Hypervolume-based Multi-Objective Reinforcement Learning

Kristof Van Moffaert, Madalina M. Drugan, and Ann Nowé

Computational Modeling Lab, Vrije Univesiteit Brussel,
Pleinlaan 2, 1050 Brussels, Belgium
{kvmoffae,mdrugan,anowe}@vub.ac.be

Abstract. Indicator-based evolutionary algorithms are amongst the best performing methods for solving multi-objective optimization (MOO) problems. In reinforcement learning (RL), introducing a quality indicator in an algorithm's decision logic was not attempted before. In this paper, we propose a novel on-line multi-objective reinforcement learning (MORL) algorithm that uses the hypervolume indicator as an action selection strategy. We call this algorithm the *hypervolume-based MORL* algorithm or *HB-MORL* and conduct an empirical study of the performance of the algorithm using multiple quality assessment metrics from multi-objective optimization. We compare the hypervolume-based learning algorithm on different environments to two multi-objective algorithms that rely on scalarization techniques, such as the linear scalarization and the weighted Chebyshev function. We conclude that HB-MORL significantly outperforms the linear scalarization method and performs similarly to the Chebyshev algorithm without requiring any user-specified emphasis on particular objectives.

Keywords: multi-objective optimization, hypervolume unary indicator, reinforcement learning

1 Introduction

Multi-objective optimization (MOO) is the process of simultaneously optimizing multiple objectives which can be complementary, conflicting or independent. MOO is omnipresent in real-life and comprises a large part of the current research landscape involving optimization techniques.

Most of the research concerning this domain is being focused on evolutionary algorithms (EAs), such as NSGA-II [1]. A popular approach to solving MOO problems is to transform the multi-objective problem into a single-objective problem by employing *scalarization* functions. These functions provide a single figure indicating the quality over a combination of objectives, which allows a simpler and fast ordering of the candidate solutions. Recently, quality indicators, such as the hypervolume measure that are usually used for performance assessment, are introduced into the decision making process of these EAs. Searching the decision space using quality indicators is a fruitful technique in EAs, but in

reinforcement learning, this approach remained untouched. This paper fills this gap by proposing a novel reinforcement learning algorithm based on Q-learning that uses the hypervolume metric as an action selection strategy.

Contributions. There exist several algorithms that focus on *multi-objective reinforcement learning* (MORL) [2,3,4], but they only take into account the linear scalarization function. We combine ideas from two machine learning techniques (i.e. optimization and reinforcement learning) that have different goals in exploiting the multi-objective environments. We propose a novel multi-objective reinforcement learning algorithm where the hypervolume unary indicator is used to evaluate action selection. We call it *hypervolume MORL* (HB-MORL) and conceptually compare it two two scalarization-based MORL algorithms on environments consisting of two and three objectives. The experimental results show that HB-MORL outperforms the linear scalarization algorithm and performs similar to the Chebyshev-based algorithm.

Outline. In Section 2, we provide an overview of background concepts such as multi-objective optimization and we introduce reinforcement learning in Section 3. Subsequently, in Section 4, we reveal our novel algorithm, HB-MORL and conduct experiments in Section 5. Finally, we draw conclusions in Section 6.

2 Preliminaries

A multi-objective optimization problem optimizes a vector function whose elements represent the objectives. A maximization multi-objective problem is $\max \mathbf{F}(x) = \max\{f^1(x), f^2(x), ..., f^m(x)\}$, where m is the number of objectives, and f^i is the value for the i-th objective. A solution x_1 is said to *dominate* another solution x_2, $\mathbf{F}(x_2) \prec \mathbf{F}(x_1)$, iff for all objectives j, $f^j(x_2) \leq f^j(x_1)$, and there exists a objective i, for which $f^i(x_2) < f^i(x_1)$.

2.1 Scalarization functions

Scalarization functions transform a multi-objective problem to a single-objective problem. The scalarization functions often take into consideration weighting coefficients which allow the user some control over the chosen policy, by placing more or less emphasis on each objective. In this paper, we consider two instances of scalarization functions:

Linear scalarization function. In the linear weighted-sum method a weighted coefficient w_i is associated with each objective function. A weighted-sum is performed over all objectives and their corresponding weights. The value of a solution x is $\sum_{i=1}^{m} w_i f_i(x)$. The benefit of the linear scalarization functions is its simplicity and intuitive representation.

Chebyshev scalarization function. Also for this scalarization, we have weights associated to each objective. The Chebyshev metric [5] calculates for each objective the weighted distance between a reference point, $\mathbf{z}^*$ and a point

of interest in the multi-objective environment. For maximization problems, it chooses the greatest of these distances. The scalarized value for a solution x is $\max_{i=1,\ldots,m} w_i|f_i(x) - z_i^*|$. The reference point $\mathbf{z}^*$ is a parameter that is constantly being updated with the best value for each objective of solutions in the current Pareto set plus a small constant, i.e. $\mathbf{z}^*$ is $z_i^* = f_i^{best}(x) + \epsilon$, where ϵ is a small number.

