Better Algorithms for Unfair Metrical Task Systems and Applications

Amos Fiat*
Dept. of Computer Science
Tel-Aviv University
fiat@math.tau.ac.il

Manor Mendel†
Dept. of Computer Science
Tel-Aviv University
mendel@math.tau.ac.il

ABSTRACT

Unfair metrical task systems are a generalization of online metrical task systems. In this paper we introduce new techniques to combine algorithms for unfair metrical task systems and apply these techniques to obtain the following results:

1. Better randomized algorithms for unfair metrical task systems on the uniform metric space.
2. Better randomized algorithms for metrical task systems on general metric spaces, $O(\log^2 n(\log\log n)^2)$ competitive, improving on the best previous result of $O(\log^5 n \log\log n)$.
3. A tight randomized competitive ratio for the k-weighted caching problem on $k+1$ points, $O(\log k)$, improving on the best previous result of $O(\log^2 k)$.

1. INTRODUCTION

Metrical task systems, introduced by Borodin, Linial, and Saks [12], can be described as follows: A server in some internal configuration receives *tasks* that have a service cost associated with each of the internal configurations. The server may switch configurations, paying a cost given by a metric space defined on the configuration space, and then pays the service cost associated with the new configuration.

Metrical task systems have been the subject of a great deal of study. A large part of the research into online algorithms can be viewed as a study of some particular metrical task system. In modeling some of these problems as metrical task systems, the set of permissible tasks is constrained to fit the particulars of the problem. In this paper we consider the original definition of metrical task systems where the set of tasks can be arbitrary.

A deterministic algorithm for any n-configuration metrical task system with a competitive ratio of $2n-1$ was already given in the original paper [12], along with a matching lower bound for any metric space. However, randomized algorithms for metrical tasks systems in the oblivious adversary model are not fully understood. The first randomized algorithm for general metrical task systems better than the deterministic was presented by Irani and Seiden [16], $\approx 1.58n$ competitive algorithm. Bartal, Blum, Burch, and Tomkins [5] gave the first sublinear randomized competitive ratio, $O(\log^6 n/\log\log n)$. Bartal [3] improves this to $O(\log^5 n \log\log n)$. In this paper we obtain an $O(\log^2 n (\log\log n)^2)$ upper bound on the competitive ratio.

A lower bound on the randomized competitive ratio for an arbitrary metric space of $\Omega(\log\log n)$ was given by Karloff, Rabani, and Ravid [18]. The best lower bound currently known is $\Omega(\sqrt{\log n/\log\log n})$ due to Blum, Karloff, Rabani and Saks [11].

The basic paging problem is the online problem of deciding what page to evict upon a page fault. The performance measure is the number of page faults. This problem has tight deterministic and randomized bounds on the competitive ratio of k and $\approx \ln k$, where k is the number of page slots in memory [22; 15; 20; 1].

Weighted caching is the paging problem when all pages sizes are the same but there is a different cost to fetch different pages. Deterministically, a competitive ratio of k [13; 23; 24] is achievable, with a matching deterministic lower bound following from the k-server bound in [19]. No randomized algorithm is known to have a competitive ratio better than the deterministic ratio of k. However, in some special cases progress has been made. Irani [private communication] has shown an $O(\log k)$ competitive algorithm when page fetch costs are one of two possible values. Blum, Furst, and Tomkins [9] have given an $O(\log^2 k)$ competitive algorithm for arbitrary page costs, when the total number of pages is $k+1$, they also give a lower bound of $\Omega(\log k)$. In this paper we obtain an $O(\log k)$ competitive algorithm for the weighted caching problem on $k+1$ pages. This is tight up to a constant factor.

To obtain the results above we make use of *unfair metrical task systems* [21; 5], we apply algorithms for unfair metrical task systems on the uniform metric space so as to obtain algorithms for *hierarchically well separated trees* (HST) [2], this technique is due to [2; 5]. In [2; 3] it is shown how to

*Partly supported by United States Israel Bi-national Science Foundation Grant 96-00247/1.

†Partly supported by United States Israel Bi-national Science Foundation Grant 96-00247/1.

STOC 2000 Portland Oregon USA

reduce problems on general metric spaces to HST metrics.

1.1 Contributions of this Paper

We introduce a general notation and technique for combining algorithms for unfair metrical task systems. This technique is an improvement on the previous methods [11; 21; 5] and we believe that it is of independent interest beyond the applications we have given in this paper. We also believe that using some notation similar to ours is imperative to avoid possible confusion in combining algorithms from different spaces.

Using this technique, we obtain randomized algorithms for unfair metrical task systems on the uniform metric space that are better than the algorithm of [5]. Using the algorithm for unfair metrical task systems on uniform metric space and the new method for combining algorithms, we obtain $O(\log^2 n(\log\log n)^2)$-competitive randomized algorithm for metrical task systems on any metric space, improving on the best previous result of $O(\log^5 n \log\log n)$. Using the same techniques in a slightly different manner, we obtain a tight randomized competitive ratio for the k-weighted caching problem on $k+1$ points, $O(\log k)$, improving on the best previous result of $O(\log^2 k)$.

2. PRELIMINARIES

Unfair metrical task systems (UMTSs) [21; 5] are a generalization of metrical task systems [12], the terminology is that of [21]. A UMTS $U = (s, M, r_1, \ldots, r_b)$ consists of a *distance ratio* $s \in \mathbb{R}^+$, a metric space M on b configurations, $v_1, \ldots, v_b$ with a distance matrix d_M, and a sequence of *cost ratios* $r_1, r_2, \ldots, r_b \in \mathbb{R}^+$.

Given some UMTS U, the associated online problem is defined as follows. An online algorithm A occupies some configuration $v_i \in M$. When receiving some sequence of tasks, a task is a vector $(c_1, c_2, \ldots, c_b)$, algorithm A can choose a new configuration $v_j \in M$. The cost for A associated with servicing the task is $s \cdot \mathrm{d}_M(v_i, v_j) + r_j c_j$. The cost for A associated with servicing a sequence of tasks σ is simply the sum of costs for servicing the individual tasks of the sequence consecutively and is denoted by $\mathrm{cost}_A(\sigma)$. An online algorithm makes its decisions based only upon tasks seen so far.

An off-line player is defined that services the same sequence of tasks over U. The cost of an off-line player, if it were to do exactly as above, would be $\mathrm{d}_M(v_i, v_j)+c_j$. Thus, the concept of *unfairness*, the costs for doing the same are different.

Given a sequence of tasks σ we define the *work function* [14] at v, $w_{\sigma,U}(v)$, to be the minimal cost, for any off-line player, to start at the initial configuration in U, deal with all tasks in σ, and end up in configuration v. We omit the use of the subscript U if the UMTS is clear from the context. We note that for all $u, v \in M$, $w_\sigma(u) - w_\sigma(v) \leq \mathrm{d}_M(u, v)$. If $w_\sigma(u) = w_\sigma(v) + \mathrm{d}_M(u, v)$, u is said to be *supported* by v. We say that $u \in M$ is *supported* if there exists some $v \in M$ such that u is supported by v.

We define $\mathrm{cost}_{\mathrm{OPT}}(\sigma)$ to be $\min_v w_\sigma(v)$. This is simply the minimal cost, for any off-line player, to start at the initial configuration and process σ. Define a *competition vector* $\alpha = (\alpha(v_1), \alpha(v_2), \ldots, \alpha(v_b))$ to be a real valued vector where $\sum_i \alpha(v_i) = 1$ and $\alpha(v_i) \geq 0$ for all $1 \leq i \leq b$. We define the α-optimal-cost of a sequence of tasks σ to be $\mathrm{cost}_{\alpha\text{-OPT}}(\sigma) = < \alpha, w_\sigma >$, where $< x, y >$ is the inner product of x and y. We observe that $\mathrm{cost}_{\alpha\text{-OPT}}(\sigma) \leq \mathrm{cost}_{\mathrm{OPT}}(\sigma) + \Delta(M)$, where $\Delta(M) = \max_{u,v \in M} \mathrm{d}_M(u, v)$ is the diameter of the M.

A randomized online algorithm A for unfair metrical task is an online algorithm that decides upon the next configuration using a random process. The expected cost of a randomized algorithm A on a sequence σ is denoted by $E[\mathrm{cost}_A(\sigma)]$. A randomized online algorithm is called r competitive against an oblivious adversary [22; 17; 6] if exists c such that for all task sequences σ, $E[\mathrm{cost}_A(\sigma)] \leq r\ \mathrm{cost}_{OPT}(\sigma)+c$. Note that an algorithm is r competitive if and only if exists c' such that for all task sequences σ, $E[\mathrm{cost}_A(\sigma)] \leq r\ \mathrm{cost}_{\alpha\text{-OPT}}(\sigma) + c'$.

