ε-approximate Pareto optimal set of arms identification in multi-objective multi-armed bandits

Mădălina M. Drugan MDRUGAN@VUB.AC.BE
Computational Modeling group, Artificial Intelligence Lab, Pleinlaan 2, B-1050, Brussels

Ann Nowé ANOWE@VUB.AC.BE
Computational Modeling group, Artificial Intelligence Lab, Pleinlaan 2, B-1050, Brussels

Keywords: Multi-objective optimization, Multi-armed bandits, ϵ-dominance relation, Best arm identification

Abstract

Many real-world stochastic environments are inherently multi-objective environments with multiple possibly conflicting objectives. Techniques from multi-objective optimization are imported into the multi-armed bandits (MAB) problem for efficient exploration/exploitation mechanisms of reward vectors. We introduce the ε-approximate Pareto MAB algorithm that uses the ε-dominance relation such that its upper confidence bound does not depend on the number of best arms, an important feature for environments with relatively many optimal arms. We experimentally show that the ε-approximate Pareto MAB algorithms outperform the performance of the Pareto UCB1 algorithm on a multi-objective Bernoulli problem inspired by a real world control application.

1. Introduction

There are many interesting applications in the field of automatic control where one wants to simultaneously fulfill different criteria (or objectives). Objectives can be aligned as well as conflicting, resulting in complex Pareto fronts that cannot be assumed to be convex. Furthermore, it is hard to a priori weight this criteria and an approach that combines different objectives using predefined weights is not feasible. Multi-armed bandits (Auer et al., 2002) is a machine learning paradigm used to study and analyze resource allocation in stochastic and noisy environments. A variant of multi-armed bandits that uses reward vectors instead of reward values was introduced in (Drugan & Nowe, 2013) and has been named *multi-objective multi-armed bandits* (MO-MABs). Some of these techniques were also imported in other related learning paradigms like multi-objective reinforcement learning (van Moffaert et al., 2013; Wang & Sebag, 2012; Roijers et al., 2013).

Multi-objective MABs lead to important differences to the standard MABs. There can be several arms considered to be the best according to their reward vectors and, thus, dominance relations specific for the multi-objective search spaces, i.e. Pareto or scalarized dominance relation, should be considered. Moreover, when the multi-objective environments are large and complex, i.e. ϵ-Pareto sets, can be considered to efficiently explore the Pareto front.

The exploitation/exploration trade-off is important in both EAs (Evolutionary Algorithms) and MABs but with different meaning. *Exploration* in MABs means to choose suboptimal actions, whereas in EAs exploration means to generate solutions in unexplored regions of the search space. *Exploitation* in MAB means to select the best option from an available set, while in EAs exploitation means to generate new solutions in promising regions of the search space using the commonalities of current solutions.

Main contributions. In this paper, we propose a Pareto MAB algorithm that improves the exploration/exploitation properties of the Pareto UCB1 algorithm (Drugan & Nowe, 2013) for large sets of optimal arms. The main idea of this Pareto MAB algorithm is inspired by: i) the MAB algorithms that successively remove the sub-optimal arms (Audibert et al., 2010), and ii) Pareto ϵ-dominance optimization algorithm where a subset of Pareto optimal arms are considered. The best arm identification MAB algorithm (Audibert et al., 2010) identifies a single optimal arm, whereas in its generalization (Bubeck et al., 2013) the m-best arms are identified. We extend this concept for multi-objective environments.

Preliminary work. Under review for BENELEARN 2014. Do not distribute.

The *ϵ-approximate Pareto optimal set of arms identification algorithm* (POAI) is a variant of the best arm identification algorithm where the Pareto ϵ-dominance relation is used. The multi-objective environment is considered a hypergrid, and the arms are considered part of the hypercubes inside the hypergrid. At first, the algorithm assigns arms to hypercubes in order to deterministically delete dominated hypercubes rather than the dominated arms. The resulting algorithm is considered a naive PAC algorithm and the probability that an arm is assigned to the wrong box is bounded. Second, the Pareto MAB algorithm selects a single non-dominated arm in each hypercube using a similar approach with the Pareto optimal set of arms identification algorithm but without the severe assumption that the size of the Pareto optimal set of arms is known beforehand. The upper confidence bound of the POAI algorithm does not depend on the size of the Pareto optimal set of arms. It depends on the number of non-dominated hypercubes instead, and this is a parameter that can be tuned by the user.

In the experimental section, we consider the *Hoeffding race* (Maron & Moore, 1994) as the baseline for comparison with the two Pareto optimal set of arms identification algorithms proposed here. The three MO-MAB algorithms are tested on data from a real world application that comes from control theory. As expected, the two algorithms introduced here outperform the Hoeffding race algorithm, and the ϵ-Pareto optimal set of arms identification algorithm is the best performing algorithm.