2.2 Indicator-based Evolutionary Algorithms

Indicator-based evolutionary algorithms or IBEA is the class of algorithms that rely on a quality indicators in their selection process. Let $\Psi \subseteq 2^X$ be the set of all possible Pareto set approximations. A unary quality indicator is a function $I : \Psi \to \mathbb{R}$, assigning a Pareto set approximations, A_1, a real value $I(A_1)$. Many quality indicators exist, but the one that is most interesting for our context is the hypervolume indicator. This metric calculates the volume of the area between a reference point and the Pareto set obtained by a specific algorithm.

The hypervolume measure is of particular interest in this context as it is the only single set quality measure known to be strictly increasing with regard to Pareto dominance. The drawback of calculating the exact hypervolume remains its computation time, as it is an NP-hard problem [6]. Over the years, several hypervolume-based EAs for MOO have been proposed, such as MO-CMA-ES [7] and SMS-EMOA [8].

3 Multi-objective reinforcement learning

Evolutionary methods optimize an explicit objective function where reinforcement learning (RL) optimizes an *implicit* objective function. More precisely, RL involves an agent operating in a certain environment and receiving reward or punishment for certain behaviour. The focus of this paper is on multi-objective reinforcement learning (MORL) and how to combine it with the hypervolume unary indicator. In the following sections, we give a brief overview of existing multi-objective reinforcement learnings algorithms that utilize scalarization functions to transform the multi-objective search space of a problem into a single-objective environment.

Markov decision process. The principal structure for RL is a Markov Decision Process (MDP). An MDP can be described as follows. Let the set $S = \{s_1, \ldots, s_N\}$ be the state space of a finite Markov chain $\{x_l\}_{l\geq 0}$ and $A = \{a_1, \ldots, a_r\}$ the action set available to the agent. Each combination of starting state s_i, action choice $a_i \in A_i$ and next state s_j has an associated transition probability $T(s_j, s_i, a_i)$ and and immediate reward $R(s_i, a_i)$. The goal is to learn a policy π, which maps each state to an action so that the expected discounted reward is maximized. [9] proposed Q-learning, an algorithm that expresses this goal by using Q-values which explicitly store the expected discounted reward for every state-action pair. Each entry contains the value for $\hat{Q}(s, a)$ which represents

the learning agent's current hypothesis about the actual value of $Q(s, a)$. The $\hat{Q}$-values are updated according to the following update rule:

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

where α_t is the learning rate at time step t and $r(s, a)$ is the reward received for performing action a in state s.

Multi-objective MDPs. In MOO, MDPs are replaced by multi-objective MDPs or MO-MDPs [10]. These extend MDPs by replacing the single reward signal by a vector of rewards, i.e. $\vec{R}(s_i, a_i) = (R_1(s_i, a_i), \ldots R_m(s_i, a_i))$, where m represents the number of objectives. Since the reward vector consists of multiple components, each representing different objectives, it is very likely that conflicts arise between them. In such case, trade-offs between these objectives have to be learned, resulting in a set of different policies compared to a single optimal policy in single-objective learning. The overall goal of solving MO-MDPs is to find a set of policies that optimize different objectives. The set of optimal policies for each objective or a combination of objectives is referred to as the *Pareto optimal set*.

Multi-objective reinforcement learning. There are several MORL frameworks proposed in literature. For instance, [3] suggests a multi-objective algorithm that uses a lexicographic ordering of the objectives. More precisely, by placing minimal thresholds on certain objectives, policies are discovered that take into account these constraints. [4] proposes a batch Convex Hull Value Iteration algorithm that learns all policies in parallel, defining the convex hull of the optimal Pareto set. [2] also proposes a batch MORL approach, based on the linear scalarization function, to identify which actions are favoured in which parts of the objective space. Notwithstanding their results, they all consists of *off-line* algorithms, which involve sweeping over a set of collected data. Therefore, the aspects of these algorithms on using and adapting their policy during the learning process (i.e. *on-line* learning) were not studied.

Scalarization-based MORL. To the best of our knowledge, all MORL algorithms are currently focusing on the linear scalarization function. Therefore, the most general MORL algorithm that allows a fair comparison in this paper is an on-line multi-objective Q-learning algorithm (MO Q-learning) employed with the linear and the Chebyshev scalarization functions, presented in Section 2. These novel multi-objective reinforcement learning algorithms [11] are an extenstion to the single-objective Q-learning algorithm [9] that can accommodate for any scalarization function. The main change compared to standard Q-learning and the work in [2] is the fact that scalarization functions are applied on Q-values in contrast to reward signals. Thus, the standard Q-table, used to store the expected reward for the combination of state s and action a, is extended to incorporate objectives, i.e. $Q(s, a, o)$. This has the advantage that non-linear functions, such as the Chebyshev function, can be utilized in the same framework.