Given a randomized online algorithm A for an UMTS U with configurations $v_1, \ldots, v_b$, and a sequence of tasks σ, we define $p_{\sigma,A}$ to be the vector of probabilities $(p_{\sigma,A}(v_1), \ldots, p_{\sigma,A}(v_b))$ where $p_{\sigma,A}(v_i)$ is the probability that A is in configuration v_i after serving the request sequence σ. We drop the subscript A if the algorithm is clear from the context.

Let $x \circ y$ denote the concatenation of sequences x and y. Let U be a UMTS over the metric space M with distance ratio s. Given two successive probability distributions on the configurations of U, p_σ and $p_{\sigma\circ e}$, where e is the next task, we define the set of transfer matrices from p_σ to $p_{\sigma\circ e}$, denoted $T(p_\sigma, p_{\sigma\circ e})$, as the set of all matrices $T = (t_{ij})_{1\leq i,j\leq b}$ with non negative real entries, where

$$\sum_{j=1}^{b} t_{ij} = p_\sigma(v_i),\ 1 \leq i \leq b; \quad \sum_{i=1}^{b} t_{ij} = p_{\sigma\circ e}(v_j),\ 1 \leq j \leq b.$$

We define the *unweighted moving cost* from p_σ to $p_{\sigma\circ e}$:

$$\mathrm{mcost}_M(p_\sigma, p_{\sigma\circ e}) = \min_{\substack{(t_{ij})\in \\ T(p_\sigma, p_{\sigma\circ e})}} \sum_{i,j} t_{ij}\, \mathrm{d}_M(v_i, v_j),$$

the *moving cost* is defined as $\mathrm{mcost}_U(p_\sigma, p_{\sigma\circ e}) = s \cdot \mathrm{mcost}_M(p_\sigma, p_{\sigma\circ e})$, and the *local cost* on $e = (c_1, \ldots, c_b)$ is defined as $\sum_j p_{\sigma\circ e}(v_j) c_j r_j$. Due to linearity of expectation, $E[\mathrm{cost}_A(\sigma\circ e)] - E[\mathrm{cost}_A(\sigma)]$ is equal to the sum of moving costs from p_σ to $p_{\sigma\circ e}$ plus the local cost on e. Hence we can view A as a deterministic algorithm that maintains the probability distributions on the configurations whose cost on task e given after sequence σ is

$$\mathrm{cost}_A(\sigma \circ e) - \mathrm{cost}_A(\sigma) = \mathrm{mcost}_U(p_\sigma, p_{\sigma\circ e}) + \sum_{j=1}^{b} p_{\sigma\circ e}(v_j) c_j r_j. \quad (1)$$

In the sequel we will use the terminology of changing probabilities, with the understanding that we are referring to a deterministic algorithm charged according to Equation (1).

Elementary tasks are task vectors with only one non-zero entry, we use the notation (v, δ), $\delta \geq 0$, for an elementary task of cost δ at configuration v. Tasks $(v, 0)$ can simply be ignored by the algorithm.

Definition 1. A *continuous* randomized online algorithm for metrical task systems is an online algorithm so that for any request sequence σ, and any $v \in M$, the function $f(\delta) = p_{\sigma\circ(v,\delta)}(v)$ is continuous for $\delta \geq 0$. A *semi-reasonable* algorithm is an algorithm that never assigns a positive probability to a supported configuration.

Definition 2. A *reasonable adversary sequence* for algorithm A, is a sequence of tasks that obeys the following:

1. All tasks are elementary.
2. For all σ, the next task (v,δ) must obey that for all $\delta > \delta' \geq 0$, $p_{\sigma\circ(v,\delta')}(v) > 0$.

It follows that a reasonable adversary sequence for A never includes tasks (v,δ), $\delta > 0$, if the current probability of A on v is zero.

The following lemma is from [5, Assumption 1 and Theorem 2].

LEMMA 1. *Given a continuous algorithm A that obtains a competitive ratio of r when the adversary sequences are limited to being reasonable adversary sequences for A, then, for all $\epsilon > 0$, there also exists a randomized algorithm A' that obtains a competitive ratio of $r+\epsilon$ on all possible sequences.*

Observation 1. For a semi-reasonable and continuous online algorithm A competing against a reasonable adversary, any elementary task $e = (v,\delta)$ causes $w(v)$ to increase by δ. This follows because v would not have been supported following any alternative request (v,δ'), $\delta' < \delta$. See [5, lemma 1].

Definition 3. An online algorithm A is said to be *reasonable* and r-competitive on UMTS $U = (s, M, r_1, \ldots, r_b)$ against a reasonable adversary if it obeys the following:

1. A is continuous and semi-reasonable.
2. The probabilities that A assigns to the different configurations is purely a function of the work function.
3. Associated with A are a competition vector α_A and a bounded potential function Φ_A such that
   - Φ_A is work function based, $\Phi_A : \mathbb{R}^b \mapsto \mathbb{R}^+$.
   - For all task sequences σ and all tasks e,

$$\text{cost}_A(\sigma \circ e) - \text{cost}_A(\sigma) + \Phi_A(w_{\sigma\circ e}) - \Phi_A(w_\sigma) \leq r \cdot < \alpha_A, w_{\sigma\circ e} - w_\sigma > . \quad (2)$$

For a potential function Φ we define $|\Phi| = \sup_w \Phi(w)$.

For the remainder of this paper, we will only deal with reasonable algorithms competing against reasonable adversaries.

Definition 4. A reasonable r competitive algorithm A for UMTS $U = (s, M, r_1, \ldots, r_b)$ with associated potential function Φ is called (β,η)-constrained, $0 \leq \beta \leq 1$, $0 \leq \eta$, if the following hold:

1. For all $u, v \in M$: if $w(u) - w(v) \geq \beta\, \mathrm{d}_M(u,v)$ then the probability that A assigns to u is zero ($p_{w,A}(u) = 0$).
2. $|\Phi| \leq \eta\, \Delta(M) r$.

The concept of constrained algorithms is a useful tool to get semi-reasonable algorithms that are combined from constrained algorithms.

Observation 2. For (β,η)-constrained algorithm competing against reasonable adversary,

$$\forall u, v \in M,\ |w(u) - w(v)| \leq \beta\ \mathrm{d}_M(u,v).$$

Observation 3. If A is (β,η)-constrained then it is trivially (β',η')-constrained for all $\beta \leq \beta' \leq 1$ and $\eta \leq \eta'$.

3. COMBINING ALGORITHMS FOR UNFAIR METRICAL TASK SYSTEMS

We are given a metric space M, composed of sub-spaces $M_1, \ldots, M_b$, with minimum distances between sub-spaces comparable to the diameters of the sub-spaces. A metrical task system on M induces a metrical task system on M_i. Assume that for every induced MTS on M_i we have a $\hat{r}_i$-competitive algorithm A_i. Our goal is to combine the A_i algorithms so as to obtain an algorithm for the original MTS defined on M. To do so we make use of a "combining algorithm" $\hat{A}$. $\hat{A}$ has the role of determining which of the M_i sub-spaces contains the server. Viewed this way, it is natural that $\hat{A}$ should be an algorithm for the UMTS $\hat{U} = (s, \hat{M}, \hat{r}_1, \ldots, \hat{r}_b)$, where $\hat{M} = \{z_1, \ldots, z_b\}$ is a space with points corresponding to the sub-spaces and distances that are roughly the distances between the corresponding sub-spaces. Tasks for the original MTS are translated to tasks for the M_i induced metrical task systems simply by restriction. It remains to define how one translates tasks for the original MTS to tasks for $\hat{U}$.

Previous papers [11; 21; 5] use the average cost of the task on the sub-space M_i as the cost for z_i in the task for $\hat{U}$. This way the cost for the online algorithm is $\hat{r}_i$ times the cost for the optimum (we assume reasonableness of the algorithm and the adversary), however, *this is true only in the amortized sense.* In order to bound the amortization effect they have to assume that the diameters of the sub-space are small compared with the distances between M_i sub-spaces. We take a different approach: the cost for a point $z_i \in \hat{U}$ is (an upper bound for) *the cost of A_i on the corresponding task, divided by $\hat{r}_i$.* In this way the amortization problem disappears, and we are able to combine sub-spaces with relatively large diameter. Following is a formal description of the construction.

