \{2, ..., N\}$ controls the power output of a generator $n \in N$ at each hour $m \in M$, and the first generator is a slack generator. The local cost $f_c^L(n,m)$ and emissions $f_e^L(n,m)$ terms for generator n over hour m are calculated as:

$$f_c^L(n,m) = a_n + b_n P_{nm} + c_n(P_{nm})^2 + |d_n sin\{e_n(P_n^{min} - P_{nm})\}| \tag{20}$$

$$f_e^L(n,m) = E(\alpha_n + \beta_n P_{nm} + \gamma_n(P_{nm})^2 + \eta \exp \delta P_{nm}) \tag{21}$$

where a_n, b_n, c_n, d_n and e_n are the cost coefficients for each generator, α_n, β_n, γ_n, η_n and δ_n are the emission coefficients for each generator, P_{nm} is the power output from generator n at time m, P_n^{min} is the minimum permissible power output of generator n, and $E = 10$ is the emissions scaling factor.

The global cost and emissions for hour m may then be calculated as the summation of $f_c^L(n,m)$ and $f_e^L(n,m)$ respectively over the $N = 10$ generators in the system:

$$f_c^G(m) = \sum_{n=1}^{N} f_c^L(n,m) \tag{22}$$

$$f_e^G(m) = \sum_{n=1}^{N} f_e^L(n,m) \tag{23}$$

The total power output in a given hour must be equal to the sum of the power demand and transmission losses:

$$\sum_{n=1}^{N} P_{nm} = P_{Dm} + P_{Lm} \qquad \forall m \in M \tag{24}$$

where P_{Dm} is the power demand over hour m and P_{Lm} is the transmission loss over hour m.

There are two inequality constraints which any potential solutions are subject to: the operating limits and the ramp limits for each power generator in the station. The operating limits specify the minimum and maximum possible power output of a generator, while the ramp limits determine the maximum allowed increase or decrease in the power output of a generator from one hour to the next.

$$P_n^{min} \leq P_{nm} \leq P_n^{max} \tag{25}$$

$$P_{nm} - P_{n(m-1)} \leq UR_n \tag{26a}$$

$$P_{n(m-1)} - P_{nm} \leq DR_n \tag{26b}$$

where P_n^{min} and P_n^{max} refer to the minimum and maximum power output of each generator, P_{nm} is the power output for $n \in N$ and $m \in M$, and UR_n and DR_n are the ramp up and ramp down limits for generator n.

In order to satisfy the equality constraint described by Eqn. 24, the first generator $n = 1$ is a slack generator. The power outputs of the other 9 generators are set directly by individual agents, and the slack generator makes up any shortfall in the combined power output. The settings for the slack generator are therefore dependant variables during the optimisation process, while the outputs of the other $N - 1$ generators are independent variables. The power output of the slack generator for a single hour, P_{1m}, may be calculated by rearranging Eqn. 24:

$$P_{1m} = P_{Dm} + P_{Lm} - \sum_{n=2}^{N} P_{nm} \tag{27}$$

The loss in the transmission lines between generators, P_{Lm}, over hour m is calculated as follows:

$$P_{Lm} = \sum_{n=2}^{N}\sum_{j=2}^{N} P_{nm}B_{nj}P_{jm} + 2P_{1m}(\sum_{n=2}^{N} B_{1n}P_{nm}) + B_{11}(P_{1m})^2 \tag{28}$$

where B is the matrix of transmission line loss coefficients [31]. Combining Eqn. 27 with Eqn. 28 produces the following quadratic equation:

$$\begin{gathered} 0 = B_{11}(P_{1m})^2 + (2\sum_{n=2}^{N} B_{1n}P_{nm} - 1)P_{1m} + \\ (P_{Dm} + \sum_{n=2}^{N}\sum_{j=2}^{N} P_{nm}B_{nj}P_{nm} - \sum_{n=2}^{N} P_{nm}) \end{gathered} \tag{29}$$

Solving this quadratic equation using basic algebra will give the reactive power of the slack generator, P_{1m}, at each hour.

