A Regularization Approach to Metrical Task Systems

Jacob Abernethy[1], Peter Bartlett[1], Niv Buchbinder[2], and Isabelle Stanton[1]

[1] UC Berkeley - {jake,bartlett,isabelle}@eecs.berkeley.edu
[2] Microsoft Research - nivbuchb@microsoft.com

Abstract. We address the problem of constructing randomized online algorithms for the Metrical Task Systems (MTS) problem on a metric δ against an oblivious adversary. Restricting our attention to the class of "work-based" algorithms, we provide a framework for designing algorithms that uses the technique of *regularization*, in which the algorithm's distribution on a given round is chosen as the solution to an objective that involves a curved regularization function. For the case when δ is a uniform metric, we exhibit two algorithms that arise from this framework, and we prove a bound on the competitive ratio of each. We show that the second of these algorithms is $\ln n + O(\log \log n)$ competitive, which is the current state-of-the art for the uniform MTS problem. We discuss a novel method for potentially solving the optimal MTS algorithm for general metrics.
*This paper is eligible for the E.M. Gold award for Isabelle and Jacob

1 Introduction

Consider the problem of driving on a congested multi-lane highway with the goal of getting home as fast as possible. You are always able to estimate the speed of all of the lanes, and must pick some lane in which to drive. At any time you are able to switch lanes, but pay an additional penalty for doing so proportional to the distance from your current lane. How should you pick lanes and when should you switch?

This is a concrete example of the *metrical task system* (MTS) problem, first introduced by Borodin, Linial and Saks [1]. The problem is defined on a space of n states with an associated distance metric function. The input to the problem is a series of cost vectors $\mathbf{c} \in \mathbb{R}^n_+$. An MTS algorithm must choose a state i after seeing each $\mathbf{c}$ and must pay the service cost c_i. In addition, the algorithm pays a cost for switching between states that is equal to their distance in the given metric. An alternative model, and the focus of the present work, is to imagine a *randomized* algorithm that maintains a distribution over the states on each round, and pays the expected switching and servicing cost.

Metrical task systems form a very general framework in which many well-known online problems can be posed. The k-server problem on an n-state metric [2], for example, can be modeled as a metrical task system problem with $\binom{n}{k}$ states although this reduction of making each k-subset a state leads to non-optimal bounds. Another example is process migration over a compute cluster - in this view each node is a state, the distance metric represents the amount of time it takes for a process to migrate from one node to another and the cost vector represents the current load on the machine.

The randomized MTS problem looks strikingly similar to one much more familiar to the learning community: the "hedge" or "expert" setting [3]. The hedge problem also requires choosing a distribution on $[n]$ on each of a sequence of rounds, witnessing a cost vector, and paying the associated expected cost of the selected distribution. The two primary distinctions are that (a) no switching cost is paid in the hedge problem and (b) the MTS *comparator*, i.e. the offline strategy that the only algorithm competes against, is given much more flexibility. In the hedge problem, the algorithm is only compared to an offline algorithm that must fix its state throughout the game, whereas the MTS offline comparator may choose the cheapest sequence of states knowing all service cost vectors in advance.

The most common measure of the quality of an MTS algorithm is the *competitive ratio*, which takes the performance of the online algorithm on a worst-case sequence of cost vectors a nd divides this by the cost of the optimal offline comparator on the same sequence. This is a notable departure from the notion of *regret*, which measures the *difference* between the worst-case online and offline cost, and is a much more common metric for evaluating learning algorithms. This extension is necessary because the complexity of the MTS comparator grows over time.

Prior Work

Borodin, Linial and Saks [1] showed that the lower bound on the competitive ratio of any deterministic algorithm over any metric is $2n - 1$. They also designed an algorithm, the Work-Function algorithm, that achieves exactly this bound. This algorithm was further analyzed by Schafer and Sivadasan using the smoothed analysis techniques of Spielman and Teng to show that the average competitive ratio can be improved to $o(n)$ when the topological features of the metric are taken into account [4].

Results improve dramatically when randomization is allowed. The first result for general metrics was an algorithm that achieved a competitive ratio of $\frac{e}{e-1}n - \frac{1}{e-1}$, [5] by Irani and Seiden. In a breakthrough result, Bartal, Blum, Burch and Tompkins [6] gave the first poly-logarithmic competitive algorithm for all metric spaces. This algorithm uses Bartal's result for probabilistically embedding general metric spaces into hierarchically well-separated trees [7, 8]. Fiat and Mendel [9] improved this result further to the currently best competitive algorithm that is $O(\log^2 n \log\log n)$. Recently, Bansal, Buchbinder and Naor [10] proposed another algorithm for general metrics based on a primal-dual approach that is only $O(\log^3 n)$-competitive, but has an optimal competitive factor with respect to service costs. The best known lower bound on the competitive ratio for general metrics is $\Omega(\log n / \log^2 \log n)$ [11]. This improves upon the previous bound of $\Omega(\sqrt{\log n / \log^2 \log n})$ [12]. A widely believed conjecture is that an $O(\log n)$-competitive algorithm exists for all metric spaces.

Better bounds are known for some special metrics. For example, for the line metric a slightly better result of $O(\log^2 n)$ is known [9]. Other metrics for which better results are known are the weighted star metric for which an $O(\log n)$-competitive algorithm is known [9, 13]. The best understood, and most extensively studied metric space is the uniform metric. For the uniform case, Bartal, Linial and Saks [1] showed a lower bound on the competitive ratio for any algorithm of H_n, the nth harmonic number. They [1] also designed an algorithm, Marking, that has competitive ratio $2H_n$. An alternate algorithm, Odd-Exponent [6], bears some similarity to one of the algorithms in this paper, and has a $4\log n+1$ competitive ratio on the uniform metric. This upper bound was further improved by the Exponential algorithm [5] to $H_n + O(\sqrt{\log n})$. Recently, the Wedge algorithm [14] was introduced with competitive ratio of $\frac{3}{2}H_n - \frac{1}{2n}$. They claim that this achieves a better competitive ratio when $n < 10^8$. Bansal, Buschbinder and Naor [15, 16] designed an algorithm for the uniform metric that is based on a previous primal-dual approach and has near optimal competitive ratio.

Our Contributions We make several contributions to the randomized metrical task system problem. In Section 2, we propose a clear and coherent framework for developing and analyzing algorithms for the MTS problem. We appeal to the class of *work-based* algorithms for which the probability distribution is chosen as a function of the *work vector*, to be defined in the Preliminaries. We provide the most comprehensive set of analytical tools for bounding the competitive ratio of work-based algorithms.

