Machine Learning (2023) 112:217–240
https://doi.org/10.1007/s10994-022-06271-z

Multi-armed bandits with censored consumption of resources

Viktor Bengs[1] · **Eyke Hüllermeier**[1,2]

Received: 15 December 2021 / Revised: 30 August 2022 / Accepted: 7 October 2022 /
Published online: 16 November 2022

Abstract
We consider a resource-aware variant of the classical multi-armed bandit problem: In each round, the learner selects an arm and determines a resource limit. It then observes a corresponding (random) reward, provided the (random) amount of consumed resources remains below the limit. Otherwise, the observation is censored, i.e., no reward is obtained. For this problem setting, we introduce a measure of regret, which incorporates both the actual amount of consumed resources of each learning round and the optimality of realizable rewards as well as the risk of exceeding the allocated resource limit. Thus, to minimize regret, the learner needs to set a resource limit and choose an arm in such a way that the chance to realize a high reward within the predefined resource limit is high, while the resource limit itself should be kept as low as possible. We propose a UCB-inspired online learning algorithm, which we analyze theoretically in terms of its regret upper bound. In a simulation study, we show that our learning algorithm outperforms straightforward extensions of standard multi-armed bandit algorithms.

Keywords Algorithm selection · Bivariate feedback · Censored feedback · Exploration · Exploitation

1 Introduction

Multi-armed bandit (MAB) problems constitute an important branch of machine learning research. Their popularity largely stems from an appealing combination of theoretical tractability and practical relevance. In fact, MABs cover a wide range of real-world sequential decision problems, where an agent takes actions (metaphorically considered as "pulling

Editors: Krzysztof Dembczynski and Emilie Devijver.

✉ Viktor Bengs
viktor.bengs@lmu.de

Eyke Hüllermeier
eyke@lmu.de

[1] Institute of Informatics, LMU Munich, Munich, Germany

[2] Munich Center for Machine Learning, Munich, Germany

arms") in order to optimize a specific evaluation criterion, simultaneously exploring the set of actions available and exploiting the feedback resulting from the actions taken. The latter typically comes in the form of (numerical) rewards, generated by the pulled arm according to an underlying probability distribution.

In spite of its versatility, the complexity of real-world problems or the availability of additional side information may suggest further extensions of the basic MAB setting. Indeed, several variants of the basic setting have been developed in order to model specific real-world problem scenarios more appropriately, including $\mathcal{X}$-armed (Bubeck et al. 2011), linear (Auer 2002; Abe et al. 2003), dueling (Yue and Joachims 2009), combinatorial (Cesa-Bianchi and Lugosi 2012), or threshold bandits (Abernethy et al. 2016), just to name a few—for a more detailed overview we refer to Lattimore and Szepesvári (2020). In this paper, we introduce yet another extension of the basic MAB problem, again motivated by practical considerations. More specifically, we consider applications in which the execution of an action requires resources, and will not be successful unless enough resources are provided. Thus, instead of observing a (noisy) reward in every round, the reward is only generated if the resources consumed by the pulled arm remain below a resource limit specified by the learner. Consequently, the learner needs to make two choices in every round of the decision process: the arm to be pulled and the resources allocated to that arm. Since we assume that resources are costly, the value of an outcome produced as a result decreases with the resources consumed. Additionally, the learner might be penalized for allocating a resource limit such that no reward is generated.

Our setting is largely (though not exclusively) motivated by the problem of algorithm selection (Kerschke et al. 2019), which has gained increasing attention in the recent past. Here, the arms are algorithms that can be run on a specific problem instance, for example different solvers that can be applied to an optimization problem or different machine learning algorithms that can be run on a data set. Given a problem instance, the task of the learner is to choose an algorithm that appears most appropriate, and the reward depends on the quality of the result achieved, typically measured in terms of a performance metric (e.g., the generalization performance of a model trained on a data set). Even if this metric is of major concern, one should not overlook that different algorithms have different runtimes or different memory consumptions. For example, training a deep neural network is way more costly than training a decision tree. Depending on the application context, these resource requirements might be important, too. In automated machine learning, for example, many algorithms—or even complete "machine learning pipelines"—are tried, one by one, before a final decision must be made after a certain cutoff time (Hutter et al. 2019). The more costly the algorithms are, the less can be tried.

In cases like those just discussed, the learner needs to find the right balance between two competing targets: an as high as possible reward and an as low as possible consumption of resources. As a consequence, the learner might be willing to sacrifice reward if it helps to keep the overall consumption of resources low, or the other way around, be willing to allocate more resources if this significantly increases the chance of realizing a high reward. In light of this, the underlying correlations between the reward distribution of an arm and the distribution of resource consumption need to be learned, in order to ascertain to which degree the target values conform to each other. Moreover, the learner needs to cope with possibly censored feedback, in case the chosen arm did not return any reward under the allocated resource limit.

In this paper, we model sequential decision problems of the above kind formally and introduce a reasonable measure of regret (or loss) capable of capturing the additional trade-off between realizable reward and consumption of resources as well as the risk of

overexciting the allocated resources (Sect. 2). In Sect. 3, we fist study this problem under the restriction that the possible resource limits can only be chosen within a fixed finite set and describe how the problem can be naïvely tackled by a standard MAB learner. Next, we define a suitable estimate for the target value in the considered problem, which extracts all available learning information from the possibly censored type of feedback. With this, we propose a UCB-inspired bandit learning algorithm, the Resource-censored Upper Confidence Bound (RCUCB) algorithm, for which we derive an upper bound on its cumulative regret. Our result reveals in particular why RCUCB is in general superior to straightforward modifications of well-established standard multi-armed bandit learning algorithms for the considered type of bandit problem. By modifying the RCUCB algorithm in a suitable way, leading to the z − RCUCB algorithm, we show in Sect. 4 how one can deal with the case where the possible resource limits can be chosen as any value within a left-open interval. Further, we experimentally confirm RCUCB's superiority to the straightforward standard bandit reduction approaches in an experimental study (Sect. 5). Finally, we discuss other bandit problems related to ours (Sect. 6), prior to concluding the paper (Sect. 7). For the sake of convenience, we provide a list of symbols used in the paper in the supplementary material, where we also provide all proofs of the theoretical results.

2 The bandit problem

In the following, we specify the bandit problem described in Sect. 1 in a formal way and motivate it using the example of algorithm selection, where the role of an arm is played by a concrete configuration of a learning algorithm, e.g., a neural network with a specific parametrization (network structure, weights, etc.). In particular, we provide a working example where we consider the scenario of a company which provides an on-the-fly machine learning service, where the customers can submit a learning task in form of a data set and some performance metric, for which a suitable machine learning model is returned. The payment agreement between the customer and the company provides for the customer to pay the company an amount of money depending on the performance of the returned machine learning model, while the company has a fast-track-promise and will pay the client some amount of money if a suitable machine learning model cannot be provided within a certain time.

2.1 Learning process

The learning process proceeds over T many rounds, where $T \in \mathbb{N}$ is not necessarily known beforehand. For each round $t \in [T] := \{1, 2, \ldots, T\}$, there is a maximal resource limit $\tau_{\max} \in \mathbb{R}_+$, which is fixed and known beforehand. In (online) algorithm selection, for instance, each round corresponds to a time step, in which an incoming task specified by a data set comprising of a training and test data set and some performance metric needs to be solved. Each algorithm consumes resources for a given task, e.g., the energy consumption or simply the time for the training phase. Due to external constraints, the consumed resources should not exceed some specific limit, e.g., a maximal energy consumption level or a time limit.

2.2 Arms

We assume a finite number of n arms, where $n \in \mathbb{N}$. For sake of convenience, we identify the arms by the set $[n] = \{1, \ldots, n\}$. Each arm $i \in [n]$ is associated with two distributions: a *reward distribution* $P_i^{(r)}$ with support[1] in [0, 1] and a *consumption of resources distribution* $P_i^{(c)}$ with support in $\mathbb{R}_+$ characterized by the cumulative distribution function $F_i^{(c)}$. The joint distribution of an arm's reward and resource consumption is denoted by $P_i^{(r,c)}$ and is *not* necessarily the product of $P_i^{(r)}$ and $P_i^{(c)}$, i.e., an arm's reward and consumption of resources are not assumed to be independent. In particular, this allows for stochastic dependencies between rewards and resource consumptions.

For instance, running a specific configuration of a learning algorithm on an incoming task generates a reward, e.g., the accuracy on the test data or a monetary conversion thereof, and consumes resources, e.g., the energy consumption or simply the time for the training phase. If the data set is generated by some unknown random mechanism (a random training/test split) both the reward and the resource consumption are random as well. Moreover, both observations are likely to be correlated, because the more complex the configuration of a learning algorithm is, e.g., a neural network with a large number of neurons and weights, the higher its accuracy (reward) in general, but also the higher its resource consumption due to its high complexity.

Example 1 Coming back to the working example, let us assume for sake of simplicity that the company has three possible machine learning models available each representing an arm, so that $n = 3$. Suppose for simplicity reasons that the payoff for the returned model is 1:1 to its performance, so that the reward distribution is equivalent to the general performance distribution of a model on possible learning tasks. The only resources consumed is the wall-clock time for running the model. In Fig. 1 the reward, consumption of resources and their joint distribution of the three models are illustrated.

We see that the first model yields high rewards (general performance), but also has a high consumption of resources (running time), e.g. a very complex model such as a large deep neural network. The second model has mediocre rewards (general performance), while consuming fewer resources as the first one, e.g. a random forest. Finally, the third model has both low rewards and low resource use, e.g. a simple linear regression model. All three models show a positive correlation between rewards and resource consumption, which makes sense in this case due to the complexity of the models, because the higher the resource consumption, the higher the reward.