Algorithm 1 Scalarized ϵ-greedy action selection, *scal*-ϵ-greedy()

1: $SQList \leftarrow \{\}$
2: **for** each action $a_i \in A$ **do**
3: $\quad \vec{o} \leftarrow \{Q(s, a_i, o_1), \ldots, Q(s, a_i, o_m)\}$
4: $\quad SQ(s, a) \leftarrow scalarize(\vec{o})$ ▷ Scalarize Q-values
5: $\quad$ Append $SQ(s, a)$ to $SQList$
6: **end for**
7: **return** ϵ-greedy($SQList$)

In Algorithm 1, we present the scalarized action selection strategy for MO Q-learning. At line 4, the *scalarize* function can be instantiated by any scalarization function to obtain a single indication for the quality of the combination of state s and action a, $SQ(s, a)$ over the Q-values for each objective. Furthermore, the standard ϵ-greedy strategy from RL can be applied after we transform the multi-objective problem to a single-objective problem and decide the appropriate action, based on these individual indications in $SQList$. The new multi-objective Q-learning algorithm is presented in Algorithm 2. At line 1, the Q-values for each triple of states, actions and objectives are initialized. Each episode, the agent starts in state s (line 3) and chooses an action based on the multi-objective action selection strategy of Algorithm 1 at line 5. Upon taking action a, the agent is being transitioned into the new state s' and the environment provides it with the vector of rewards $\vec{r} \in \vec{\mathbb{R}}$. At line 10, the $Q(s, a, o)$ are updated with a multi-objective version of Eq. 1. This process is repeated until the Q-values converge.

Algorithm 2 MO Q-learning algorithm

1: Initialize $Q(s, a, o)$ arbitrarily
2: **for** each episode T **do**
3: $\quad$ Initialize state s
4: $\quad$ **repeat**
5: $\quad\quad$ Choose action a from s using policy derived from Q (e.g. *scal*-ϵ-greedy)
6: $\quad\quad$ Take action a and observe state $s' \in S$, reward vector $\vec{r} \in \vec{\mathbb{R}}$
7: $\quad\quad$ $\max_{a'} \leftarrow$ Call Scal. greedy action selection ▷ Get best scal. action in s'
8:
9: $\quad\quad$ **for** each objective o **do** ▷ Update Q-values for each objective
10: $\quad\quad\quad$ $Q(s, a, o) \leftarrow Q(s, a, o) + \alpha[\vec{r}(s, a, o) + \gamma Q(s', \max_{a'}, o) - Q(s, a, o)]$
11: $\quad\quad$ **end for**
12:
13: $\quad\quad$ $s \leftarrow s'$ ▷ Proceed to next state
14: $\quad$ **until** s is terminal
15: **end for**

Algorithm 3 Greedy Hypervolume-based Action Selection, HBAS(s, l)

1: $volumes \leftarrow \{\}$ ▷ The list collects hv contributions for each action
2: **for** each action $a_i \in A$ of state s **do**
3: $\quad \vec{o} \leftarrow \{Q(s, a_i, o_1), \ldots, Q(s, a_i, o_m)\}$
4: $\quad hv \leftarrow calculate\_hv(l + \vec{o})$ ▷ Compute hv contribution of a_i to l
5: $\quad$ Append hv to $volumes$
6: **end for**
7: **return** $\text{argmax}_a\ volumes$ ▷ Retrieve the action with the maximal contribution

4 Hypervolume-based Multi-Objective RL

In this section, we present our novel *hypervolume-based* MORL algorithm (HB-MORL) that combines the hypervolume unary indicator as a novel action selection mechanism. This action selection mechanism is similar to the selection strategy utilized for the MO Q-learning algorithm (Algorithm 1).

The proposed strategy is presented in Algorithm 3, while the entire HB-MORL algorithm is presented in Algorithm 4. The outline of the HB-MORL algorithm is similar to the MO Q-learning algorithm in Algorithm 2, but has an additional parameter, l. Each episode, the agent maintains a list l of Q-values of already visited states and actions. Initially, this list is empty (Algorithm 4, line 3).

In the action selection strategy, the agent consults this list (Algorithm 3) by employing the hypervolume metric. For each action a_i of state s, the vector of Q-values is retrieved from the table at line 3, whereafter the contribution of each action to the list of visited state-action pairs is calculated (line 4) and stored in the *volumes* list. In the greedy selection case, the action with the largest contribution is retrieved from *volumes* and selected (line 7), while in the ϵ-greedy case a random action is selected with a probability of ϵ (not shown in Algorithm 3). Subsequently, the Q-values of the selected action are appended to the list l (line 8, Algorithm 4) and the learning proceeds.