Outline. Section 2 introduces some background knowledge. In Section 3, we present the Pareto UCB1 algorithm. Section 4 introduces the ϵ-Pareto optimal problem. Section 5 shows the usage of MAB with the ϵ-Pareto optimal problem. Section 6 introduces a combination between the ϵ-Pareto optimal problem and the best arm identification algorithm. In Section 7, we compare the proposed algorithms with Pareto UCB1 algorithm. Section 8 concludes the paper.

2. Background

Let's consider the definition of a multi-armed bandit algorithm where only one arm is played at a time and there are fixed equal range stochastic reward vectors for each arm. When arm i is played at time steps t_1,t_2,..., the corresponding reward vectors $\mathbf{X}_i^{t_1}$, $\mathbf{X}_i^{t_2}$, ... are independently and identically distributed according to an unknown law with unknown expectation vector. The independence also holds between the arms.

Let's consider a K-armed bandit, $K \geq 2$, and let $\mathcal{I}$ be the set of K arms. In the multi-objective setting, the expected reward of each bandit i is multi-dimensional, $\mu_i = (\mu_i^1, \ldots, \mu_i^D)$, where D is a fixed number of dimensions, or objectives. We consider the general case where a mean reward vector can be better than another mean reward vector in one dimension, and worse in another dimension. This means that the objectives might be conflicting as well as correlated.

The Pareto dominance relation (Zitzler et al., 2003) orders the reward vectors. A reward vector μ is considered better than, or *dominating*, another reward vector ν, $\nu \prec \mu$, if and only if there exists at least one dimension j for which $\nu^j < \mu^j$, and for all other dimensions o, we have $\nu^o \leq \mu^o$. A reward vector μ is considered *incomparable* with another reward vector ν, $\nu \| \mu$, if and only if there exists at least one dimension j for which $\nu^j < \mu^j$, and there exists another dimension o, for which $\nu^o > \mu^o$. We say that μ is *non-dominated* by ν, $\nu \nsucc \mu$, if and only if there exists at least one dimension j for which $\nu^j < \mu^j$.

Let $\mathcal{I}^*$ be the Pareto optimal set of arms, i.e. non-dominated by all other arms, and let the *Pareto optimal reward set* $\mathcal{O}^*$ be their reward vectors. Although these two sets are equivalent, we will use them alternatively when convenient.

2.1. The Pareto regret metric

The *Pareto regret* is introduced in (Drugan & Nowe, 2013) and it estimates the distance between a suboptimal arm and the Pareto optimal set of arms $\mathcal{I}^*$.

To approximate the smallest distance between $\mathcal{O}^*$ and μ_i, we construct a *virtual* reward vector that is incomparable with *all* the reward vectors from $\mathcal{O}^*$. We add to μ_i a positive value ϵ in all objectives, resulting in a so called virtual reward vector $\nu_{i,\epsilon}$, where $\nu_{i,\epsilon}^j = \mu_i^j + \epsilon^j$ and $\epsilon^j > 0$, $\forall j$. For simplicity, we assume the ϵ has equal values in all dimensions, $\forall j, k$, $\epsilon^j = \epsilon^k$. The virtual optimal reward for the arm i, ν_i^* has the minimum value for ϵ for which $\nu_{i,\epsilon}$ is incomparable to all the rewards in $\mathcal{O}^*$.

The distance between the virtual optimal reward vector of the arm i, ν_i^*, and the mean reward vector of the same arm, μ_i, is denoted with

$$\Delta_i = \sqrt{\sum_{j=1}^{D} (\nu_i^{*j} - \mu_i^j)^2} = \sqrt{D}\epsilon_i \tag{1}$$

Since by definition ϵ_i is always positive, this regret is always positive. Note that the distance between the Pareto optimal arms and $\mathcal{O}^*$ is 0 because the corresponding virtual rewards coincide with the optimal reward vector itself.

An algorithm A selects the next arm to play based on the list of past plays and obtained reward vectors. Let $T_i(N)$ be the number of times a suboptimal arm i has been played by A during the first N plays. The expected regret after the first N plays is $\sum_{i \notin \mathcal{I}^*} \Delta_i \cdot \mathbb{E}[T_i(N)]$, the expected

Algorithm 1 Pareto UCB1

```
Play each arm $i$ once
$n \leftarrow K; n_i \leftarrow 1, \forall i; t \leftarrow 0$
while stopping criteria NOT met do
    Find the current Pareto optimal set $I^{*(t)}$ for all $i \in I^{(t)}$
    and $\widehat{\boldsymbol{\mu}}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D|I^*|})}{n_i}}$
    Pull arm $h$ uniform randomly chosen from $I^{*(t)}$
    Update $\widehat{\boldsymbol{\mu}}_h$; $n \leftarrow n+1$; $n_h \leftarrow n_h + 1$; $t \leftarrow t+1$
end while
```

loss due to the play of suboptimal arms, where $I\!\!E[\cdot]$ is the expectation and Δ_i as in Equation 1.