2.3 Learner

A learner (or bandit algorithm) in this setting is a possibly non-deterministic procedure, which, in each round $t \in [T]$, chooses an arm $I_t \in [n]$ and a resource limit $\tau_t \in \mathcal{M}$ depending on the history of previously chosen arms, resource limits, and observed feedback (specified below). Here, $\mathcal{M}$ is a subset of $(0, \tau_{\max}]$ specifying the admissible resource range. In the usual algorithm selection setting, the learner is essentially a mechanism deciding on

[1] It is straightforward to extend our algorithmic solution to σ-sub-Gaussian reward distributions.

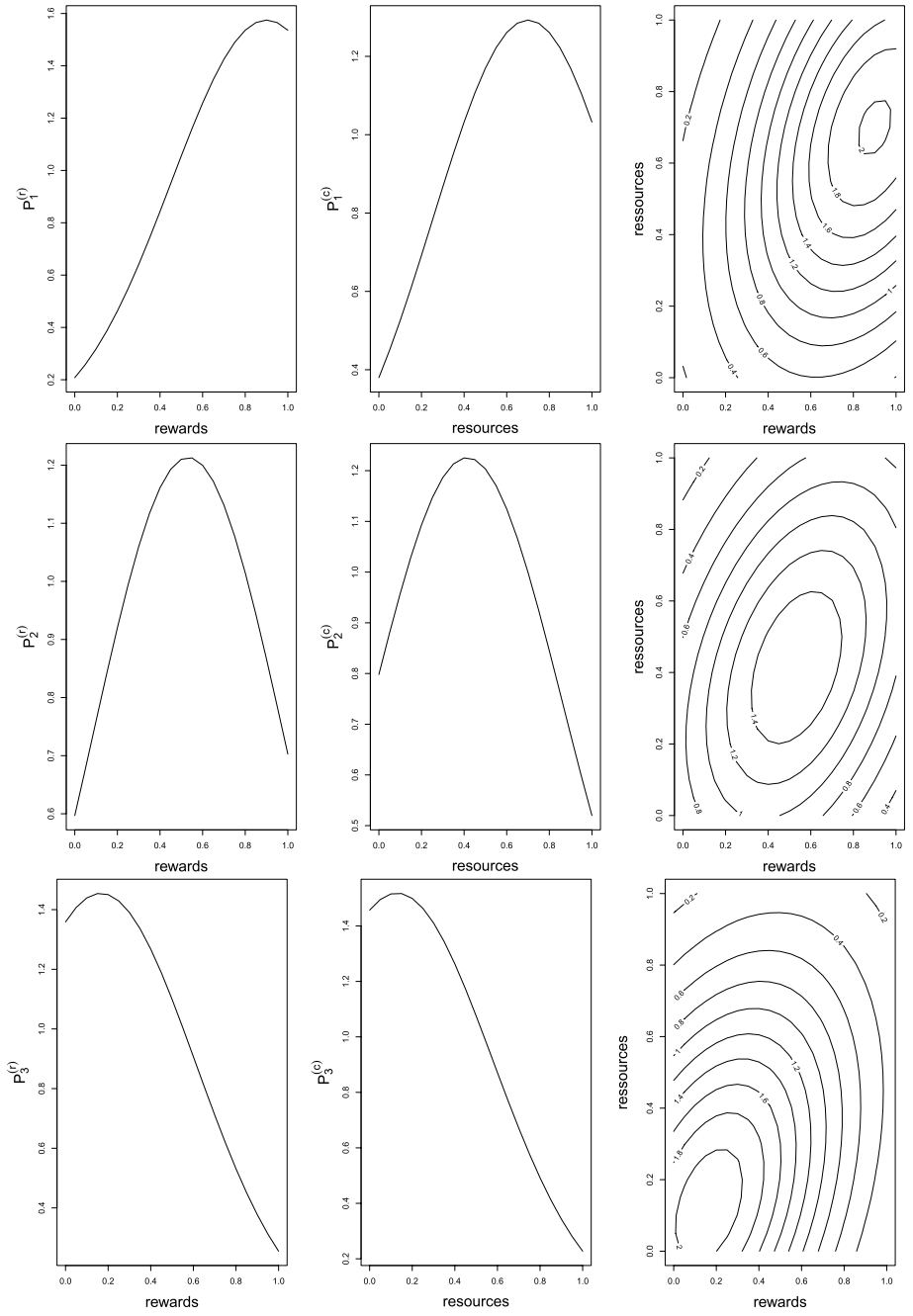

Fig. 1 Exemplary reward (left column), consumption of resource (middle column) and joint distribution for three arms (ith arm corresponds to ith row for $i = 1, 2, 3$) representing the machine learning models in Example 1

which algorithm to choose for the incoming task based on the history of observations seen so far. In our setting, on the other hand, the learner has a more challenging decision to make, as it needs to decide on the most suitable algorithm/resource-limit pair for the given task.

2.4 Feedback

The feedback observed by the learner in round t, if I_t is the chosen arm and τ_t the resource limit, is

$$X_{I_t,t} = \begin{cases} (R_{I_t,t}, C_{I_t,t}), & \text{if } C_{I_t,t} \leq \tau_t \\ (\varnothing, (\tau_t, \infty]), & \text{else} \end{cases}, \tag{1}$$

where $(R_{I_t,t}, C_{I_t,t}) \sim P_{I_t}^{(r,c)}$. In words, if the (noisy) consumption of resources $C_{I_t,t}$ of the chosen arm I_t is within the scheduled resource limit τ_t of the learner, the corresponding reward of the arm $R_{I_t,t}$ is observed (realized), and the corresponding consumption of resources $C_{I_t,t}$ as well. Otherwise, neither the consumption of resources $C_{I_t,t}$ nor the corresponding arm reward $R_{I_t,t}$ is observed (or realized), which we represent by the left-open interval $(\tau_t, \infty]$ resp. $\varnothing$.[2] Here, it is worth noting that although the learner does not observe direct feedback in the latter case, it still observes a valuable information in the form of censored feedback, namely that the consumption of resources $C_{I_t,t}$ exceeded the resource limit τ_t, i.e., the latter is an element in $(\tau_t, \infty]$.

In our scenario, we assume that the observed feedback $X_{I_t,t}$ in round t is independent of the past given I_t and τ_t. This assumption is reasonable from a practical point of view, as, for instance, the run of one specific configuration of a learning algorithm on a randomly split training set is independent of the run of the same learning algorithm configuration on another randomly split training set.

2.5 Profit and loss account

The task of the learner is to select, in each round t, an arm as well as a resource limit such that in expectation an as high as possible reward can be realized within the specified resource limit, while simultaneously keeping the expected consumption of resources of the round as small as possible. To this end, we assume that the learner is provided with two monotonic increasing functions, namely

- A *cost function* $c : \mathbb{R}_+ \to [0, 1]$, which specifies the cost generated by the consumed resources;
- A *penalty function* $\lambda : \mathcal{M} \to \mathbb{R}_+$ which specifies the penalty for exceeding the allocated resources.

The cost function is in the first place mapping the consumption of resources on the same scale as the rewards in order to make them comparable[3] The penalty function maps the

[2] Here, $\varnothing$ is interpreted as a symbol for a dummy variable indicating that no reward information was received.

[3] In particular, the cost function is in fact a mapping $c : \mathbb{R}_+ \to \text{supp}(P^r)$, where $\text{supp}(P^r)$ is the common support of the reward distributions, which is assumed to be [0, 1] for sake of simplicity.

allocated resources (in case of exceeding them) on the same scale as the rewards as well, but in addition gives the learner an incentive to choose resource limits smaller than the maximal possible resource limit. Leveraging the prevalent way of profit and loss accounting in economics, we define for each possible decision pair of a learner $(i, \tau) \in [n] \times \mathcal{M}$ its *penalized expected gain* $\nu_{i,\tau}$ via

$$\nu_{i,\tau} := \mathbb{E}_{P_i^{(r,c)}}\left(\left(X_i^{(r)} - c(X_i^{(c)})\right) \cdot \mathbb{1}_{\{X_i^{(c)} \leq \tau\}}\right) - \mathbb{E}_{P_i^{(c)}}\left(\lambda(\tau)\mathbb{1}_{\{X_i^{(c)} > \tau\}}\right), \tag{2}$$

where $(X_i^{(r)}, X_i^{(c)})^\top \sim P_i^{(r,c)}$. In words, the quality of a decision pair $(i, \tau) \in [n] \times \mathcal{M}$ is measured by means of its expected gain (first term in (2)), which counts the expected profit against the expected loss, while taking an expected "fine" or penalty for possibly exceeding the allocated resources into account.

With this, the task of the learner is to select in each learning round an arm/resource-limit pair having the maximal penalized expected gain, i.e.,

$$(i^*, \tau^*) \in \operatorname{argmax}_{(i,\tau)\in[n]\times\mathcal{M}} \nu_{i,\tau}. \tag{3}$$

The "negative part" or the expected cost part in (2), i.e.,

$$\mathbb{E}_{P_i^{(c)}}\left(c(X_i^{(c)}) \cdot \mathbb{1}_{\{X_i^{(c)} \leq \tau\}} + \lambda(\tau)\mathbb{1}_{\{X_i^{(c)} > \tau\}}\right)$$

allows one to recover common performance metrics considered for algorithm selection problems (Kerschke et al. 2019), such as the so-called *penalized average running times* if the consumption of resources correspond to runtimes of algorithms. For instance, the expected value of the popular PAR10 score corresponds to the choice of $c(x) = x$ and $\lambda(x) = 10x$[4]

In general, one can note that depending on the concrete form of the functions, the learner can either be urged to focus on arms with a small consumption of resources, but possibly slightly smaller expected rewards (c and/or λ grow quickly), or to almost exhaust the available resources in a single round in order to realize the presumably high rewards of arms with high consumption of resources (c and/or λ grow slowly).