The next environmental state for each agent i is defined as a vector containing the change in power demand ΔP_D since the previous timestep, and the previous power output of the generator n, P_{nm}. The change in power demand at time m is calculated as:

$$\Delta P_{Dm} = P_{Dm} - P_{D(m-1)} \tag{30}$$

Therefore the state vector for agent i (controlling generator n) at time m is:

$$s_{im} = [\Delta P_{Dm}, P_{n(m-1)}] \tag{31}$$

The action chosen by agent i at each timestep determines the power output of the generator n under its control. However, the power output constraints in Eqn. 25 must be satisfied for each generator. Therefore the possible action set for agent i consists of:

$$A_i = \{P_n^{min}, ..., P_n^{max}\} \tag{32}$$

At any hour m, when the ramp limits in Eqns. 26a and 26b are imposed, an agent's action set is constrained to:

$$A_{im} = \{P_{n(m-1)} - UR_n \geq P_n^{min}, ..., P_{n(m-1)} - UR_n \leq P_n^{max}\} \tag{33}$$

In order to discretise this continuous action space, we use an abstraction A^* of the action space, where each agent has a set of 101 possible actions $A^* = \{0, 1, ..., 99, 100\}$. Each action represents a different percentage value of the operating range of the generator, so generators with wider operating ranges have larger increments. The power output from generator n for action a_i^* is calculated as:

$$P_n = P_n^{min} + a_i^* \left(\frac{P_n^{max} - P_n^{min}}{100} \right) \qquad i = n \tag{34}$$

The power output selected by an agent is still subject to the ramp limits, as per Eqns. 26a, 26b and 33, so a^* selections that would violate these limits are not allowed. Agents select actions using the ϵ-greedy strategy, where a random action is selected with probability ϵ, and the highest valued action is selected with probability $1 - \epsilon$.

We must also consider how to handle the power limits and ramp limits of the slack generator, $n = 1$. We develop a global penalty function f_p^G based on the static penalty method [33] to capture violations of these constraints:

$$f_p^G(m) = \sum_{v=1}^{V} C(|h_v + 1|\delta_v) \tag{35}$$

$$h_1 = \begin{cases} P_{1m} - P_1^{max} & \text{if } P_{1m} > P_1^{max} \\ P_1^{min} - P_{1m} & \text{if } P_{1m} < P_1^{min} \\ 0 & \text{otherwise} \end{cases} \tag{36}$$

$$h_2 = \begin{cases} (P_{1m} - P_{1(m-1)}) - UR_1 & \text{if } (P_{1m} - P_{1(m-1)}) > UR_1 \\ (P_{1m} - P_{1(m-1)}) + DR_1 & \text{if } (P_{1m} - P_{1(m-1)}) < -DR_1 \\ 0 & \text{otherwise} \end{cases} \tag{37}$$

where $V = 2$ is the number of constraints handled using this method (one possible violation each for slack generator power and ramp limits over hour m), $C = 10E6$ is the violation constant, h_v is the violation of each constraint, and $\delta_v = 0$ if there is no violation in a given constraint and $\delta_v = 1$ if the constraint is violated. The violation constant $C = 10E4$ was selected so that the output of the penalty function will have a similar magnitude to that of the cost function f_c^G. The penalty function is an additional objective that must be optimised, in addition to cost and emissions.

The global cost, emissions and penalty functions are combined into a single global reward signal using linear scalarisation:

$$G(m) = -\left[\mathbf{w}_c f_c^G(m) + \mathbf{w}_e f_e^G(m) + \mathbf{w}_p f_p^G(m)\right] \tag{38}$$

Note that the return from this reward function is negative, as the objectives in the DEED domain must be minimised. When applying PBRS, the scalarised return from G is shaped as per Eqns. 3 and 4.