THEOREM 1. *Given an UMTS $U = (s, M, r_1, \ldots, r_n)$, where M is a metric space on n points. Consider an arbitrary partition of the points of M, $P = (M_1, M_2, \ldots, M_b)$, where $|M_j| = m_j$. $U_j = (s, M_j, r_{i_1^j}, \ldots, r_{i_{m_j}^j})$ is the UMTS induced by U on the subspace M_j. Let $\hat{M}$ be a metric space defined over the set of points $z_1, z_2, \ldots, z_b$ with a distance metric $\mathrm{d}_{\hat{M}}(z_i, z_j) \geq \max\{\mathrm{d}_M(u,v) : u \in M_i, v \in M_j\}$.*

Assume that

- *For all j, there is a (β_j,η_j)-constrained $\hat{r}_j$-competitive algorithm A_j for the UMTS U_j.*
- *There is a $(\hat{\beta},\hat{\eta})$-constrained r-competitive algorithm $\hat{A}$ for the UMTS $\hat{U} = (s, \hat{M}, \hat{r}_1, \ldots, \hat{r}_b)$.*

Let $\Delta(M)$ denote the diameter of a metric space M, $d_M(p,q)$ denote the distance in M between p and q, and define

$$\beta = \max\Big\{\max_i \beta_i,\ \max_{i\neq j} \frac{\hat{\beta}\,\mathrm{d}_{\hat{M}}(z_i,z_j) + \beta_j\,\Delta(M_j) + \beta_i\,\Delta(M_i) + \eta_i\Delta(M_i)}{\min_{p\in M_i, q\in M_j} \mathrm{d}_M(p,q)}\Big\}, \quad (3)$$

and

$$\eta = \hat{\eta}\frac{\Delta(\hat{M})}{\Delta(M)} + \max_i \eta_i \frac{\Delta(M_i)}{\Delta(M)}. \quad (4)$$

If $\beta \leq 1$, then there exists a (β,η)-constrained and r-competitive algorithm, A, for the UMTS U, against a reasonable adversary.

3.1 The Construction of Alg' A from Thm 1

For $1 \leq j \leq b$, denote by Φ_j and α_j the associated potential function and competition vector of algorithm A_j, respectively. Similarly, denote by $\hat{\Phi}$ and $\hat{\alpha}$ the associated potential function and competition vector of algorithm $\hat{A}$, respectively.

Given a sequence of tasks $\sigma = (v_1, \delta_1) \circ (v_2, \delta_2) \circ \cdots \circ (v_{|\sigma|}, \delta_{|\sigma|})$, $v_i \in M$, we define the sequences

$\gamma^\ell(\sigma) = (u_1^\ell, \delta_1^\ell) \circ (u_2^\ell, \delta_2^\ell) \circ \cdots \circ (u_{|\sigma|}^\ell, \delta_{|\sigma|}^\ell)$, where

- $u_j^\ell = v_j$ and $\delta_j^\ell = \delta_j$, if $v_j \in M_\ell$.
- u_j^ℓ is an arbitrary point in M_ℓ and $\delta_j^\ell = 0$, if $v_j \notin M_\ell$.

Define

$$\chi_\ell(w) =< \alpha_\ell, w > -\Phi_\ell(w)/\hat{r}_\ell. \tag{5}$$

For $u \in M$, define $\chi(u) = i$ iff $u \in M_i$. We define the sequence

$$\chi(\sigma) = (z_{\chi(v_1)}, \hat{\delta}_1) \circ (z_{\chi(v_2)}, \hat{\delta}_2) \circ \cdots \circ (z_{\chi(v_{|\sigma|})}, \hat{\delta}_{|\sigma|}),$$

inductively. Let $e = (v, \delta)$, $\chi(v) = \ell$, then $\chi(\sigma \circ e) = \chi(\sigma) \circ (z_\ell, \hat{\delta})$ where

$$\hat{\delta} = \chi_\ell(w_{\gamma^\ell(\sigma \circ e), U_\ell}) - \chi_\ell(w_{\gamma^\ell(\sigma), U_\ell}). \tag{6}$$

Note that $\hat{\delta} \geq 0$. This is so since A_ℓ is reasonable, and $\chi(\sigma)$ is a reasonable adversary sequence (see Claim 2); it follows from inequality (2) that $\hat{\delta} \geq$ the cost of A_ℓ for the task (v, δ), divided by $\hat{r}_\ell$, which is always ≥ 0.

ALGORITHM A. The algorithm works as follows:

1. It simulates algorithm A_ℓ on the task sequence $\gamma^\ell(\sigma)$, for $1 \leq \ell \leq b$.
2. It also simulates algorithm $\hat{A}$ on the task sequence $\chi(\sigma)$.
3. Its distribution is computed so that the probability assigned to a point $v \in M_\ell$ is the product of the probability assigned by A_ℓ to v and the probability assigned by $\hat{A}$ to z_ℓ. (*i.e.*, $p_{\sigma,A}(v) = p_{\gamma^\ell(\sigma),A_\ell}(v) \cdot p_{\chi(\sigma),\hat{A}}(z_\ell)$.)

We remark that the simulations above can be performed in an online fashion.

3.2 Proof of Theorem 1

To simplify notation and without loss of generality we consider an arbitrary sequence σ and arbitrary task e. With respect to σ we define

$w = w_{\sigma,U}$;	$w^e = w_{\sigma \circ e,U}$;
$w_k = w_{\gamma^k(\sigma),U_k}$, $1 \leq k \leq b$;	$w_k^e = w_{\gamma^k(\sigma \circ e),U_k}$, $1 \leq k \leq b$;
$\hat{w} = w_{\chi(\sigma),\hat{U}}$;	$\hat{w}^e = w_{\chi(\sigma \circ e),\hat{U}}$.

Define p, p_k, and $\hat{p}$ to be the probability distributions on the configurations of U, U_k and $\hat{U}$ as induced by algorithms A, A_k and $\hat{A}$ on the sequences σ, $\gamma^k(\sigma)$, and $\chi(\sigma)$, $1 \leq k \leq b$. Likewise, we define p^e, p_k^e and $\hat{p}^e$ where the sequences are $\sigma \circ e$, $\gamma^k(\sigma \circ e)$, and $\chi(\sigma \circ e)$.

We observe that algorithm A is continuous because the probabilities it assigns are the product of the probabilities assigned by two continuous algorithms.

CLAIM 2. *If the adversary sequence given to algorithm A on U is reasonable, then the simulated task sequences for algorithms A_i on U_i and the simulated task sequence for algorithm $\hat{A}$ on $\hat{U}$ are also reasonable task sequences.*

PROOF. If the adversary issues a task $e = (v, \delta)$ for A, it implies that had we replaced task e by any other task $e' = (v, \delta')$ where $0 \leq \delta' < \delta$, then $p^{e'}(v)$ would have been strictly greater than zero. This implies that $p_\ell^{e'}(v)$ would also have been strictly greater than zero, since $p^{e'}(v) = p_\ell^{e'}(v)\hat{p}^{e'}(z_\ell)$ (where $v \in M_\ell$). A similar argument implies the same for the $\chi(\sigma)$ sequence for $\hat{A}$. □

CLAIM 3. *For all σ and for all ℓ, $\hat{w}(z_\ell) = \chi_\ell(w_\ell)$.*

PROOF. From Claim 2 we know that the adversary sequence $\chi(\sigma)$ for $\hat{A}$ is reasonable. As $\hat{A}$ is reasonable and from Observation 1 we know that $\hat{w}(z_\ell)$ is exactly the sum of costs in $\chi(\sigma)$ for z_ℓ. By the definition of $\chi(\sigma)$ in Equation (6) it follows that this sum is $\chi_\ell(w_\ell)$. □

CLAIM 4. *If for all $1 \leq \ell \leq b$, $u \in M_\ell$, it holds that $w(u) = w_k(u)$ then any configuration $u \in U$ for which there exists a configuration v such that $w(u) - w(v) \geq \beta\, \mathrm{d}_M(u, v)$ has $p(u) = 0$.*

PROOF. Consider configurations u and v as above, *i.e.*, $w(u) - w(v) \geq \beta\, \mathrm{d}_M(u, v)$. We now consider two cases:

1. $u \in M_i$ and $v \in M_i$. We want to show that $w_i(u) - w_i(v) \geq \beta_i\, \mathrm{d}_{M_i}(u, v)$, as A_i is (β_i, η_i)-constrained this implies that $p_i(u) = 0$, which implies that $p(u) = 0$. From the conditions above we get

$$\begin{aligned} w_i(u) - w_i(v) &= w(u) - w(v) \\ &\geq \beta\, \mathrm{d}_M(u, v) \geq \beta_i\, \mathrm{d}_{M_i}(u, v). \end{aligned}$$

2. We now deal with the case where $u \in M_i$, $v \in M_j$, $i \neq j$. Our goal now will be to show that $\hat{w}(z_i) - \hat{w}(z_j) \geq \hat{\beta}\, \mathrm{d}_{\hat{M}}(z_i, z_j)$, as this implies that $\hat{p}(z_i) = 0$ which implies that $p(u) = 0$.

   A lower bound on $\hat{w}(z_i)$ is

$$\begin{aligned} \hat{w}(z_i) &= \chi_i(w_i) =< \alpha_i, w_i > -|\Phi_i|/r_i & (7) \\ &\geq w_i(u) - \beta_i \Delta(M_i) - |\Phi_i|/r_i & (8) \\ &= w(u) - \beta_i \Delta(M_i) - \eta_i \Delta(M_i). & (9) \end{aligned}$$

   To justify (7) one uses the definitions and Claim 3, (8) follows because a convex combination of values is at least an arbitrary value minus the maximal difference. The maximal difference between work function values is bounded by β_i times the distance, see observation 2. The last equation, (9), follows from our assumption that the work functions are equal and from the definition of η_i.