Differences between MORL algorithms. The HB-MORL algorithm, presented in this paper, resembles in quite a few places to the scalarization framework, presented in Algorithm 2. They are both based on Watkins' Q-learning algorithm and its update rule. This offers the advantage that we can rely on the same convergence proof and no exotic or problem-specific algorithm is proposed. On the contrary, their correspondence allows the same generality that Q-learning has been offering for decades. As presented, the main difference to the scalarization framework lies in the action selection strategy. The scalarization framework transforms the vector of Q-values into a single indicator, whereas the hypervolume-based algorithm performs searches directly into the objective space. Furthermore, HB-MORL does not rely on weights, defined a priori to guide the search process, as opposed to the scalarized algorithms. When the policies obtained by different runs of the algorithm are collected, the user can

Algorithm 4 Hypervolume-based Q-learning algorithm

Initialize $Q(s, a, o)$ arbitrarily
for each episode T **do**
  Initialize s, $l = \{\}$
  repeat
    Choose a from s using policy derived from Q (e.g. ϵ-greedy HBAS(s, l))
    Take action a and observe state $s' \in S$, reward vector $\vec{r} \in \vec{\mathbb{R}}$
    $\vec{o} \leftarrow \{Q(s, a, o_1), \ldots, Q(s, a, o_m)\}$
    Add $\vec{o}$ to l ▷ Add Q-values of selected action a to l
    $max_{a'} \leftarrow$greedy HBAS(s', l) ▷ Get greedy action in s' based on new l

    for each objective o **do** ▷ Update Q-values for each objective
      $Q(s, a, o) \leftarrow Q(s, a, o) + \alpha[\vec{r}(s, a, o) + \gamma Q(s', \max_{a'}, o) - Q(s, a, o)]$
    end for

    $s \leftarrow s'$ ▷ Proceed to next state
  until s is terminal
end for

still make her/his decision on which policies or trade-offs are preferred, but the advantage is that emphasis on particular objectives is not required beforehand.

5 Results

In this section, we experimentally evaluate the performance of the HB-MORL algorithm on two benchmark environments for different quality measures. These results are then compared to two instances of MORL algorithms that use scalarization-based action selection strategies, i.e. the linear and the Chebyshev Q-learning algorithm.

5.1 Testing environments

Recently, [12] proposed empirical evaluation techniques for multi-objective reinforcement learning, together with a few benchmark environments. We build further on this work and perform our experiments on the same worlds, such as the *Deep Sea Treasure* and the *Multi-Objective Mountain Car* environments to compare the two scalarization functions and the HB-MORL algorithm in detail. The optimal Pareto sets of each world were provided by the same researchers.

5.2 Parameter setting

In the experiments, presented below, we relied on identical configurations for each of the testing environments. We applied an ϵ-greedy exploration strategy with ϵ set to 0.1 and the Q-values were initialized randomly for each objective.

Table 1: The reference points (RP) used to calculate the hypervolume indicator in the learning algorithm and in the quality assessment for each environment.

	Deep Sea Treasure	**MO-Mountain Car**
Learning RP	$(-25.0, -5.0)$	$(-10.0, -10.0, -10.0)$
Quality Assessment RP	$(-25.0, 0.0)$	$(-350.0, -250.0, -500.0)$

The learning rate α was set to 0.1 and the discount factor γ to 0.9. Results are collected and averaged over 50 trials of each 500 runs.

Each scalarization-based MORL algorithm was tested using different sets of the weights sets ranging from 0 to 1, with a steps of 0.1 for each objective, i.e. there are 11 and 64 tuples of weights (and experiments) for the worlds with two and three objectives, respectively. Each scalarized MORL algorithm using a particular weight tuple could be regarded as an individual learning agent. As the hypervolume metric does not require any prior knowledge and to ensure a fair comparison in the results, we ran the HB-MORL algorithm for exactly the same number of experiments as the scalarized algorithms. Per iteration, each agent individually tests its learned policy by greedily selecting actions, whereafter each agents' policy is stored in a set. This set is called the Pareto approximation set for a particular iteration number of the learning phase.

We employed the hypervolume metric also as a quality assessment tool in the comparisons of the different algorithms. Table 1 presents the reference points (RP) used for calculating the hypervolume in both the learning phase and the testing phase for each of the environments. These values were determined empirically by examining the bounds on the reward structure of each testing environment in a straightforward way.

5.3 Performance experiment

In this experiment, we compare the performance of the linear and Chebyshev-based algorithms to our novel hypervolume-based MORL algorithm. In Table 2 and 3, we relied on the Wilcoxon rank test [13] to indicate a significant difference on the mean performance between the indicator-based MORL algorithm and both scalarized methods on each environment. We present the learning curves for each of the environments in Fig. 1(a) and 1(b) by applying the hypervolume indicator for quality assessment purposes. For the Deep Sea Treasure world (Fig. 1(a)), the linear scalarization-based MORL is not capable of improving the hypervolume after 100 runs whereas the Chebyshev-based algorithm slightly improves its performance until 250 runs. The HB-MORL is gradually improving its policy and improves the Chebyshev algorithm after 400 runs. It is interesting to note that the HB-MORL algorithm is increasingly its performance gradually until the end of the learning phase. In other preliminary tests, not included in this paper, we ran the experiments for a longer period of time, but no algorithm was able to further improve its policy after 500 runs.