In Section 3, we develop an approach to the MTS problem using a *regularization* framework. This provides a generic template for constructing randomized MTS algorithms based on certain parameters of the regularized objective. For the case of the uniform metric, we employ the entropy function as a regularizer and exhibit two novel algorithms. The second of these achieves the current state-of-the-art competitive ratio of $H_n + O(\log\log n)$. We discuss potential methods for constructing general-metric algorithms as well.

1.1 Preliminaries

The set $[n] := \{1,\ldots,n\}$ is a *metric space* if there exists a distance metric $\delta : [n] \times [n] \to \mathbb{R}_+$. The primary feature of metrics that we will use is that they satisfy the triangle inequality.

Given two distributions $\mathbf{p}^1, \mathbf{p}^2 \in \Delta_n$, where Δ_n is the n-dimensional probability simplex, we define the Earth Mover Distance, or Wasserstein Distance, $\text{dist}_\delta(\mathbf{p}^1,\mathbf{p}^2)$, as the least expensive way to transition between $\mathbf{p}^1$ and $\mathbf{p}^2$. Precisely, it is the solution to

$$\begin{array}{ll} \text{min} & \sum_{i,j\in[n]} \delta(i,j)x_{i,j} \\ \text{subject to} & \mathbf{1}_n^\top [x_{i,j}] = \mathbf{p}^1 \\ & [x_{i,j}]\mathbf{1}_n = \mathbf{p}^2 \\ & x_{i,j} \geq 0 \quad \forall i,j \in [n] \end{array}$$

In one case, the Earth Mover Distance is rather easy to compute. In fact, for the uniform metric, the Earth Mover Distance is exactly the variational distance. The proof of the following Lemma is straightforward.

Lemma 1. *Assume we are given $\mathbf{p}^1, \mathbf{p}^2 \in \Delta_n$ with the property that $\mathbf{p}^1$ dominates $\mathbf{p}^2$ at every coordinate but i, that is $p^1_j \geq p^2_j$ whenever $j \neq i$. Then*

$$dist_\delta(\mathbf{p}^1,\mathbf{p}^2) = \sum_{j\in[n]\setminus\{i\}} (p^1_j - p^2_j)\delta(i,j)$$

The Randomized Metrical Task Systems problem The Metrical Task Systems (MTS) problem is formally defined as follows. Given n states and a metric δ over $[n]$, a randomized algorithm is given a sequence of *service cost vectors* $\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T \in \mathbb{R}^n_+$ as input and must choose a sequence of distributions $\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^T \in \Delta_n$ as a response. The *cost* of some algorithm A is the total expected servicing cost plus the total moving cost, i.e.

$$\mathrm{cost}_A(\mathbf{c}^1, \ldots, \mathbf{c}^T) := \sum_{t=1}^{T} \left(\mathbf{p}^t \cdot \mathbf{c}^t + \mathrm{dist}_\delta(\mathbf{p}^t, \mathbf{p}^{t-1})\right)$$

where $\mathbf{p}^0$ is set to some default distribution, which we assume to be $\langle 1, 0, \ldots, 0\rangle$ by convention.

An *offline* MTS algorithm may select $\mathbf{p}^t$ with knowledge of the entire sequence of cost vectors $\mathbf{c}^1, \ldots, \mathbf{c}^T$. We refer to the optimal offline algorithm by $\mathrm{OPT}(\mathbf{c}^1, \ldots, \mathbf{c}^T)$. In this Section we discuss how OPT can be computed easily with a simple dynamic program.

An *online* MTS algorithm can select $\mathbf{p}^t$ with knowledge only of $\mathbf{c}^1, \ldots, \mathbf{c}^t$. Notice that, unlike in the usual "expert setting", we let an online algorithm have access to the cost vector $\mathbf{c}^t$ before the distribution $\mathbf{p}^t$ is chosen and the cost $\mathbf{p}^t \cdot \mathbf{c}^t$ is paid.

We measure the performance of an online algorithm by its *Competitive Ratio* (CR), which is the ratio of the cost of this algorithm relative to the cost of the optimal offline algorithm on a worst-case sequence. More precisely, the CR is the minimal value $C > 0$ for which there is some b such that, for any T and any sequence $\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T$,

$$\mathrm{cost}_A(\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T) \quad \leq \quad C \cdot \mathrm{cost}_{\mathrm{OPT}}(\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T) + b$$

The additive term b, which can depend on the fixed parameters of the problem, is included to deal with potential fixed "startup costs". Indeed, without affecting the definition of competitive ratio we can assume that b is $o(\mathrm{cost}_{\mathrm{OPT}}(\mathbf{c}^1, \ldots))$, although here (and in most work) b is assumed to be constant.

The Work Function We observe that the offline algorithm OPT need not play in a randomized fashion because the optimal distributions $\mathbf{p}^t$ will occur at the corners of the simplex. Hence, computing OPT is not difficult, and can be reduced to a simple dynamic programming problem. The elements of this dynamic program are fundamental to all of the results in this paper, and we now define it precisely. Given a sequence $\mathbf{c}^1, \ldots, \mathbf{c}^T$, we define the *work function vector* $\mathbf{W}^t$ at time t by the following recursive definition:

$$\begin{aligned} \mathbf{W}^0 &:= \langle 0, \infty, \ldots, \infty\rangle \\ W_i^t &:= \min_{j \in [n]} \left\{ W_j^{t-1} + \delta(i,j) + c_j^t \right\} \end{aligned}$$

The work function value W_i^t is exactly the smallest total cost incurred by an offline algorithm for which $\mathbf{p}^t = \mathbf{e}_i$, i.e. one which must be at location i at time t. Indeed, if we define

$$W_{\min}^t := \min_i W_i^t,$$

then we see that OPT$(\mathbf{c}^1, \ldots, \mathbf{c}^T) = W^T_{\min}$.

If we think of the work vector $\mathbf{W}^t$ as a function from $[n]$ to $\mathbb{R}$, where $\mathbf{W}^t(i) := W^t_i$, then it is easily checked that $\mathbf{W}^t$ is *1-Lipschitz* with respect to the metric δ. That is, for all $i, j \in [n]$, $|W^t_i - W^t_j| \leq \delta(i, j)$. We define a notion of a *supported state* which occurs when this Lipschitz constraint becomes tight.