Example 2 Recall that in our working example, the company has a fast-track-promise and pays a compensation, say L, to the customer if no suitable model can be provided within a specific amount of time $\tilde{\tau}$. Suppose that the wall-clock time of running the model (consumption of resources) only generates energy costs, which correspond to one tenth of the wall-clock time, i.e., $c(x) = x/10$.. Thus, the penalty function is $\lambda(x) = c(x)\mathbb{1}_{\{x\leq\tilde{\tau}\}} + Lx\mathbb{1}_{\{x>\tilde{\tau}\}}$, where the first term accounts for the (energy) costs of running the model. For the three available models with distributions as in Example 1 we illustrate in Fig. 2 the penalized expected gain for two cases, a strict fast-track-promise case with $L = 10$ and $\tilde{\tau} = 0.5$, and a soft fast-track-promise case with $L = 5$ and $\tilde{\tau} = 0.99$.

We see that in the case of a strict fast-track-promise (left plot) the simple model is most lucrative for most choices of τ and being overtaken by the medium-complex model only near the $\tilde{\tau}$, while the complex model is completely unsuitable.[5] For the soft

[4] Here, we assume for sake of simplicity that the resource consumption (runtime) is already on the proper scale for comparing it with the rewards.

[5] We leave out the cases for which $\tau > 0.5$ as $\nu_{i,\tau}$ is negative in this case.

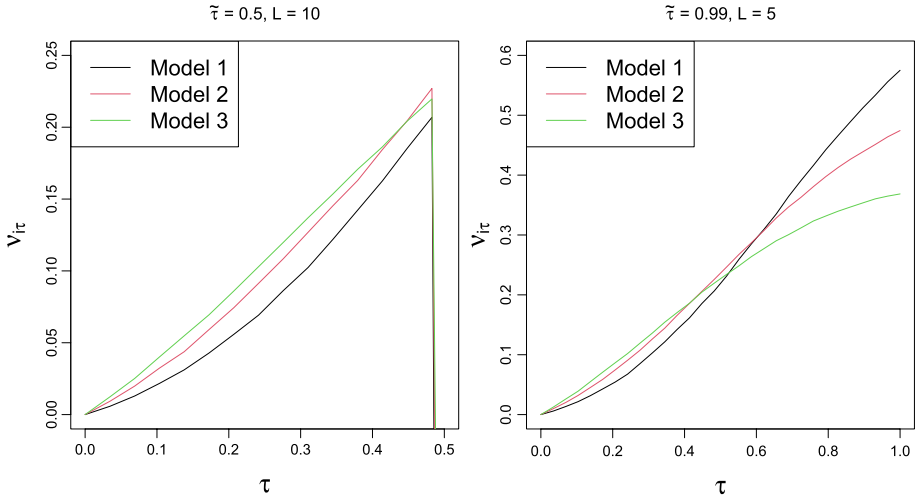

Fig. 2 Penalized expected gain $v_{i,\tau}$ for cost function $c(x) = x/10$ and penalty function $\lambda(x) = c(x)\mathbb{1}_{\{x\leq\tilde{\tau}\}} + Lx\mathbb{1}_{\{x>\tilde{\tau}\}}$ for $\tilde{\tau} = 0.5, L = 10$ (left) and $\tilde{\tau} = 0.99, L = 5$ (right) for the three arms (machine learning models) of Example 1

fast-track-promise case (right plot), the most complex model attains the highest penalized expected gain, for which a larger choice of τ is necessary due to its high chance of returning a censored feedback by exceeding the allocated resources.

2.6 Quality of a learner

Having defined the quality of a decision pair, one can compare the decision made by a learner with the optimal decision to obtain a natural measure for the (sub-)optimality of the learner's decision in each round by means of a notion of regret. Indeed, if the learner chooses the arm I_t and the resource limit τ_t in round t, define the *instantaneous (pseudo-) regret* as the difference between the optimal penalized expected gain and the penalized expected gain of the chosen pair, i.e., $r_t := v^* - v_{I_t,\tau_t}$, where v^* is the maximum value in (3). Hence, the cumulative (pseudo-) regret is given by

$$\mathcal{R}_T := \sum_{t=1}^{T} r_t = v^* T - \sum_{t=1}^{T} \mathbb{E}\, v_{I_t,\tau_t}, \tag{4}$$

where $(I_t, \tau_t)_{t=1}^T$ are the actions chosen by the learner during the T rounds. Note that, in general, it is possible to have multiple optimal pairs $(i, \tau) \in [n] \times \mathcal{M}$, such that the instantaneous regret r_t vanishes. However, without loss of generality, we subsequently assume that there is only one unique pair (i^*, τ^*) such that $v^* = v_{i^*,\tau^*}$ holds, as having multiple optimal pairs only makes the learning problem easier.

Remark 1 The considered bandit problem can recover the standard MAB problem by assuming that $\mathcal{M} = \{0\}$ and each arm's consumption of resources distribution is the Dirac measure on $\mathcal{M}$, while the cost and the penalty function are both the zero function. Moreover, it is also possible to consider problem scenarios in which there is only one fixed resource limit, say $\tilde{\tau}$, by setting $\mathcal{M} = \{\tilde{\tau}\}$.

3 Finite number of resource limits

In this section, we assume $\mathcal{M}$ to be a finite set of grid points within the admissible resource range of each round $(0, \tau_{\max}]$. In the following, we denote by I_t the chosen arm and by τ_t the resource limit set by a learner in round $t \in [T]$, where the learner should be clear from the context.

3.1 Reduction to classical MAB problem

For any pair $(i, \tau) \in [n] \times \mathcal{M}$, define the sub-optimality gap of this pair by means of

$$\Delta_{i,\tau} := v^* - v_{i,\tau}. \tag{5}$$

With this, it is straightforward to show that the cumulative (pseudo-) regret in (4) admits a regret decomposition similar to the pseudo-regret in the classical MAB problem (see Lemma 4.5 in Lattimore and Szepesvári (2020)):

$$\mathcal{R}_T = \sum\nolimits_{(i,\tau)\in[n]\times\mathcal{M}} \Delta_{i,\tau}\, \mathbb{E}(T_{i,\tau}(T+1)), \tag{6}$$

where

$$T_{i,\tau}(t) = \sum\nolimits_{s=1}^{t-1} \mathbb{1}_{\{I_s = i \wedge \tau_s = \tau\}}$$

is the number of times the pair $(i, \tau) \in [n] \times \mathcal{M}$ has been chosen till round $t \in \{1, \ldots, T\}$. In light of this, one might be tempted to cast the considered bandit problem into a classical MAB problem by considering each pair $(i, \tau) \in [n] \times \mathcal{M}$ as an arm ("virtual arm") which generates the "reward" sequence

$$\Big((R_{i,t} - c(C_{i,t}))\mathbb{1}_{\{C_{i,t} \le \tau\}} - \lambda(\tau)\mathbb{1}_{\{C_{i,t} > \tau\}} \Big)_{t=1,\ldots,T}, \tag{7}$$

each having expected value $v_{i,\tau}$. Thus, the bandit problem at hand can be naïvely considered as an unstructured class of (classical) multi-armed bandits $\mathcal{E} = \times_{i\in[n],\tau\in\mathcal{M}} \mathcal{P}_{i,\tau}$ (see Section 4.3 in Lattimore and Szepesvári 2020), where $\mathcal{P}_{i,\tau}$ is a set of bivariate probability distributions on $[0, 1] \times \mathbb{R}_+$.

As indicated by the reduction, the considered bandit problem can in principle be tackled by any bandit algorithm for the classical MAB problem by means of interpreting each pair $(i, \tau) \in [n] \times \mathcal{M}$ as an (virtual) arm. However, as we shall see in the following section, exemplified on the basis of UCB (Auer et al. 2002), this straightforward reduction seems to be sub-optimal, as the available information of the possibly censored type of feedback is not incorporated in an appropriate way. This is in fact not surprising, because the problem

at hand is actually not an unstructured bandit problem, as for each fixed arm $i \in [n]$, there is a relationship between the probability distributions $(\mathcal{P}_{i,\tau})_{\tau \in \mathcal{M}}$ due to the joint reward and resource consumption distribution $P_i^{(r,c)}$. This relationship is lost by the reduction.

3.2 Penalized expected gain estimates

Considering the desired value v^*, which arises from (3), one certainly needs to estimate the penalized expected gain $\nu_{i,\tau}$ in (2) in a suitable way. Regarding their form, one needs for each pair $(i, \tau) \in [n] \times \mathcal{M}$ suitable estimates for both the expected gain

$$g_{i,\tau} := \mathbb{E}_{P_i^{(r,c)}}\left(\left(X_i^{(r)} - c(X_i^{(c)})\right) \cdot \mathbb{1}_{\{X_i^{(c)} \leq \tau\}}\right)$$

and the expected penalty term

$$\Lambda_{i,\tau} := \mathbb{E}_{P_i^{(c)}}\left(\lambda(\tau)\mathbb{1}_{\{X_i^{(c)} > \tau\}}\right) = \lambda(\tau)\,(1 - P_i^{(c)}(\tau)).$$

For the expected gain we define for any round $t \in [T]$ the estimate

$$\hat{g}_{i,\tau}(t) = \frac{\sum_{s=1}^{t-1}(R_{i,s} - c(C_{i,s})) \cdot \mathbb{1}_{\{C_{i,s} \leq \tau\}} \cdot \mathbb{1}_{\{I_s = i \wedge \tau_s \geq \tau\}}}{N_{i,\tau}(t)}, \tag{8}$$

where

$$N_{i,\tau}(t) = \sum_{s=1}^{t-1} \mathbb{1}_{\{I_s = i \wedge \tau_s \geq \tau\}}.$$

The expected penalty term in round $t \in [T]$ can be estimated via

$$\hat{\Lambda}_{i,\tau}(t) = \lambda(\tau)\,\frac{\sum_{s=1}^{t-1} \mathbb{1}_{\{C_{i,s} > \tau\}} \cdot \mathbb{1}_{\{I_s = i\}}}{N_{i,0}(t)}, \tag{9}$$

which is simply the empirical survival function estimate. Thus, combining (8) and (9), our suggested penalized expected gain estimate for a pair $(i, \tau) \in [n] \times \mathcal{M}$ is

$$\hat{\nu}_{i,\tau}(t) = \hat{g}_{i,\tau}(t) - \hat{\Lambda}_{i,\tau}(t). \tag{10}$$