The expected value of each arm is computed by averaging the samples observed over time. The mean of arm i is estimated as $\widehat{\mu}_i(n) = \sum_{s=1}^{T_i(n)} \mathbf{X}_k(s)/T_k(n)$, is the s-th sample observed for arm i.

3. The Pareto UCB1 algorithm

Pareto UCB1 (Drugan & Nowe, 2013) is a straightforward generalization of UCB1 where the Pareto dominance relation is used. By definition, the index for each arm has two terms: i) the mean vector, and ii) a term related to the size of a one-sided confidence interval of the average reward according with the Chernoff-Hoeffding bounds.

As initialization step, each arm is played once. Each iteration, for each arm, i, we add its estimated mean reward vector and its associated confidence interval, $\widehat{\boldsymbol{\mu}}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D|I^*|})}{n_i}}$. The Pareto optimal set of arms for the time step t, $I^{*(t)}$, is calculated from this index. Thus, for all not Pareto optimal arms $i \notin I^{*(t)}$, there exists a Pareto optimal arm $h \in I^{*(t)}$ that dominates the arm i:

$$\widehat{\boldsymbol{\mu}}_h + \sqrt{\frac{2\ln(n\sqrt[4]{D|I^*|})}{n_h}} \succ \widehat{\boldsymbol{\mu}}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D|I^*|})}{n_i}}$$

We now select uniform at random a Pareto optimal arm from $I^{*(t)}$ and pull it. After selection, the mean value of the selected arm $\widehat{\boldsymbol{\mu}}_h$ and the corresponding counters are updated. A possible stopping criteria is a fixed number of iterations n.

An arm that is closer to the Pareto optimal set of arms I^* is more often selected than an arm that is further away from I^*. Note that, by design, the Pareto UCB1 algorithm is fair in selecting Pareto optimal arms.

Consider the Pareto regret defined in Equation 1. The expected Pareto regret of a policy π after any number of n plays is at most

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

where I^* is the set of Pareto optimal arms.

For any suboptimal arm i, $I\!\!E[T_i(N)] \leq \frac{8}{\Delta_i^2}\ln(N\sqrt[4]{D|\mathcal{I}^*|})$ 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 Pareto UCB1 is logarithmic in the number of plays N. Unlike the single objective UCB1, this expected bound is in addition logarithmic with the number of dimensions D and the number of optimal arms $|\mathcal{I}^*|$. The worst-case performance of this algorithm is when the number of Pareto optimal arms is approximately equal with the total number of arms $|\mathcal{I}^*| \approx K$, a probable situation in many objective environments.

4. The ϵ-approximate Pareto MAB problem

An alternative Pareto dominance relation assumes that exists a set of representative vectors that is a good approximation of a large Pareto optimal reward set. This technique is imported from multi-objective optimization and it is called Pareto ϵ-dominance (Laumanns et al., 2002) and its usage in an MAB with reward vectors is described in Section 5. In Section 6, we combine a best arm identification-like algorithm with the Pareto ϵ-dominance relation in order to obtain an MAB algorithm with an efficient exploitation/exploration trade off.

The Pareto ϵ-dominance relation. We consider the definition of additive Pareto ϵ-dominance relation. The reward vector μ *ϵ-dominates* another reward vector ν, $\mu \succ_{\boldsymbol{\epsilon}} \nu$ if and only if for all the objectives j, we have $\mu^j + \boldsymbol{\epsilon}^j \geq \nu^j$ and exists an objective for which $\mu^j + \boldsymbol{\epsilon}^j > \nu^j$. We take $\boldsymbol{\epsilon}^j$ positive constants defined for each dimension, $\boldsymbol{\epsilon}^j > 0$. If $\boldsymbol{\epsilon}^j = 0, \forall j$, we have the classical definition of Pareto dominance.