Definition 1. *Given some work vector $\mathbf{W}^t$ with respect to a metric δ, the state i is* supported *if there exists a $j \neq i$ such that $W^t_i = W^t_j + \delta(i, j)$.*

Throughout this text, when it is unnecessary, we will drop the superscript t from $\mathbf{W}^t$, W^t_i, $\mathbf{p}^t$ and p^t_i.

2 The Work-Based MTS Framework

In this Section we lay out a framework for designing randomized MTS algorithms. This framework imposes a number of significant restrictions on the algorithm, and makes relatively strong assumptions about the types of inputs the algorithm will observe.

The Simplified MTS Framework

1. *The algorithm will be "work-based",* that is, we choose $\mathbf{p}^t = \mathbf{p}(\mathbf{W}^t)$ for some fixed function $\mathbf{p}$ regardless of the sequence of cost vectors that resulted in $\mathbf{W}$.
2. *All cost vectors are "elementary":* every $\mathbf{c}^t$ has the form $\alpha\mathbf{e}_i$ for some $\alpha > 0$ and some i
3. *The algorithm will be "reasonable":* whenever $W_i = W_j + \delta(i, j)$ for some j, i.e. i is a *supported state*, then it must be that $p_i(\mathbf{W}) = 0$.
4. *The cost vectors will be "reasonable" as well:* Given a current work vector $\mathbf{W}$, if a cost $\mathbf{c} = \alpha\mathbf{e}_i$ is received then $\alpha \leq W_j - W_i + \delta(i, j)$ for all j
5. *The algorithm will be "conservative":* whenever a cost $\mathbf{c} = \alpha\mathbf{e}_i$ is received given current work vector $\mathbf{W}$, then for each $j \neq i$ we have $p_j(\mathbf{W}) \leq p_j(\mathbf{W} + \alpha\mathbf{e}_i)$ – that is, the probabilities at other locations can not decrease.

This paper focuses entirely on the construction of work-based algorithms, where the algorithm can forget about the sequence of cost vectors $\mathbf{c}^1, \ldots, \mathbf{c}^t$ and simply use $\mathbf{W}^t$ to choose $\mathbf{p}^t$. This algorithmic restriction has been used as early as [1] and appears elsewhere. It has not been shown, to the best of our knowledge, that this restriction is made *without sacrificing optimality*. We conjecture this to be true.

Conjecture 1. There is an optimal randomized MTS algorithm that is work-based. In other words, there is an optimal algorithm such that, after receiving $\mathbf{c}^1, \ldots, \mathbf{c}^t$, the probability $\mathbf{p}^t$ need only depend on the resulting work vector $\mathbf{W}^t$.

Strictly speaking, we need not settle this conjecture to proceed with developing algorithms within this restricted class. However, if it were settled in the affirmative this would suggest that the algorithmic design problem can be safely restricted to this smaller class of algorithms. Indeed, by making this assumption we gain a number of other simple and appealing properties as we discuss below.

For the remaining assumptions/restrictions in our simplified framework, each has been previously justified in other works–that is, each is made without loss of generality.

The restriction of using elementary cost vectors is discussed by Irani and Seiden in [5], which was first published in 1995. They argue that this is an equivalent formulation of the problem. Once we make the work-based assumption, the fact that our algorithm must satisfy the reasonableness property is straightforward – whenever this property is broken an adversary can induce an unbounded competitive ratio [6]. Again, making the work-based assumption, it is shown by [9] that it is sufficient to consider reasonable cost sequences – any unreasonable cost vectors can be truncated without changing either the online algorithm's cost or the cost of OPT. The conservative property is used throughout the literature and is easy to justify.

Before proceeding, we mention one useful fact that results from our framework, whose proof we omit but can be easily checked.

Lemma 2. *Under the assumption that the sequence of cost vectors* $\mathbf{c}^1, \dots, \mathbf{c}^t$ *is reasonable, the work vector is precisely* $\mathbf{W}^t = \mathbf{c}^1 + \ldots + \mathbf{c}^t$.

2.1 Relationship to the Experts Setting

Before proceeding, let us show why the proposed framework brings us much closer to a much more well-understood problem: the so-called "expert" or "hedge" setting [3]. Here, the algorithm must choose a probability distributions $\mathbf{p}^t \in \Delta_n$ on each round t, and an adversary then chooses a loss vector $\boldsymbol{l}^t \in [0,1]^n$. Let us write $\mathbf{L}^t = \sum_{s=1}^t \boldsymbol{l}^s$. Then the goal of the algorithm is to minimize $\sum_{t=1}^T \mathbf{p}^t \cdot \boldsymbol{l}^t$ relative to the loss of the "best expert", i.e. $\min_i L_i^t$.

Within our MTS framework, the story is quite similar. The algorithm and adversary choose $\mathbf{p}^t$ and $\mathbf{c}^t$ on each round, with the goal of minimizing $\sum_{t=1}^T \left(\mathbf{p}^t \cdot \mathbf{c}^t + \text{dist}_\delta(\mathbf{p}^t, \mathbf{p}^{t-1})\right)$. By Lemma 2, $\mathbf{W}^T = \sum_{t=1}^T \mathbf{c}^t$, and the algorithm's goal is to pay as little as possible relative to $\min_i W_i^t$.

These problems have a strong resemblance, yet there are several critical differences:

- The MTS algorithm has *one-step lookahead*, i.e. it can select $\mathbf{p}^t$ with knowledge of $\mathbf{c}^t$
- An additional penalty $\text{dist}_\delta(\mathbf{p}^{t-1}, \mathbf{p}^t)$ for moving is added to the objective for MTS
- The algorithm must be "reasonable", requiring that the probability p_i^t must vanish under certain conditions of $\mathbf{W}^t$

While the first point would appear quite advantageous for MTS, this benefit is spoiled by the latter two. As is well-known in the expert setting, we can ensure that the average cost of the algorithm approaches the comparator $\min_i L_i^t$ using an algorithm like Hedge, whereas in the MTS setting a lower bound shows that this ratio is at least $\Omega(\log n / \log^2 \log n)$ for the work function comparator [11]. At a high level, this is because charging the algorithm for adjusting its distribution *and* requiring that the probability vanishes on certain states causes the algorithm to pay a huge amount in transportation.