Table 2: The Wilcoxon rank test denoted a significant difference on the mean performance of the linear scalarized algorithm and HB-MORL on both testing worlds for a threshold p value of 0.05.

	Linear scalarization	HB-MORL	p-value	Significant?
Deep Sea Treasure	762	1040.24	7.0338^{-4}	$\surd$
MO-Mountain Car	15727946.18	23984880.12	$5.9282e^{-5}$	$\surd$

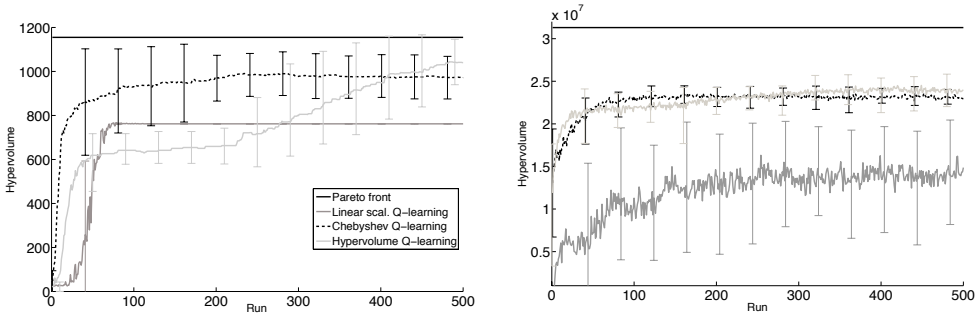
Table 3: Also for the Chebyshev-based algorithm and HB-MORL, a significant difference is noted by the Wilcoxon rank test on the two benchmark instances (threshold p = 5%).

	Chebyshev	HB-MORL	p-value	Significant?
Deep Sea Treasure	938.29	1040.24	$1.5256e^{-17}$	$\surd$
MO-Mountain Car	23028392.94	23984880.12	$5.0379e^{-16}$	$\surd$

Finally, we observed the largest difference between the two scalarized algorithms in the complex 3D MO-Mounting Car world (Fig. 1(b)). In this benchmark environment, the Chebyshev-based algorithm is stabilized after 100 runs, whereas the linear scalarization-based MORL algorithm slowly increases until the last learning steps. Nevertheless, a considerable difference in quality is kept between the two scalarized MORL methods. The hypervolume-based Q-learning algorithm is stabilising approximately as fast as the Chebyshev algorithm, but the gradual improvement phase is also observed in this complex world and a significant improved performance is achieved.

In Fig. 2(a) and 2(b), we elaborate into more detail the gradual learning phase achieved by each of the learning methods. We restricted the information in the learning curves to capture the performance every 100 runs. For each of the environments, the linear scalarized algorithm is performing the worst and its performance stagnates after only a few iterations. The Chebyshev method is able to escape the local maxima in a much better way and is improving until the end of the learning phase. Finally, the HB-MORL algorithm is able to improve even further and achieves an enhanced performance in its finals runs.

In Fig. 2(c), we show the frequency probability of the 10 Pareto dominating goals (i.e. treasures) reached in the Deep Sea world. This plot provides us with an idea on the spread that each learning method can obtain amonst Pareto dominating solutions. We note that the linear scalarization-based algorithm only finds extreme solution, i.e. solutions that maximize only one of the objectives. More precisely, the two results found are the treasures with value 1 and 124, i.e. the treasures that minimize the *time* objective and maximize the *treasure* objective, respectively, but no real compromising solutions were obtained. The Chebyshev algorithm however, obtains a larger spread in the results compared



(a) Deep Sea Treasure world (b) MO Mountain Car world

Fig. 1: Learning curves on the Deep Sea Treasure and the MO Mountain Car worlds, respectively

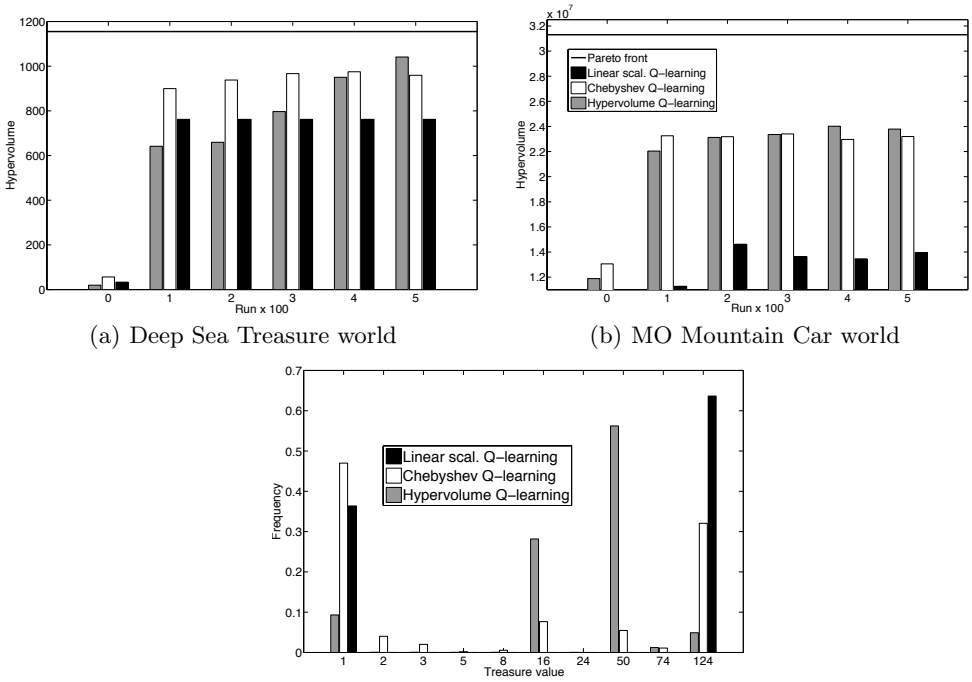
to the linear case. Out of the 10 possible goals, it found 8 on a regular basis, with increasing probability near the extreme solutions. Without being initialized with any prior preferences, the hypervolume-based MORL performed acceptable and focused on 5 solutions.