   Similarly, to obtain an upper bound on $\hat{w}(z_j)$, we derive

$$\begin{aligned} &\hat{w}(z_j) = \chi_j(w_j) = \\ &\quad =< \alpha_j, w_j > -|\Phi_j|/r_j \leq w(v) + \beta_j \Delta(M_j). \end{aligned} \tag{10}$$

It now follows from Inequalities (9) and (10) that,

$$\begin{aligned}&\hat{w}(z_i) - \hat{w}(z_j) \geq \\ &\geq (w(u) - w(v)) - \beta_i \Delta(M_i) - \beta_j \Delta(M_j) - \eta_i \Delta M_i \\ &\geq \beta d_M(u,v) - \beta_i \Delta(M_i) - \beta_j \Delta(M_j) - \eta_i \Delta M_i \\ &\geq \hat{\beta}\, \mathrm{d}_{\hat{M}}(z_i, z_j).\end{aligned}$$

The last inequality follows from Eq. (3). □

CLAIM 5. *For all σ, ℓ, and $v \in M_\ell$: $w_\ell(v) = w(v)$.*

PROOF. Proof by contradiction.

Let $\sigma \circ e$ where $e = (v, \delta)$ be the shortest task sequence for which $w_\ell^e(v) \neq w^e(v)$. As the sequence $\gamma^\ell(\sigma \circ e)$ is a reasonable adversary sequence (from Claim 2) and A_ℓ is semi-reasonable, it follows that $w_\ell^e(v) = w_\ell(v) + \delta$. But $w^e(v) \leq w(v) + \delta$ so it follows that $w_\ell^e(v) > w^e(v)$.

Let $e_x = (v, x)$, define $\delta' = \sup\{x : w^{e_x}(v) = w_\ell^{e_x}(v)\}$. Obviously, $0 \leq \delta' \leq \delta$. Define $e' = (v, \delta')$. By continuity of the work function $w^{e'}(v) = w_\ell^{e'}(v)$ and thus $\delta' < \delta$. Therefore v is supported in $w^{e'}$, and thus meets the conditions of Claim 4 (here we use the assumption that $\beta \leq 1$). Hence $p^{e'}(v) = 0$ and as the sequence σ is reasonable for A it follows that $\delta \leq \delta'$. A contradiction. □

LEMMA 6. *For all σ, and all tasks $e = (v_i, \delta)$, $v_i \in M_\ell$,*

$$\mathrm{cost}_A(\sigma \circ e) - \mathrm{cost}_A(\sigma) \leq \mathrm{cost}_{\hat{A}}(\chi(\sigma \circ e)) - \mathrm{cost}_{\hat{A}}(\chi(\sigma)).$$

PROOF. We split the cost of A into two main components, the moving cost $\mathrm{mcost}_U(p, p^e)$, and the local cost $r_i \hat{p}^e(z_\ell) p_\ell(v_i)\delta = r_i \hat{p}^e(z_\ell) p_\ell(v_i)\delta$ (see Equation (1)).

We give an upper bound on the moving cost of A by considering a possibly suboptimal algorithm that works as follows:

1. Move probabilities between the different M_j subspaces. *I.e.*, change the probability $p(v) = \hat{p}(z_j)p_j(v)$ for $v \in M_j$ to an intermediate stage $\hat{p}^e(z_j)p_j(v)$. The moving cost for A to produce this intermediate probability is bounded by $\mathrm{mcost}_{\hat{U}}(\hat{p}, \hat{p}^e)$ as the distances in $\hat{M}$ are an upper bound on the real distances for A ($\mathrm{d}_{\hat{M}}(z_i, z_j) \geq \mathrm{d}_M(u,v)$ for $u \in M_i$, $v \in M_j$). We call this cost the inter-space cost for A.

2. Move probabilities within the M_j subspaces. *I.e.*, move from the intermediate probability $\hat{p}^e(z_j)p_j(v)$, $v \in M_j$ to the probability $p^e(v) = \hat{p}^e(z_j)p_j^e(v)$. As all algorithms A_j, $j \neq \ell$, get a task of zero cost, $p_j^e = p_j$, $j \neq \ell$. The moving cost for A to produce $p^e(v)$, $v \in M_\ell$, from the intermediate stage , is no more than $\hat{p}^e(z_\ell) \cdot \mathrm{mcost}_{U_\ell}(p_\ell, p_\ell^e)$. We call this cost the intra-space cost for A.

Taking the local cost for A and the intra-space cost for A:

$$\begin{aligned}&r_i \hat{p}^e(z_\ell) p_\ell(v_i)\delta + \hat{p}^e(z_\ell) \cdot \mathrm{mcost}_{U_\ell}(p_\ell, p_\ell^e) \\ &\quad = \hat{p}^e(z_\ell)\,(cost_{A_\ell}(\sigma \circ e) - \mathrm{cost}_{A_\ell}(\sigma)) \qquad (11) \\ &\quad \leq \hat{p}^e(z_\ell)\hat{r}_\ell\big((< \alpha_\ell, w^e > - \Phi_\ell(w^e)/\hat{r}_\ell) \\ &\qquad\qquad - (< \alpha_\ell, w > - \Phi_\ell(w)/\hat{r}_\ell)\big) \qquad (12) \\ &\quad = \hat{p}^e(z_\ell)\hat{r}_\ell(\chi_\ell(w^e) - \chi_\ell(w)). \qquad (13)\end{aligned}$$

To obtain (11) we use the definition of online cost (see Eq. (1)). To obtain (12) we use the fact that A_ℓ is $\hat{r}_\ell$ competitive and reasonable (see Eq. (2)). The last equality (13) follows from the definition of χ_ℓ (see Eq. (5)).

Let $\hat{e}$ be the last task in $\chi(\sigma \circ e)$. Formula (13) is simply the local cost for algorithm $\hat{A}$ on task $\hat{e}$. Thus, we've bounded the cost for algorithm A on task e to be no more than the cost for algorithm $\hat{A}$ on task $\hat{e}$. □

PROOF OF THEOREM 1. We associate a competition vector α and a bounded potential function Φ with algorithm A, where

$$\begin{aligned}&\alpha(v) = \hat{\alpha}(z_\ell)\alpha_\ell(v) \quad \text{for } v \in M_\ell; \\ &\Phi(w) = \hat{\Phi}(\hat{w}) + r\sum_i \hat{\alpha}(z_i)\Phi_i(w_i)/\hat{r}_i\end{aligned}$$

We remark that from Claim 3 and Claim 5 it follows that $\hat{w}$ and w_i are determined by w, so $\Phi(w)$ is well defined.

We derive the following upper bound on the cost of A:

$$\begin{aligned}&\mathrm{cost}_A(\sigma \circ e) - \mathrm{cost}_A(\sigma) \\ &\quad \leq \mathrm{cost}_{\hat{A}}(\chi(\sigma \circ e)) - \mathrm{cost}_{\hat{A}}(\chi(\sigma)) \qquad (14) \\ &\quad \leq r\Big(\sum_i \hat{\alpha}(z_i)\hat{w}^e(z_i) - \sum_i \hat{\alpha}(z_i)\hat{w}(z_i)\Big) \\ &\qquad - \Big(\hat{\Phi}(\hat{w}^e) - \hat{\Phi}(\hat{w})\Big) \qquad (15) \\ &\quad = r\Big(\sum_i \sum_{v \in M_i} \hat{\alpha}(z_i)\alpha_i(v)w_i^e(v) \\ &\qquad\qquad - \sum_i \sum_{v \in M_i} \hat{\alpha}(z_i)\alpha_i(v)w_i(v)\Big) \\ &\qquad - \Big((\hat{\Phi}(\hat{w}^e) + r\sum_i \hat{\alpha}(z_i)\Phi_i(w_i^e)/\hat{r}_i) \\ &\qquad\qquad - (\hat{\Phi}(\hat{w}) + r\sum_i \hat{\alpha}(z_i)\Phi_i(w_i)/\hat{r}_i)\Big) \qquad (16) \\ &\quad = r(< \alpha, w^e > - < \alpha, w >) - (\Phi(w^e) - \Phi(w)). \qquad (17)\end{aligned}$$

Inequality (14) follows from Lemma 6, inequality (15) is implied as $\hat{A}$ is a reasonable r competitive algorithm. We obtain (16) by replacing $\hat{w}^e(z_i)$ by $\chi_i(w_i^e)$ and replacing $\hat{w}(z_i)$ by $\chi_i(w_i)$, this is possible because of Claim 3, we then substitute the definition of $\chi_\ell(w)$ (see Equation (5)), and rearranging the summands. Equation (17) follows from the definition of α and Φ above.