These estimates admit a simple update rule for the chosen arm $I_t \in [n]$ in round t. Indeed, it holds that $\hat{\Lambda}_{I_t,\tau}(t+1) = \frac{\lambda(\tau)}{N_{i,0}(t)+1}\left(\frac{N_{i,0}(t)\,\hat{\Lambda}_{I_t,\tau}(t)}{\lambda(\tau)} + \mathbb{1}_{\{C_{I_t,t} > \tau\}}\right)$ and

- If $\tau > \tau_t$, then $\hat{g}_{I_t,\tau}(t+1) = \hat{g}_{I_t,\tau}(t)$,
- If $\tau \leq \tau_t$, then $\hat{g}_{I_t,\tau}(t+1) = \begin{cases} \frac{N_{I_t,\tau}(t)\cdot\hat{g}_{I_t,\tau}(t) + (R_{I_t,t} - c(C_{I_t,t}))}{N_{I_t,\tau}(t)+1}, & C_{I_t,t} \leq \tau, \\ \frac{N_{I_t,\tau}(t)\cdot\hat{g}_{I_t,\tau}(t)}{N_{I_t,\tau}(t)+1}, & C_{I_t,t} > \tau, \end{cases}$

  as well as $N_{I_t,\tau}(t+1) = N_{I_t,\tau}(t) + 1$.

Note that this update has a complexity of $O(|\mathcal{M}|)$. Moreover, $\hat{g}_{I_t,\tau}$ is updated for all resource limits τ below the currently chosen one (i.e., τ_t), even though the feedback was possibly

censored, i.e., in the case where $C_{I_t,t} > \tau$. This is in particular advantageous compared to a standard plug-in estimate (i.e., see (31) in the appendix), which does not adapt the estimate value for censored observations.

Besides their appealing property of extracting all available feedback information, these estimates also allow for deriving suitable confidence intervals by exploiting results from the theory of martingales and using a peeling argument. Indeed, for confidence lengths defined for each pair $(i, \tau) \in [n] \times \mathcal{M}$ in round $t \in [T]$ by

$$\begin{aligned}\mathfrak{c}_{i,\tau}(t;\alpha) &= \mathfrak{c}_{i,\tau}^{(g)}(t;\alpha) + \mathfrak{c}_{i,\tau}^{(\Lambda)}(t;\alpha) \\ &= \sqrt{(2\alpha \log(t))/N_{i,\tau}(t)} + \lambda(\tau)\sqrt{(2\alpha \log(t))/N_{i,0}(t)}, \quad \alpha > 1,\end{aligned}$$

we obtain the following result (cf. Section A for the proof).

Proposition 1 *Let $(i, \tau) \in [n] \times \mathcal{M}$ and $\alpha > 1$. Then, for any round $t \in [T]$, it holds that $\mathbb{P}\big(\hat{\nu}_{i,\tau}(t) - \nu_{i,\tau} > \mathfrak{c}_{i,\tau}(t;\alpha)\big) \leq 2\left(1 + \frac{\log(t)}{\log(\frac{\alpha+1}{2})}\right) t^{-\frac{2\alpha}{\alpha+1}}$, and the right-hand side is also an upper bound for $\mathbb{P}\big(\hat{\nu}_{i,\tau}(t) - \nu_{i,\tau} < -\mathfrak{c}_{i,\tau}(t;\alpha)\big)$.*

3.3 Resource-censored upper confidence bound

Another appealing property of the estimates introduced in the previous section and especially the underlying counter variables $N_{i,\tau}$ is the possibility to refine the regret decomposition in (6), which in turn will provide insights into the question why a learner revolving around $\hat{\nu}_{i,\tau}$ will in general improve upon a naïve reduction to the standard MAB problem. More specifically, for $\tau \neq \tau_{\max}$ let

$$\mathrm{up}(\tau) = \min\nolimits_{\tilde{\tau} \in \mathcal{M} \setminus \{\tau_{\max}\}} \{\tilde{\tau} > \tau\}$$

and $T^+ = T + 1$. Then, we can write the cumulative regret $\mathcal{R}_T$ as

$$\begin{aligned}\mathcal{R}_T = &\sum_{\substack{\tau \in \mathcal{M} \setminus \{\tau_{\max}\} \\ i \in [n]}} \Delta_{i,\tau}\big(\mathbb{E}(N_{i,\tau}(T^+)) - \mathbb{E}(N_{i,\mathrm{up}(\tau)}(T^+))\big) \\ &+ \sum_{i \in [n]} \Delta_{i,\tau_{\max}} \mathbb{E}(T_{i,\tau_{\max}}(T^+)),\end{aligned} \tag{11}$$

where we used that $T_{i,\tau}(T^+) = N_{i,\tau}(T^+) - N_{i,\mathrm{up}(\tau)}(T^+)$ for any (i, τ) with $\tau \neq \tau_{\max}$. Thus, to keep the number of sub-optimal arm pulls of a specific arm/resource-limit pair low (i.e., $T_{i,\tau}$), one can play the same arm but with the next larger resource limit (i.e., increase $N_{i,\mathrm{up}(\tau)}$), which in turn will increase $N_{i,\tilde{\tau}}$ for all $\tilde{\tau} \leq \tau$, but simultaneously improve the estimation accuracy of $(\hat{\nu}_{i,\tilde{\tau}})_{\tilde{\tau} \leq \tau}$. In some sense, this consideration suggests that a certain generosity regarding the choice of the resource limit might be favorable.

Inspired by these insights, we define the Resource-censored Upper Confidence Bound (RCUCB) algorithm: In the first n rounds, each arm is chosen once with the maximal available resource limit, i.e., $(I_t, \tau_t) = (t, \tau_{\max})$ for $t \in [n]$. Then, in each subsequent round $t \in \{n + 1, \ldots, T\}$, the arm and the resource limit are chosen as follows:

$$\big(I_t, \tau_t\big) \in \operatorname*{argmax}_{(i,\tau)\in[n]\times\mathcal{M}} \big(\hat{\nu}_{i,\tau}(t) + \mathfrak{c}_{i,\tau}(t;\alpha)\big), \tag{12}$$

where ties are broken arbitrarily and $\alpha > 0$ is a fixed parameter of choice.

Note that unless the penalty function λ increases too drastically from τ to $\mathrm{up}(\tau)$, the confidence length $\mathfrak{c}_{i,\tau}$ of any arm $i \in [n]$ is likely to be smaller than $\mathfrak{c}_{i,\mathrm{up}(\tau)}$ in cases where $N_{i,\tau} > N_{i,\mathrm{up}(\tau)}$ holds.[6] Thus, RCUCB's exploration behavior tends to be biased towards higher resource limits, which in turn is preferable regarding the discussion above.[7]

Note that the main novelty of RCUCB lies primarily in the composition of the underlying exploitation term $\nu_{i,\tau}$, since it consists of two components that are already complex terms in themselves. Indeed, the first component, the expected gain estimate $\hat{g}_{i,\tau}$, is designed such that information from the potentially censored feedback is still extracted while ensuring the construction of valid confidence intervals. The second component, the estimator of the expected penalty term $\hat{\Lambda}_{i,\tau}$, is an empirical survival function estimate and correspondingly more complex than a classical empirical mean.

We obtain the following upper bound on the cumulative regret of RCUCB (see Section B for the proof).

Theorem 1 *Let $\alpha > 1$ in (12). Then, for any number of rounds T, $\varepsilon \in (0,1)$, and any $\delta \in (0, 1/2)$ such that $(n|\mathcal{M}|-1)^{1-\delta}T^{2\delta} \le T$, it holds that*

$$\mathcal{R}_T^{\mathrm{RCUCB}} \le \sum_{(i,\tau)\in[n]\times\mathcal{M}} \Delta_{i,\tau}\, u_{i,\tau}(T,\alpha) - \mathbb{P}(A_\varepsilon) \sum_{(i,\tau)\in[n]\times\mathcal{M}\setminus\{\tau_{\max}\}} \Delta_{i,\tau}\, l_{i,\mathrm{up}(\tau)}(T,\alpha),$$

where

$$\begin{aligned}
u_{i,\tau}(T,\alpha) &:= \frac{8\alpha(1+\lambda(\tau))^2\log(T)}{\Delta_{i,\tau}^2} + 1 + \frac{8}{\log\left(\frac{\alpha+1}{2}\right)}\left(\frac{\alpha+1}{\alpha-1}\right)^2,\\
l_{i,\mathrm{up}(\tau)}(T,\alpha) &:= \frac{\alpha\,\varepsilon\,\delta\,\log(T)}{H_{i,\mathrm{up}(\tau)}(\alpha)},\\
A_\varepsilon &:= \bigcap_{(i,\tau)\in[n]\times\mathcal{M},\ t\in[T]} A_{i,\tau,t,\varepsilon},\\
A_{i,\tau,t,\varepsilon} &:= \{\hat{\nu}_{i,\tau}(t) + (1-\varepsilon)\,\mathfrak{c}_{i,\tau}(t;\alpha) \ge \nu_{i,\tau} \ge \hat{\nu}_{i,\tau}(t) - \mathfrak{c}_{i,\tau}(t;\alpha)\},
\end{aligned}$$

and $H_{i,\tau}(\alpha) := \max_{(j,\tau')\in[n]\times\mathcal{M}:\, j\neq i\vee\tau'\neq\tau} \left(\frac{8(1+\lambda(\tau')^2\alpha)}{\Delta_{i,\tau}-\Delta_{j,\tau'}} + 1\right)^2 (\Delta_{i,\tau} - \Delta_{j,\tau'})^2$.