5.4 Quality indicator comparison

In multi-objective research, quality indicator studies are a popular approach for conducting algorithm comparisons. The performance indicators applied in this experimental study are the (inverted) generational distance , the generalized spread indicator, the cardinality and the hypervolume distance. The former three are minimization metrics, while the latter two are to be maximized. In detail, the generational distance and the inverted generational distance were both proposed by [14]. The former measures how far the elements in the set of non-dominated vectors, generated by a learning algorithm, are from those in the Pareto optimal set. The latter calculates how far each element of the Pareto optimal set is from the non-dominated vectors in the approximation set. The spread indicator [1] on the other hand is a diversity metric that measures the extent of spread achieved among the obtained solutions. The cardinality measure simply counts the number of elements found in the Pareto set.

The results are presented in Table 4. On the Mountain Car (MC) world, the HB-MORL obtained overall the best results out of the three algorithms, except for the fact that the Chebyshev method found one extra solution. The linear scalarized algorithm obtained the best value for the generalized spread, but as this metric only uses the members on the boundaries of the Pareto optimal set (i.e. the extreme solutions) in its calculations, this metric is biased towards the linear method that exclusively finds these solutions (see Fig. 2(c)).

On the Deep Sea (DS) world, the Chebyshev method found 8 out of 10 distinct results and obtained the best value for the inverted generational distance. Closely followed by HB-MORL that without any prior information (i.e.



(a) Deep Sea Treasure world

(b) MO Mountain Car world

(c) Frequency of goals in the Deep Sea world

Fig. 2: Fig. 2(a) and 2(b) depict the performance of each learning algorithm each 100 runs. In Fig. 2(c), the frequency probabilities of each of the 10 Pareto dominating goals in the Deep Sea Treasure world are presented. The Chebyshev-based algorithm obtained the best spread, closely being followed by HB-MORL. The linear scalarized algorithm only found two (extreme) solutions.

no weights) obtained 5 distinct results, but a larger hypervolume was obtained. This means that HB-MORL was much more consistent in finding good solutions frequently (i.e. the increased hypervolume), but its results were not as spread around in the search space as the Chebyshev-based algorithm (i.e. the cardinality). The linear scalarized algorithm only obtained 2 (extreme) results that are located at the largest possible distance from each other, resulting in the best generalized spread value. Each of the results found by any of the algorithms were an element of the optimal Pareto set, meaning that the generational distance is 0.

To conclude, on each environments, HB-MORL outperformed the linear scalarization algorithm and obtained the best results on the most important quality indicator, i.e. the hypervolume metric. On the other indicators in the Deep Sea Treasure world, the HB-MORL algorithm obtained good results but was not always the best performing algorithm. We can conclude that the HB-MORL al-

Table 4: Five quality indicator for each of the three algorithms on the two benchmark instances. The first three are to be minimized, while the latter two are maximization indicators. The best values are depicted in bold face.

		Linear	**Chebyshev**	**HB-MORL**
Inverted Generational distance	**DS**	0.128	**0.0342**	0.0371
	MC	0.012	0.010	**0.005**
Generalized spread	**DS**	**3.14e^{-16}**	0.743	0.226
	MC	**0.683**	0.808	0.701
Generational distance	**DS**	0	0	0
	MC	0.0427	0.013824	**0.013817**
Hypervolume	**DS**	762	959.26	**1040.2**
	MC	15727946	23028392	**23984880**
Cardinality	**DS**	2	**8**	5
	MC	15	**38**	37

gorithm was very consisting in finding solutions that maximize the hypervolume metric, but could be improved by more spread results.

