Exploration versus exploitation trade-off in infinite horizon Pareto Multi-armed bandits algorithms

Madalina Drugan[1] and Bernard Manderick[1]
[1]*Artificial Intelligence Lab, Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels, Belgium*
{*Madalina.Drugan, Bernard.Manderick*}*@vub.ac.be*

Keywords: Multi-armed bandits, Multi-objective optimisation, Pareto dominance relation, Infinite horizon policies

Abstract: Multi-objective multi-armed bandits (MOMAB) are multi-armed bandits (MAB) extended to reward vectors. We use the Pareto dominance relation to assess the quality of reward vectors, as opposite to scalarization functions. In this paper, we study the exploration vs exploitation trade-off in infinite horizon MOMABs algorithms. Single objective MABs explore the suboptimal arms and exploit a single optimal arm. MOMABs explore the suboptimal arms, but they also need to exploit fairly all optimal arms. We study the exploration vs exploitation trade-off of the Pareto UCB1 algorithm. We extend UCB2 that is another popular infinite horizon MAB algorithm to rewards vectors using the Pareto dominance relation. We analyse the properties of the proposed MOMAB algorithms in terms of upper regret bounds. We experimentally compare the exploration vs exploitation trade-off of the proposed MOMAB algorithms on a bi-objective Bernoulli environment coming from control theory.

1 Introduction

Multi-armed bandits (MAB) is a machine learning paradigm used to study and analyse resource allocation in stochastic and noisy environments. The multi-armed bandit problem considers multi-objective rewards and imports techniques from multi-objective optimisation into the multi-armed bandits algorithms. We call this the *multi-objective multi-armed bandits* (MOMAB) problem and it is an extension of the standard MAB-problem to reward vectors. [1] MOMAB also has K arms, $K \geq 2$, and let I the set of these K arms. But since we have multiple objectives, a random vector of rewards is received, one component per objective, when one of the arms is pulled. The random vectors have a stationary distribution with support in the D-dimensional hypercube $[0,1]^D$ but the vector of true expected rewards $\mu_i = (\mu_i^1, \ldots, \mu_i^D)$, where D is the number of objectives, is unknown. All rewards $\mathbf{X}_t^i$ obtained from any arm i are independently and identically distributed according to an an unknown law with unknown expectation vector $\mu_i = (\mu_i^1, \ldots, \mu_i^D)$. Reward values obtained from different arms are also assumed to be independent. A MAB algorithm chooses the next machine to play based on the sequence of past plays and obtained reward values.

MOMAB leads to important differences compared to the standard MAB. *Pareto dominance* Zitzler et al. (2003) allows to maximize the reward vectors directly in the vector reward space. A reward vector can optimize one objective and be sub-optimal in the other objectives, leading to many vector rewards of the same quality. Thus, there could be several arms considered to be the best according to their reward vectors. We call the set of optimal arms of the same quality the *Pareto front*. An adequate regret definition for the Pareto MAB algorithm measures the distance between a suboptimal reward vector and the Pareto front. We call this class of algorithms the *Pareto MAB* problem.

The main goal of this paper is to study the exploration vs exploitation trade-off in several Pareto MAB algorithms. Exploration means pulling the suboptimal arms that might have been unlucky, whereas exploitation means pulling as much as possible the optimal arms. The exploration vs exploitation trade-off is different for single objective MABs and for MOMABs. For single objective MABs, we are concerned with the exploration of the suboptimal arms and the exploitation of a single optimal arm. In MOMABs, by design, we should pull equally often

[1] Some of these techniques were also imported in other related learning paradigms: multi-objective Markov Decision Processes Lizotte et al. (2010); Wiering and de Jong (2007), and multi-objective reinforcement learning van Moffaert et al. (2013); Wang and Sebag (2012).

all the arms in the Pareto front. Thus, the exploitation now means the fair usage of Pareto optimal arms.

This difference in exploitation vs exploration trade-off reflects on all aspects of Pareto MAB algorithmic design. There are two regret metrics for the MOMAB algorithms Drugan and Nowe (2013). One performance metric, i.e. the Pareto projection regret metric, measures the amount of times *any* Pareto optimal arm is used. Another performance metric, i.e. the Pareto variance regret metric, measures the variance in using *all* Pareto optimal arms. Background information on MOMABs, in general, and Pareto MABs, in particular, are given in Section 2.

We propose several Pareto MAB algorithms that are an extension of the classical single objective MAB algorithms, i.e. UCB1 and UCB2 Auer et al. (2002), to reward vectors. The proposed algorithms focus on either the exploitation or the exploration mechanisms. We consider the Pareto UCB1 Drugan and Nowe (2013) to be an exploratory variant of this algorithm because each round only *one* Pareto optimal arm is pulled. In Section 3, we propose an exploitative variant of the Pareto UCB1 algorithm where, each round, *all* the Pareto optimal arms are pulled. We show that the analytical properties, i.e. upper confidence bound of the Pareto projection regret, for the exploitative Pareto UCB1 are improved when compared with the exploratory variant of the same algorithm because this bound is independent of the cardinality of the Pareto front.

Section 4 proposes two multi-objective variants of UCB2 corresponding to the two exploitation vs exploration mechanisms described before. The exploitative Pareto UCB2 is an extension of UCB2 where, each epoch, all the Pareto optimal arms are pulled equally often. This algorithm is introduced in Section 4.1. The exploratory Pareto UCB2 algorithm, see Section 4.2, pulls each epoch a single Pareto optimal arm. We compute the upper bound of the Pareto projection regret for the exploitative Pareto UCB2 algorithm.

Our motivating example is a bi-objective wet clutch Vaerenbergh et al. (2012) that is a system with one input characterised by a hard non-linearity when the piston of the clutch gets in contact with the friction plates. These clutches are typically used in power transmissions of off-road vehicles, which operate under strongly varying environmental conditions. The validation experiments are carried out on a dedicated test bench, where an electro-motor drives a flywheel via a torque converser and two mechanical transmissions. The goal is to learn by minimising simultaneously: i) the optimal current profile to the electro-hydraulic valve, which controls the pressure of the oil to the clutch, and ii) the engagement time. The output data is stochastic because the behavior of the machine varies with the surrounding temperature that cannot be exactly controlled. Section 5 experimentally compares the proposed MOMAB algorithms on a bi-objective Bernoulli reward distribution generated on the output solutions of the wet clutch.

Section 6 concludes the paper.

2 The multi-objective multi-armed bandits problem

We consider the general case where a reward vector can be better than another reward vector in one objective, and worse in another objective. Expected reward vectors are compared according to the **Pareto dominance relation** Zitzler et al. (2003).

The following dominance relations between two vectors μ and ν are used. A vector μ is *dominating*, another vector ν, $\nu \prec \mu$, if and only if there exists at least one objective o for which $\nu^o < \mu^o$ and for all other objectives j, $j \neq i$, we have $\nu^j \leq \mu^j$. A reward vector μ is *incomparable* with another vector ν, $\nu \| \mu$, if and only if there exists at least one objective o for which $\nu^o < \mu^o$, and there exists another objective j, $j \neq i$, for which $\nu^j > \mu^j$. Finally, the vector μ is *non-dominated* by ν, $\nu \not\prec \mu$, if and only if there exists at least one objective o for which $\nu^o < \mu^o$. Let $\mathcal{A}^*$ be the Pareto front, i.e. non-dominated by any arm in $\mathcal{A}$.

2.1 The exploration vs exploitation trade-off in Pareto MABs

A Pareto MAB-algorithm selects an arm to play based on the previous plays and the obtained reward vectors and it tries to maximize the total expected reward vectors. *The goal of a MOMAB algorithm is to simultaneously minimise the regret of not selecting the Pareto optimal arms by fairly playing all the arms in the Pareto front.*

In order to measure the performance of these algorithms, we define two Pareto regret metrics. The first regret metric measures the loss in pulling arms that are not Pareto optimal and is called the *Pareto projection regret*. The second metric, the *Pareto variance regret*, measures the variance [2] in pulling each arm from the Pareto front $\mathcal{A}^*$.