In Section 3, we borrow some tools from the hedge setting such as *entropy regularization* and potential functions. Algorithms from the hedge setting have been used on the MTS problem before, most notably by [17]. Their approach is quite different from the one we take. They imagine competing against a "switching expert" and modify known results developed by [18]. Their approach, while quite interesting, is not a work-based algorithm and does not achieve an optimal bound.

2.2 Bounding Costs using Potential Functions

We turn our attention to bounding the cost of a work-based MTS algorithm $\mathbf{p}$ on a worst-case sequence of costs. First, we make a simple observation about work-based algorithms that adhere to our framework. Given a work vector $\mathbf{W}$, consider the cost to the algorithm when vector $\mathbf{c} = \epsilon \mathbf{e}_i$ is received and the work vector becomes $\mathbf{W}^1 = \mathbf{W} + \epsilon \mathbf{e}_i$. The probability distribution transitions to $\mathbf{p}(\mathbf{W}^1)$, and the service cost is $\mathbf{p}(\mathbf{W}^1) \cdot \mathbf{c} = \epsilon p_i(\mathbf{W}^1)$. By the conservative assumption, we compute the switching cost by appealing to Lemma 1. Hence, the total cost is

$$\mathbf{p}(\mathbf{W}^1) \cdot \mathbf{c} + \mathrm{dist}_\delta(\mathbf{p}(\mathbf{W}), \mathbf{p}(\mathbf{W}^1)) = \epsilon p_i(\mathbf{W}^1) + \sum_{j \in [n] \setminus \{i\}} (p_j(\mathbf{W}^1) - p_j(\mathbf{W}))\delta(i,j). \tag{1}$$

In the present work, we will consider designing algorithms with $\mathbf{p}(\mathbf{W})$ which are both continuous and differentiable. With this in mind, we can take (1) a step further and let $\epsilon \to 0$ to get the instantaneous increase in cost to the algorithm as we add cost to state i. Using continuity, we see that $\mathbf{W}^1 \to \mathbf{W}$ as $\epsilon \to 0$, which gives that the instantaneous cost at $\mathbf{W}$ in the direction of $\mathbf{e}_i$ as

$$p_i(\mathbf{W}) + \sum_{j \in [n] \setminus \{i\}} \frac{\partial p_j(\mathbf{W})}{\partial W_i} \delta(i,j)$$

Ultimately, we need to bound the total cost of the algorithm on any sequence. The typical way to achieve this is with a potential function that maintains an upper bound on the worst case sequence of cost vectors that results in the current $\mathbf{W}$. There is a natural "best" potential function $\Phi^*_{\mathbf{p}}(w)$ for a given algorithm $\mathbf{p}$, which we now construct.

For any measurable function $I : \mathbb{R}_+ \to [n]$, we can define a continuous path through the space of work vectors by $\mathbf{W}_I(s) = \int_0^s \mathbf{e}_{I(\alpha)} d\alpha$. This is exactly the continuous version of Lemma 2. The function $I(s)$ specifies which coordinate of $\mathbf{W}_I(s)$ is increasing at time s. Let $\rho(\mathbf{W})$ be the set of all functions I which induce paths starting at $\mathbf{0}$ that lead to $\mathbf{W}$. We now construct a potential function,

$$\Phi^*_{\mathbf{p}}(\mathbf{W}) = \sup_{I \in \rho(\mathbf{W})} \int_0^{T: \mathbf{W}_I(T) = \mathbf{W}} \left(p_{I(s)}(\mathbf{W}_I(s)) + \sum_{j \neq I(s)} \frac{\partial p_j(\mathbf{W}_I(s))}{\partial \mathbf{W}_I(s)} \delta(i,j) \right) ds.$$

This potential function measures precisely the worst case cost of arriving at a work vector $\mathbf{W}$.

Lemma 3. *For any sequence of reasonable elementary vectors $\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^T$ with $\mathbf{W} = \sum_t \mathbf{c}^t$, the cost to algorithm $\mathbf{p}$ is no more than $\Phi^*_{\mathbf{p}}(\mathbf{W})$. Furthermore, $\Phi^*_{\mathbf{p}}(\mathbf{W})$ is the supremal cost over all possible cost sequences $\{\mathbf{c}^t\}$ that lead to $\mathbf{W}$.*

Proof. This fact is straightforward and we sketch the proof. For $\mathbf{c} = \epsilon \mathbf{e}_i$ and any $\mathbf{W}$, the cost to the algorithm is stated in Equation (1). On the other hand, if we apply the cost to state i in a continuous fashion, then the cost is

$$\int_0^\epsilon \left(p_i(\mathbf{W} + s\mathbf{e}_i) + \sum_{j \neq i} \frac{\partial p_j(\mathbf{W} + s\mathbf{e}_i)}{\partial W_i} \delta(i,j) \right) ds.$$

By the conservative property, this is clearly an upper bound on Equation (1). In addition, for any sequence of $\mathbf{c}^t$'s, we can construct an associated smooth path to $\mathbf{W}$ in the same way. But $\Phi^*_{\mathbf{p}}(\mathbf{W})$ was defined as the supremum cost over such paths. Thus, both the lower and upper bound follow.

Once we have $\Phi^*_{\mathbf{p}}$, the competitive ratio of $\mathbf{p}$ has the following characterization.

Lemma 4. *The competitive ratio of algorithm* $\mathbf{p}$ *is the infimal value* C *such that* $\Phi^*_{\mathbf{p}}(\mathbf{W}) - CW_{\min}$ *is bounded away from* $+\infty$ *for all* $\mathbf{W}$.

Certain work-based algorithms, which we will call *shift-invariant* algorithms, satisfy $\mathbf{p}(\mathbf{W}) = \mathbf{p}(\mathbf{W} + c\mathbf{1})$ for any $\mathbf{W}$ and any c.

Lemma 5. *The competitive ratio of a shift-invariant algorithm is* $\mathbf{1} \cdot \nabla \Phi^*_{\mathbf{p}}(\mathbf{W})$ *for any* $\mathbf{W}$.

Finding the optimal $\Phi^*_{\mathbf{p}}$ for the algorithm $\mathbf{p}$ may be difficult. To prove an upper bound on the competitive ratio, however, we need only construct a *valid* Φ. Precisely, define $\Phi(\mathbf{W})$ to be valid with respect to the algorithm $\mathbf{p}$ if, for all $\mathbf{W}$ and all i, we have

$$\frac{\partial \Phi(\mathbf{W})}{\partial W_i} \geq p_i(\mathbf{W}) + \sum_{j \neq i} \frac{\partial p_j(\mathbf{W})}{\partial W_i} \delta(i,j)$$

Lemma 6. *Given any* $\mathbf{p}$ *and any valid potential* Φ*,* C *is an upper bound on the competitive ratio if* $\Phi(\mathbf{W}) - CW_{\min}$ *is bounded away from* $+\infty$.