Theorem 1 reveals why RCUCB is in general superior to the straightforward mapping to the standard MAB problem. The terms $u_{i,\tau}$ correspond (up to multiplicative constants) to the upper bounds on the expected sub-optimal arm pulls of the naïve UCB variant (cf. Corollary 1), from which $l_{i,\mathrm{up}(\tau)}$ is subtracted (with a probabilistic weight $\mathbb{P}(A_\varepsilon)$). The term $l_{i,\mathrm{up}(\tau)}$ is a lower bound for the expected number of sub-optimal arm pulls of an (virtual) arm's "higher resource neighbor", i.e., the (virtual) arm corresponding to $(i, \mathrm{up}(\tau))$.

[6] Recall that $N_{i,\tau} \ge N_{i,\mathrm{up}(\tau)}$ always holds.

[7] The exploration behavior of UCB-type algorithms is mainly driven by the confidence intervals.

Remark 2 The term $H_{i,\tau}$ occurring in the lower bound for the expected number of sub-optimal arm pulls is essentially the largest difference between the sub-optimality gap of the corresponding arm/resource-limit pair and any other sub-optimality gap of an arm/resource-limit pair. In particular, this term is small (such that $l_{i,\mathrm{up}(\tau)}$ is large) if the sub-optimality gaps of the arm/resource-limit pairs are similar, i.e., the learning problem is difficult. The event A_ϵ is the anytime concentration (up to some ϵ relaxation for the upper deviation) of the estimate's confidence bounds around the ground-truth value. Using a union bound and Hoeffding's inequality it is straightforward to show that A_ϵ has strictly positive mass.

Remark 3 Note that the terms ϵ and δ could be chosen appropriately such that the second term in the regret bound is as large as possible. Furthermore, it is worth mentioning that the term $(1 + \lambda(\tau))$ occurring in the upper as well as lower bound terms is due to the crude estimate $1 \leq N_{i,0}(T^+)$ used to bound $(1 + \lambda(\tau)/N_{i,0}(T^+))$. We conjecture that this bound can be refined to $(1 + \tilde{C}\,\lambda(\tau)/\log(T))$, for an appropriate constant $\tilde{C} > 0$.

Regarding the update complexity of RCUCB, we can derive the following result, which is proven in Section D.

Proposition 2 RCUCB *has a worst case update complexity of order* $O(n|\mathcal{M}|)$.

Note that the update complexity for most of the state-of-art bandit algorithms combined with the reduction in Sect. 3.1 is of order $O(n|\mathcal{M}|)$ as well, since these are usually linear in the number of arms, which are $n|\mathcal{M}|$ many in light of the reduction.

4 Arbitrary resource limits

We now turn to the case in which $\mathcal{M}$ equals $(0, \tau_{\max}]$. Obviously, the challenge in this variant is to cope with the infinite size of the decision set $[n] \times \mathcal{M}$. To this end, we will follow the ideas of the *zooming* algorithm (Kleinberg et al. 2008) or the StoOO algorithm (Munos 2014) and maintain finite subsets of $\mathcal{M}$, one for each arm, which will be refined successively in order to include resource limits τ, where the penalized expected gain of an arm is believed to be large. The exploration-exploitation behavior will be up to an additional bias-correction guided by the upper confidence term considered in RCUCB.

4.1 Zooming-RCUCB

A zooming algorithm seeks to find a good approximation of the optimum of an unknown stochastic function $f : \mathcal{X} \to \mathbb{R}$ over a (semi-)metric space $\mathcal{X}$ with (semi-)metric d using a numerical budget T for the maximal number of function evaluations. For this purpose, a zooming algorithm constructs a hierarchical partitioning of $\mathcal{X}$ into nested subsets in an online manner. Each subset is associated with a specific point, usually the center of the subset, at which the function f may be evaluated. For each subset the algorithm maintains an estimate of the function value at the center point as well as an confidence interval representing the uncertainty in the estimation. Further, by assuming structural properties on the expected value of f such as Lipschitz continuity (w.r.t. d) the algorithm maintains a bias

correction term for the function value at the center point depending on the size of the associated subset. In each time step the algorithm chooses to evaluate the function at the center point with the highest potential to be an optimal point by taking the confidence interval and the bias correction into account. Once the width of the confidence interval is smaller than the bias term of f at a center point, the corresponding subset is refined into smaller subsets. This adds new center points to the set of considered center points, at which the estimate of f, as well as the confidence width and potential bias is computed, while the center point responsible for the refinement is left out of the consideration (or is used as the center point for a smaller subset). The rationale behind this approach is to "zoom" successively into regions of $\mathcal{X}$, where the optimum of f is located.

Following the idea of zooming algorithms, we maintain for any arm $i \in [n]$ a time-dependent grid set $\mathcal{M}_t^{(i)}$, where $\mathcal{M}_0^{(i)} = \{\tau_{\max}\}$. For any arm $i \in [n]$ and $l = 1, \ldots, |\mathcal{M}_t^{(i)}|$ denote by $\tau_l^{(i)}(t)$ the grid points in $\mathcal{M}_t^{(i)}$. Each point is representing the left-open interval $(\text{down}_i(\tau_l^{(i)}(t)), \tau_l^{(i)}(t)]$, where

$$\text{down}_i(\tau) = \max_{\tilde{\tau} \in \mathcal{M}_t^{(i)}} \{\tilde{\tau} \leq \tau\}$$

is the next smallest grid point to τ in $\mathcal{M}_t^{(i)}$, and we set $\text{down}_i(\tau) = 0$ if the set is empty. We say that $\mathcal{M}_t^{(i)}$ is extended at some point $\tau_l^{(i)} \in \mathcal{M}_t^{(i)}$ if the grid points of the equidistant decomposition[8] of $[\text{down}_i(\tau_l^{(i)}(t)), \tau_l^{(i)}(t)]$ with size $m - 1$ is added to $\mathcal{M}_t^{(i)}$. For an illustration of the grid point sets consider Fig. 3, where the initial grid point set $\mathcal{M}_0^{(i)}$ is illustrated in the top plot, while the middle plot shows the extension of $\mathcal{M}_0^{(i)}$ at the point $\tau_{\max}$ at some time step t_1 for $m = 4$. Here, for instance, $\text{down}_i(\tau_{\max}) = \frac{3\tau_{\max}}{4}$ and $\text{down}_i(\frac{\tau_{\max}}{4}) = 0$, so that

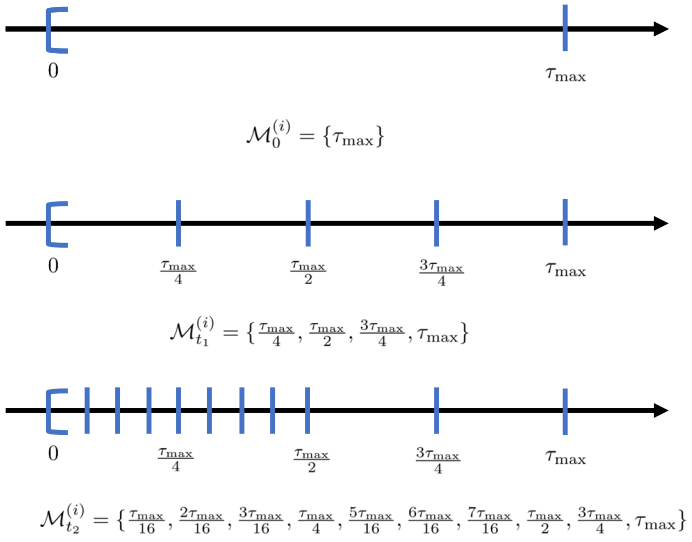

Fig. 3 Initial grid point set $\mathcal{M}_0^{(i)}$ (top plot). Extension of $\mathcal{M}_0^{(i)}$ at the point $\tau_{\max}$ at some time step t_1 for $m = 4$ (middle plot). Multiple extension of $\mathcal{M}_{t_1}^{(i)}$ at the points $\tau_{\max}/4$ and $\tau_{\max}/2$ at some time step $t_2 > t_1$ for $m = 4$ (bottom plot)

[8] All grid points are in the interior of the interval.

$\tau_{\max}$ represents the left-open interval $(\frac{3\tau_{\max}}{4}, \tau_{\max}]$, while $\frac{\tau_{\max}}{4}$ represents $(0, \frac{\tau_{\max}}{4}]$. Note that the size $m \geq 2$ is fixed and specified by the learner and the same holds true for the criterion leading to an extension of a grid set.