We now prove that A is (β, η) constrained. It follows from Claim 4 and Claim 5 that the condition on β is satisfied (see Def. 4). It remains to show the condition on η:

$$\begin{aligned}&|\Phi| \leq |\hat{\Phi}| + r\sum_i \hat{\alpha}(z_i)|\Phi_i|/\hat{r}_i \qquad (18) \\ &\quad \leq \hat{\eta} r \cdot \Delta(\hat{M}) + r\sum_i \hat{\alpha}(z_i)\eta_i \hat{r}_i \cdot \Delta(M_i)/\hat{r}_i \qquad (19) \\ &\quad \leq r \cdot \Delta(M)\big(\hat{\eta}\tfrac{\Delta(\hat{M})}{\Delta(M)} + \max_i\{\eta_i \tfrac{\Delta(M_i)}{\Delta(M)}\}\big) \\ &\quad = r \cdot \Delta(M)\eta.\end{aligned}$$

Inequality (18) follows by the definition of Φ, (19) follows because $\hat{A}$ is $(\hat{\beta}, \hat{\eta})$-constrained and A_i is (β_i, η_i)-constrained, $1 \leq i \leq b$.

We have therefore shown that A is a (β, η)-constrained and r-competitive algorithm. □

3.3 More Constrained Algorithms

Theorem 1 assumed constrained algorithms. In this section we show how to obtain such algorithms.

Definition 5. Fix a metric space M on b configurations and cost ratios $r_1, \ldots, r_b$. Assume that for all $s > 0$ there is a (β, η) constrained $f(s)$ competitive algorithm A_s for the UMTS $U_s = (s, M, r_1, \ldots, r_b)$ against a reasonable adversary. For $\rho > 0$ we define the ρ-variant of A_s (if it exists) to be a $(\beta\rho, \eta\rho)$ constrained $f(s/\rho)$ competitive algorithm for U_s.

LEMMA 7. *Under the assumptions of Definition 5, for all $\rho > 0$ such that $\beta\rho \leq 1$, and for all $s > 0$, the ρ-variant of A_s exists.*

The proof of this lemma is motivated by similar ideas from [21; 5], see the appendix. □

4. THE UNIFORM METRIC SPACE

Let M_b^d denote the uniform metric space on b points where all pairwise distances are d.

We define algorithm R_1 on UMTS $U = (s, M_b^d, r_1, \ldots, r_b)$, $b \geq 2$. Without loss of generality assume $r_1 = \max_i r_i$. Algorithm R_1 is from [5, Strategy 1] (also called ODDEXPONENT in [7]). The following lemma applies our terminology to the results of [5].

LEMMA 8. *Algorithm R_1 is $(1,1)$-constrained, and $(r_1 + 6s\ln b)$-competitive.*

PROOF. Algorithm R_1, competing against a reasonable adversary, allocates for configuration v the probability $p(v) = \frac{1}{b} + \frac{1}{b}\sum_u \left(\frac{w(u)-w(v)}{d}\right)^t$. t is chosen to be an odd integer in the range $[\ln b, \ln b + 2)$.

Bartal *et. al.* [5] prove that R_1 is reasonable, $(r_1 + 6s\ln b)$-competitive and that the associated potential function $|\Phi_1| \leq (r_1/(t+1)+s)d \leq (1/\lceil \ln b \rceil)(r_1 + 6s\ln b)d$. This implies that R_1 is $(1, 1/\lceil \ln b \rceil)$-constrained. □

We define algorithm R_2 on UMTS $U = (s, M_2^d, r_1, r_2)$. Algorithm R_2 is from [21, TWO STABLE] and [5, Strategy 2]; the algorithm also appears implicitly in [11]. The following lemma applies our terminology to the results of [21; 5].

LEMMA 9. *Algorithm R_2 is $(1,4)$-constrained, and r competitive where*

$$r = r_1 + \frac{r_1 - r_2}{e^{(r_1-r_2)/s} - 1} = r_2 + \frac{r_2 - r_1}{e^{(r_2-r_1)/s} - 1}.$$

PROOF. Algorithm R_2 works as follows: Let $y = w(v_1) - w(v_2)$, and $z = r_1 - r_2$. The probability on point v_1 is $p(v_1) = \left(e^{\frac{z}{s}} - e^{\frac{z}{s}(\frac{1}{2} + \frac{y}{2d})}\right)/(e^{z/s} - 1)$. Algorithm R_2 is shown to be reasonable and r competitive in [5; 21] and the potential function associated with R_2, Φ_2, obeys $|\Phi_2| \leq (2r_2 + s)d$.

It remains to show that $|\Phi_2| \leq 4rd$. We use the fact that if $|x| \leq 1/2$ then $1/2 \leq x/(e^x - 1)$, and do a simple case analysis. If $\max\{r_1, r_2\} > \frac{1}{2}s$ then $|\Phi_2| \leq (2r_2 + s)d \leq (2r + 2r)d \leq 4rd$. Else, let $z' = (r_2 - r_1)/s$, and notice that $|z'| \leq 1/2$, so $r = r_2 + \frac{z'}{e^{z'}-1}s \geq r_2 + \frac{s}{2}$. Hence $|\Phi_2| \leq 2rd$. □

To gain insight about the competitive ratio of R_2, we have the following claim, whose proof appears in the appendix:

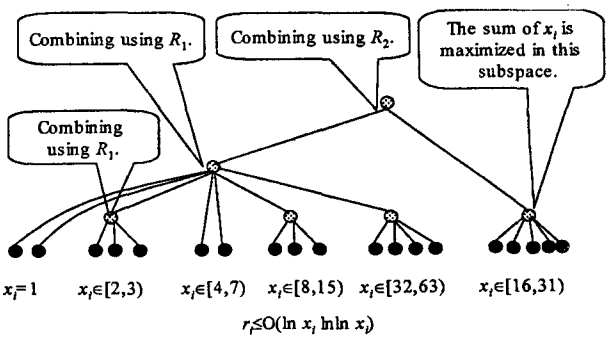

Figure 1: Schematic description of Algorithm R_3.

CLAIM 10. *Let $f(s, r_1, r_2) = r_1 + (r_1 - r_2)/\left(e^{(r_1-r_2)/s} - 1\right)$. Let $x_1, x_2 \in \mathbb{R}^+$ such that $r_1 \leq 2s(\ln x_1 + 1)$ and $r_2 \leq 2s(\ln x_2 + 1)$. Then $f(s, r_1, r_2) \leq 2s(\ln(x_1 + x_2) + 1)$.*

We now describe algorithm R_3 as defined on an UMTS $U = (s, M_b^d, r_1, \ldots, r_b)$ (see also Fig. 1). This algorithm is inspired by Strategy 3 from [5].

ALGORITHM R_3. Let x_i be the minimal real number such that $r_i \leq 100s\ln x_i \ln\ln x_i$ and $x_i \geq e^{e^6+1}$, and let x denotes $\sum_i x_i$. For a set $S \subset M_b^d$ let $U(S)$ denote the UMTS induced by U on S.

Let $M_b^d = \{v_1, \ldots, v_b\}$, where v_i has cost ratio r_i. We partition the points of M_b^d as follows: let $Q_\ell = \{v_i : e^{\ell-1} \leq x_i < e^\ell\}$. Let $P = \{Q_\ell : |Q_\ell| \geq \ln x\} \cup \{\{v\} : v \in Q_\ell \text{ and } |Q_\ell| < \ln x\}$, P is a partition of M_b^d. For $S \in P$ let $x(S) = \sum_{v_i \in S} x_i$. Without loss of generality we can assume $P = \{S_1, S_2, \ldots, S_{b'}\}$ where $b' = |P|$ and $x(S_j) \geq x(S_{j+1})$, $1 \leq j \leq b' - 1$.

We associate with every set S_i an algorithm $A(S_i)$ on UMTS $U(S_i)$. If $|S_i| \geq \ln x$ we choose $A(S_i)$ to be (1/10)-variant of R_1. If $|S_i| < \ln x$ then $|S_i| = 1$ and we choose $A(S_i)$ to be the trivial algorithm on one point, this algorithm has a competitive ratio equal to the cost ratio, it is also $(0,0)$-constrained. Let $r(S_i)$ denote the competitive ratio of $A(S_i)$ on $U(S_i)$.

If $b' = 1$ we choose R_3 to be $A(S_1)$ and we are done. If $b' \geq 2$, let $\tilde{M} = \cup_{i=2}^{b'} S_i$. We want to construct an algorithm, $A(\tilde{M})$, for $U(\tilde{M})$. If $b' = 2$, we choose $A(\tilde{M})$ to be $A(S_2)$. Otherwise, we apply Theorem 1 on $\tilde{M}$ with the partition $\{S_2, \ldots, S_{b'}\}$. We define $\hat{M}$ from Theorem 1 to be $M_{b'-1}^d$. Likewise, $\hat{A}$ from Theorem 1 is the application of the (1/5)-variant of R_1 on $\hat{U} = (s, M_{b'-1}^d, r(S_2), \ldots, r(S_{b'}))$. Let $r(\tilde{M})$ denote the competitive ratio of $\hat{A}$.