5.2. Experimental Procedure (Finite-Horizon)

We apply multiple individual Q-learners with ϵ-greedy exploration to the DEED MOSG, learning using the scalarised global reward function with and without PBRS. We ran these algorithms with 5 different weight vectors,

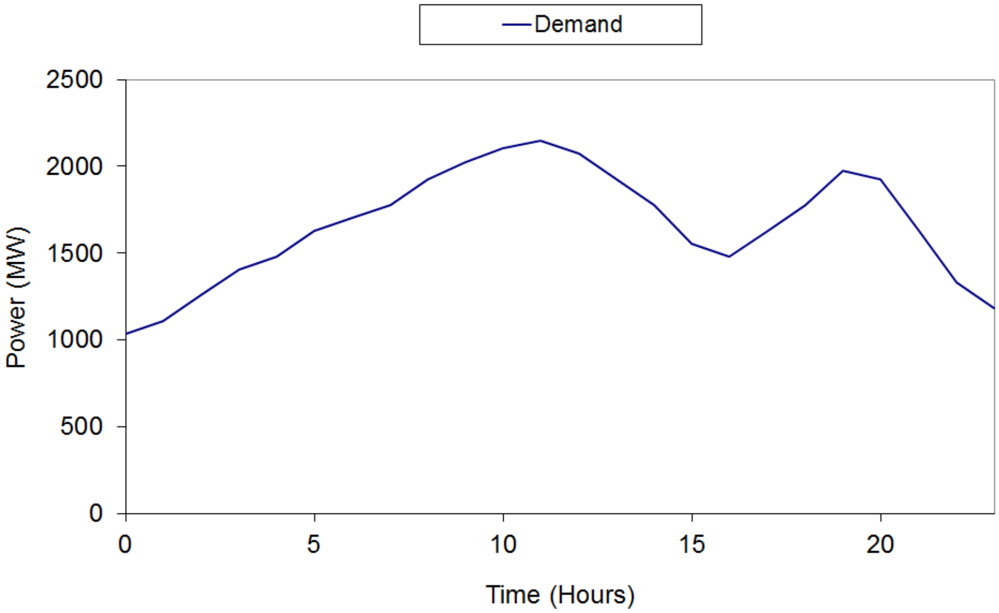

Figure 9: 24 hour power demand for the DEED domain

varying the weights of the cost and emissions objectives so that a variety of solutions would be learned. The weight vectors used were: [0.200,0.300,0.5], [0.225,0.275,0.500], [0.250,0.0.250,0.500], [0.275,0.225,0.500], [0.300,0.200,0.500]. The best non-dominated policies were then combined to calculate hypervolumes for a single run. Agents learned for $20,000$ episodes, and the duration of each episode was $M = 24$ hours. At the beginning of each run, all action values in all states were initialised to 0. The learning parameters for all agents were set as follows: $\alpha = 0.10$, $\gamma = 0.75$, $\epsilon = 0.05$. These are the same values that were used by Mannion et al. [15] in their work on formulating the DEED MOSG. The customer power demand profile used in our experiments is shown in Fig. 9. All values for cost coefficients, emission coefficients, ramp limits, generator capacity limits, power demands and transmission line loss coefficients can be found in the work of Basu [31].

To test the effect of PBRS in this problem domain, we use three different heuristics:

- **High:** All agents are encouraged to select high power values. This heuristic is expected to quickly reduce the cost and emissions values during learning, and lead to good solutions.

$$\Phi_{High}(i,m) = -\left(100+\left(\frac{P_{n,m-1} - P_n^{min}}{P_n^{max} - P_n^{min}}\times 100\right)\right)\times 10^6 \qquad i = n \quad (39)$$

- **Low:** All agents are encouraged to select low power values. This heuristic is also expected to increase learning speed, although encouraging agents to select low power values will increase loading on the slack generator, which may negatively affect the running costs and emissions produced.

$$\Phi_{Low}(i,m) = -\left(100-\left(\frac{P_{n,m-1} - P_n^{min}}{P_n^{max} - P_n^{min}}\times 100\right)\right)\times 10^6 \qquad i = n \quad (40)$$