Algorithm 1 The z-RCUCB algorithm

Input: $\alpha > 1$, $m \geq 2$, $\delta \in (0,1)$, $T \in \mathbb{N}$

Initialization: $\mathcal{M}_t^{(i)} = \{\tau_{\max}\}\ \forall i \in [n]$, $\hat{\nu}_{i,\tau} = 0$, $\mathfrak{c}_{i,\tau} = \infty\ \forall (i,\tau) \in [n] \times \mathcal{M}_t^{(i)}$

for $t \in \{1, 2, \ldots, T\}$ **do**
  Choose (I_t, τ_t) according to (13)
  Observe $X_{I_t,t}$ (cf. (1))
  Update $\hat{\nu}_{I_t,\tau}$, $\mathfrak{c}_{I_t,\tau}\ \forall \tau \in \mathcal{M}_t^{(I_t)}$
  for $\tau \in \mathcal{M}_t^{(I_t)}$ **do**
    if $(\tau - \mathrm{down}_{I_t}(\tau)) \geq \mathfrak{c}_{I_t,\tau}$ **then**
      $\mathcal{M}_t^{(I_t)} \leftarrow \mathtt{extend}(\mathcal{M}_t^{(I_t)}, m, \tau)$
      Initialize $\hat{\nu}_{I_t,\tau}$ $\mathfrak{c}_{I_t,\tau}$ for all new grid points τ in $\mathcal{M}_t^{(I_t)}$
Let $(\hat{i}, \hat{\tau})$ be the arm/resource-limit pair with the finest grid level among $(\mathcal{M}_t^{(i)})_{i\in[n]}$
return $(\hat{i}, \hat{\tau})$

In light of this, we suggest the z − RCUCB (zooming-RCUCB, Algorithm 1), which adapts the choice criterion (12) of RCUCB and additionally refines the finite grid points by "zooming" into subsets of $\mathcal{M}$, where the penalized expected gain of an arm seems to be large. More precisely, for some $\delta \in (0,1)$ the arm/resource limit pair in time step t chosen by z − RCUCB is

$$\big(I_t, \tau_t\big) \in \operatorname*{argmax}_{(i,\tau)\in[n]\times\mathcal{M}_t^{(i)}} \big(\hat{\nu}_{i,\tau}(t) + \mathfrak{c}_{i,\tau}(n\,T^2/\delta;\alpha) + (\tau - \mathrm{down}_i(\tau))\big). \tag{13}$$

The set of grid points $\mathcal{M}_t^{(I_t)}$ is extended at some point $\tau \in \mathcal{M}_t^{(I_t)}$ if

$$(\tau - \mathrm{down}_{I_t}(\tau)) \geq \mathfrak{c}_{I_t,\tau}(n\,T^2/\delta;\alpha) \tag{14}$$

holds. Roughly speaking, the extension criterion is used in the case where the discretization bias (represented by the left-hand side in (14)) for the left-open interval $(\mathrm{down}_{I_t}(\tau), \tau]$ is larger than the uncertainty of its representing grid point τ (represented by the confidence interval $\mathfrak{c}_{I_t,\tau}$). Note that as all counter variables $N_{I_t,\tilde{\tau}}$ such that $\tilde{\tau} \in \mathcal{M}_t^{(I_t)}$ and $\tilde{\tau} \leq \tau_t$ hold are incremented in round t, it could be that more than one grid point in $\mathcal{M}_t^{(I_t)}$ is extended in one round t. Such a multiple extension is illustrated in the bottom plot of Fig. 3.

In comparison to (12), the choice criterion in (13) incorporates an additional term $(\tau - \mathrm{down}_i(\tau))$, which can be interpreted as a bias-correction due to the discretization of $\mathcal{M}$ by $\mathcal{M}_t^{(i)}$. Moreover, the (α root of the) confidence level is set to $n\,T^2/\delta$, which requires the knowledge of T. In case the number of learning rounds T is not known beforehand, one can use the well-known *doubling trick* (Cesa-Bianchi and Lugosi 2006) to obtain an algorithm that preserves the theoretical guarantees of an algorithm that needs to know the number of learning rounds. Note that the parameters of

z − RCUCB are the exploration constant $\alpha > 1$, the confidence level $\delta \in (0, 1)$, the grid refinement size $m \geq 2$ and the total number of learning rounds T.

4.2 Theoretical guarantees

Similarly as for the infinite multi-armed bandit case or $\mathcal{X}$-armed bandits (Bubeck et al. 2011; Munos 2014), we will focus the theoretical analysis on the loss after T many rounds (simple regret) given by $L_T := \nu^* - \nu_{\hat{i}(T),\hat{\tau}(T)}$, where $(\hat{i}(T), \hat{\tau}(T))$ is the arm/resource-limit pair with the finest grid level among $(\mathcal{M}_T^{(i)})_{i\in[n]}$. For this purpose, we will make the following assumption on the local smoothness of the optimal penalized expected gain (interpreted as a function of τ):

$$\nu_{i^*,\tau^*} - \nu_{i^*,\tau} \leq |\tau^* - \tau|, \qquad \forall \tau \in \mathcal{M}.$$

Such an assumption is common in $\mathcal{X}$-armed bandits and one of the weakest assumptions in this regard (Grill et al. 2015).

Next, we need the notion of η-near-optimality dimension to capture the possible rate of convergence of the resulting estimates using the successive discretization process via the finite grid points above for the problem at hand.

Definition 1 The η-near-optimality dimension is the smallest $d \geq 0$ such that there exists a constant $C > 0$ (the η-near-optimality constant) such that for all $\varepsilon > 0$, the maximal number of disjoint sets of the form

$$\{j\} \times (a_j, b_j], \qquad j \in [n], 0 \leq a_j < b_j \leq \tau_{\max}$$

such that $|b_j - a_j| \leq \eta\varepsilon$ and (j, b_j) is an element of $\{(i, \tau) \in [n] \times \mathcal{M} \mid \nu_{i,\tau} \geq \nu_{i^*,\tau^*} - \epsilon\}$, is less than $C\varepsilon^{-d}$.

Finally, we introduce for any $l \geq 1$ the equidistant grid points with granularity m^{-l} via

$$\tau_{(l,j)} = \frac{o_j}{m}(\tau_{(l-1,s_j)} - \tau_{(l-1,s_j-1)}), \quad j \in \{1, \ldots, m^l\}$$

where $s_j = \lfloor j/m \rfloor + 1$ and

$$o_j = \begin{cases} j \mod m, & \text{if } j \mod m \neq 0 \\ m, & \text{else .} \end{cases}$$

Here, we set $\tau_{(l,0)} = 0$ and $\tau_{(l,m^l)} = \tau_{\max}$ for any l. Note that for any $l \geq 0$ and any $j, k \in \{0, \ldots, m^l\}$ such that $|j - k| \leq 1$ it holds $|\tau_{(l,j)} - \tau_{(l,k)}| \leq m^{-l}$. With this, and assuming the local smoothness, we obtain the following result for the loss of z − RCUCB.

Theorem 2 *Let d be the 1/3-near-optimality of $\{(i, \tau) \in [n] \times \mathcal{M} \mid \nu_{i,\tau} \geq \nu_{i^*,\tau^*} - \epsilon\}$, i.e., the set of all ϵ-best arm/resource-limit pairs, with corresponding near optimality constant $C > 0$. Then, for any $\delta \in (0, 1)$, $\alpha > 1$ it holds with probability at least $1 - \delta$ that*

$$L_T^{\text{z-RCUCB}} \leq \tilde{C}\left(\log(T^2/\delta)/T\right)^{\frac{1}{d+2}},$$

where

$$\tilde{C} = \left(\frac{4\alpha\, C\, m^2\, (H_T + (1 + \lambda(\tau_{\max}))^2)}{3^d (1 - m^{-(d+2)})} \right)^{1/(d+2)},$$

$$H_T = \max_{l \geq 0, 1 \leq j \leq m^l - 1} \left((1 + \lambda(\tau_{(l,j)}))^2 - \frac{1}{m^2} \left(1 + \lambda(\tau_{(l-1,s_j)})\, c_T \right)^2 \right),$$

$$c_T = \sqrt{\frac{2\alpha \log(nT^2/\delta) \left(1 + \lambda(\tau_{\max})\right)^2}{\tau_{\max}^2 T}}.$$

Moreover, setting $\delta = 1/T$ *yields* $\mathbb{E}[L_T^{\mathtt{z-RCUCB}}] = O\left((\log(T)/T)^{\frac{1}{d+2}} \right)$.

Note that the speed of convergence is basically the same as the one obtained for the $\mathcal{X}$-armed bandit setting. The difference is only regarding the constant $\tilde{C}$, which leads to an improvement over a straightforward application of an $\mathcal{X}$-armed bandit algorithms on $\mathcal{X} = [n] \times \mathcal{M}$. Indeed, using a straightforward application of some $\mathcal{X}$-armed bandit algorithm such as StoOO (Munos 2014) on $\mathcal{X} = [n] \times \mathcal{M}$, one can derive a theoretical guarantee on the loss of StoOO as in Theorem 1 with

$$\tilde{H}_T = \max_{l \geq 0, 1 \leq j \leq m^l - 1} (1 + \lambda(\tau_{(l,j)}))^2$$

replacing H_T. However, it obviously holds that $\tilde{H}_T \geq H_T$ and the gap between $\tilde{H}_T$ and H_T can be large depending on the underlying penalty function λ.

Regarding the update complexity of $\mathtt{z-RCUCB}$, we can derive the following result, which is proven in Section D.

Proposition 3 $\mathtt{z-RCUCB}$*'s update complexity is in* $O(|\mathcal{M}_t^{(I_t)}| + (m-1))$.

5 Experimental study

In this section, we present experimental results for our learning algorithm and compare it with variants of the Upper Confidence Bound algorithm (UCB) and Thompson Sampling (TS), adapted to the considered type of bandit problem in the spirit of Sect. 3.1. For further details see Section E.

5.1 Synthetic data

We consider three different problem instances **PosCorr**, **NegCorr**, and **Indep**, each consisting of $n = 10$ arms, where the correlation between the reward and resource consumption distribution of the arms is positive, negative, and zero,[9] respectively. The arm distributions for **PosCorr** are similar as in Fig. 1 in the sense that the correlation level is the same for all arms and only the arm's means are different, while for **NegCorr** the correlation structure of **PosCorr** is simply reversed and for **Indep** no correlation structure is present at all. The explicit choice of the distributions is detailed in Section E. For all problem instances we consider the admissible resource range (0, 1], i.e., $\tau_{\max} = 1$ and an equidistant

[9] In fact, the distributions are even independent for **Indep**.