The Pareto projection regret expresses the expected loss due to the play of suboptimal arms. For

[2] Not to be confused with the variance of random variables.

this purpose, it uses the Euclidean distance between the mean reward vector μ_i of an arm i and its projection ν_i into the Pareto front. This projection is obtained as follows: A vector ε_i with equal components ε_i, i.e. $\varepsilon_i = (\varepsilon_i, \varepsilon_i, \cdots, \varepsilon_i)$, is added to μ_i such that ε_i is the smallest value for which $\nu_i = \mu_i + \varepsilon_i$ becomes Pareto optimal. The Euclidean distance Δ_i between μ_i and its projection ν_i into the Pareto front equals:

$$\Delta_i = \|\nu_i - \mu_i\|_2 = \|\varepsilon_i\|_2 = \sqrt{D}\varepsilon_i \quad (1)$$

where the last equality holds because we have D objectives and all components of ε_i are the same.

Since by definition Δ_i is always non-negative, the resulting regret is also non-negative. Note that the for a Pareto optimal arm $\nu_i = \mu_i$ and $\Delta_i = 0$.

Let $T_i(n)$ be the number of times that arm i has been played after n plays in total. Then the Pareto projection regret $R_p(n)$ after n plays is defined as:

$$R_p(n) = \sum_{i \notin \mathcal{A}^*} \Delta_i \mathbb{E}[T_i(n)] \quad (2)$$

where Δ_i is defined in Equation 1 and where $\mathbb{E}$ is the expectation operator. A similar regret metric was introduced in Drugan and Nowe (2013).

The Pareto variance regret metric measures the variance of a Pareto-MAB algorithm in pulling all optimal arms. Let $T_i^*(n)$ be the number of times an optimal arm i is pulled during n total arm pulls. Let $\mathbb{E}[T_i^*(n)]$ the expected number of times the Pareto optimal arm i is pulled. The Pareto variance regret is defined as

$$R_v(n) = \frac{1}{|\mathcal{A}^*|} \sum_{i \in \mathcal{A}^*} (\mathbb{E}[T_i^*(n)] - \mathbb{E}[T^*(n)]/|\mathcal{A}^*|)^2 \quad (3)$$

where $\mathbb{E}[T^*(n)]$ is the expected number of times that *any* Pareto optimal arm is selected, and $|\mathcal{A}^*|$ is the cardinality of the Pareto front $\mathcal{A}$.

If all Pareto optimal arms are played in a fair way, i.e. an equal number of times, then $R_v(n)$ is minimized. For a perfect fair, or equal, usage of the Pareto optimal arms, we have $R_v(n) \leftarrow 0$. If a Pareto MAB-algorithm identifies only a subset of $\mathcal{A}^*$, then $R_v(n)$ is large. A similar measure, called unfairness, was proposed in Drugan and Nowe (2013) to measure variance of a Pareto-MAB algorithm in pulling all Pareto optimal arms.

3 Exploration vs exploitation trade-off in Pareto UCB1

The Pareto UCB1 algorithm Drugan and Nowe (2013) is an UCB1 algorithm using the Pareto dominance relation to partially order the reward vectors.

Algorithm 1 Exploitative Pareto UCB1

1: Play each arm i once
2: $t \leftarrow 0$; $n \leftarrow K$; $n_i \leftarrow 1, \forall i$
3: **while** the stopping criteria is NOT met **do**
4: $t \leftarrow t + 1$
5: Select the Pareto front at the round t, $\mathcal{A}^{*(t)}$, such that $\forall i \in \mathcal{A}^{*(t)}$ the index $\hat{\mu}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D})}{n_i}}$ is non-dominated
6: Pull each arm i once, where $i \in \mathcal{A}^{*(t)}$
7: $\forall i \in \mathcal{A}^{*(t)}$, update $\hat{\mu}_i$, and $n_i \leftarrow n_i + 1$
8: $n \leftarrow n + |\mathcal{A}^{*(t)}|$
9: **end while**

Like for the classical single-objective UCB1 Auer et al. (2002), the index for a Pareto UCB1 algorithm has two terms: the mean reward vector, and the second term related to the size of a one-sided confidence interval of the average reward according to the Chernoff-Hoeffding bounds.

In this section, we propose a Pareto UCB1 algorithm with an improved exploration vs exploitation trade-off because its performance does not depend on the size of Pareto front. In each round, all the Pareto optimal arms are pulled once instead of pulling only one arm. This means that the proposed Pareto UCB1 algorithm has an aggressive exploitation mechanism of Pareto optimal arms that improves it upper regret bound. We denote this algorithm with the *exploitative Pareto UCB1* algorithm as opposite with the Pareto UCB1 algorithm from Drugan and Nowe (2013), denoted as *exploratory Pareto UCB1 algorithm*.

3.1 Exploitative Pareto UCB1

The pseudo-code for the *exploitative Pareto UCB1* is given in Algorithm 1. To initialise the algorithm, each arm is played once. Let $\hat{\mu}_i$ be the estimation of the true but unknown expected reward vector μ_i of an arm i. In each iteration, we compute for each arm i its index, i.e. the sum of the estimated reward vector $\hat{\mu}_i$ and the associated confidence value of arm i

$$\hat{\mu}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D})}{n_i}} =$$

$$\left(\hat{\mu}_i^1 + \sqrt{\frac{2\ln(n\sqrt[4]{D})}{n_i}}, \ldots, \hat{\mu}_i^D + \sqrt{\frac{2\ln(n\sqrt[4]{D})}{n_i}}\right)$$

At each time step t, the Pareto front $\mathcal{A}^{*(t)}$ is determined using the indexes $\hat{\mu}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D})}{n_i}}$. Thus, for

all arms not in the Pareto front $i \notin \mathcal{A}^{*(t)}$, there exists a Pareto optimal arm $h \in \mathcal{A}^{*(t)}$ that dominates arm i:

$$\hat{\mu}_h + \sqrt{\frac{2\ln(n\sqrt[4]{D})}{n_h}} \succ \hat{\mu}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D})}{n_i}}$$

Each iteration, the exploitative Pareto UCB1 algorithm selects *all* Pareto optimal arm from $\mathcal{A}^{*(t)}$ and pull them. Thus, by design, this algorithm is fair in selecting Pareto optimal arms. Next, the estimated vector of the selected arm $\hat{\mu}_h$ and the corresponding counters are updated. A possible stopping criteria is a given fix number of iterations.

The following theorem provides an upper bound for the Pareto regret of the efficient Pareto UCB1 strategy. The only difference is that a suboptimal arm is pulled $|\mathcal{A}^*|$ times less often than in the exploratory Pareto UCB1 algorithm. This fact is reflected by the multiplicative constant, $\sqrt[4]{D}$, in the index of the algorithm.