In the following Section, we show how to design algorithms and construct potentials for the case of uniform metrics using regularization techniques.

3 Work-Based Algorithms via Regularization

We begin this Section by providing a general tool for the construction of work-based MTS algorithms. We present a *regularization* approach, very common in the adversarial online learning community, which we modify to ensure the required conditions for the MTS setting. We then present two algorithms for the uniform metric that arise from this framework, with associated potential functions, and we prove a bound on the competitive ratio of each. We finish with a discussion on how to extend this approach to general metric spaces.

3.1 Regularization and Achieving Reasonableness

We now turn our attention to the problem of designing competitive work-based algorithms for the case when δ is the uniform metric. The uniform metric is such that all states are the same distance from each other–that is, we assume without loss of generality $\delta(i, j) = 1$ whenever $i \neq j$ and 0 otherwise.

To obtain a competitive work-based algorithm, we need to find a function $\mathbf{p}$ and construct an associated potential function Φ with the following properties:

– (Conservativeness) We require that $\frac{\partial p_j(\mathbf{W})}{\partial W_i} \geq 0$ for any $\mathbf{W}$ and $\forall j \neq i$
– (Reasonableness) The probability $p_i(\mathbf{W})$ must vanish whenever i is a supported state for $\mathbf{W}$, i.e. when $W_i = W_j + \delta(i, j)$ for some j
– (Valid Potential) For any $\mathbf{W}, i$, the potential Φ must satisfy $\frac{\partial \Phi(\mathbf{W})}{\partial W_i} \geq p_i(\mathbf{W}) - \frac{\partial p_i(\mathbf{W})}{\partial W_i}$

Notice that the term $-\frac{\partial p_i(\mathbf{W})}{\partial W_i}$ has replaced $\sum_{j\neq i} \frac{\partial p_j(\mathbf{W})}{\partial W_i}\delta(i, j)$ in the last expression. These two quantities are equal when δ is the uniform metric, precisely because for any j we have $\sum_i \frac{\partial p_i(\mathbf{W})}{\partial W_j} = 0$ since $\sum_i p_i(\mathbf{W}) = 1$.

In order to obtain an algorithm with a low competitive ratio, we must construct a slowly-changing $\mathbf{p}(\mathbf{W})$ and a valid potential $\Phi(\mathbf{W})$ that controls the motion of $\mathbf{p}(\mathbf{W})$ as $\mathbf{W}$ varies in each direction. In other words, we would like to enforce a level of stability in $\mathbf{p}(\mathbf{W})$. Stability is a central concept within both the batch-learning and the adversarial online-learning literature. The most common and thoroughly analyzed approach is to employ *regularization*. To describe this approach, let us return our attention to the experts setting discussed in Section 2.1. Recall that, at time t, a distribution $\mathbf{p}^t \in \Delta_n$ is to be chosen with knowledge of $\boldsymbol{l}^1, \ldots, \boldsymbol{l}^{t-1}$. This can be achieved by solving the following regularized objective,

$$\mathbf{p}^t = \underset{\mathbf{p}\in\Delta_n}{\operatorname{argmin}} \left(R(\mathbf{p}) + \lambda \sum_{s=1}^{t-1} \mathbf{p} \cdot \boldsymbol{l}^s\right) \tag{2}$$

where generally the "regularizer" R is selected as some smooth convex function and λ is a learning parameter. Exactly how to select the correct regularizer is a major area of research, but for the experts setting the most common is the negative of the *entropy function*, $R(\mathbf{p}) := \sum_{i\in[n]} p_i \log p_i$. This choice leads to the well-known exponential weights:

$$p_i^t = \frac{\exp\left(-\lambda \sum_{s=1}^{t-1} l_i^s\right)}{\sum_j \exp\left(-\lambda \sum_{s=1}^{t-1} l_j^s\right)} \tag{3}$$

Regularization in online learning appears in the literature at least as early as [19] and [20], and more modern analyses can be found in [21] and [22].

In this paper, we use the regularization framework to produce an algorithm $\mathbf{p}(\mathbf{W})$. It is tempting to suggest solving the equivalent objective of Equation (2), where we treat $\mathbf{W}$ as the cumulative costs; this leads to setting

$$\mathbf{p}(\mathbf{W}) = \underset{\mathbf{p}}{\operatorname{argmin}} \left(R(\mathbf{p}) + \lambda \mathbf{p} \cdot \mathbf{W}\right). \tag{4}$$

This approach can indeed guarantee stability with the correct R, and it's easy to check that the objective induces a conservative algorithm. Unfortunately, it does *not* enforce the reasonableness property that we require. (It has been shown that an unreasonable work-based algorithm must admit an unbounded competitive ratio [17].)

The question we are thus left with is, how can we adjust the objective to maintain stability and ensure reasonableness? Recall, when δ is the uniform metric, the reasonableness property requires that $p_i(\mathbf{W}) \to 0$ whenever $1 + W_j - W_i$ approaches 0 for any j, or equivalently when $1 + W_{\min} - W_i \to 0$. To guarantee this behavior, we propose replacing the term $\mathbf{p} \cdot \mathbf{W}$ in Equation (4) with $\sum_i p_i f_i(\mathbf{W}, \lambda)$ where the function $f_i(\mathbf{W}, \lambda)$ will be a *Lipschitz penalty*: for any metric δ on $[n]$ and any $1-$Lipschitz vector $\mathbf{W}$ with respect to δ, we say that $f_i(\mathbf{W}, \lambda)$ is a Lipschitz penalty function if $f_i(\mathbf{W}, \lambda) \to \infty$ as $\min_j \big(W_j - W_i + \delta(i,j)\big) \to 0$. λ is a learning parameter that may be tuned. Hence, we propose the following method to find $\mathbf{p}(\mathbf{W})$:

$$\mathbf{p}(\mathbf{W}) = \underset{\mathbf{p}}{\operatorname{argmin}} \, (R(\mathbf{p}) + \sum_i p_i f_i(\mathbf{W}, \lambda)). \tag{5}$$

For both algorithms in the following Section, we employ the entropy function for our regularizer $R(\mathbf{p})$.