A set of reward vectors $O_{\boldsymbol{\epsilon}}$ is called an *ϵ-approximate* Pareto reward set of O, if any reward vector ν is ϵ-dominated by at least one reward vector $\mu \in O_{\boldsymbol{\epsilon}}$. Thus,

$$\forall\ \nu \in O\ :\ \exists \mu \in O_{\boldsymbol{\epsilon}}\ \text{ such that }\ \mu \succ_{\boldsymbol{\epsilon}} \nu$$

Note that the set $O_{\boldsymbol{\epsilon}}$ is not unique. The simplest way to generate $O_{\boldsymbol{\epsilon}}$ is to consider the multi-objective reward space as a hyper-grid with equal non-intersecting hypercubes.

For simplicity, we assume that the upper and lower value bounds are equal in all objectives. Let m and M be the upper and the lower bound in each objective. Let $\lceil \frac{M-m}{\boldsymbol{\epsilon}} \rceil$ be the number of equal non-intersecting segments of length $\boldsymbol{\epsilon}$ in each objective. The number of hypercubes

Algorithm 2 Assign arms to the hypergrid (ArmToHypercubes)

```
Require: $\xi > 0, \delta > 0$
  $C_{\epsilon} \to \emptyset$
  for all arms $i \in \mathcal{I}$ do
    Play it for $t_i = \frac{4}{\xi^2} \ln \frac{2DN}{\delta}$ times
    Let $\widehat{\mu}_i$ be its average reward vector
    Assign it to the hypercube $\mathbf{o} \cdot \boldsymbol{\epsilon}$ that contains $\widehat{\mu}_i$
    If $\mathbf{o} \cdot \boldsymbol{\epsilon} \notin C_{\epsilon}$, then $C_{\epsilon} \to C_{\epsilon} \cup \{\mathbf{o} \cdot \boldsymbol{\epsilon}\}$
  end for
  Remove the everywhere dominated hypercubes from $C_{\epsilon}$
```

in $\boldsymbol{\epsilon}$-approximative Pareto sets $O^*_{\boldsymbol{\epsilon}}$ is upper bounded by $|O^*_{\boldsymbol{\epsilon}}| \le (m-1) \cdot \frac{M-m}{\boldsymbol{\epsilon}}$.

We define an hypercube using its lowest right vertex defined as an inner product $\mathbf{o} \cdot \boldsymbol{\epsilon} = (o^1 \cdot \epsilon, \dots, o^D \cdot \epsilon)$, where $\forall j,\ 0 \le o^j \le \lceil \frac{M-m}{\boldsymbol{\epsilon}} \rceil$. An hypercube is the Cartesian product of intervals $\times_{j=1}^{m-1} [o^j \cdot \epsilon, (o^j+1) \cdot \epsilon[$. Thus, the highest vertex of the hypercube is the following inner product $(\mathbf{o}+1) \cdot \boldsymbol{\epsilon} = ((o^1+1) \cdot \epsilon, \dots, (o^D+1) \cdot \epsilon)$.

A hypercube $\mathbf{o}_1 \cdot \boldsymbol{\epsilon}$ is not everywhere dominated by another hypercube $\mathbf{o}_2 \cdot \boldsymbol{\epsilon}$, iff $\exists j$, for which $o_1^j \ge o_2^j$. Note that, by definition, the not everywhere-dominated relation is more relaxed than the definition of non-dominated relation. That is because two adjacent hypercubes can contain non-dominated arms.

The optimal set of not everywhere dominated hypercubes is denoted with $\mathcal{O}^*_{\boldsymbol{\epsilon}}$. The arms inside a hypercube are ordered using the non-dominated relation, but only one arm would be further used.

5. Assign arms to hypercubes

This is a sequential problem because each arm is individually assigned to a hypercube. Intuitively, an arm i is assigned to the hypercube that contains its mean reward vector, μ_i. We want to bound the probability that the arm i is assigned to the wrong hypercube. Thus, the arms that are in the middle of the hypercube are easier to assign, whereas the arms that are close to the border of a hypercube are more difficult to assign. The goal of this algorithm is to deterministically delete hypercubes that are dominated in all objectives rather than dominated arms.

The pseudo-code for this algorithm is given in Algorithm 2. An arm is assigned to a single hypercube, but to a hypercube, more than one arm can be assigned. Let $C_{\boldsymbol{\epsilon}}$ be the set of non-empty hypercubes to which at least one arm is assigned. At initialization, $C_{\boldsymbol{\epsilon}}$ is the empty set. Each arm is sampled for $\frac{4}{\xi^2} \ln \frac{2DN}{\delta}$ times and then the arm is assigned to the hypercube that contains its mean. As last, the hypercubes that are dominated in all objectives by at least one other hypercube from $C_{\boldsymbol{\epsilon}}$ are deleted.