Weights vs. quality indicator. In the following test, we investigate into more detail the results of HB-MORL to the results obtained for each weighted tuple for the scalarization-based algorithms (Table 5). It is important to note that the HB-MORL algorithm is not set up with any information on how the objectives should be balanced or weighed. Therefore, in the table, its values remain identical. Note that the generational distance was omitted because every result obtained was an element of the Pareto set. We focus on the differences between the Chebyshev method and the HB-MORL algorithm and notice that there is still a significant portion of the weighted tuples for which the Chebyshev algorithm achieved better performance in term of the inverted generational distance than was presumed in Table 4. Although, there are four cases (weights W6, W8, W9 and W10) where the HB-MORL algorithm obtained improved performance and 2 tuples (weights W3 and W4) that perform similarly.

The hypervolume measure indicated that for some weights, the Chebyshev algorithm obtained a large portion of the Pareto set, but on the larger portion of the experiments the results are less efficient. Especially when assigning a very low weight to the *treasure* objective (e.g. weight W10), a limited hypervolume is achieved. In those cases, the *time* objective is being minimized with the result that treasures with a limited value, located near the starting position to minimize the time and distance, are favored. The generalized spread indicator showed that when focusing on the performance of particular tuples of weights, the values become more clear and the HB-MORL algorithm is performing intrinsically better. Note that the Chebyshev algorithm found the two extreme points in the objective space for weight W9, thus resulting in the best possible value of 0. The same can be concluded for the cardinality indicator as for particular weights, very few solution points are obtained.

To conclude, based on the empirical results, for a large portion of weights, the Chebyshev MORL algorithm is considered a well-performing algorithm by many quality indicator measures, such as spread and cardinality (see Table 4). We have seen that for some weights (Table 5), the Chebyshev algorithm obtained

Table 5: Five quality indicator for each of the 11 weights in the Deep Sea world. The inverted generational distance (IGD) and spread indicator are to be minimized, while the other two are maximization indicators. The best values are depicted in bold face.

	Weight	**Linear**	**Chebyshev**	**HB-MORL**
IGD	W0	0.3234	**0.0227**	0.0371
	W1	0.3234	**0.0253**	0.0371
	W2	0.3234	**0.0296**	0.0371
	W3	0.3234	**0.0365**	0.0371
	W4	0.3234	**0.0360**	0.0371
	W5	0.32344	**0.0260**	0.0371
	W6	0.32344	0.0451	**0.0371**
	W7	0.2201	**0.0260**	0.0371
	W8	0.2201	0.0616	**0.0371**
	W9	0.2201	0.1279	**0.0371**
	W10	0.2201	0.2201	**0.0371**

	Weight	**Linear**	**Chebyshev**	**HB-MORL**
HV	W0	744	**1145**	1130
	W1	744	**1143**	1130
	W2	744	**1136**	1130
	W3	744	1094	**1130**
	W4	744	**1140**	1130
	W5	744	1024	**1130**
	W6	744	1082	**1130**
	W7	24	**1140**	1130
	W8	24	1018	**1130**
	W9	24	762	**1130**
	W10	24	24	**1130**

	Weight	**Linear**	**Chebyshev**	**HB-MORL**
Spread	W0	1	0.5481	**0.2257**
	W1	1	0.4003	**0.2257**
	W2	1	0.3714	**0.2257**
	W3	1	0.6234	**0.2257**
	W4	1	0.6830	**0.2257**
	W5	1	0.4159	**0.2257**
	W6	1	**0.1949**	0.2257
	W7	1	0.4159	**0.2257**
	W8	1	0.7121	**0.2257**
	W9	1	**0**	0.2257
	W10	1	1	**0.2257**

	Weight	**Linear**	**Chebyshev**	**HB-MORL**
Cardinality	W0	1	**8**	5
	W1	1	**7**	5
	W2	1	**6**	5
	W3	1	**6**	5
	W4	1	**6**	5
	W5	1	**7**	5
	W6	1	4	**5**
	W7	1	**7**	5
	W8	1	4	**5**
	W9	1	2	**5**
	W10	1	1	**5**

very good results for many quality indicators, while for other weights the method operates ineffectively. Thus, the principal drawback of these scalarization-based algorithm remains the fact that the user has to predefine its preferences by placing greater or lesser emphasis on each objective. This is a task that should not be underestimated. The main benefit of scalarization techniques remains their simplicity, but this does not compensate for the inconvenience of manually guiding the nature of the policy and the fact that these methods are very biased to the actual weights used.

5.5 Discussion

By randomly initializing the Q-values, the HB-MORL method obtains an acceptable notion of spread and found a large portion of Pareto dominating solutions. Thus, approaching the Chebyshev method for the spread indicator. But more importantly, HB-MORL improved the hypervolume indicator on every benchmark, indicating the method's robustness as good results are found frequently. Furthermore, HB-MORL does not require any direct input from the user on its actual preferences and solves this burden by employing the hypervolume qual-

ity indicator directly in its search process. This makes the main advantage of the HB-MORL algorithm its simplicity. Also, unlike the Chebyshev method, no specific reference point z^* is to be specified and updated in every run of the algorithm, making it easier for the developer to conduct experiments with HB-MORL instead of scalarization-based methods.