3.2 Two Resulting Algorithms for the Uniform Metric

We will consider the following two Lipschitz penalty functions, and analyze the resulting algorithms:

$$\begin{array}{ll} \text{(Alg 1)} & f_i(\mathbf{W}, \lambda) = -\lambda \log(1 + W_{\min} - W_i) \\ \text{(Alg 2)} & f_i(\mathbf{W}, \lambda) = -\log(e^{\lambda(1+W_{\min}-W_i)} - 1) \end{array}$$

The analysis of both algorithms proceeds by solving the regularization function to find p_i as a function of $\mathbf{W}$ and then using the potential function technique of Section 2.2 to bound the switching and servicing costs regardless of which state receives cost. For both, we separate the analysis into two cases: when increasing W_i causes $W_{\min} = \min_j W_j$ to increase, and when increasing W_i does not affect $W_{\min}$.

Algorithm 1

```
1: Input: $\mathbf{W}, \lambda$
2: Set
       $p_i = \frac{(1 + W_{\min} - W_i)^\lambda}{\sum_{j=1}^n (1 + W_{\min} - W_j)^\lambda}$
```

Theorem 1. *Algorithm 1 results from regularizing with* $f_i(\mathbf{W}, \lambda) = -\lambda \log(1+W_{\min}-W_i)$ *with* $\lambda = \log n$ *and has a competitive ratio of* $e \log n + 1$ *for the uniform metric.*

Proof. The algorithm that results from regularizing with $f_i(\mathbf{W}, \lambda) = -\lambda \log(1 + W_{\min} - W_i)$ is:

$$p_i = \frac{(1 + W_{\min} - W_i)^\lambda}{\sum_{j=1}^n (1 + W_{\min} - W_j)^\lambda}$$

We will show that each of the components of the cost of the algorithm is bounded by a multiple of the following potential function:

$$\Phi(\mathbf{W}) = cW_{\min} - \log \sum_{i=1}^n (1 + W_{\min} - W_i)^\lambda$$

The parameters will be set so that $c = e(\log n - 1) + 1$ and $\lambda = \log n$. We will show that these have been tuned optimally.

As discussed in the beginning of this section, we must show that $p_i - \frac{\partial p_i}{\partial W_i} \le \frac{\partial \Phi}{\partial W_i}$ for all i. We will vary from that slightly and show that when $i \neq \min$, $p_i - \frac{\partial p_i}{\partial W_i} \le (1 + \frac{1}{\lambda}) \frac{\partial \Phi}{\partial W_i}$ and if $i = \min$ then $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \le \frac{\partial \Phi}{\partial W_{\min}}$. Combining these facts, the competitive ratio will be upper bounded by $(1 + \frac{1}{\lambda})c$.

First, we will show that if $i \neq \min$, $p_i - \frac{\partial p_i}{\partial W_i} \le (1 + \frac{1}{\lambda}) \frac{\partial \Phi}{\partial W_i}$.

$$\begin{aligned} p_i - \frac{\partial p_i}{\partial W_i} &= \frac{(1 + W_{\min} - W_i)^\lambda}{\sum_j (1 + W_{\min} - W_j)^\lambda} + \frac{\lambda (W_{\min} - W_i + 1)^{\lambda - 1}}{(\sum_j (1 + W_{\min} - W_j)^\lambda)^2} \\ &\le \frac{(1 + W_{\min} - W_i)^{\lambda - 1} (\lambda + 1 + W_{\min} - W_i)}{\sum_j (1 + W_{\min} - W_j)^\lambda} \\ &\le \frac{(\lambda + 1)(W_{\min} - W_i + 1)^{\lambda - 1}}{\sum_j (1 + W_{\min} - W_j)^\lambda} = \frac{\lambda + 1}{\lambda} \frac{\partial \Phi}{\partial W_i} \end{aligned}$$

Next, we consider $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \le \frac{\partial \Phi}{\partial W_{\min}}$. Notice that $p_{\min} = \frac{1}{Z}$ where $Z = \sum_j (1 + W_{\min} - W_j)^\lambda$. We have

$$p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} = \frac{1}{Z} + \frac{1}{Z^2} \frac{\partial Z}{\partial W_{\min}} \le 1 + \frac{1}{Z^2} \frac{\partial Z}{\partial W_{\min}}$$

In addition, we see that

$$\frac{\partial \Phi}{\partial W_{\min}} = c - \frac{1}{Z} \frac{\partial Z}{\partial W_{\min}}$$

In order to show that $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \le \frac{\partial \Phi}{\partial W_{\min}}$, using the above two statements it is equivalent to show that

$$\frac{1}{Z} \frac{\partial Z}{\partial W_{\min}} + \frac{1}{Z^2} \frac{\partial Z}{\partial W_{\min}} \le c - 1$$

We now show this fact. First, let $\alpha_j := 1 + W_{\min} - W_j$. Now we need to maximize

$$\left(1 + \frac{1}{1 + \sum_{j \neq \min} \alpha_j^\lambda}\right) \frac{\lambda \sum_{j \neq \min} \alpha_j^{\lambda - 1}}{1 + \sum_{j \neq \min} \alpha_j^\lambda}$$

This expression is maximized when $\alpha_j = (\frac{\lambda-1}{n-1})^{1/\lambda}$ and attains a maximum value of $\frac{\lambda+1}{\lambda}(\lambda-1)(n-1)^{1/\lambda}(\lambda-1)^{-1/\lambda}$. This can be seen by first noting that it is maximized when all α_j are some value α and then taking the derivative with respect to α and setting it equal to 0.

We note that as $\lambda \to \infty$, $(\lambda-1)^{-1/\lambda} \to 1$, as does $\frac{\lambda+1}{\lambda}$. Thus, we only concern ourselves with the limit of $(n-1)^{1/\lambda}$. Let this quantity be L. By L'Hopital's rule:

$$\lim_{n\to\infty} \log L = \lim_{n\to\infty} \frac{\log(n-1)}{\lambda} = \lim_{n\to\infty} \frac{\frac{1}{n-1}}{\frac{d\lambda}{dn}}$$

If we let $\lambda = \log n$ then we have $\frac{1}{(n-1)}/\frac{1}{n} \to 1$. Thus, $L = e$ and $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \leq \frac{\partial \Phi}{\partial W_{\min}}$ if $c - 1 > (\lambda-1)(n-1)^{1/\lambda} = e(\log n - 1)$. Therefore, $c = e(\log n - 1) + 1$.