- **Mixed:** This heuristic encourages a mixture of the two different behaviours above. The agents $n = 2$ to $n = 5$ are encouraged to select low power values using the low heuristic, whereas the agents from $i = 6$ to $i = 10$ are encouraged to select high power values using the high heuristic. The design of this mixed heuristic is based on the intuition that it may be beneficial to keep some generators in a group working at close to full power continuously to satisfy the baseline power demand, while the rest of the generators will only increase their power output during peaks of high demand.

$$\Phi_{Mixed}(i,m) = \begin{cases} \Phi_{Low}(i,m) & \text{if } i \leq 5 \\ \Phi_{High}(i,m) & \text{otherwise} \end{cases} \quad (41)$$

5.3. Results (Finite-Horizon)

The online hypervolume of cost and emissions for best non-dominated policies learned by each approach in the DEED domain is shown in Fig. 10. This plot includes 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. Error bars are included on this plot at 1000 episode intervals. The plot shows the average performance across the 30 statistical runs that were conducted at 100 episode intervals. Fig. 11 shows the Pareto fronts for each approach. These fronts are comprised of the best non-dominated policies

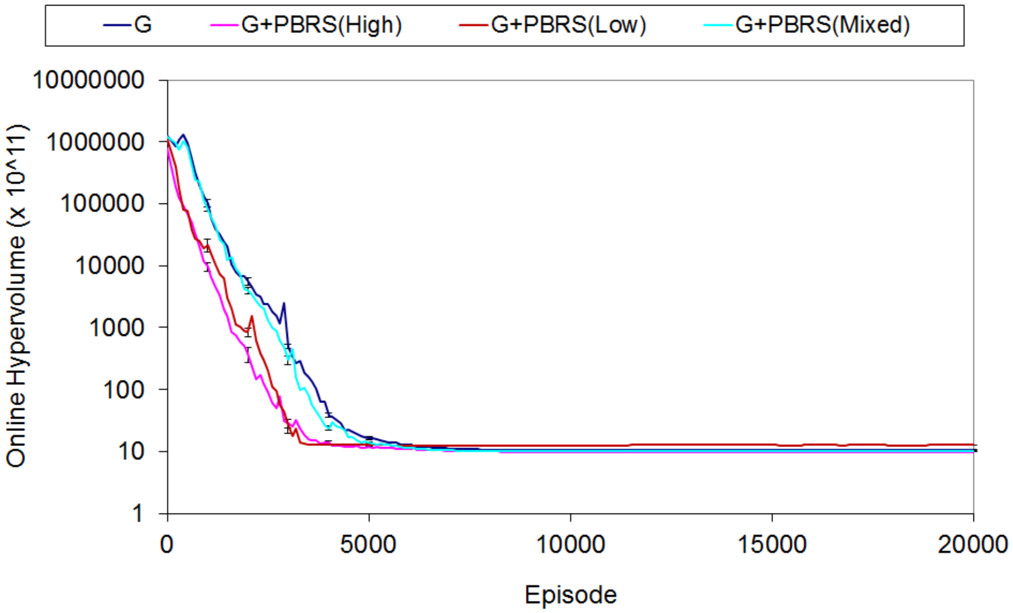

Figure 10: Online hypervolume of cost and emissions for non-dominated policies learned in the DEED environment

learned over the 30 runs conducted. Table 3 shows the average online hypervolume of cost and emissions over the last 2000 episodes of the 30 runs that were conducted. All claims of statistical significance are supported by two-tailed t-tests assuming unequal variances, with $p = 0.05$ selected as the threshold for significance.