On both benchmark instances, the linear scalarization algorithm failed to achieve decent performance and got stuck in local optima while obtaining only two extreme solution points. These outcomes are in accordance with previous findings [12] on the linear scalarization algorithm, stating that the method is unable of finding solutions near non-convex regions of the optimal Pareto set. Independent of the running time, the algorithm gets stuck in local optima from which it can not escape. The Chebyshev-based algorithm and HB-MORL are able to gradually improve their hypothesis. Especially the latter is able to improve in final stages of its training phase.

Note that the HB-MORL algorithm is more of a slow starter compared to the Chebyshev-based algorithm. We believe that this is caused by the lack of diversity in the Pareto set explored with the hypervolume based indicator. This is also noticed in other approaches that conduct searches using the hypervolume metric and therefore [15] proposes to include a mechanism to increase the diversity of the Pareto sets and to encourage exploration. The reason why the Chebyshev-based method does not have this problem, is because the algorithm is restarted with different weights forcing the exploration of different regions of the multi-objective environment.

6 Conclusions

In this paper, we have successfully built a bridge between two machine learning techniques that rely on different solution approaches given an certain environment. More precisely, we have included a technique from multi-objective optimization, i.e. the hypervolume unary based indicator, into reinforcement learning. We have conceptually and experimentally compared our novel hypervolume-based MORL (HB-MORL) algorithm to two other scalarization-based learning algorithms, which require weights to be defined beforehand. In contrast, the HB-MORL algorithm does not contain preference-based parameters to be specified. For our experiments, we performed performance assessment tests on two benchmark instances with two and three objectives. We have noted that the suggested algorithm significantly improved the linear scalarization-based algorithm and performed similarly to the Chebyshev-based algorithm. Especially on indicators that asses the robustness of an algorithm on finding high-quality solutions frequently, the hypervolume-based algorithm turned out to be the best performing. We believe that HB-MORL is especially useful in cases where it is difficult to define user-preferences beforehand or in cases where it is complex to tune an algorithm specifically for a particular problem instance. In those situations, HB-MORL would allow to obtain a significant amount of high-quality solutions without requiring any weights parameters to be defined.

Acknowledgement

This research is supported by the IWT-SBO project PERPETUAL (grant nr. 110041).

References

1. Deb, K.D., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm : NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2) (April 2002) 182–197
2. Lizotte, D.J., Bowling, M., Murphy, S.A.: Efficient reinforcement learning with multiple reward functions for randomized controlled trial analysis. In: Proceedings of the Twenty-Seventh International Conference on Machine Learning (ICML). (2010) 695–702
3. Gábor, Z., Kalmár, Z., Szepesvári, C.: Multi-criteria reinforcement learning. In Shavlik, J.W., ed.: ICML, Morgan Kaufmann (1998) 197–205
4. Barrett, L., Narayanan, S.: Learning all optimal policies with multiple criteria. In: Proceedings of the 25th international conference on Machine learning. ICML '08, New York, NY, USA, ACM (2008) 41–47
5. Miettinen, K.: Nonlinear Multiobjective Optimization. Volume 12 of International Series in Operations Research and Management Science. Kluwer Academic Publishers, Dordrecht (1999)
6. Bringmann, K., Friedrich, T.: Approximating the volume of unions and intersections of high-dimensional geometric objects. Comput. Geom. Theory Appl. **43**(6-7) (August 2010) 601–610
7. Igel, C., Hansen, N., Roth, S.: Covariance matrix adaptation for multi-objective optimization. Evol. Comput. **15**(1) (March 2007) 1–28
8. Beume, N., Naujoks, B., Emmerich, M.: Sms-emoa: Multiobjective selection based on dominated hypervolume. European Journal of Operational Research **181**(3) (2007) 1653–1669
9. Watkins, C.: Learning from Delayed Rewards. PhD thesis, University of Cambridge,England (1989)
10. Wiering, M.A., de Jong, E.D.: Computing Optimal Stationary Policies for Multi-Objective Markov Decision Processes. In: 2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning, IEEE (April 2007) 158–165
11. Van Moffaert, K., M. Drugan, M., Nowé, A.: Multi-objective reinforcement learning using scalarization functions. Technical report, Computational Modeling Lab, Vrije Universiteit Brussel, Brussels, Belgium (2012)
12. Vamplew, P., Dazeley, R., Berry, A., Issabekov, R., Dekker, E.: Empirical evaluation methods for multiobjective reinforcement learning algorithms. Machine Learning **84**(1-2) (2010) 51–80
13. Gibbons, J., Chakraborti, S.: Nonparametric Statistical Inference. Statistics, Textbooks and monographs. Marcel Dekker (2003)
14. Veldhuizen, D.A.V., Lamont, G.B.: Multiobjective evolutionary algorithm research: A history and analysis (1998)
15. Ulrich, T., Bader, J., Zitzler, E.: Integrating decision space diversity into hypervolume-based multiobjective search. In: Proceedings of the 12th annual conference on Genetic and evolutionary computation. GECCO '10, New York, NY, USA, ACM (2010) 455–462