Now we choose the partition $\{S_1, \tilde{M}\}$ of M_b^d. We combine the two algorithms $A(S_1)$ and $A(\tilde{M})$ using a (1/10) variant of R_2 (this is the $\hat{A}$ required in Theorem 1) on UMTS $(s, M_2^d, r(S_1), r(\tilde{M}))$ (the UMTS $\hat{U}$ of Theorem 1). We denote the competitive ratio of $\hat{A}$ by r. The resulting combined algorithm, $A(M)$, is Algorithm R_3.

LEMMA 11. *Given that $x = \sum_i x_i$, $r_i \leq 100s\ln x_i \ln\ln x_i$, and $e^{e^6+1} \leq x_i$, Algorithm R_3 for UMTS $U = (s, M_b^d, r_1, \ldots, r_b)$ is $(1,1)$-constrained and r-competitive, where $r \leq 100s\ln x \ln\ln x$.*

PROOF. First we calculate the constraints of the algorithm.

From Lemma 8 and Lemma 7, $A(S_i)$ is $(1/10, 1/10)$-constrained, for every $1 \le i \le b'$. We would like to show that $A(\tilde{M})$ is $(1/2, 3/10)$-constrained. If $b' = 2$ then it obviously $(1/10, 1/10)$-constrained. Else ($b' > 2$), the combining algorithm for $\tilde{M}$ is a $(1/5)$-variant of R_1 which is $(1/5, 1/5)$-constrained. Hence, from Eq. (3), $\beta \le (d/5 + d/10 + d/10 + d/10)/d = 1/2$, and from Eq. (4), $\eta \le (d/5 + d/10)/d = 3/10$. From Theorem 1, $A(\tilde{M})$ is $r(\tilde{M})$ competitive.

The (β, η)-constraints of algorithm R_3 are calculated as follows: $(1/10)$-variant of R_2 is $(1/10, 4/10)$ constrained, therefore $\beta = (d/10 + d/10 + d/2 + 3d/10)/d = 1$ and $\eta = (4d/10 + 3d/10)/d = 7/10$. From Theorem 1, $A(M)$ is r-competitive.

To summarize, R_3 is $(1, 7/10)$-constrained and r-competitive algorithm for the UMTS U.

It remains to prove the bound on r. First we show that $r(S_j) \le 100s \ln x(S_j) \ln\ln x(S_j)$ for all $1 \le j \le b'$.

If $|S_j| = 1$, we are done. Otherwise, $|S_j| \ge \ln x$, and $S_j = Q_\ell$ for some ℓ. Note that $\ell \le \ln x$, so

$$
\begin{aligned}
& r(S_j) \\
& \quad \le 100s \ln e^\ell \ln\ln e^\ell + 6 \cdot 10s \ln |S_j| && (20)\\
& \quad \le 100s\big(\ln e^{\ell-1} \ln\ln e^{\ell-1} + \ln \ell + \tfrac{1}{\ell} \ln e^{\ell-1}\big) + 60s \ln |S_j| \\
& \quad \le 100s\big(\ln e^{\ell-1} \ln\ln e^{\ell-1} + \ln\ln x + \tfrac{60}{100} \ln |S_j| + 1\big) \\
& \quad \le 100s\big(\ln e^{\ell-1} \ln\ln e^{\ell-1} + 2 \ln |S_j|\big) && (21)\\
& \quad \le 100s \ln(|S_j| e^{\ell-1}) \ln\ln(|S_j| e^{\ell-1}) \\
& \quad \le 100s \ln x(S_j) \ln\ln x(S_j). && (22)
\end{aligned}
$$

We derive inequality (20) by noting that we apply a $(1/10)$ variant of R_1 to $S_j = Q_\ell$. This implies that $r_i \le 100s \ln e^\ell \ln\ln e^\ell$ for all $v_i \in S_j$. By the bound on the competitive ratio of the $(1/10)$ variant of R_1 (See Lemma 8 and Lemma 7) we get the inequality. Inequality (21) follows because $\ln |S_j| \ge \ln\ln x$, and $\ln\ln x \ge 6$. The last inequality follows because $e^{\ell-1}$ is a lower bound on x_i for $v_i \in S_j$ and thus $|S_j| e^{\ell-1} \le x(S_j)$.

We note that $b' \le \ln^2 x$ as there are at most $\ln x$ sets Q_i, and each such set contributes at most $\ln x$ sets S_i to P. We now derive a bound on $r(\tilde{M})$.

$$
\begin{aligned}
r(\tilde{M}) &\le \max_{2 \le i \le b'} r(S_i) + 6 \cdot 5 \cdot s \cdot \ln(b' - 1) && (23)\\
&\le 100s \cdot \ln x(S_2) \ln\ln x + 30 \cdot s(2 \ln\ln x) && (24)\\
&= 100s(\ln x(S_2) + 0.6) \ln\ln x
\end{aligned}
$$

Inequality (23) follows since the algorithm used is a $(1/5)$ variant of R_1. Inequality (24) follows by using the previously derived bound on $r(S_i)$ and noting that $x(S_2)$ is maximal amongst $x(S_2), \ldots, x(S_{b'})$ and that $x(S_i) \le x$.

From Lemma 7 we know that the competitive ratio of the $(1/10)$-variant of R_2 is $f(10s, r(S_1), r(\tilde{M}))$ where f is the function as given in Lemma 9. We give an upper bound on $f(10s, r(S_1), r(\tilde{M}))$ using Claim 10. To do this we need to find values y_1 and y_2 such that

$$
\begin{aligned}
r(S_1) &\le 100s \ln x(S_1) \ln\ln x = 2(10s)(\ln y_1 + 1) \\
r(\tilde{M}) &\le 100s(\ln x(S_2) + 0.6) \ln\ln x = 2(10s)(\ln y_2 + 1).
\end{aligned}
$$

Indeed, the following values satisfy the conditions above: $y_1 = x(S_1)^{\frac{100}{20} \ln\ln x}/e$ and $y_2 = (e^{0.6} x(S_2))^{\frac{100}{20} \ln\ln x}/e$. Using Claim 10 we get a bound on r as follows

$$
\begin{aligned}
r &\le 2 \cdot 10s(\ln(y_1 + y_2) + 1) && (25)\\
&\le 20s \ln\big(x(S_1)^{\frac{100}{20} \ln\ln x} + (e^{0.6} x(S_2))^{\frac{100}{20} \ln\ln x}\big) \\
&= 20s \ln\big(x(S_1)^{5 \ln\ln x} + 2^{\frac{0.6}{\ln 2} 5 \ln\ln x} \cdot x(S_2)^{5 \ln\ln x}\big) \\
&\le 20s \ln\big(x(S_1)^{5 \ln\ln x} + (2^{5 \ln\ln x} - 2) x(S_2)^{5 \ln\ln x}\big) && (26)\\
&\le 20s \ln\big((x(S_1) + x(S_2))^{5 \ln\ln x}\big) && (27)\\
&\le 100s \ln x \ln\ln x
\end{aligned}
$$

Inequality (25) follows from Claim 10. Inequality (26) follows because $\ln\ln x \ge 6$. Inequality (27) follows since, in general, $a^z + (2^z - 2)b^z \le (a + b)^z$, for $a \ge b > 0$ and $z \ge 1$ — see Claim 15 in the appendix. □

Remark 1. Strategy 3 from [5] gives (implicitly) an r-competitive algorithm for a UMTS $U = (s, M_b^d, r_1, \ldots, r_b)$, but with the weaker guarantee that if $r_i \le c\, s \log^2 x_i$, then $r \le c\, s \log^2(\sum_i x_i)$.

We now present a better algorithm when all the cost ratios but one are equal.

LEMMA 12. *Given a UMTS $U = (s, M_b^d, r_1, r_2, \ldots, r_b)$ with $r_2 = r_3 = \cdots = r_b$, there exists a $(1,1)$ constrained r competitive online algorithm, R_4, where*

$$r = 30s\Big(\ln\big(e^{\frac{r_1}{30s} - \frac{1}{3}} + (b-1)e^{\frac{r_2}{30s} - \frac{1}{3}}\big) + \tfrac{1}{3}\Big).$$

PROOF SKETCH. We define x_1, x_2, such that

$$
\begin{aligned}
r_1 &= 30s(\ln x_1 + \tfrac{1}{3}) = 2 \cdot 5 \cdot s(\ln x_1^3 + 1), \\
r_2 &= 30s(\ln x_2 + \tfrac{1}{3}) = 2 \cdot 5 \cdot s(\ln x_2^3 + 1).
\end{aligned}
$$

Let $\tilde{M} = \{v_2, \ldots v_b\}$. We use a $(1/5)$ variant of R_1 on the UMTS $U(\tilde{M})$. The competitive ratio of this algorithm is at most

$$
\begin{aligned}
r(\tilde{M}) &\le r_2 + 30s \ln(b-1) \\
&\le 30s\big(\ln((b-1)x_2) + \tfrac{1}{3}\big) = 10s\big(\ln((b-1)x_2)^3 + 1\big)
\end{aligned}
$$

and it is $(1/5, 1/5)$ constrained. We combine it with the trivial algorithm for $U(\{v_1\})$ using a $(1/5)$ variant of algorithm R_2, the resulting algorithm is $(1,1)$ constrained, and by Claim 10 we have

$$
\begin{aligned}
r &\le 10s(\ln(x_1^3 + ((b-1)x_2)^3 + 1) \\
&\le 10s(\ln(x_1 + (b-1)x_2)^3 + 1) \\
&= 30s(\ln(x_1 + (b-1)x_2) + \tfrac{1}{3}).
\end{aligned}
$$

Substituting for x_i gives the required bound. □

5. APPLICATIONS

5.1 An $O(\log^2 n \log^2 \log n)$ Competitive algorithm for MTSs

Bartal [2] defines the following:

Definition 6. A *k-hierarchical well separated tree* (k-HST) is a rooted tree with the following properties.