Note that this algorithm is a variant of the naive $(\xi, \delta)-$PAC algorithm (Even-Dar et al., 2006) where the probability that the expected reward vector of an arm i, μ_i, does not belong to the same hypercube as its estimated reward vector, $\widehat{\mu}_i$, is bounded with confidence interval δ. Thus, we want to bound the probability of the event $|\mu_i - \widehat{\mu}_i| \not\prec \xi\mathbf{1}$. Thus, $\nexists j$, for which $|\mu_i^j - \widehat{\mu}_i^j| > \xi$ and $\mathbf{1}$ the unity vector.

Theorem 1 *Let Algorithm 2 be run on a K-armed multi-objective bandit problem, $K > 1$, having arbitrary reward distributions $\mathbf{P}_1, \dots \mathbf{P}_K$ with support in $[0,1]^D$.*

This algorithm is a naive (ξ, δ)-PAC with sample complexity $\mathcal{O}(\frac{N}{\xi^2} \log \frac{2DN}{\delta})$.

Proof. Let's prove that this algorithm is a naive (ξ, δ)-PAC algorithm. Let's consider the hypercube with the lowest vertex $\mathbf{o} \cdot \boldsymbol{\epsilon}$ be the hypercube that contains i, $\mu_i \in \mathbf{o} \cdot \boldsymbol{\epsilon}$. We consider that when the arm i is at a certain distance from the bounds of $\mathbf{o} \cdot \boldsymbol{\epsilon}$, then i can be assigned to that hypercube. Thus, $\mu_i \not\prec \boldsymbol{\epsilon} \cdot \mathbf{o} + \xi$ and $\mu_i \not\succ \boldsymbol{\epsilon} \cdot (\mathbf{o} + \mathbf{1}) - \xi$. This means that $\exists j$, such that $\mu_i^j > \boldsymbol{\epsilon}^j \cdot o^j + \xi$ and $\mu_i^j < \boldsymbol{\epsilon}^j \cdot (o^j+1) - \xi$.

The probability that the estimated reward vector is not in the same hypercube like μ_i, means that $\widehat{\mu}_i \not\succ \boldsymbol{\epsilon}\mathbf{o}$ or $\widehat{\mu}_i \not\prec (\boldsymbol{\epsilon} + \mathbf{1})\mathbf{o}$. This means that $\exists j$, such that $\widehat{\mu}_i^j < \boldsymbol{\epsilon}^j o^j$ or $\widehat{\mu}_i^j > \boldsymbol{\epsilon}^j (o^j + 1)$.

Thus, the probability of error is

$$\mathbb{P}(|\mu_i - \widehat{\mu}_i| \not\prec \xi) = \mathbb{P}(\widehat{\mu}_i \not\succ \mu_i - \xi \ \text{ or } \ \widehat{\mu}_i \not\prec \mu_i + \xi)$$

$$\le \mathbb{P}(\widehat{\mu}_i \not\succ \mu_i - \xi) + \mathbb{P}(\widehat{\mu}_i \not\prec \mu_i + \xi)$$

$$= \sum_{j=1}^{D} \mathbb{P}(\widehat{\mu}_i^j < \mu_i^j - \xi) + \mathbb{P}(\widehat{\mu}_i^j > \mu_i^j + \xi) \le 2De^{-\xi^2 \ell/2} = \frac{\delta}{N}$$

where the last inequality uses the Hoeffding inequality. Choosing $\ell = \frac{2}{\xi^2} \ln \frac{2DN}{\delta}$ assures that $\mathbb{P}(|\mu_i - \widehat{\mu}_i| \not\prec \xi) \le \delta/N$. Summing over all the arms, we have that the error probability is at most δ.

The above bound does not depend on the number of optimal arms but depends on the number of dimensions D. Thus, the number of times an arm should be pulled is thus longer compared with the standard naive (ξ, δ)-PAC algorithm from (Even-Dar et al., 2006).

The last step of the algorithm deletes the hypercubes that are dominated in all objectives and it is deterministic. The computational complexity of testing the dominance relations between the non-empty hypercubes is the same with the complexity of sorting. Thus, $\mathcal{O}(c \log c)$, where $c = |C_{\boldsymbol{\epsilon}}|$.