Theorem 1. *Let exploitative Pareto UCB1 from Algorithm 1 be run on a K-armed D-objective bandit problem, $K > 1$, having arbitrary reward distributions $\mathbf{P}_1, \ldots \mathbf{P}_K$ with support in $[0,1]^D$. Consider the Pareto regret defined in Equation 1. The expected Pareto projection regret of after any number of n plays is at most*

$$\sum_{i \notin \mathcal{A}^*} \frac{8 \cdot \ln(n\sqrt[4]{D})}{\Delta_i} + (1 + \frac{\pi^2}{3}) \cdot \sum_{i \notin \mathcal{A}^*} \Delta_i$$

Proof. The prove follows closely the prove from Drugan and Nowe (2013). Let $\mathbf{X}_{i,1}, \ldots, \mathbf{X}_{i,n}$ be random D-dimensional variables generated for arm i with common range $[0,1]^D$. The expected reward vector for the arm i after n pulls is

$$\bar{\mathbf{X}}_{i,n} = 1/n \cdot \sum_{t=1}^{n} \mathbf{X}_{i,t} \;\Rightarrow\; \forall j,\; \bar{X}^j_{i,n} = 1/n \cdot \sum_{t=1}^{n} X^j_{i,t}$$

Chernoff-Hoeffding bound. We use a straightforward generalization of the standard Chernoff-Hoeffding bound for D dimensional spaces. Consider that $\forall j, 1 \leq j \leq D$, $\mathbb{E}[\mathbf{X}^j_{i,t} \mid \mathbf{X}^j_{i,1}, \ldots, \mathbf{X}^j_{i,t-1}] = \mu^j_i$. There, $\bar{\mathbf{X}}_{i,n} \not\prec \mu_i + a$ if there exists *at least* a dimension j for which $\bar{X}^j_{i,n} > \mu^j_i + a$. Translated in Chernoff-Hoeffding bound, using union bound, for all $a \geq 0$, we have

$$\mathbb{P}\{\left(\bar{\mathbf{X}}_{i,n} \not\prec \mu_i + a\right)\} = \tag{4}$$

$$\mathbb{P}\{\left(\bar{X}^1_{i,n} > \mu^1_i + a\right) \vee \ldots \vee \left(\bar{X}^D_{i,n} > \mu^D_i + a\right)\} \leq De^{-2na^2}$$

Following the same line of reasoning

$$\mathbb{P}\{\left(\bar{X}^1_{i,n} < \mu^1_i - a\right) \vee \ldots \vee \left(\bar{X}^D_{i,n} < \mu^D_i - a\right)\} \leq De^{-2na^2} \tag{5}$$

Let $\ell > 0$ an arbitrary number. We take $c_{t,s} = \sqrt{2 \cdot \ln(t\sqrt[4]{D})/s}$, and we upper bound $T_i(n)$ on any sequence of plays by bounding for each $t \geq 1$ the indicator $(I_t = i)$. We have $(I_t = i) = 1$ if arm i is played at time t and $(I_t = i) = 0$ otherwise. We use the superscript $*$ when we mean a Pareto optimal arm. Thus, $T^*_h(n)$ means that the arm h is Pareto optimal, $h \in \mathcal{A}^*$. Then,

$$T_i(n) = 1 + \sum_{t=K+1}^{n} \{I_t = i\} \leq$$

$$\ell + \sum_{t=K+1}^{n} \{I_t = i, T_i(t-1) \geq \ell\} \leq \ell + \sum_{t=K+1}^{n} \frac{1}{|\mathcal{A}^*|} \cdot$$

$$\sum_{h=1}^{|\mathcal{A}^*|} \{\bar{\mathbf{X}}^*_{h,T^*_h(t-1)} + c_{t-1,T^*_h(t-1)} \not\prec \bar{\mathbf{X}}_{i,T_i(t-1)} + c_{t-1,T_i(t-1)}\}$$

$$\leq \begin{array}{c} s^*_h \leftarrow T^*_h(t-1) \\ s_i \leftarrow T_i(t-1) \end{array} \ell +$$

$$\sum_{t=1}^{\infty} \sum_{s=1}^{t-1} \sum_{s_i=\ell}^{t-1} \frac{1}{|\mathcal{A}^*|} \sum_{h=1}^{|\mathcal{A}^*|} \{\bar{\mathbf{X}}^*_{h,s^*_h} + c_{t-1,s^*_h} \not\prec \bar{\mathbf{X}}_{i,s_i} + c_{t-1,s_i}\} \tag{6}$$

From the straightforward generalization of Chernoff-Hoeffding bound to D objectives, we have that

$$\mathbb{P}\{\bar{\mathbf{X}}^{(t)}_i \not\prec \mu_i + c^{(t)}_s\} \leq \frac{D}{D} \cdot t^{-4} = t^{-4}$$

and

$$\mathbb{P}\{\bar{\mathbf{X}}^{*(t)}_h \not\succ \mu^*_h - c^{(t)}_{s^*_h}\} \leq t^{-4}$$

For $s_i \geq \frac{8 \cdot \ln(n\sqrt[4]{D})}{\Delta^2_i}$, we have that

$$\nu^*_i - \mu_i - 2 \cdot c_{t,s_i} = \nu^*_i - \mu_i - 2 \cdot \sqrt{\frac{2 \cdot \ln(n\sqrt[4]{D})}{s_i}} \geq \nu^*_i - \mu_i - \Delta_i$$

Thus, we take $\ell = \lceil \frac{8 \cdot \ln(n\sqrt[4]{D})}{\Delta^2_i} \rceil$, and we have

$$\mathbb{E}[T_i(n)] \leq \lceil \frac{8 \cdot \ln(n\sqrt[4]{D})}{\Delta^2_i} \rceil + \sum_{t=1}^{\infty} \sum_{s=1}^{t-1} \sum_{s_i = \lceil \frac{8 \cdot \ln(n\sqrt[4]{D})}{\Delta^2_i} \rceil}$$

$$\sum_{h=1}^{|\mathcal{A}^*|} (\mathbb{P}\{\bar{\mathbf{X}}^{*(t)}_h \not\succ \mu^*_h - c^{(t)}_{s^*_h}\} + \mathbb{P}\{\bar{\mathbf{X}}^{(t)}_i \not\prec \mu_i + c^{(t)}_{s_i}\})$$

$$\leq \frac{8 \cdot \ln(n \cdot \sqrt[4]{D})}{\Delta^2_i} + 1 + \sum_{t=1}^{\infty} \sum_{s=1}^{t} \sum_{s_i=1}^{t} \sum_{h=1}^{|\mathcal{A}^*|} t^{-4} \frac{t^{-4}}{|\mathbb{A}^*|} =$$

$$\leq \frac{8 \cdot \ln(n\sqrt[4]{D})}{\Delta^2_i} + 1 + 2 \cdot \sum_{t=1}^{\infty} t^2 \cdot |\mathcal{A}^*| \frac{t^{-4}}{|\mathcal{A}^*|} =$$

$$\frac{8 \cdot \ln(n\sqrt[4]{D})}{\Delta^2_i} + 1 + 2 \cdot \sum_{t=1}^{\infty} t^{-2}$$

Approximating the last term with the Riemann zeta function $\zeta(2) = \sum_{t=1}^{\infty} t^{-2} \approx \frac{\pi^2}{6}$ we obtain the bound from the theorem. □

For a suboptimal arm i, we have $\mathbb{E}[T_i(n)] \leq \frac{8}{\Delta_i^2} \ln(n\sqrt[4]{D})$ plus a small constant. Like for the standard UCB1, the leading constant is $8/\Delta_i^2$ and the expected upper bound of the Pareto regret for the exploitative Pareto UCB1 is logarithmic in the number of plays n. Unlike exploratory Pareto UCB1 Drugan and Nowe (2013), this expected bound does not depend on the cardinality of the Pareto front $\mathcal{A}^*$. This is an important improvement for the exploratory Pareto UCB1 since the size of the Pareto optimal arms is: i) usually not known beforehand, and ii) increases with the number of objectives.

Note that the algorithm reduces to the standard UCB1 for $D = 1$. Thus, exploitative Pareto UCB1 performs similarly with the standard UCB1 for small number of objectives. Consider that almost all the arms K are Pareto optimal arms, $|\mathcal{A}|^* \approx K$. Then, each iteration, the exploitative Pareto UCB1 algorithm pulls once (almost) all arms.