- Successive edge lengths on any path from the root to a leaf decrease by a factor of at least k.

- For any vertex, the lengths of the edges to its children are all equal.

The metric space induced by a k-HST T has one point for each leaf of the tree, with distances given by the tree path lengths, let $M(T)$ denote this metric space.

Bartal [2; 3] shows how to approximate arbitrary metric spaces using an efficiently constructable probability distribution over a set of k-HST spaces[1], resulting in the following theorem.

THEOREM 2 ([3]). *Suppose there is a r-competitive algorithm for any n-points k-HST metric space. Then there exists an $O(rk \log n \log\log n)$-competitive randomized algorithm for* any *n-point metric space.*

We seek an online algorithm for a metrical task system where the underlying metric space is a k-HST. Following [5] we use the unfair MTS model to obtain an online algorithm for a MTS over a k-HST metric space.

ALGORITHM RHST. We define the RHST(T) algorithm on the metric space $M(T)$, where T is a k-HST. The RHST(T) algorithm is defined inductively on the depth of the underlying HST, T. In fact, we also require that $k \geq 8$, later we show that $k \geq 8$ is not a real restriction.

When $|M(T)| = 1$, RHST(T) serves all task sequences optimally. It is $(0,0)$-constrained. Otherwise, let the children of the root of T be $v_1, \ldots, v_b$, and let T_i be the subtree rooted at v_i. Let the distance between the root of T' and v_i be $D/2$. As the subtrees T_i are themselves HSTs it follows that $\Delta(M(T_i)) \leq D/k + D/k^2 + \cdots \leq D/(k-1)$. Every algorithm RHST($T_i$) is an algorithm for the UMTS $U_i = (1, M(T_i), 1, \ldots, 1)$

We construct a metric space $\hat{M} = M_b^d$ where $d = Dk/(k-1)$, we define cost ratios $r_1, \ldots, r_b$ where $r_i = r(T_i)$, the competitive ratio of RHST(T_i). We now use Theorem 1 to combine algorithms RHST(T_i). The role of $\hat{A}$ is played by the (1/2) variant of algorithm R_3 on the unfair metrical task system $\hat{U} = (1, \hat{M}, r_1, \ldots, r_b)$. The combined algorithm is a RHST(T) on the UMTS $(1, M(T), 1, \ldots, 1)$.

We remark that the application of Theorem 1 requires that the algorithms be reasonable and constrained, we show that this is true in the following lemma.

LEMMA 13. *The algorithm RHST(T) is $O(\ln n \ln\ln n)$ competitive against a reasonable adversary, where $n = |M(T)|$.*

PROOF. Let $n' = e^{e^6+1} n$. We prove by induction on the depth of the tree that RHST(T) is $(1,1)$-constrained and $200 \ln n' \ln\ln n'$-competitive.

When $|M(T)| = 1$, it is obvious. Otherwise, let $n_i = |M(T_i)|$, $n_i' = e^{e^6+1} n_i$, and $n' = \sum_i n_i'$. We assume inductively that each of the RSHT(T_i) algorithms is $(1,1)$-constrained and $200 \ln n_i' \ln\ln n_i'$ competitive on $M(T_i)$. The combined algorithm, RSHT(T), is (β, η)-constrained. From Eq. (3), and given that $k \geq 8$, we get that

$$\beta \leq \max\left\{1, \frac{1/2 \cdot Dk/(k-1) + D/(k-1) + D/(k-1) + D/(k-1)}{D}\right\} \leq \max\{1,1\} = 1.$$

From Eq. (4) we get that $\eta \leq 1/2 \cdot \frac{Dk/(k-1)}{D} + \frac{D/(k-1)}{D} \leq 5/7$, for $k \geq 8$. This proves that the algorithm is well defined and $(1,1)$ constrained.

We now bound the competitive ratio using Lemma 11, we apply a (1/2) variant of R_3, from Lemma 7 this means that the competitive ratio obtained by the (1/2) variant of R_3 on $(1, \hat{M}, r_1, \ldots, r_b)$ is the same as the competitive ratio attained by R_3 on $(2, \hat{M}, r_1, \ldots, r_b)$. Note that x_i computed by R_3 is at most n_i', hence by Lemma 11 it follows that the competitive ratio is at most $100 \cdot 2 \ln x \ln\ln x \leq 200 \ln n' \ln\ln n'$, since $x = \sum_i x_i$. □

Remark 2. Every k-HST T can be approximated by a 8-HST T' so the the distances in $M(T')$ dominate the distances in $M(T)$ and the distortion is at most $O(k/(k-1))$. This means that we have an $O(\frac{k}{k-1} \ln n \ln\ln n)$ competitive algorithm for any k-HST T with $|M(T)| = n$ and $k > 1$.

Combining Theorem 2 with Lemma 13 and by choosing $k = 8$, it follows that

THEOREM 3. *For any MTS over an n-point metric space, the randomized competitive ratio is $O(\log^2 n \log^2\log n)$.*

5.2 A Tight Competitive Ratio for k-Weighted Caching on $k+1$ Points

The k-weighted caching problem is a paging problem where the cost to retrieve a page varies from page to page. It is known (*e.g.*, [23; 9]) that this problem is equivalent to a problem of k-server on a star metric.[2] It is well known that the k-server problem on a metric space on $k+1$ points is a special case of a metrical task system on the same metric space, and hence any upper bound for the metrical task system translates to an upper bound for the corresponding k-server problem.

Given a star metric space M, we construct an 8-HST T and map the points of the star metric space, under mapping m, to the leaves of the T. T has the special structure that for every internal vertex, all children except perhaps one, are leaves. As before, let $M(T)$ denotes the metric space induced by T. It is not hard to see[3] that one can find such a tree T such that for any $u, v \in M$, $\mathrm{d}_M(u,v) \leq \mathrm{d}_{M(T)}(m(u), m(v)) \leq 8 \cdot 15/7 \cdot \mathrm{d}_M(u,v)$.

We now follow the construction of RHST given in the previous section, on an 8-HST T, except that we make use of (1/2)-variant of R_4 rather than (1/2)-variant of R_3. The special structure of T implies that all the children of an inner vertex, except perhaps one, have trivial 1-competitive algorithms on their subspaces, and hence we can apply R_4. Using induction on the depth of the tree and Lemma 12, it is easy to bound the competitive ratio on $k+1$ leaves tree to be at most $60(\ln(k+1) + 1/3)$.

Combining the above with the lower bound of [9] gives us:[4]

[1] The approximation is only in the expectation.

[2] The star metric is derived from a depth one tree with distances on the edges, the points of the metric space are the leaves of the tree and the distances between a pair of points is the length of the (2 edge) path between them.

[3] Essentially, the vertices furthest away from the root (up to a factor of 8) in the star are children of the root of T and the last child of the root is a recursive construction for the rest of the star metric points.

[4] For the purpose of k-weighted caching on $k+1$ points, it is possible to replace the usage of R_1 in R_4 by algorithms for paging, such as of [20; 1], and have better constant factor in the competitive ratio.

THEOREM 4. *The competitive ratio for the k-weighted caching problem on $k+1$ points is $\Theta(\log k)$.*

6. FURTHER RESEARCH

The problem of finding an optimal algorithm for the UMTS on uniform metric space remains open (even for an MTS on the uniform metric space, no optimal algorithm is known, see [16]). Such a tight algorithm may give an $O(\log n)$-competitive algorithm for HSTs and if so will improve the bound for any MTS to $O(\log^2 n \log\log n)$. In order to further improve this bound, one needs to deviate from the black box usage of Theorem 2.

Analogously to the combining algorithms technique presented in this paper, it is possible to define appropriate adversaries and combine them in a similar way.[5] If we would have matching adversaries, analogous to algorithms R_1 and R_2, it would be possible to get $\Omega(\frac{k-1}{k}\log n/\log\log n)$ lower bound on the competitive ratio for k-HST on n points. Such a result would imply improved $\Omega(\log n/(\log\log n)^2)$ lower bound on the competitive ratio for MTS on any metric space. Indeed, Seiden [21] proves a matching lower bound for UMTSs on two points, however, it is still open whether exists a $r_1+\Omega(\log b)$ lower bound on the competitive ratio for UMTSs on uniform metric space with b points and equal cost ratios of r_1.