Algorithm 3 ϵ-Pareto optimal set of arms identification (POAI)

```
A non-empty not everywhere dominated set of hypercubes $C_\epsilon \leftarrow$ ArmToHypercubes$(\xi, \delta)$
$\mathcal{I}^*_\epsilon \leftarrow \emptyset$
for all hypercubes $c = 1, \dots, |C_\epsilon|$ do
  Let $A_1 = \mathcal{I}$, and $n_0 = 0$, and $n_k = \left\lceil \frac{1}{\overline{\log}(K)} \cdot \frac{N-K}{K+1-k} \right\rceil$
  for all rounds $k = 1, 2, \dots, K-1$ do
    (1) $\forall i \in A_k$, select arm $i$ for $n_k - n_{k-1}$ rounds
    (2) Let $argmin_{i \in A_k} \widehat{\mu}_i$ be the arm to dismiss
    (3) $A_{k+1} \rightarrow A_k \setminus argmin_{i \in A_k} \widehat{\mu}_i$
  end for
  Let the remaining set of arm be $i^* \leftarrow A_{K-1}$
  $\mathcal{I}^*_\epsilon \leftarrow \mathcal{I}^*_\epsilon \cup \{i^*\}$
end for
```

For a meaningful algorithm, we need to ensure that $\xi \ll \epsilon$. Only for the arms that are at the border of hypercubes, the probability to select the wrong hypercube is large. But the Pareto dominance relation between arms in one hypercube is invariable with the process of assigning these arms to the right hypercube. Thus, we consider here $\xi = \epsilon$.

The budget for this algorithm is now

$$n_1 = \sum_{i=1}^{K} t_i = \frac{4K}{\epsilon^2} \ln \frac{2DN}{\delta} = \frac{4K}{\epsilon^2} \ln \frac{N}{\delta} + \frac{4K}{\epsilon^2} \ln 2D$$

$$\approx \frac{4K}{\epsilon^2} \ln \frac{N}{\delta}$$

where the dominant term of this sum is given by $\frac{4K}{\epsilon^2} \ln \frac{N}{\delta}$ and the second term is a constant that is further ignored. Note that this budget n_1 increases when δ decreases, thus when the confidence that a certain arm is assigned to the correct hypercube increases.

6. The ϵ-approximate Pareto optimal arm identification algorithm

The goal of the best arm identification class of algorithms (Gabillon et al., 2012) is to delete suboptimal arms in such a way that the probability to delete the optimal arm is bounded. The main idea of the ϵ-approximate Pareto optimal arm identification (POAI) algorithm is, for each not everywhere dominated hypercube, to successively delete dominated arms until all the arms that were not dismissed are non-dominated arms. The only difference between the standard, i.e. single objective, successive rejects and this algorithm is the usage of Pareto dominance relations to qualitatively classify the arms. The pseudo-code for the POAI algorithm is given in Algorithm 3.

The POAI algorithm starts with dividing the time in $K-1$ phases. The length of a phase is carefully chosen to ensure logarithmic regret. Thus, the length of the k-th phase is $n_k = \left\lceil \frac{1}{\overline{\log}(K)} \frac{N-K}{K+1-k} \right\rceil$, where we have the following approximation for the logarithm function $\overline{\log}(K) = \frac{1}{2} + \sum_{i=2}^{K} \frac{1}{i}$. At the end of each phase, the algorithm dismisses the arm with the lowest empirical mean reward. During the next phase, each arm that is not dismissed yet will be pulled equally often. The probability of erroneously removing a Pareto optimal arm is shown to be upper bounded by $\binom{K-1}{2} e^{-\frac{N-K}{\overline{\log}(K) H_2}}$, where a complexity measure is $H_2 = \max_{i \in \mathcal{I}} i \Delta_i^{-2}$. This type of exploration/exploitation trade-off is especially useful when the regrets are close to 0, meaning there are near Pareto optimal solutions.

For each hypercube, we assume that only one non-dominated arm represents that hypercube. This algorithm has an upper confidence bound on the probability that *all* Pareto optimal arms are deleted in a hypercube, and thus the selected arm is suboptimal. Like for the standard successive reject algorithm, the probability of wrongly deleting *all* Pareto optimal arms in a hypercube after any number of N plays is at most

$$e_N \leq cD \binom{K-1}{2} \cdot e^{-\frac{(N/c-K)}{H_2(\overline{\log}(K)+1)}}$$

The proof results immediately from the proof in (Audibert et al., 2010).

The error from the POAI algorithm, cf. Algorithm 3, depends on the number of non-dominated hypercubes, c, a parameter that is controlled by the user.

In the limit, it can be at most K when all arms are in a hypercube and the hypergrid is considered too coarse. The smallest number of arms in a hypercube is 1. If almost all non-empty not everywhere dominated hypercubes have only one arm, then the hypergrid is considered too fine.

If the hypergrid is fine, there are many not everywhere dominated hypercubes, each of them containing a small number of arms. When $c \approx |\mathcal{I}^*|$, the POAI algorithm, cf Algorithm 3, has a performance similar with the successive rejects algorithm. If the hypergrid is coarse, there are few not everywhere dominated hypercubes, each of them containing many arms. If $c \approx 1$, then the POAI algorithm selects only one non-dominated arm.