3.2 Exploratory Pareto UCB1

The exploratory version of Pareto UCB1 algorithm was introduced in Drugan and Nowe (2013) and it is a straightforward extension of the UCB1 algorithm to reward vectors. The main difference between the exploratory Pareto UCB1 and the exploitative Pareto UCB1, cf Algorithm 1, is in lines $6-8$ of the algorithm. For the exploratory Pareto UCB1 algorithm, each iteration, a single Pareto optimal arm is selected uniformly at random and pulled. The counters are updated accordingly, meaning that $n \leftarrow n+1$.

Another difference is the index associated to the mean vector that is larger than for the exploitative Pareto UCB1. Thus, the Pareto set is now the non-dominated vectors $\hat{\mu}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D|\mathcal{A}^*|})}{n_i}}$.

The regret bound for the exploratory Pareto UCB1 algorithm using Pareto regrets is logarithmic in the number of plays for a suboptimal arm and in the size of the reward vectors, D. In addition, this confidence bound is also logarithmic in the cardinality of Pareto front, $|\mathcal{A}^*|$. This indicates a poor behavior of the exploratory Pareto UCB1 for a large Pareto front approaching the number of total arms, which is usually the case for large number of objectives.

Algorithm 2 Exploitative Pareto UCB2

Require: $0 < \alpha < 1$; the length of a epoch r is an exponential function $\tau(r) = \lceil (1+\alpha)^r \rceil$

1: Play each arm once
2: $n \leftarrow K$; $r_i \leftarrow 1, \forall i$
3: **while** the stopping condition is NOT met **do**
4: Select the Pareto front at the epoch r, $\mathcal{A}^{*(r)}$, such that $\forall i \in \mathcal{A}^{*(r)}$, the index $\hat{\mu}_i + a_n^{\tau(r_i)}$ is non-dominated
5: **for all** $i \in \mathcal{A}^{*(t)}$ **do**
6: Pull the arm i exactly $\tau(r_i+1) - \tau(r_i)$
7: Update $\hat{\mu}_i$, and $r_i \leftarrow r_i + 1$
8: $r \leftarrow r+1$ and $n \leftarrow n + \tau(r+1) - \tau(r)$
9: **end for**
10: **end while**

4 The exploration vs exploration trade-off in Pareto UCB2

In this section, we propose Pareto MAB algorithms that extend of the standard UCB2 algorithm to reward vectors. Like for the standard UCB2, these Pareto UCB2 algorithms play the optimal arms in epochs. These epochs are exponential with the number of plays in order to allow the gradual selection of good arms to be played longer each epoch. In single objective MABs, the UCB2 algorithm is acknowledged to have a better upper regret bound than the UCB1 algorithm Auer et al. (2002). We show that Pareto UCB2 algorithms have a better upper Pareto projection regret bound than the Pareto UCB1 algorithms, considering the same exploitation vs exploration trade-off.

The first proposed Pareto UCB2 algorithm, see Section 4.1, plays in an epoch *all* Pareto optimal arms equally often. We call this algorithm an exploitative Pareto UCB2 algorithm. The second Pareto UCB2 algorithm introduced in Section 4.2 plays only *one* Pareto optimal arm per epoch. We call this algorithm an exploratory Pareto UCB2 algorithm.

4.1 Exploitative Pareto UCB2

In this section, we present the *exploitative Pareto UCB2* algorithm and we analyze its upper confidence bound. The pseudo-code for this algorithm is given in Algorithm 2.

As an initial step, we play each arm once. The plays are divided in epochs, r, of exponential length until a stopping criteria is met a fix number of arm' pulls. The length of an epoch is an exponential function $\tau(r) = \lceil (1+\alpha)^r \rceil$. In each epoch, we compute for

each arm i an index given by with the sum of expected rewards plus a second term for the confidence value

$$\hat{\mu}_i + a_n^{\tau(r_i)} \leftarrow \left(\hat{\mu}_i^1 + a_n^{\tau(r_i)}, \ldots, \hat{\mu}_i^D + a_n^{\tau(r_i)}\right)$$

where $a_n^{\tau(r_i)} = \sqrt{\frac{(1+\alpha)\cdot\ln(e\cdot n/(D\cdot\tau(r_i)))}{2\cdot\tau(r_i)}}$, and r_i is the number of epochs played by the arm i. A Pareto front $\mathcal{A}^{*(r)}$ is selected from all vectors $\hat{\mu}_i + a_n^{\tau(r_i)}$. Thus, $\forall i \in \mathcal{A}$, exists $h \in \mathcal{A}^{*(t)}$, such that we have

$$\hat{\mu}_h + a_n^{\tau(r_h)} \succ \hat{\mu}_i + a_n^{\tau(r_i)}$$

Each arm $i \in \mathcal{A}^{*(t)}$ is selected and played $\tau(r_i+1) - \tau(r_i)$ consecutive times. The mean value and the epoch counter for all Pareto optimal arms are updated accordingly, meaning that $r_i \leftarrow r_i + 1$. The total epoch counter, r, and the total number of arms' pulls n are also updated.

The following theorem bounds the expected regret for the Pareto UCB2 strategy from Algorithm 2.

Theorem 2. *Let exploitative Pareto UCB2 from Algorithm 2 be run on K-armed bandit, $K > 1$, having arbitrary reward distributions $\mathbf{P}_1, \ldots \mathbf{P}_K$ with support in $[0,1]^D$. Consider the regret defined in Equation 1.*

The expected regret of a strategy π after any number of $n \geq \max_{\hat{\mu}_i \notin \mathcal{A}^*} \frac{D}{2\cdot\Delta_i^2}$ plays is at most

$$\sum_{i:\hat{\mu}_i \notin \mathcal{A}^*} \left(D \cdot \frac{(1+\alpha)\cdot(1+4\cdot\alpha)\cdot\ln(2\cdot e\cdot\Delta_i^2\cdot n/D)}{2\cdot\Delta_i} + \frac{c_\alpha}{\Delta_i} \right)$$

where

$$c_\alpha = 1 + \frac{D^2\cdot(1+\alpha)\cdot e}{\alpha^2} + D^{\alpha+2}\cdot\left(\frac{\alpha+1}{\alpha}\right)^{(1+\alpha)}\cdot\left(1+\frac{11\cdot D\cdot(1+\alpha)}{5\cdot\alpha^2\cdot\ln(1+\alpha)}\right)$$

Proof. This prove is based on the homologue prove of Auer et al. (2002). We consider $n \geq \frac{D}{2\cdot\Delta_i^2}$, for all i. From the definition of $\tau(r)$ we can deduce that $\tau(r) \leq \tau(r-1)\cdot(1-\alpha)+1$.

Let $\tau(\tilde{r}_i)$ be the largest integer such that

$$\tau(\tilde{r}_i - 1) \leq \frac{D\cdot(1+4\cdot\alpha)\cdot\ln(2\cdot e\cdot n\cdot\Delta_i^2/D)}{2\cdot\Delta_i^2}$$

We have that for an suboptimal arm i

$$T_i(n) \leq 1 + \frac{1}{|\mathcal{A}^*|}\cdot\sum_{r\geq 1}(\tau(r)-\tau(r-1))\cdot\{\text{arm } i \text{ finished its } r\text{-th epoch }\}$$

$$\leq \tau(\tilde{r}_i) + \frac{1}{|\mathcal{A}^*|}\cdot\sum_{r>\tilde{r}_i}(\tau(r)-\tau(r-1))\cdot\{\text{arm } i \text{ finished its } r\text{-th epoch }\}$$