Finally, we note that we have both requirements, $p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} \leq \frac{\partial \Phi'}{\partial W_{\min}}$ and $p_i - \frac{\partial p_i}{\partial W_i} \leq \frac{\partial \Phi'}{\partial W_i}$ for $\Phi' = (1+\frac{1}{\lambda})\Phi$. Therefore, the total cost of this algorithm is bounded by $(1+\frac{1}{\lambda})c\text{OPT} = (1+\frac{1}{\log n})(e(\log n - 1)+1)\text{OPT} \leq (e\log n + e + 1)\text{OPT}$.

The previous algorithm demonstrates our analysis technique for a very simple and natural Lipschitz-penalty function. However, it has a somewhat unsatisfying competitive ratio of $e\log n$. Even the very simple Marking algorithm has a better competitive ratio of $2H_n$. Next, we will show that a different Lipschitz penalty function, $f_i(\mathbf{W}, \lambda) = \log(\exp(\lambda(1 + W_{\min} - W_i)) - 1)$, produces an algorithm that achieves the current best competitive ratio for the uniform MTS problem.

Algorithm 2

```
1: Input: $\mathbf{W}, \lambda$
2: Set
        $$p_i = \frac{e^{\lambda(1+W_{\min}-W_i)} - 1}{\sum_j \left(e^{\lambda(1+W_{\min}-W_j+1)} - 1\right)}$$
```

Theorem 2. *Algorithm 2 results from the Lipschitz penalty* $f_i(\mathbf{W}, \lambda) = -\log(\exp(\lambda(1+W_{\min} - W_i)) - 1)$ *with* $\lambda = \log n + 2\log\log n$ *and has a competitive ratio of* $\log n + O(\log\log n)$ *for the uniform metric.*

Proof. Solving the regularization problem when $f_i(\mathbf{W}, \lambda) = \log(\exp(\lambda(1 + W_{\min} - W_i)) - 1)$ results in

$$p_i = \frac{e^{\lambda(1+W_{\min}-W_i)} - 1}{\sum_j e^{\lambda(W_{\min}-W_j+1)} - 1}$$

We will show that the *switching and servicing* costs are bounded by the following potential function:

$$\Phi(\mathbf{W}) = cW_{\min} - \frac{1+\lambda}{\lambda}\log\sum_{i=1}^{n}(e^{\lambda(1+W_{\min}-W_i)} - 1).$$

This analysis requires tuning the parameter λ, which we will do at the end.

In the same vein as the previous proof, we will show that $p_i - \frac{\partial p_i}{\partial W_i} \le \frac{\partial \Phi}{\partial W_i}$. We will break this up into two steps, one where $i \ne \min$ and when $i = \min$.

Let us consider the case when $i \ne \min$. Let $Z = \sum_j (e^{\lambda(1+W_{\min}-W_j)} - 1)$, the normalization term of the above distribution. For any $i \ne \min$, we see that

$$\begin{aligned} p_i - \frac{\partial p_i}{\partial W_i} &= p_i + \frac{\lambda e^{\lambda(1+W_{\min}-W_i)}}{Z} + \frac{1}{Z^2}\frac{\partial Z}{\partial W_i}(e^{\lambda(1+W_{\min}-W_i)} - 1) + \frac{\lambda}{Z} - \frac{\lambda}{Z} \\ &= p_i + \frac{\lambda(e^{\lambda(1+W_{\min}-W_i)} - 1)}{Z} + \frac{\lambda}{Z} + p_i \frac{1}{Z}\frac{\partial Z}{\partial W_i} \\ &= (1 + \lambda + \frac{1}{Z}\frac{\partial Z}{\partial W_i})p_i + \frac{\lambda}{Z} \quad \le \quad (1+\lambda)p_i + \frac{\lambda}{Z} \end{aligned}$$

Notice that the final inequality follows since $\frac{\partial Z}{\partial W_i} \le 0$.

Then, we consider $\frac{\partial \Phi}{\partial W_i}$.

$$\begin{aligned} \frac{\partial \Phi}{\partial W_i} &= \frac{\lambda+1}{\lambda}\frac{1}{Z}\frac{\partial Z}{\partial W_i} \\ &= \frac{\lambda+1}{\lambda}\frac{1}{Z}(\lambda e^{\lambda(W_{\min}-W_i+1)}) \\ &= \frac{1+\lambda}{Z}e^{\lambda(W_{\min}-W_i+1)} + \frac{1+\lambda}{Z} - \frac{1+\lambda}{Z} \\ &= (1+\lambda)(p_i + 1/Z) \end{aligned}$$

$p_i - \frac{\partial p_i}{\partial W_i} \le \frac{\partial \Phi}{\partial W_i}$ follows immediately.

Now let $i = \min$. Notice that $p_{\min} = \frac{e^{\lambda}-1}{Z}$, so we have

$$p_{\min} - \frac{\partial p_{\min}}{\partial W_{\min}} = p_{\min} + (e^{\lambda} - 1)\frac{1}{Z^2}\frac{\partial Z}{\partial W_{\min}} = p_{\min}\left(1 + \frac{1}{Z}\frac{\partial Z}{\partial W_{\min}}\right)$$

Furthermore,

$$\frac{\partial \Phi}{\partial W_{\min}} = c - \frac{1+\lambda}{\lambda}\frac{1}{Z}\frac{\partial Z}{\partial W_{\min}}$$

We compute

$$\begin{aligned} \frac{1}{Z}\frac{\partial Z}{\partial W_{\min}} = \frac{\lambda}{Z}\sum_{j\ne\min} e^{\lambda(W_{\min}-W_j+1)} &= \frac{\lambda}{Z}\sum_{j\ne\min}(e^{\lambda(W_{\min}-W_j+1)} - 1) + \lambda\frac{n-1}{Z} \\ &= \lambda\left(1 - p_{\min} + \frac{n-1}{Z}\right) \end{aligned}$$

Putting the last three statements together, we can restate $p_{\min} - \frac{\partial p_{\min}}{\partial W_i} \le \frac{\partial \Phi}{\partial W_{\min}}$ as

$$\begin{aligned} & p_{\min}\left(1 + \lambda\left(1 - p_{\min} + \frac{n-1}{Z}\right)\right) \le c - (1+\lambda)\left(1 - p_{\min} + \frac{n-1}{Z}\right) \\ \iff \quad & \frac{n-1}{Z}(1 + \lambda + \lambda p_{\min}) + 1 + \lambda(1 - p_{\min}^2) \le c \end{aligned}$$

Noting that $Z \geq e^\lambda - 1$ and $\lambda p_{\min} \leq \lambda$, it is equivalent to show that $\frac{(2\lambda+1)n}{e^\lambda - 1} + 1 + \lambda \leq c$. Setting $\lambda = \log n + 2 \log \log n$ gives that the first term is $o(1)$, and we can then set $c = \lambda + 1 + o(1)$. Thus the competitive ratio of this algorithm is $\log n + O(\log \log n)$, the best achieved thus far.