The learning curves in Fig. 10 show an improvement in learning speed for all heuristics tested, when compared to unshaped G (note that the y-axis uses a $\log_{10}$ scale). The high heuristic gave the greatest improvement in learning speed, followed by the mixed and low heuristics. The high heuristic also achieved the lowest average hypervolume (9.8612×10^{11}), although the mixed heuristic achieved a similar value (9.9918×10^{11}). The difference in the mean hypervolumes of the high and mixed heuristics was found to be statistically insignificant ($p = 0.05954$). The next lowest hypervolume was that of unshaped G (10.5743×10^{11}), and the difference in the means of G and $G + PBRS(Mixed)$ was found to be statistically significant ($p = 8.98 \times 10^{-12}$). $G + PBRS(Low)$ had the highest hypervolume (12.6627×10^{11}), and the difference between its mean and that of G was found to be statistically

Table 3: DEED online hypervolumes (averages over last 2000 episodes)

	Online Hypervolume ($\times 10^{11}$)
$G + PBRS(High)$	9.8612
$G + PBRS(Mixed)$	9.9918
G	10.5743
$G + PBRS(Low)$	12.6627

significant ($p = 3.57 \times 10^{-20}$).

The plot of Pareto optimal policies learned in Fig. 11 shows that $G + PBRS(High)$ performed the best overall, learning joint policies that dominate the best solutions found by the other approaches. All Pareto optimal solutions found by $G + PBRS(Mixed)$ dominate those of G and $G + PBRS(Low)$. Overall, we have seen that the high heuristic offers the best performance in this domain, in terms of learning speed and Pareto optimal policies learned. Furthermore, two of the three PBRS heuristics that we have tested outperform unshaped G in every respect, demonstrating that PBRS incorporating suitable domain knowledge is a useful technique to improve performance in multi-agent MORL domains.

As the true Pareto front for the DEED MOSG is not known, we cannot evaluate the performance of each variant with respect to the hypervolume of the true NDS. However, we have demonstrated that well-designed PBRS heuristics can improve both learning speed and the quality of the final joint policies learned in multi-objective SGs. This is a similar finding to that of Devlin et al. [16] in their empirical study of PBRS in single-objective SGs. While PBRS does not alter the set of Nash equilibria or the Pareto relation between policies in multi-agent MORL, it can alter the points of equilibrium that are reached, as we have demonstrated in this study.

5.4. Experimental Procedure (Infinite-Horizon)

In addition to the above empirical study, we also tested the effect of PBRS on an infinite-horizon version of the DEED problem. This second study uses the same experimental parameters and reward functions as defined in Section 5.2 above. In practice, the continuous operation of a group of generators is in fact an infinite-horizon control problem, and this section aims to assess the suitability of our PBRS heuristics to develop policies for such a scenario. Thus, instead of an episode ending and all agents being reset

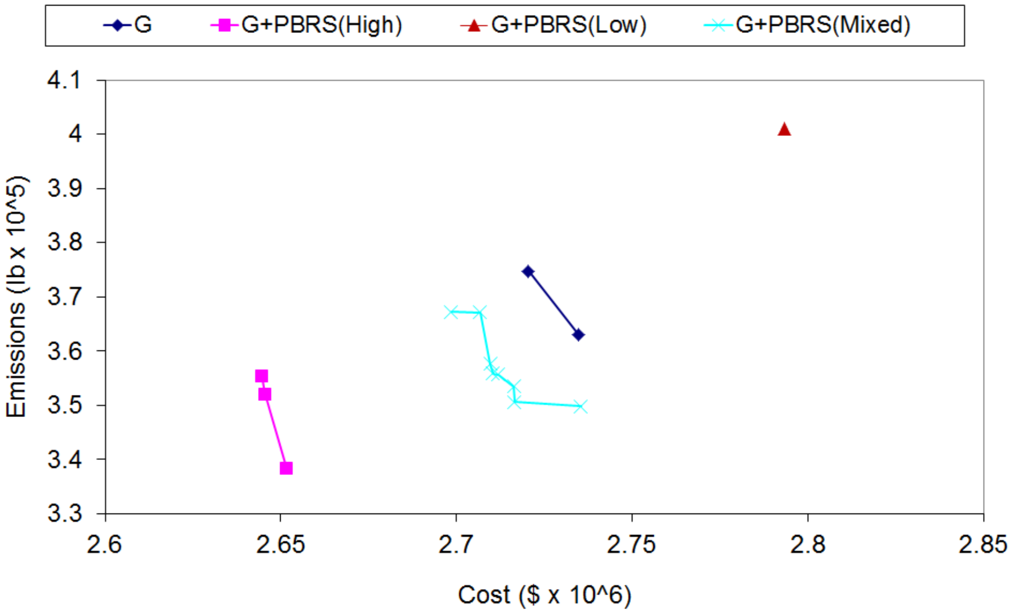

Figure 11: Pareto fronts showing the best non-dominated policies learned in the DEED environment