The assumption $n \leq D/(2\cdot\Delta_i^2)$ implies $\ln(2e\cdot n\Delta_i^2/D) \geq 1$. Therefore, for $r > \tilde{r}_i$, we have

$$\tau(r-1) > \frac{D\cdot(1+4\alpha)\cdot\ln(2e\cdot n\Delta_i^2/D)}{2\cdot\Delta_i^2} \quad (7)$$

and

$$a_n^{\tau(r-1)} = \sqrt{\frac{(1+\alpha)\ln(e\cdot n/(D\cdot\tau(r-1)))}{2\tau(r-1)}} \leq_{Eq\ 7} \frac{\Delta_i}{\sqrt{D}}\cdot\sqrt{\frac{(1+\alpha)\ln(e\cdot n/(D\cdot\tau(r-1)))}{(1+4\alpha)\ln(2e\cdot n\Delta_i^2/D)}} \leq \frac{\Delta_i}{\sqrt{D}}\cdot\sqrt{\frac{(1+\alpha)\ln(2e\cdot n\Delta_i^2/D))}{(1+4\alpha)\ln(2e\cdot n\Delta_i^2/D)}} \leq \frac{\Delta_i}{\sqrt{D}}\cdot\sqrt{\frac{1+\alpha}{1+4\alpha}}$$

Because $a_t^{\tau(r)}$ is increasing in t, by definition, if the suboptimal arm j finishes to play the r-th epoch then $\forall h,\ 1 \leq h \leq |\mathcal{A}^{*(r)}|,\ \exists s_h \geq 0,\ \exists t \geq \tau(r-1) + \tau(s_h)$ such that arm i is non-dominated by any of the Pareto optimal arms in $|\mathcal{A}^{*(r)}|$. This means that

$$\bar{\mathbf{X}}_h^{*\tau(s_h)} + a_t^{s_h} \not\succ \bar{\mathbf{X}}_i^{\tau(r-1)} + a_t^{\tau(r-1)}$$

implies that one of the following conditions holds

$$\bar{\mathbf{X}}_i^{\tau(r-1)} + a_n^{\tau(r-1)} \not\succ \nu_i^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}$$

or

$$\bar{\mathbf{X}}_h^{*\tau(s_h)} + a_{\tau(r-1)+\tau(s_h)}^{\tau(s_h)} \not\succ \mu_h^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}$$

Then,

$$I\!E[T_i(n)] \leq \tau(\tilde{r}_i) + \sum_{r\geq\tilde{r}_i}\frac{\tau(r)-\tau(r-1)}{|\mathcal{A}^*|}\cdot \quad (8)$$

$$\sum_{h=1}^{|\mathcal{A}^*|} I\!P\{\bar{\mathbf{X}}_i^{\tau(r-1)} + a_n^{\tau(r-1)} \not\succ \nu_i^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\} + \sum_{i\geq 0}\sum_{r\geq 1}\frac{\tau(r)-\tau(r-1)}{|\mathcal{A}^*|}\cdot \sum_{h=1}^{|\mathcal{A}^*|} I\!P\{\bar{\mathbf{X}}_{s_h}^{\tau(r-1)} + a_{\tau(r-1)-\tau(s_h)}^{\tau(s_h)} \not\succ \mu_h^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\}$$

Let's expand Inequation 8 using Chernoff and union bound. For the first term between the parenthesis, we have that

$$I\!P\{\bar{\mathbf{X}}_i^{\tau(r-1)} + a_n^{\tau(r-1)} \not\succ \nu_i^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\} =$$

$$\sum_{j=1}^{D} I\!P\{\bar{X}_i^{j\tau(r-1)} + a_n^{\tau(r-1)} > \mu_i^j + \Delta_i - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\} \leq$$

$$D\cdot e^{-2\cdot\tau(r-1)\cdot\Delta_i^2\cdot(1-\frac{\alpha}{2\cdot\sqrt{D}}-\frac{1}{\sqrt{D}}\cdot\frac{1+\alpha}{1+4\cdot\alpha})^2}$$

$$\leq_{\alpha<1/10} D\cdot e^{-\frac{\tau(r-1)\cdot\Delta_i^2\cdot\alpha^2}{2\cdot D}}$$

If $g(x)=\frac{x-1}{1+\alpha}$ and $c=\frac{\Delta_i^2\cdot\alpha^2}{D}$, and $g(x)\leq\tau(r-1)$ then

$$\sum_{r\geq 1}\frac{\tau(r)-\tau(r-1)}{|\mathcal{A}^*|}\cdot$$

$$\sum_{h=1}^{|\mathcal{A}^*|} I\!P\{\bar{\mathbf{X}}_i^{\tau(r-1)} + a_n^{\tau(r-1)} \not\prec \mathbf{v}_i^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\} \leq$$

$$\sum_{r\geq 1}\sum_{i\geq 0}\frac{\tau(r)-\tau(r-1)}{|\mathcal{A}^*|}\cdot\sum_{h=1}^{|\mathcal{A}^*|} D\cdot e^{-\tau(r-1)\cdot\Delta_i^2\cdot\alpha^2/D} =$$

$$D\cdot|\mathcal{A}^*|\cdot\sum_{r\geq 1}\sum_{i\geq 0}\frac{\tau(r)-\tau(r-1)}{|\mathcal{A}^*|}\cdot e^{-\tau(r-1)\cdot\Delta_i^2\cdot\alpha^2/D} \leq$$

$$\frac{D}{|\mathcal{A}^*|}\cdot|\mathcal{A}^*|\cdot\int_0^{\infty} e^{-c\cdot g(x)}dx \leq \frac{D^2\cdot(1+\alpha)\cdot e}{\Delta_i^2\cdot\alpha^2}$$

Let's now expand the second term of the parenthesis in Inequation 8

$$I\!P\{\bar{\mathbf{X}}_s^{\tau(r-1)} + a_{\tau(r-1)-\tau(s)}^{\tau(s)} \not\succ \mu_h^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\} =$$

$$\sum_{j=1}^{D} I\!P\{\bar{X}_s^{j\tau(r-1)} + a_{\tau(r-1)-\tau(s)}^{\tau(s)} < \mu_h^{j*} - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\} \leq$$

$$D\cdot e^{-\tau(i)\cdot\frac{\alpha^2\cdot\Delta_i^2}{D\cdot 2}}\cdot e^{-(1+\alpha)\cdot\ln\frac{e\cdot(\tau(r-1)+\tau(i))}{D\cdot\tau(i)}} \leq$$

$$D^{\alpha+2}\cdot e^{-\tau(i)\cdot\frac{\alpha^2\cdot\Delta_i^2}{D\cdot 2}}\cdot\left(\frac{\tau(r-1)+\tau(i)}{\tau(i)}\right)^{-(1+\alpha)}$$

Thus,

$$\sum_{i\geq 0}\sum_{r\geq 1}\frac{\tau(r)-\tau(r-1)}{|\mathcal{A}^*|}\cdot$$

$$\sum_{h=1}^{|\mathcal{A}^*|} I\!P\{\bar{\mathbf{X}}_s^{\tau(r-1)} + a_{\tau(r-1)-\tau(s)}^{\tau(s)} \not\succ \mu_h^* - \frac{\alpha\cdot\Delta_i}{\sqrt{D}\cdot 2}\} \leq$$

$$D^{\alpha+2}\cdot\sum_{i\geq 0} e^{-\tau(i)\cdot\frac{\alpha^2\cdot\Delta_i^2}{D\cdot 2}}\cdot\int_0^{\infty}\left(1+\frac{x-1}{(1+\alpha)\cdot\tau(i)}\right)^{-(1+\alpha)}dx \leq$$

$$D^{\alpha+2}\cdot\frac{\alpha}{(1+\alpha)-1}\cdot\left(\frac{\alpha+1}{\alpha}\right)^{(1+\alpha)}\cdot\sum_{i\geq 0}\tau(i)\cdot e^{-\tau(i)\cdot\frac{\alpha^2\cdot\Delta_i^2}{D\cdot 2}}$$

Following the rationale from the prove of Theorem 2 from Auer et al. (2002), we can bound further the first term of Inequation 8 to

$$\sum_{i\geq 0}\tau(i)\cdot e^{-\tau(i)\cdot\frac{\alpha^2\cdot\Delta_i^2}{D\cdot 2}} \leq 1+\frac{11\cdot D\cdot(1+\alpha)}{5\alpha^2\cdot\Delta_i^2\cdot\ln(1+\alpha)}$$