3.3 Extending to general metrics

While it may appear that the entropy function was chosen out-of-the-blue as a regularization, it has been well established that entropy is ideal when we want to control the L1-stability of our hypothesis and, for the uniform metric, $\text{dist}_\delta(\mathbf{p}^1, \mathbf{p}^2) = \|\mathbf{p}^1 - \mathbf{p}^2\|_1/2$. But notice that the algorithmic template we propose in (5) does not rely on the uniform metric, and can be posed in general. Trying to immediately extend our approach to general metrics unfortunately does not lead to an algorithm with an $O(\log n)$-competitive ratio, the major goal since the randomized MTS problem was introduced nearly 20 years ago.

For other metrics, it is clear that entropy is not at all the correct regularizer. Instead, what is needed is a regularization function that controls the stability of $\mathbf{p}$ with respect to the norm induced by the Earth Mover Distance $\text{dist}_\delta(\cdot, \cdot)$. It would be of particular interest if such a function existed and could be constructed.

Conjecture 2. For any metric δ on $[n]$, there is some regularization function $R(\cdot)$ such that the algorithm resulting from Equation (5) is $O(\log n)$-competitive.

The choice of Lipschitz penalty $f_i(\cdot, \cdot)$ may need to be tuned as well.

4 Conclusions and Open Problems

We have introduced a framework for developing and analyzing algorithms for the metrical task system problem. This framework presupposes that an optimal algorithm that is a function of the work vector exists and we conjecture that this is true. Given this framework we are able to use the popular entropy regularization approach to develop state-of-the-art algorithms. We believe this system gives good insight into how to develop algorithms for the general metric case.

Our work leaves open several important questions. The most obvious are the answers to our conjectures - is it actually true that assuming that the algorithm will be work vector based does not preclude optimality? All of the current algorithms for general metrics rely on embedding the metric into a hierarchical search tree and then using MTS algorithms for this metric space and none are known to be based on the work vector.

There is also an open question with regards to the regularization approach. It is known that entropy is a good regularizer when the movement cost is measured by the L1 norm. However, in the general case, the switching costs are measured according to the Earth Mover Distance. What is the correct regularization function for general distance metrics? We believe that an algorithm for the general metric with even a poly$\log n$ bound on the competitive ratio that is worse than the current results achieved by metric embedding would be interesting due to it's potential relative simplicity.

Acknowledgements We would like to thank Sylvain Arlot for his thoughtful comments and discussion and the anonymous reviewers.

References

1. Borodin, A., Linial, N., Saks, M.: An optimal on-line algorithm for metrical task system. JACM: Journal of the ACM **39(4)** (1992) 745–763
2. Manasse, M., McGeoch, L., Sleator, D.: Competitive algorithms for server problems. J. Algorithms **11** (1990) 208–230
3. Freund, Schapire: A decision-theoretic generalization of on-line learning and an application to boosting. JCSS: Journal of Computer and System Sciences **55** (1997)
4. Schafer, G., Sivadasan, N.: Topology matters: Smoothed competitiveness of metrical task systems. TCS: Theoretical Computer Science **341** (2005)
5. Irani, S., Seiden, S.: Randomized algorithms for metrical task systems. Theoretical Computer Science **194** (1998)
6. Bartal, Y., Blum, A., Burch, C., Tomkins, A.: A polylog(n)-competitive algorithm for metrical task systems. In: Symposium on Theory Of Computing (STOC). (1997) 711–719
7. Bartal, Y.: On approximating arbitrary metrics by tree metrics. In: Symposium Theory Of Computing (STOC). (1998) 161–168
8. Fakcharoenphol, J., Rao, S., Talwar, K.: A tight bound on approximating arbitrary metrics by tree metrics. In: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing (STOC). (2003) 448–455
9. Fiat, A., Mendel, M.: Better algorithms for unfair metrical task systems and applications. SIAM Journal on Computing **32** (2003)
10. Bansal, N., Buchbinder, N., Naor, S.: Metrical task systems and the k-server problem on hsts. In: Manuscript. (2009)
11. Bartal, Y., Bollobás, B., Mendel, M.: A ramsey-type theorem for metric spaces and its applications for metrical task systems and related problems. In: IEEE Symposium on Foundations of Computer Science (FOCS). (2001) 396–405
12. Blum, A., Karloff, H., Rabani, Y., Saks, M.: A decomposition theorem and lower bounds for randomized server problems. SIAM Journal on Computing **30** (2000) 1624–1661
13. Bansal, N., Buchbinder, N., Naor, J.: A primal-dual randomized algorithm for weighted paging. In: IEEE Symposium on Foundations of Computer Science (FOCS). (2007)
14. Bein, W., Larmore, L., Noga, J.: Uniform metrical task systems with a limited number of states. IPL: Information Processing Letters **104** (2007)
15. Bansal, N., Buchbinder, N., Naor, S.: Towards the randomized k-server conjecture: A primal-dual approach. In: ACM-SIAM Symposium on Discrete Algorithms (SODA). (2010)
16. Buchbinder, N., Naor, S.: The design of competitive online algorithms via a primal-dual approach. Foundations and Trends in Theoretical Computer Science **3** (2009) 93–263
17. Blum, A., Burch, C.: On-line learning and the metrical task system problem. Machine Learning **39** (2000) 35–58
18. Herbster, M., Warmuth, M.K.: Tracking the best expert. Machine Learning **32** (1998) 151
19. Kivinen, J., Warmuth, M.: Exponentiated gradient versus gradient descent for linear predictors. Information and Computation (1997)
20. Gordon, G.: Regret bounds for prediction problems. In: Proceedings of the twelfth annual conference on Computational learning theory, ACM New York, NY, USA (1999) 29–40
21. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, New York, NY, USA (2006)
22. Rakhlin, A.: Lecture Notes on Online Learning DRAFT. (2009)