Thus, in order to obtain an efficient algorithm, we need to tune parameter c. In the experimental section, we experimentally select a value for c.

Let's consider that the POAI algorithm has a fixed budget of N pulls. The algorithm that assigns arms to hypercubes, cf. Algorithm 2, spends n_1 pulls to assign arms to hypercubes, and the rest of the plays $n_2 = N - n_1$ in the POAI algorithm, cf Algorithm 3, are spent on identifying *a* Pareto optimal arm in each hypercube.

Thus, the number of pulls for Algorithm 3 is $n_2 = N - n_1 = N - \frac{4K}{\epsilon^2} \ln \frac{N}{\delta}$. This can be tuned also with different values for ϵ and δ. In general, the amount of pulls needed for each of the two algorithms depends on the characteristics of the environment. When there are many Pareto optimal arms, we want to spend less arm pulls on assigning arms to the hypergrid than on removing arms from a hypercube. For many suboptimal arms, the importance of correctly assigning arms to hypercubes is obvious.

7. Experiments

In this section, we consider the multi-objective multi-armed bandits framework to find the Pareto optimal set of settings of a control application like the wet clutch (Vaerenbergh et al., 2012).

The description of the wet clutch. In order to optimize the functioning of the clutch it is necessarily to simultaneously minimize the optimal current profile to the electro-hydraulic valve that controls the pressure of the oil to the clutch, and 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 being characterized by a hard non-linearity. Additionally, external factors that cannot be exactly controlled such as the surrounding temperature making the outcome of a control application stochastic. Thus, the goal of such systems will be to minimize the clutch's profile and the engagement time in varying environmental conditions. These clutches are typically used in power transmissions of off-road vehicles, which operate under strongly varying environmental conditions.

Currently, Pareto (near)-optimal solutions for this problem are generated using a standard multi-objective evolutionary algorithm (Zhong et al., 2012) that considers only the mean of the solutions. However, the classical multi-objective evolutionary algorithm has no mechanism to handle the stochastic information. The evolutionary algorithms for dynamic and uncertain environments assumes a dynamic support of the distribution of a solution, and the system updates its operators at certain moments in time. In our case, the underlying distribution is stochastic and the adaptation step is expensive and might lead to inaccurate results. Thus, we need an optimizer suited for stochastic environments with stationary underlying distributions.

In Figure 1 a), we give 50 points generated with the wet clutch application, each point representing a trial of the machine and the jerk time obtained in the given time. In Figure 1 a), we have a minimization problem that we transform into a maximization problem, see Figure 1 b), by first

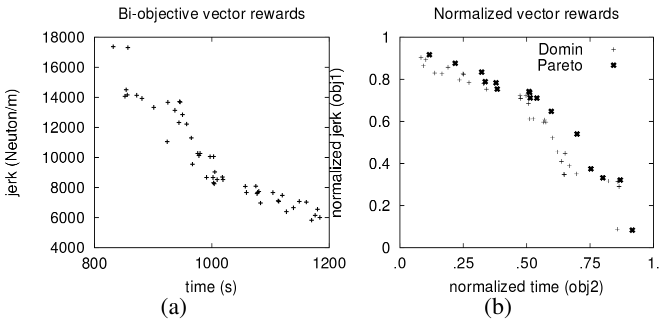


Figure 1. a) All the points generated by the bi-objective wet-clutch application. b) Transforming this points from a maximization problem into a minimization problem.

normalizing each objective with values between 0 and 1, and then transforming it into a maximization problem for each of the two objectives. The best set of incomparable reward vectors is called the Pareto optimal reward set, e.g. in Figure 1 b), there are 16 such reward vectors. In our example, the Pareto optimal set $\mathcal{O}^*$ is about one-third from the total number of arms, i.e. 16/50, and is a mixture of convex and non-convex regions.

Parameter settings. The scope is to experimentally compare the behavior of three instances of multi-objective MAB algorithms:

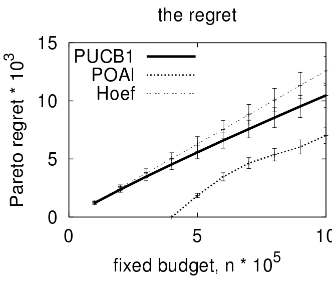
PUCB1 The Pareto UCB1 algorithm described in Section 3;

POAI The ϵ-Pareto optimal set of arms identification algorithm described in Section 6;

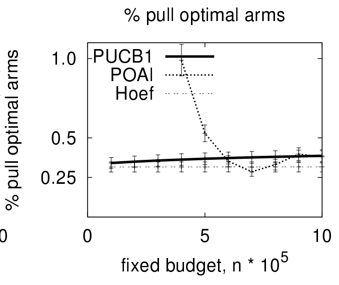
Hoef As a basedline algorithm, we use an adaptation to the multi-objective spaces of the Hoeffding race algorithm (Maron & Moore, 1994) where all the arms are pulled equally often and then the arms with the Pareto non-dominated empirical mean are chosen.