to the starting state, in this study agents operate continuously for a period of 480,000 hours. The same power demand curve shown in Fig. 9 is looped continuously during the 480,000 hour period. This time we use only a single vector [0.250,0.0.250,0.500] for linear scalarisation of the objective functions. As before we conducted 30 statistical runs for each reward function, and the results presented are the average of the 30 runs. The first 240,000 hours of each run are treated as a training period to allow the agents to converge to good joint policies, and the second 240,000 hours are used to test the quality of the policies learned.

5.5. Results (Infinite-Horizon)

The average results for the infinite-horizon DEED experiments are shown in Table 4. We measure the performance of the approaches using two different metrics: accumulated cost and accumulated emissions over the 240,000 hour testing period. Similar to the results for the finite-horizon study, we see that PBRS can also alter the joint policies learned in infinite-horizon problems.

Considering the accumulated cost metric, again $G + PBRS(High)$ and $G + PBRS(Mixed)$ both outperform unshaped G, while $G + PBRS(Low)$

Table 4: DEED infinite-horizon results (over last $240,000$ hours)

	Cost (\$ $\times 10^{10}$)	Emissions (lb $\times 10^{9}$)
$G + PBRS(High)$	2.7537	3.8931
$G + PBRS(Mixed)$	2.7973	3.8197
G	2.8244	4.0602
$G + PBRS(Low)$	2.9590	5.6805

is the worst performer. $G+PBRS(High)$ offers the lowest accumulated cost on average, and performs significantly better than $G + PBRS(Mixed)$ on this metric ($p = 1.68 \times 10^{-14}$). $G + PBRS(Mixed)$ does however offer a significant improvement in accumulated cost when compared to unshaped G ($p = 6.55 \times 10^{-10}$). The accumulated cost result for $G + PBRS(Low)$ is higher than that of G, and the difference in their performance was found to be significant ($p = 6.97 \times 10^{-29}$).

On the accumulated emissions metric, $G+PBRS(Mixed)$ offers the best performance, and is significantly better than $G + PBRS(High)$ ($p = 2.54 \times 10^{-81}$). $G + PBRS(High)$ significantly outperforms unshaped G on this metric ($p = 2.36 \times 10^{-32}$), while unshaped G outperforms $G + PBRS(Low)$ ($p = 1.49 \times 10^{-5}$). We see that both heuristics that performed better than unshaped G in the finite-horizon study also outperform it in this study, and that again $G + PBRS(Low)$ offers the worst performance.

The results from this study again highlight the effect of the quality of the heuristic used on the final performance reached by multiple agents in MORL domains. An interesting finding from this study is the notion that different PBRS heuristics may improve system performance on different objectives, as we see when we compare the results for $G + PBRS(High)$ and $G+PBRS(Mixed)$. We previously suspected that certain types of heuristics could lead agents to policies that favour one objective over another, however this study represents the first empirical evidence that this is in fact the case. Therefore, when designing PBRS heuristics for multi-objective problems this factor must be taken into account, especially in the case of multi-agent domains where the use of PBRS can change which Nash equilibrium is reached upon convergence. Due to this effect, it may be possible to use PBRS as a mechanism to incorporate user preferences in multi-criteria sequential decision making problems, by designing potential functions that bias an agent's exploration to favour one objective over another.