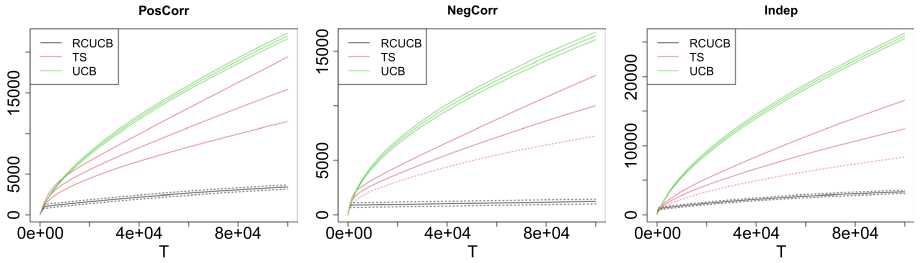

Fig. 4 Mean cumulative regret (solid lines) for UCB ($\alpha = 1$), TS and RCUCB ($\alpha = 1$) for the **PosCorr**, **NegCorr**, and **Indep** problem instances. The dashed lines depict the empirical confidence intervals, using the standard error

grid point set $\mathcal{M}$ for the admissible resource range (0, 1] of size 10. We also consider varying grid point sizes as well as varying numbers of arms in Section F. In light of our running example in Sect. 2, we use for the cost function $c(x) = x/10$ and for the penalty function $\lambda(x) = c(x)\mathbb{1}_{\{x \leq 0.5\}} + 10x\mathbb{1}_{\{x > 0.5\}}$. All considered learning algorithms proceed over a total number of $T = 100,000$ rounds.

Figure 4 illustrates the mean cumulative regret over 100 repetitions for these problem scenarios. It is clearly visible that RCUCB distinctly outperforms both UCB and TS on each problem instance. Regarding the impact of the correlation on the performance, we see that RCUCB reveals a much better performance, if there is a correlation - either negative or positive - present between the reward and the consumption of resource distribution compared to the considered baselines. Indeed, in Fig. 4, we see that the relative gap between RCUCB and its competitors is larger for the correlated problem settings than for the uncorrelated one. Thus, the learning behavior of RCUCB seems to profit from available correlations of the two distributions.

Finally, the following table reports the (mean) proportion of censored rounds, i.e., where the resource limit was exceeded or equivalently no reward was observed, as well as the probability of observing a censored observation for the optimal arm/resource-limit pair (i^*, τ^*).

	RCUCB	TS	UCB	$1 - P_{i^*}^{(c)}(\tau^*)$
PosCorr	0.6399	0.7516	0.7993	0.6116
NegCorr	0.7700	0.8510	0.8894	0.7631
Indep	0.4462	0.5749	0.5455	0.4404

As expected, both UCB and TS seemingly fail to process the censored feedback in an appropriate way, as the proportion of rounds with censored rewards is much higher than the actual ground truth probability of obtaining censored rewards for the optimal arm/limit pair. Thus, RCUCB is again preferable over the two competing algorithms.

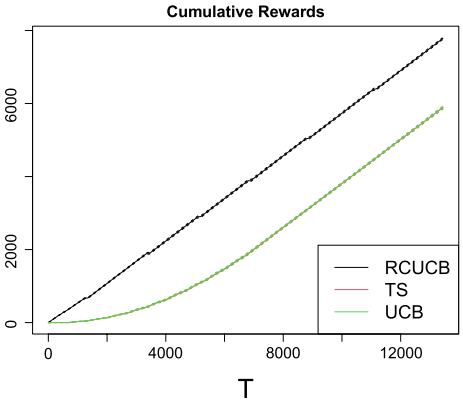

Fig. 5 Mean cumulative rewards for the algorithm configuration task of Random Forest on the AmesHousing data set

5.2 Algorithm configuration

We consider the problem of configuring a Random Forest for regression over a variety of tuning parameters (arms) in an on-the-fly manner, within a reasonable time for the training (resource) on a specific data set. To this end, we consider the *AmesHousing* data set,[10] which is randomly split into a 70:30 training/test set in each learning round, and each learner chooses a Random Forest parameter configuration as well as a time limit for the training. As reward for the learner, we use 1 minus the (normalized) root-mean-squared error on the test data, provided the learner's predefined limit for the training is not exceeded. Otherwise, the learner obtains a reward of zero, i.e., the feedback is censored.

The considered set of possible parameters of the Random Forest consists of

- The number of trees: $\{100, 200, \ldots, 700\}$,
- The number of variables to randomly sample as candidates at each split: $\{20, 22, \ldots, 30\}$,
- The minimal node size: $\{3, 5, 7, 9\}$,
- The fraction of training samples for bagging: $\{0.55, 0.632, 0.75, 0.8\}$.

The remaining parameters of the Random Forest are set as the default parameters as specified in the R-package 'ranger'.[11] Each combination is treated as an arm, resulting in $n = 672$ arms in total.

For the admissible range of time limits for training, we have used an equidistant grid of size $m = 10$ of the interquartile range of the obtained training times if each possible configuration is run once. Motivated by the PAR10 loss in algorithm configuration problems (Kerschke et al. 2019), we use $c(x) = x$ for the cost function, while the penalty function is $\lambda(x) = 10x$. The total number of rounds is set to $T = 2\,n\,m$, and the number of repetitions to 10. All these experiments were conducted on a machine featuring Intel(R) Core(TM) i7-8550U@1.80 GHz CPUs with 4 cores and 16 GB of RAM.

The mean cumulative rewards over 10 repetitions of the algorithms is shown in Fig. 5, in which we see that the cumulative reward of `RCUCB` exceeds the cumulative rewards of

[10] https://cran.r-project.org/package=AmesHousing

[11] https://cran.r-project.org/web/packages/ranger/index.html

UCB and TS throughout. Note that both UCB and TS have almost the same mean cumulative rewards, so they are barely distinguishable in the plot. The proportion of rounds, where the time limit for the training was exceeded, is 0.0589 for RCUCB, 0.2843 for TS, and 0.2853 for UCB. Again the obtained results are in favor of RCUCB.

6 Related work

Various authors have considered bandit problems in which each arm is equipped with a multivariate distribution, i.e., in which the learner receives potentially vectorial type of feedback. In the *bandit problem with delayed feedback* (Joulani et al. 2013; Mandel et al. 2015; Vernade et al. 2017; Pike-Burke et al. 2018), each arm possesses a reward and a reward-time generation distribution, and rewards of previously chosen arms can be observed in a later round. This is fundamentally different from our setting, where an arm's reward is only observable in the round it is played. Moreover, the two distributions occurring in the bandits problem with delayed feedback are usually assumed to be independent, while we do not make an independence assumption on the reward and consumption of resources distribution of the arms.

Multivariate feedback of a played arm is at the core of the *multi-objective multi-armed bandit* (MOMAB) problem. Due to the possibly competing objectives encoded in the vectorial payoffs, different approaches have been considered to specify an optimal decision in the MOMAB problem. Each objective is considered as a different multi-armed bandit problem by Gabillon et al. (2011), and the aim is to find the optimal arm for each objective separately. In the majority of works, the Pareto front with respect to the mean vector is used to determine the optimality of an arm (see Auer et al. 2016 or Drugan 2019) as well as references therein). Finally, by aggregating the vectorial payoff by means of the generalized Gini index, a single objective to be optimized can again be obtained (Busa-Fekete et al. 2017).

In the *bandits with knapsacks problem* (Badanidiyuru et al. 2013; Slivkins 2019; Cayci et al. 2020), each arm is associated with a reward and a (multivariate) resource consumption distribution as well. Although the original problem does not involve censoring rewards, variations of this problem have recently been considered in which the learner also has the option of setting a round-by-round limit on an arm's resource consumption that leads to censored rewards if exceeded (Cayci et al. 2019; Sharoff et al. 2020). Nonetheless, the learning process in both the original bandits with knapsacks problem as well as the censored variant is substantially different from the one considered in this work: There is a predefined overall resource budget, which once exhausted leads to a termination of the entire learning process. This in turn leads to a different notion of cumulative regret and consequently different approaches regarding its theoretical analysis.

Censored feedback due to thresholding has been considered by Abernethy et al. (2016), Jain and Jamieson (2018) and Verma et al. (2019) within a bandit learning setting as well, albeit without multivariate distributions of the arms. Also, the threshold values are either specified in each round by the environment or unknown but fixed among the arms, whereas in our setting, the learner chooses the threshold itself.

Resource allocation in a combinatorial bandit scenario has been the subject of research by Lattimore et al. (2014) and Dagan and Koby (2018) as well as for a contextual variant by Lattimore et al. (2015). However, in all these scenarios, the reward distributions

are Bernoulli with a specific shape of the success probability depending on the allocated resources.

Bidding in online auctions (Cesa-Bianchi et al. 2014) is also concerned with choosing a suitable resource limit (reserve price for auctions) and obtaining possibly censored feedback from bidders. This scenario is different from ours, as all available bidders are involved in a learning round (auction), while in our setting the learning algorithm has to pick one of the bidders in a metaphorical sense.

Ephemeral resource-constrained optimization problems (ERCOPs) are dealing with a dynamic constrained optimization problem, in which both the objective function and the set of feasible solutions is static and certain time-dependent constraints may exist such that certain solutions may be temporarily unavailable for evaluation (Allmendinger and Knowles 2010; Allmendinger and Knowles 2011; Allmendinger and Knowles 2013; Allmendinger and Knowles 2015). These dynamic constraints are referred to as ephemeral resource constraints and account for a possible temporary unavailability of the resources needed to evaluate a solution. This results in an online learning problem similar to ours, where evaluations of solutions (an *arm-resource pair* in our terminology) may be incomplete (or fail). However, existing work on ERCOPs explicitly considers only scenarios where it can be checked a priori whether a solution will be incomplete (*censored* in our terminology) without actually trying it. This leads to entirely different learning/optimization approaches than ours, as the evaluability of a solution is stochastic in our setting and consequently excludes such prechecks. Moreover, in our case, the focus is on a specific performance measure (regret) to evaluate an optimization strategy over time, which has significant implications for the design of appropriate strategies. For example, one optimization strategy for ERCOPs is to wait until a certain solution is available (or evaluable) again; a strategy that seems to be questionable with regard to cumulative regret minimization.