Using the bounds above, we now bound the expected regret for an arm i in Algorithm 2

$$I\!E[T_i(n)] \leq \tau(\tilde{r}_i) - 1 + \frac{c_\alpha}{\Delta_i^2}$$

where

$$c_\alpha = 1+\frac{D^2\cdot(1+\alpha)\cdot e}{\alpha^2}+$$

$$D^{\alpha+2}\cdot\left(\frac{\alpha+1}{\alpha}\right)^{(1+\alpha)}\cdot\left[1+\frac{11\cdot D\cdot(1+\alpha)}{5\cdot\alpha^2\cdot\ln(1+\alpha)}\right]$$

and the upper bound on $\tau(\tilde{r}_i)$

$$\tau(\tilde{r}_i) \leq \tau(\tilde{r}_i-1)(1+\alpha)+1 \leq$$

$$\frac{D\cdot(1+\alpha)\cdot(1+4\alpha)\cdot\ln(2en\Delta_i^2/D)}{2\cdot\Delta_i^2}+1$$

This concludes our prove. □

The bound of the expected regret for Pareto UCB2 is the similar with the bound for the standard UCB2 within a constant given by the number of objectives D. The intuition is that now the algorithm has to run D times longer to achieve a similar regret bound for the Pareto UCB2. For α small, the Pareto projection regret of this Pareto algorithm is bounded by $\frac{1}{2\cdot\Delta_i^2}$. This is a better bound than for the Pareto UCB1 algorithm, $\frac{8}{\Delta_i^2}$.

The difference between single objective and Pareto UCB2 is in the constant c_α which is smaller than the same constant for the standard UCB2 for $\alpha>0$. This means that the constant c_α converges faster to infinity when $\alpha\to 0$.

4.2 Exploratory Pareto UCB2

In this section, we introduce the *exploratory Pareto UCB2* algorithm. In fact, the only difference between the exploratory and exploitative variants of Pareto UCB2 is in lines 5 from Algorithm 2. Now a single arm from the Pareto front at epoch r, $\mathcal{A}^{*(r)}$, is selected and played the entire epoch, i.e. for $\tau(r_i+1)-\tau(r_i)$ consecutive times.

Since the length of the epochs is exponential, a single Pareto optimal arm is played longer and longer. Thus, the exploitation mechanism of Pareto optimal arms of the exploratory Pareto UCB2 algorithm is poor, and the upper Pareto projection regret depends on the cardinality of the Pareto front.

5 Numerical simulations

In this section, we compare the performance of five Pareto MAB algorithms: 1) a baseline algorithm,

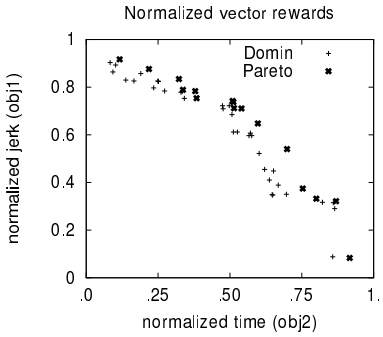

Figure 1: All the points generated by the bi-objective wet-clutch application.

2) two Pareto UCB1 algorithms and 3) two Pareto UCB2 algorithms. As announced in the introduction, the test problem is a bi-objective stochastic environment generated by a real world control application.

The algorithms. The five Pareto MAB algorithms compared are

tPUCB1 The exploitative Pareto UCB1 algorithm introduced in Section 3.1;

rPUCB1 The exploratory Pareto UCB1 algorithm summarised in Section 3.2;

tPUCB2 The exploitative Pareto UCB2 algorithm summarised in Section 4.1;

rPUCB2 The exploratory Pareto UCB2 algorithm summarised in Section 4.2;

hoef A baseline algorithm for multi-armed bandits in general is the Hoeffding race algorithm Maron and Moore (1994) where all the arms are pulled equally often and the arms with the non-dominated empirical mean reward vectors are chosen.

Each algorithm is run 100 times with a fixed budged, or arm' pulls, of $N = 10^6$. By default, we set the α parameter for the two Pareto UCB2 algorithms to 1.

The wet clutch application. In order to optimise the functioning of the wet clutch Vaerenbergh et al. (2012) it is necessary to simultaneously minimise 1) the optimal current profile of the electro-hydraulic valve that controls the pressure of the oil in the clutch, and 2) the engagement time. The piston of the clutch gets in contact with the friction plates to change the profile of the valve. Such a system is characterised by a hard non-linearity. Additionally, external factors that cannot be controlled exactly, e.g. the surrounding temperature, make this a stochastic control application. Such clutches are typically used in the power transmission of off-road vehicles that has to operate under strongly varying environmental conditions. And the goal in this control problem is to minimise both the clutch's profile and the engagement time under varying environmental conditions.

$\mu_1^* = (0.116, 0.917)$	$\mu_2^* = (0.218, 0.876)$	$\mu_3^* = (0.322, 0.834)$
$\mu_4^* = (0.336, 0.788)$	$\mu_5^* = (0.379, 0.783)$	$\mu_6^* = (0.383, 0.753)$
$\mu_7^* = (0.509, 0.742)$	$\mu_8^* = (0.512, 0.737)$	$\mu_9^* = (0.514, 0.711)$
$\mu_{10}^* = (0.540, 0.710)$	$\mu_{11}^* = (0.597, 0.647)$	$\mu_{12}^* = (0.698, 0.540)$
$\mu_{13}^* = (0.753, 0.374)$	$\mu_{14}^* = (0.800, 0.332)$	$\mu_{15}^* = (0.869, 0.321)$
	$\mu_{16}^* = (0.916, 0.083)$	
$\mu_{17} = (0.249, 0.826)$	$\mu_{18} = (0.102, 0.892)$	$\mu_{19} = (0.497, 0.722)$
$\mu_{20} = (0.251, 0.824)$	$\mu_{21} = (0.249, 0.826)$	$\mu_{22} = (0.102, 0.892)$
$\mu_{23} = (0.497, 0.722)$	$\mu_{24} = (0.251, 0.824)$	$\mu_{25} = (0.575, 0.596)$
$\mu_{26} = (0.651, 0.448)$	$\mu_{27} = (0.571, 0.607)$	$\mu_{28} = (0.083, 0.903)$
$\mu_{29} = (0.696, 0.350)$	$\mu_{30} = (0.272, 0.784)$	$\mu_{31} = (0.601, 0.521)$
$\mu_{32} = (0.341, 0.753)$	$\mu_{33} = (0.507, 0.685)$	$\mu_{34} = (0.526, 0.611)$
$\mu_{35} = (0.189, 0.857)$	$\mu_{36} = (0.620, 0.454)$	$\mu_{37} = (0.859, 0.314)$
$\mu_{38} = (0.668, 0.388)$	$\mu_{39} = (0.334, 0.782)$	$\mu_{40} = (0.864, 0.290)$
$\mu_{41} = (0.473, 0.722)$	$\mu_{42} = (0.822, 0.316)$	$\mu_{43} = (0.092, 0.863)$
$\mu_{44} = (0.234, 0.796)$	$\mu_{45} = (0.476, 0.709)$	$\mu_{46} = (0.566, 0.596)$
$\mu_{47} = (0.166, 0.825)$	$\mu_{48} = (0.646, 0.349)$	$\mu_{49} = (0.137, 0.829)$
$\mu_{50} = (0.511, 0.611)$	$\mu_{51} = (0.637, 0.410)$	$\mu_{52} = (0.329, 0.778)$
$\mu_{53} = (0.649, 0.347)$	$\mu_{54} = (0.857, 0.088)$	

Table 1: Fifty-four bi-dimensional reward vectors labelled from 1 to 54 for the wet clutch application. The first sixteen reward vectors are labeled from μ_1^* till μ_{16}^* and are Pareto optimal, while the last thirty-four reward vectors are labelled from μ_{17} till μ_{54} and they are suboptimal.