6. Conclusion & Future Work

In this paper, we have analysed the effect of PBRS in single- and multi-agent MORL problems, contributing both formal theoretical proofs and empirical studies. Our theoretical results prove that the set of Pareto optimal policies is invariant when PBRS is applied to both infinite- and finite-horizon MOMDPs and MOSGs. This result is true regardless of whether PBRS is applied to each objective individually, or to the resultant scalarised combination of the reward vector. Furthermore, our empirical results show that agents learning with and without PBRS in single-agent MORL problems converge to the same set of Pareto optimal policies, regardless of the quality of the PBRS heuristic used. When PBRS is applied to multi-agent MORL problems, we found that the joint policies learned can be altered, and that good PBRS heuristics can improve both learning speed and the quality of the Pareto-dominating solutions learned compared to agents learning without PBRS.

Numerous possibilities for further research are raised by the results presented in this paper. In the future, we plan to continue our work on reward shaping techniques for MORL problem domains. In particular, we intend to investigate the possibility of using reward shaping in conjunction with multi-policy RL algorithms (e.g. Pareto Q-learning [26]). The effect of reward shaping techniques in multi-agent MORL domains remains an under-explored topic, and we will continue to address this issue in future theoretical and empirical studies. It would be useful to develop a set of multi-agent MORL domains with known Pareto fronts which could act as a set of standardised benchmarks when evaluating multi-agent MORL algorithms, in a similar manner to the set of single-agent MORL benchmarks proposed by Vamplew et al. [24]. Recent work has made some progress towards achieving this goal [34, 35], and this line of research should be continued in order to produce a comprehensive suite of multi-agent MORL benchmark domains.

Acknowledgments

Patrick Mannion would like to acknowledge the support of the Irish Research Council, who funded his PhD work at the National University of Ireland Galway through the Government of Ireland Postgraduate Scholarship Scheme. The authors also wish to thank the editors and four anonymous reviewers for their constructive feedback which helped to improve this article.

References

[1] B. Majhi, C. Anish, Multiobjective optimization based adaptive models with fuzzy decision making for stock market forecasting, Neurocomputing 167 (2015) 502 – 511.

[2] P. Mannion, J. Duggan, E. Howley, An experimental review of reinforcement learning algorithms for adaptive traffic signal control, in: L. T. McCluskey, A. Kotsialos, P. J. Müller, F. Klügl, O. Rana, R. Schumann (Eds.), Autonomic Road Transport Support Systems, Springer International Publishing, 2016, pp. 47–66.

[3] K. Muralitharan, R. Sakthivel, Y. Shi, Multiobjective optimization technique for demand side management with load balancing approach in smart grid, Neurocomputing 177 (2016) 110 – 119.

[4] V. Pareto, Manual of political economy, Macmillan, 1971.

[5] A. Y. Ng, D. Harada, S. J. Russell, Policy invariance under reward transformations: Theory and application to reward shaping, in: Proceedings of the Sixteenth International Conference on Machine Learning, ICML '99, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999, pp. 278–287.

[6] S. Devlin, D. Kudenko, Theoretical considerations of potential-based reward shaping for multi-agent systems, in: Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2011, pp. 225–232.

[7] M. Wiering, M. van Otterlo (Eds.), Reinforcement Learning: State-of-the-Art, Springer, 2012.

[8] M. L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, 1st Edition, John Wiley & Sons, Inc., New York, NY, USA, 1994.

[9] C. J. C. H. Watkins, Learning from delayed rewards, Ph.D. thesis, King's College, Cambridge, UK (1989).

[10] C. J. Watkins, P. Dayan, Technical note: Q-learning, Machine Learning 8 (3-4) (1992) 279–292.

[11] M. Wooldridge, Introduction to Multiagent Systems, John Wiley & Sons, Inc., New York, NY, USA, 2001.

[12] K. Tumer, A. Agogino, Distributed agent-based air traffic flow management, in: Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Honolulu, HI, 2007, pp. 330–337.

[13] D. H. Wolpert, K. Tumer, Collective intelligence, data routing and braess' paradox, Journal of Artificial Intelligence Research (2002) 359–387.

[14] P. Mannion, K. Mason, S. Devlin, J. Duggan, E. Howley, Multi-objective dynamic dispatch optimisation using multi-agent reinforcement learning, in: Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2016, pp. 1345–1346.