Finally, as we take the online algorithm selection problem as a running example for our setting, it is worth mentioning that bandit-based approaches have been already considered for this problem (Gagliolo and Schmidhuber 2007; Gagliolo and Schmidhuber 2010; Degroote 2017; Degroote et al. 2018; Tornede et al. 2022). However, these focus on certain algorithmic problem classes, such as the boolean satisfiability problem (SAT) or the quantified boolean formula problem (QBF). In particular, these works consider binary reward signals (the solver has solved/not solved the problem instance) and the runtimes of their respective solvers as the consumption of resources. Extending these approaches to more general frameworks like ours with continuous reward signals or other types of resource consumptions is far from a given.

7 Conclusion and future work

In this paper, we have introduced another variant of the classical multi-armed bandit problem, where attention is paid not only to the rewards themselves, but also to the resources consumed by an arm necessary to generate the rewards within each round of the sequential decision process. The learner (bandit policy) is equipped with the ability to determine the resource limit of one round and might be willing to sacrifice optimality regarding the obtained rewards in order to keep the overall consumption of resources low, as this generates costs diminishing the overall gain. As a consequence, the learner needs to find a good compromise between the two possibly conflicting targets, namely an as high as possible

realizable reward on the one side, and an as low as possible consumption of resources for the reward generation on the other side.

To this end, we proposed a regret measure, which, by virtue of a cost and a penalty function, takes these two targets into account and allows for a suitable assessment of the expected gain with respect to the allocated resources. By defining a suitable estimate of an arm's expected gain and its probability to exceed the allocated resources, we proposed optimistic bandit strategies for dealing with a finite or an infinite subset of available resource limits.

For future work, it would be interesting to extend the considered bandit problem to a combinatorial bandit setting, in which it is possible to choose a subset of arms in each round. Moreover, the very idea of incorporating resource constraints for the feedback generation process is not restricted to feedback of numerical nature, but could also be of interest for related bandit scenarios with other types of feedback, such as the preference-based multi-armed bandit problem (Bengs et al. 2021). Last but not least, as the motivation of the considered type of bandit problem stems from practical applications, it would be of interest to investigate our algorithm for a variety of real-world problems, such as a more extensive simulation study on algorithm configuration (Schede et al. 2022).

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10994-022-06271-z.

Funding Open Access funding enabled and organized by Projekt DEAL. Not applicable.

Data availability The data set used in this paper is freely available: https://cran.r-project.org/package=AmesHousing

Code availability All authors agree upon publishing the code in case of acceptance.

References

Abe, N., Biermann, A., & Long, P. (2003). Reinforcement learning with immediate rewards and linear hypotheses. *Algorithmica, 37*(4), 263–293.

Abernethy, J., Amin, K., & Zhu, R. (2016). Threshold bandit, with and without censored feedback. In *NeurIPS* (pp. 4896–4904).

Agrawal, S., & Goyal, N. (2012). Analysis of Thompson sampling for the multi-armed bandit problem. In *COLT* (pp. 1–39).

Allmendinger, R., & Knowles, J. (2010). On-line purchasing strategies for an evolutionary algorithm performing resource-constrained optimization. In *International Conference on Parallel Problem Solving from Nature* (pp. 161–170). Springer.

Allmendinger, R., & Knowles, J. (2011). Policy learning in resource-constrained optimization. In *GECCO* (pp. 1971–1978).

Allmendinger, R., & Knowles, J. (2013). On handling ephemeral resource constraints in evolutionary search. *Evolutionary Computation, 21*(3), 497–531.

Allmendinger, R., & Knowles, J. (2015). Ephemeral resource constraints in optimization. In *Evolutionary Constrained Optimization* (pp. 95–134). Springer.

Auer, P. (2002). Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research, 3*(Nov), 397–422.
Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning, 47*(2–3), 235–256.
Auer, P., Chiang, C. K., Ortner, R., & Drugan, M. (2016). Pareto front identification from stochastic bandit feedback. In *AISTATS* (pp. 939–947).
Badanidiyuru, A., Kleinberg, R., & Slivkins, A. (2013). Bandits with knapsacks. In *Annual Symposium on Foundations of Computer Science* (pp. 207–216). IEEE.
Bengs, V., Busa-Fekete, R., El Mesaoudi-Paul, A., & Hüllermeier, E. (2021). Preference-based online learning with dueling bandits: A survey. *Journal of Machine Learning Research, 22*(7), 1–108.
Bubeck, S. (2010). *Bandits games and clustering foundations*. Ph.D. thesis, Université des Sciences et Technologie de Lille-Lille I.
Bubeck, S., Munos, R., Stoltz, G., & Szepesvári, C. (2011). X-armed bandits. *Journal of Machine Learning Research, 12*(5), 1655–1695.
Busa-Fekete, R., Szörényi, B., Weng, P., & Mannor, S. (2017). Multi-objective bandits: Optimizing the generalized Gini index. In *ICML* (pp. 625–634).
Cayci, S., Eryilmaz, A., & Srikant, R. (2019). Learning to control renewal processes with bandit feedback. *Proceedings of the ACM on Measurement and Analysis of Computing Systems, 3*(2), 1–32.
Cayci, S., Eryilmaz, A., & Srikant, R. (2020). Budget-constrained bandits over general cost and reward distributions. In *AISTATS* (pp. 4388–4398).
Cesa-Bianchi, N., Gentile, C., & Mansour, Y. (2014). Regret minimization for reserve prices in second-price auctions. *IEEE Transactions on Information Theory, 61*(1), 549–564.
Cesa-Bianchi, N., & Lugosi, G. (2006). *Prediction, learning, and games*. Cambridge University Press.
Cesa-Bianchi, N., & Lugosi, G. (2012). Combinatorial bandits. *Journal of Computer and System Sciences, 78*(5), 1404–1422.
Dagan, Y., & Koby, C. (2018). A better resource allocation algorithm with semi-bandit feedback. In *ALT* (pp. 268–320).
Degroote, H. (2017). Online algorithm selection. In *IJCAI* (pp. 5173–5174).
Degroote, H., Causmaecker, P. D., Bischl, B., & Kotthoff, L. (2018). A regression-based methodology for online algorithm selection. In *Proceedings of the Eleventh International Symposium on Combinatorial Search, SOCS 2018* (pp. 37–45).
Drugan, M. (2019). Covariance matrix adaptation for multiobjective multiarmed bandits. *IEEE Transactions on Neural Networks and Learning Systems, 30*(8), 2493–2502.
Gabillon, V., Ghavamzadeh, M., Lazaric, A., & Bubeck, S. (2011). Multi-bandit best arm identification. In *NeurIPS* (pp. 2222–2230).
Gagliolo, M., & Schmidhuber, J. (2007). Learning restart strategies. In *IJCAI* (pp. 792–797).
Gagliolo, M., & Schmidhuber, J. (2010). Algorithm selection as a bandit problem with unbounded losses. In *International Conference on Learning and Intelligent Optimization (LION)* (pp. 82–96). Springer.
Grill, J.B., Valko, M., & Munos, R. (2015). Black-box optimization of noisy functions with unknown smoothness. In *NeurIPS* (pp. 667–675).
Hutter, F., Kotthoff, L., & Vanschoren, J. (2019). *Automated machine learning: Methods, systems, challenges*. Springer.
Jain, L., & Jamieson, K. (2018). Firing bandits: Optimizing crowdfunding. In *ICML* (pp. 2206–2214).
Joulani, P., György, A., & Szepesvári, C. (2013). Online learning under delayed feedback. In *ICML* (pp. 1453–1461).
Kerschke, P., Hoos, H., Neumann, F., & Trautmann, H. (2019). Automated algorithm selection: Survey and perspectives. *Evolutionary Computation, 27*(1), 3–45.
Kleinberg, R., Slivkins, A., & Upfal, E. (2008). Multi-armed bandits in metric spaces. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing* (pp. 681–690).
Lattimore, T., Crammer, K., & Szepesvári, C. (2014). Optimal resource allocation with semi-bandit feedback. In *UAI* (pp. 477–486).
Lattimore, T., Crammer, K., & Szepesvári, C. (2015). Linear multi-resource allocation with semi-bandit feedback. In *NeurIPS* (pp. 964–972).
Lattimore, T., & Szepesvári, C. (2020). *Bandit algorithms*. Cambridge University Press.
Mandel, T., Liu, Y.E., Brunskill, E., & Popović, Z. (2015). The queue method: Handling delay, heuristics, prior data, and evaluation in bandits. In *AAAI* (pp. 2849–2856).
Munos, R. (2014). From bandits to Monte-Carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends® in Machine Learning, 7*(1), 1–129.
Pike-Burke, C., Agrawal, S., Szepesvari, C., & Grunewalder, S. (2018). Bandits with delayed, aggregated anonymous feedback. In *ICML* (pp. 4105–4113).

Schede, E., Brandt, J., Tornede, A., Wever, M., Bengs, V., Hüllermeier, E., & Tierney, K. (2022). A survey of methods for automated algorithm configuration. arXiv preprint arXiv:2202.01651.

Sharoff, P., Mehta, N., & Ganti, R. (2020). A farewell to arms: Sequential reward maximization on a budget with a giving up option. In *AISTATS* (pp. 3707–3716).

Slivkins, A. (2019). Introduction to multi-armed bandits. *Foundations and Trends ® in Machine Learning, 12*(1–2), 1–286.

Tracà, S., & Rudin, C. (2021). Regulating greed over time in multi-armed bandits. *Journal of Machine Learning Research, 22*(3), 1–99.

Tornede, A., Bengs, V., & Hüllermeier, E. (2022). Machine learning for online algorithm selection under censored feedback. *Proceedings of the AAAI Conference on Artificial Intelligence, 36*(9) 10370–10380. https://doi.org/10.1609/aaai.v36i9.21279

Verma, A., Hanawal, M., Rajkumar, A., & Sankaran, R. (2019). Censored semi-bandits: A framework for resource allocation with censored feedback. In *NeurIPS* (pp. 14526–14536).

Vernade, C., Cappé, O., & Perchet, V. (2017). Stochastic bandit models for delayed conversions. In *UAI*.

Yue, Y., & Joachims, T. (2009). Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML* (pp. 1201–1208).