In Figure 1, we give 54 points generated with the wet clutch application, each point representing a trial of the machine and the jerk time obtained in the given time. The problem was a minimisation problem that we have transformed into a maximisation problem, by first normalising each objective with values between 0 and 1, and then transforming it into a maximisation problem. The best set of incomparable reward vectors is called the Pareto optimal reward set, i.e. there are 16 such reward vectors. In our example, $|\mathcal{A}^*|$ is about one-third from the total number of arms, i.e. $16/54$, and is a mixture of convex and non-convex regions. In Table 1, we show the mean values of the 54 reward vectors.

The performance of the algorithms. We use four metrics to measure the performance of the five tested Pareto MAB algorithms. Two of these metrics are the Pareto projection regret, cf Equation 2, and the Pareto variance regret, cf Equation 3, presented in Section 2. We also use two additional metrics two explain the dynamics of the Pareto MAB algorithms.

The third metric measures the percentage of times each Pareto optimal arm is pulled. Thus, for all Pareto optimal arms, $i \in \mathcal{A}^*$, we measure $\mathbb{E}[T_i^*(n)]$ the expected number of times the arm i is pulled during n total arm pulls. Note that $\mathbb{E}[T_i^*(n)]$ is a part of Equation 3 and it gives a detailed understanding of the

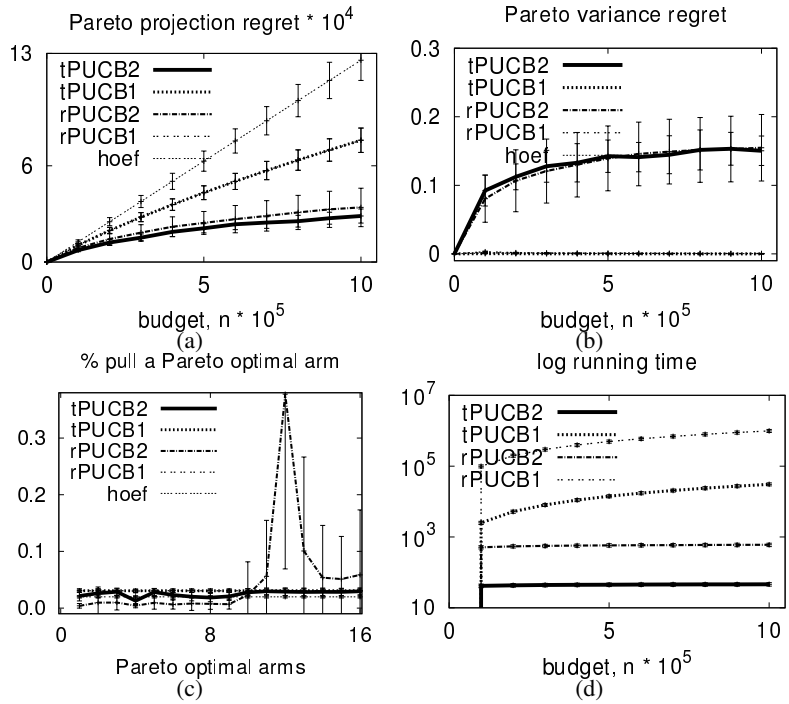

Figure 2: The performance of the five MOMAB algorithms on the wet clutch problem: a) the Pareto projection regret, b) the Pareto variance regret, c) the percentage of times each Pareto optimal arm is pulled, and d) the running time in terms of comparisons between arms and Pareto front for each MOMAB algorithm. The five algorithms are: 1) *tPUCB1* the exploitative Pareto UCB1, 2) *rPUCB1* the exploratory Pareto UCB1, 3) *tPUCB2* the exploitative Pareto UCB2, 4) *rPUCB2* the exploratory Pareto UCB2, and 5) *hoef* the Hoeffding race algorithm.

Pareto variance regret.

The last metric is a measure of the running time of each algorithm, and it is given by the number of times each arm in $\mathcal{A}$ was compared against the other arms in $\mathcal{A}$ in order to compute the Pareto front. Note that for the exploratory algorithms, i.e. *rPUCB1* and *rPUCB2*, each arm pull corresponds to one estimation of the Pareto front, whereas, for the exploitative algorithms, i.e. *tPUCB1* and *tPUCB2*, one estimation of $\mathcal{A}^*$ corresponds to the arms' pulls of the entire set.

5.1 Comparing MOMAB algorithms

In Figure 2, we compare the performance of the five MOMAB algorithms. According to the Pareto projection regret, cf. Figure 2 a), the best performing algorithm is the exploitative Pareto UCB2, cf. tPUCB2, the second best algorithm is the exploratory Pareto UCB2, cf rPUCB2, and the worst algorithm is the Hoeffding race algorithm, cf. hoef. Note that the Pareto UCB1 family of algorithms has a (almost) linear regret whereas Pareto UCB2 algorithms have a logarithmic regret, like the single objective UCB2 algorithm. The worst performance of the exploitative Pareto UCB1 algorithm can be explained by the poor explorative behaviour of the algorithm. The performance of the explorative Pareto UCB1 is in-between linear and logarithmic and can be explained by the improved exploratory technique of pulling all the Pareto optimal arms each round. Both Pareto UCB2 algorithms perform better than Pareto UCB1 algorithms because the Pareto optimal arms are explored longer each round.

In opposition, according to the Pareto variance regret, cf. Figure 2 b), the worst performing algorithms are the exploitative and exploratory Pareto UCB2 algorithms and the best algorithms are the exploratory and exploitative Pareto UCB1 algorithms but also the Hoeffding race algorithm. It is interesting to note that the difference in Pareto variance and projection regret between the exploratory and exploitative variance of the same algorithms is small. In general, Pareto UCB1 algorithms have a larger Pareto projection regret then the Pareto UCB2 algorithms, but a smaller Pareto variance regret.

Figure 2 c) explains these contradictory results with the percentage of times each of the Pareto optimal arms are pulled. As noticed in Section 4.2, the exploratory Pareto UCB2, cf rPUCB2, pulls the same Pareto optimal arm each epoch longer and longer,

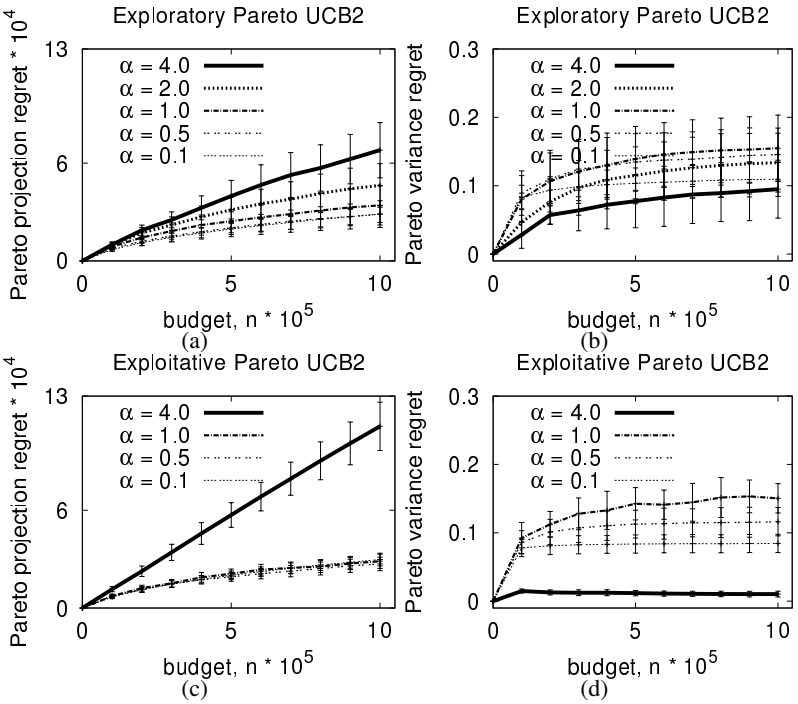

Figure 3: The performance of the two version of Pareto UCB2 algorithms, i.e. exploratory and exploitative Pareto UCB2, given for the five values of the $\alpha = \{0.1, 0.5, 1.0, 2.0, 4.0\}$ parameter.

generating the peak in the figure on one random single Pareto optimal arm. In contrast, the exploitative Pareto UCB2, cf. tPUCB2, is fair in exploiting the entire Pareto front. In the sequel, the exploratory Pareto UCB1 algorithm, cf rPUCB1, has more variance in pulling Pareto optimal arms than the exploitative Pareto UCB2 algorithm, cf tPUCB1, and this fact is reflected also in the Pareto variance regret measures from Figure 2 b).