[15] P. Mannion, K. Mason, S. Devlin, J. Duggan, E. Howley, Dynamic economic emissions dispatch optimisation using multi-agent reinforcement learning, in: Proceedings of the Adaptive and Learning Agents workshop (at AAMAS 2016), 2016.

[16] S. Devlin, M. Grzes, D. Kudenko, An empirical study of potential-based reward shaping and advice in complex, multi-agent systems, Advances in Complex Systems 14 (2) (2011) 251–278.

[17] K. Mason, P. Mannion, J. Duggan, E. Howley, Applying multi-agent reinforcement learning to watershed management, in: Proceedings of the Adaptive and Learning Agents workshop (at AAMAS 2016), 2016.

[18] L. Buşoniu, R. Babuška, B. Schutter, Multi-agent reinforcement learning: An overview, in: D. Srinivasan, L. Jain (Eds.), Innovations in Multi-Agent Systems and Applications - 1, Vol. 310 of Studies in Computational Intelligence, Springer Berlin Heidelberg, 2010, pp. 183–221.

[19] Y. Shoham, R. Powers, T. Grenager, If multi-agent learning is the answer, what is the question?, Artificial Intelligence 171 (7) (2007) 365–377.

[20] S. Devlin, Potential-based reward shaping for knowledge-based, multi-agent reinforcement learning, Ph.D. thesis, University of York, UK (2013).

[21] J. Randløv, P. Alstrøm, Learning to drive a bicycle using reinforcement learning and shaping, in: Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 463–471.

[22] S. Devlin, D. Kudenko, Dynamic potential-based reward shaping, in: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2012, pp. 433–440.

[23] P. Vamplew, J. Yearwood, R. Dazeley, A. Berry, On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts, in: W. Wobcke, M. Zhang (Eds.), AI 2008: Advances in Artificial Intelligence: 21st Australasian Joint Conference on Artificial Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 372–378.

[24] P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, E. Dekker, Empirical evaluation methods for multiobjective reinforcement learning algorithms, Machine Learning 84 (1) (2010) 51–80.

[25] L. Barrett, S. Narayanan, Learning all optimal policies with multiple criteria, in: Proceedings of the 25th international conference on Machine learning, ACM, 2008, pp. 41–47.

[26] K. Van Moffaert, A. Nowé, Multi-objective reinforcement learning using sets of pareto dominating policies, The Journal of Machine Learning Research 15 (1) (2014) 3483–3512.

[27] C. R. Shelton, Importance sampling for reinforcement learning with multiple objectives, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, USA (2001).

[28] D. M. Roijers, P. Vamplew, S. Whiteson, R. Dazeley, A survey of multi-objective sequential decision-making, Journal of Artificial Intelligence Research 48 (2013) 67–113.

[29] M. Grześ, Reward shaping in episodic reinforcement learning, in: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2017, pp. 565–573.

[30] T. Brys, A. Harutyunyan, P. Vrancx, M. E. Taylor, D. Kudenko, A. Nowé, Multi-objectivization of reinforcement learning problems by reward shaping, in: Neural Networks (IJCNN), 2014 International Joint Conference on, IEEE, 2014, pp. 2315–2322.

[31] M. Basu, Dynamic economic emission dispatch using nondominated sorting genetic algorithm-ii, International Journal of Electrical Power & Energy Systems 30 (2) (2008) 140–149.

[32] K. Mason, Avoidance techniques & neighbourhood topologies in particle swarm optimisation, Master's thesis, National University of Ireland Galway (2015).

[33] A. E. Smith, D. W. Coit, T. Baeck, D. Fogel, Z. Michalewicz, Penalty functions, Evolutionary computation 2 (2000) 41–48.

[34] P. Mannion, J. Duggan, E. Howley, A theoretical and empirical analysis of reward transformations in multi-objective stochastic games, in: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2017, pp. 1625–1627.

[35] P. Mannion, J. Duggan, E. Howley, Analysing the effects of reward shaping in multi-objective stochastic games, in: Proceedings of the Adaptive and Learning Agents workshop (at AAMAS 2017), 2017.