Each algorithm is run 100 times with a fixed budged of $N = 10^6$. We consider $\epsilon = 0.1$, the accuracy $\xi = 0.01$ and the confidence interval $\delta = 0.1$. Note that, for $K = 50$ and $|\mathcal{I}^*| = 16$, the number of pulls for the assign arms to hypercubes algorithm, cf Algorithm 2, is $n_1 \approx 32 \cdot 10^4$, and the number of pulls for the POAI algorithm, cf. Algorithm 3, is almost double $n_2 \approx 67 \cdot 10^4$. These parameters are set based on data observation from Figure 1, automatically tuning these parameters remains as future work.

Performance assessment. We evaluate the performance of the three multi-objective MAB algorithms based on two performance measures as indicated in Figure 2. Figure 2 a) gives the instantaneously Pareto regret metric for each of the three algorithms. According with this measure, Pareto



(a)



(b)

Figure 2. The performance of the three multi-objective MAB algorithms: i) Hoeffding race (Hoef), ii) Pareto UCB1 (PUCB1) and iii) ϵ-approximate Pareto optimal set of arms identification algorithm (POAI).

UCB1 and POAI are the best performing algorithms. Interesting are the slopes of the Pareto regret metric. It seems that for a larger budget, the best performing algorithm will be ϵ-Pareto optimal set of arms identification algorithm those regret grows very slowly after $n_1 \approx 32000$ pulls in which the arms are assigned to hypercubes.

Figure 2 b) shows the percentage of times any of the Pareto optimal arms from $\mathcal{I}^*$ is used. Note that the Pareto UCB1 and Hoeffding algorithms perform similarly, while POAI has a better performance in the beginning where the suboptimal arms are deleted and resembles Pareto UCB1's performance when the Pareto optimal arms in the same hypercube are deleted.

We conclude that POAI has the best performance from the three algorithms.

8. Conclusions

The multi-objective multi-armed bandits is a theoretical framework for stochastic environments. *Pareto UCB1* is a UCB1 algorithm extended to reward vectors those performance depends on the number of Pareto optimal arms. In order to improve the exploration/ exploitation trade-off and thus the theoretical bounds of the Pareto UCB1 algorithms, we incorporate techniques from multi-objective evolutionary algorithms like Pareto ϵ-dominance. The upper confidence bound of this algorithm is independent of the size of the Pareto optimal set of arms. This is an important property for environments with many objectives where relatively many arms have incomparable quality. We show that the proposed ϵ-approximate Pareto optimal arms identification algorithm can be used to identify the Pareto optimal set of solutions in a real world control problem.

References

Audibert, J.-Y., Bubeck, S., & Munos, R. (2010). Best arm identification in multi-armed bandits. *Proc of Conference on Learning Theory (COLT'10).*

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

Bubeck, S., Wang, T., & Viswanathan, N. (2013). Multiple identifications in multi-armed bandits. *Proc of International Conference on Machine Learning (ICML'13).*

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

Even-Dar, E., Mannor, S., & Mansour, Y. (2006). Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *The Journal of Machine Learning Research*, *7*, 1079–1105.

Gabillon, V., Ghavamzadeh, M., & Lazaric, A. (2012). Best arm identification: A unified approach to fixed budget and fixed confidence. *Proc of Neural Information Processing Systems (NIPS).*

Laumanns, M., Thiele, L., Deb, K., & Zitzler, E. (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, *10*, 263–282.

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

Roijers, D., Whiteson, S., & Oliehoek, F. (2013). Computing convex coverage sets for multi-objective coordination graphs. *ADT 2013: Proceedings of the Third International Conference on Algorithmic Decision Theory.*

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

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

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

Zhong, Y., Dutta, A., Wyns, B., Ionescu, C. M., Pinte, G., Symens, W., Stoev, J., & Keyser, R. (2012). Fine tuning of a wet clutch engagement by means of a genetic algorithm. In *Artificial intelligence applications and* *innovations*, vol. 381 of *IFIP Advances in Information and Communication Technology*, 58–67. Springer Berlin Heidelberg.

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