The percentage of time of the *any* Pareto optimal arms is pulled: 1) for the exploitative Pareto UCB2 is $83\% \pm 8.5$, 2) for the explorative Pareto UCB2 is $77\% \pm 10.9$, 3) for the exploitative Pareto UCB1 is $49\% \pm 4.9$, and 4) for the explorative UCB1 is $49\% \pm 4.9$. Note the large difference between the efficiency of Pareto UCB2 and Pareto UCB1 algorithms.

In Figure 2 d), we show that the running time, i.e. number of comparisons between arms, for exploratory MOMABs, i.e. the exploratory Pareto UCB1 and the exploratory Pareto UCB2, are order of magnitude larger than the exploitative MOMAB algorithms, i.e. the exploitative Pareto UCB1 and the exploitative Pareto UCB2. The running time for Pareto UCB1 algorithms which compute the Pareto front often is larger than the running time for Pareto UCB2 algorithms that compute the Pareto front once in the beginning of an epoch. The most computational efficient is the exploitative Pareto UCB2 and the worst algorithm is the exploratory Pareto UCB1.

5.2 Exploration vs exploitation mechanism in Pareto UCB2 algorithms

In our second experiment, we measure the influence of the parameter α on the performance of Pareto UCB2 algorithms. Figure 3 considers five values for this parameter $\alpha = \{0.1, 0.5, 1.0, 2.0, 4.0\}$ that indicates the length of an epoch. The largest variance in performance we have for the exploratory Pareto UCB2. The smaller is the size of an epoch, the better the performance of the exploratory Pareto UCB2 algorithm is in terms of Pareto projection regret and Pareto variance regret. Note that for epochs' length of 1, the Pareto UCB2 algorithms resemble the Pareto UCB1 algorithms, meaning that an arm or a set of arms are pulled each epoch. Of course, the two algorithms have a different exploration index. The same parameter α has little influence on the performance of exploitative Pareto UCB2 algorithm where all the Pareto arms are pulled each epoch.

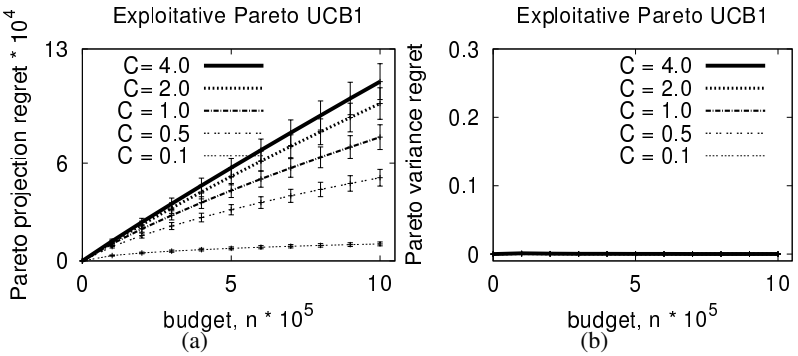

Figure 4: The performance of the exploitative Pareto UCB1 algorithm given five values of the $C = \{0.1, 0.5, 1.0, 2.0, 4.0\}$ parameter multiplying the index value.

5.3 Exploration vs exploitation trade-off in Pareto UCB1 algorithms

To study the influence of the exploration index for the Pareto UCB1 algorithm, we multiply the index of exploitative Pareto UCB1 with a constant C that takes five values $C = \{0.1, 0.5, 1.0, 2.0, 4.0\}$. Unlike for the exploitative Pareto UCB2 algorithm, the constant C has a big influence on the performance of Pareto UCB1 algorithms. The smaller is the multiplication constant, the better is the performance of the exploitative Pareto UCB1 algorithm. This means that an exploitative Pareto UCB1 algorithm performs the best with a small exploration index.

6 Conclusion

In this paper, we investigate the exploration vs exploitation trade-off in two of the infinite horizon MABs. The classical UCB1 and UCB2 algorithms are extended to reward vectors those quality is classified with Pareto dominance relation. We analytically and experimentally study the regret, i.e. the Pareto projection regret and the Pareto variance regret, of the proposed MOMAB algorithms.

We propose the exploitative Pareto UCB1 algorithm that each round pulls all the Pareto optimal arms. The exploratory version of the same algorithm uniform at random selects each round only one arm Pareto optimal arm. We show that this difference has an important impact on the upper Pareto projection regret bound of the exploitative Pareto UCB1 algorithm. Now, the upper regret bound is independent of the cardinality of the Pareto front, which is large for many objective environments, and, furthermore, unknown beforehand.

Based on the same principle, we propose the exploratory and exploitative Pareto UCB2 algorithms. The exploratory Pareto UCB2 algorithm pulls each epoch a single Pareto optimal arm selected at random. The exploitative Pareto UCB2 pulls, each epoch, equally often all the Pareto optimal arms. We upper bound the Pareto projection regret of the exploitative Pareto UCB2 algorithm.

We compare these algorithms also experimentally on a bi-objective problem coming from control theory. Our conclusion is that the exploration vs exploitation trade-off is better in the exploitative Pareto algorithms where all the Pareto optimal arms are pulled often. In opposition, the exploratory Pareto UCB2 algorithm has a small Pareto projective variance regret but a large Pareto variance regret since the algorithm pulls a single Pareto optimal arm during exponentially large epochs.

7 Acknowledgements

Madalina M. Drugan was supported by the IWT-SBO project PERPETUAL (gr. nr. 110041).

REFERENCES

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2/3):235–256.

Drugan, M. and Nowe, A. (2013). Designing multi-objective multi-armed bandits: a study. In *Proc of International Joint Conference of Neural Networks (IJCNN)*.

Lizotte, D., Bowling, M., and Murphy, S. (2010). Efficient reinforcement learning with multiple reward functions for randomized clinical trial analysis. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning (ICML)*.

Maron, O. and Moore, A. (1994). Hoeffding races: Accelerating model selection search for classification and function approximation. In *Advances in Neural Information Processing Systems*, volume 6, pages 59–66. Morgan Kaufmann.

Vaerenbergh, K. V., Rodriguez, A., Gagliolo, M., Vrancx, P., Nowe, A., Stoev, J., Goossens, S., Pinte, G., and Symens, W. (2012). Improving wet clutch engagement with reinforcement learning. In *International Joint Conference on Neural Networks (IJCNN)*. IEEE.

van Moffaert, K., Drugan, M., and Nowe, A. (2013). Hypervolume-based multi-objective reinforcement learning. In *Proc of Evolutionary Multi-objective Optimization (EMO)*. Springer.

Wang, W. and Sebag, M. (2012). Multi-objective Monte Carlo tree search. In *Asian conference on Machine Learning*, pages 1–16.

Wiering, M. and de Jong, E. (2007). Computing optimal stationary policies for multi-objective markov decision processes. In *Proc of Approximate Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 158–165. IEEE.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE T. on Evol. Comput.*, 7:117–132.