An interesting line of research would be to apply these techniques to the (much harder) k-server problem, or even for a special case such as the k-weighted caching problem, see also [8].

Acknowledgments. We would like to thank Yair Bartal and Avrim Blum for helpful discussions.

7. REFERENCES

[1] D. Achlioptas, M. Chrobak, and J. Noga. Competitive analysis of randomized paging algorithms. In *Proceedings of the 4th European Symposium on Algorithms, LNCS*, volume 1136, pages 419–430. Springer-Verlag, 1996.

[2] Y. Bartal. Probabilistic approximations of metric space and its algorithmic application. In *37th Annual Symposium on Foundations of Computer Science*, pages 183–193, Oct. 1996.

[3] Y. Bartal. On approximating arbitrary metrics by tree metrics. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 183–193, 1998.

[4] Y. Bartal, A. Blum, C. Burch, and A. Tomkins. A polylog(n)-competitive algorithm for metrical task systems. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 711–719, May 1997.

[5] Y. Bartal, A. Blum, C. Burch, and A. Tomkins. A polylog(n)-competitive algorithm for metrical task systems. manuscript (prelimanary version appeared at [4]), Oct. 1997.

[5] Blum *et al.* [11] use combined adversaries to get an $\Omega(\sqrt{\log n/\log\log n})$ lower bound for any metric space. However, as in the upper bound, their combining technique can be used effectively on k-HST only for large k.

[6] S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Wigderson. On the power of randomization in online algorithms. *Algorithmica*, 11(1):2–14, Jan. 1994.

[7] A. Blum and C. Burch. On-line learning and the metrical task system problem. In *Proceedings of the 10th Annual Conference on Computational Learning Theory*, pages 45–53, 1997.

[8] A. Blum, C. Burch, and A. Kalai. Finely-competitive paging. In *40th IEEE Symposium on Foundations of Computer Science*, 1999.

[9] A. Blum, M. L. Furst, and A. Tomkins. What to do with your free time: algorithms for infrequent requests and randomized weighted caching. manuscript, Apr. 1996.

[10] A. Blum, H. Karloff, Y. Rabani, and M. Saks. A decomposition theorem and lower bounds for randomized server problems. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, pages 197–207, 1992.

[11] A. Blum, H. Karloff, Y. Rabani, and M. Saks. A decomposition theorem and bounds for randomized server problems. manuscript, Feb. 1999. preliminary version appeared in [10].

[12] A. Borodin, N. Linial, and M. Saks. An optimal online algorithm for metrical task systems. *J. Assoc. Comput. Mach.*, 39(4):745–763, 1992.

[13] M. Chrobak, H. Karloff, T. Payne, and S. Vishwanathan. New results on server problems. In *1st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 291–300, 1990.

[14] M. Chrobak and L. L. Larmore. The server problem and on-line games. In L. A. McGeoch and D. D. Sleator, editors, *On-line Algorithms*, volume 7 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 11–64, Feb. 1991.

[15] A. Fiat, R. Karp, M. Luby, L. A. McGeoch, D. D. Sleator, and N. E. Young. Competitive paging algorithms. *Journal of Algorithms*, 12:685–699, 1991.

[16] S. Irani and S. Seiden. Randomized algorithms for metrical task systems. *Theoretical Computer Science*, 194(1–2):163–182, Mar. 1998.

[17] A. Karlin, M. Manasse, L. Rudolph, and D. D. Sleator. Competitive snoopy caching. *Algorithmica*, 3(1):79–119, 1988.

[18] H. Karloff, Y. Rabani, and Y. Ravid. Lower bounds for randomized k-server and motion-planning algorithms. *SIAM Journal on Computing*, 23(2):293–312, Apr. 1994.

[19] M. Manasse, L. A. McGeoch, and D. Sleator. Competitive algorithms for server problems. *Journal of Algorithms*, 11:208–230, 1990.

[20] L. McGeoch and D. Sleator. A strongly competitive randomized paging algorithm. *J. Algorithms*, 6:816–825, 1991.

[21] S. Seiden. Unfair problems and randomized algorithms for metrical task systems. *Information and Computation*, 148(2):219–240, Feb. 1999.

[22] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Communication of the ACM*, 28:202–208, 1985.

[23] N. Young. The k-server dual and loose competitiveness for paging. *Algorithmica*, 11(6):525–541, June 1994.

[24] N. Young. On-line file caching. In *9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 82–86, 1998.

APPENDIX

Given any metric space M, we define a new metric space ρM, $\rho > 0$, that is defined on a set of points $\{v' | v \in M\}$, where the distance $\mathrm{d}_{\rho M}(v', u') = \rho\, \mathrm{d}_M(v, u)$ for all $u', v' \in \rho M$.

LEMMA 14. *Let $0 < \beta \leq 1$ and $0 < \beta/\rho \leq 1$. Assume we are given a $(\beta/\rho, \eta/\rho)$-constrained, r-competitive online algorithm A', against a reasonable adversary on the UMTS $U' = (s/\rho, \rho M, r_1, \ldots, r_b)$. This implies that there exists an online algorithm A which is (β, η)-constrained and r competitive, against a reasonable adversary on the UMTS $U = (s, M, r_1, \ldots, r_b)$.*

PROOF. Algorithm A on UMTS U simulates algorithm A' on UMTS U' by translating every task (v, δ) to task (v', δ). The probability that A associates with configuration v is the same as the probability that algorithm A' associates with configuration v'. If the adversary sequence for A' is reasonable then the simulated adversary sequence for A' is also reasonable simply because the probabilities for v and v' are identical.

The costs of A or A' on task (v, δ) or (v', δ) can be partitioned into moving costs and local costs. As the probability distributions are identical, the local costs for A and and A' are the same. The unweighted moving costs for A are $1/\rho$ the unweighted moving costs for A' because all distances are multiplied by $1/\rho$. However, the moving costs for A' are the unweighted moving costs multiplied by a factor of s/ρ whereas the moving costs for A are the unweighted moving costs multiplied by a factor of s. Thus, the moving costs are also identical.

To show that A is (β, η)-constrained (and hence semi-reasonable) we first need to show that if the work functions in U and U' are equal, then this implies that if u and v are two configurations such that $w(u) \geq w(v) + \beta\, \mathrm{d}_M(u, v)$ then $p(u) = 0$. This is true because A' is $(\beta/\rho, \eta/\rho)$-constrained, and thus $w(u') \geq w(v') + (\beta/\rho) \cdot \mathrm{d}_{\rho M}(u', v')$ implies a probability of zero on u' for A' which implies a probability of zero on u for A. Next, one needs to show that the work functions are the same, this can be done using an argument similar to the proof of Claim 5.

As the work functions and costs are the same for the online algorithms A and A' it follows that we can use the same potential function. To show that $|\Phi| \leq \eta \cdot \Delta(M)$ we note that $|\Phi| \leq (\eta/\rho)\Delta(\rho M)$. □

Observation 4. Assume that there is a (β, η)-constrained, r-competitive algorithm A for a UMTS $U = (s, M, r_1, \ldots, r_b)$ against a reasonable adversary. Then, for all $\rho > 0$, a natural modification of A, A', is a (β, η)-constrained, r-competitive algorithm against a reasonable adversary on the UMTS $U' = (s, \rho M, r_1, \ldots, r_b)$. A task (v', δ) for U' is simulated as a task $(v, \delta/\rho)$ in U. The observation follows because there is a natural isomorphism between U and U' and between the original sequence on U and the modified sequence on U'.

PROOF OF LEMMA 7. For all $\rho > 0$ such that $\beta\rho \leq 1$:

1. By assumption, there is a (β, η)-constrained, $f(s/\rho)$-competitive algorithm for the UMTS $(s/\rho, M, r_1, \ldots, r_b)$.
2. It follows from Lemma 14 that there exists an online algorithm that is $(\rho\beta, \rho\eta)$-constrained, $f(s/\rho)$-competitive for the UMTS $(s, \rho^{-1}M, r_1, \ldots, r_b)$.
3. It now follows from Observation 4 that there exists a $(\rho\beta, \rho\eta)$-constrained, $f(s/\rho)$-competitive online algorithm for the UMTS $(s, M, r_1, \ldots, r_b)$. This means that the ρ variant of A_s exists. □

PROOF OF CLAIM 10. First we show that f is monotonic and non-decreasing as a function of both r_1 and r_2. Since the formula is symmetric in r_1 and r_2 it is enough to check monotonicity in r_1. Let $x = (r_1 - r_2)/s$, it suffices to show that $g(x) = sx + r_2 + sx/(e^x - 1)$ is monotonic in x. Taking the derivative

$$g'(x) = s \cdot \frac{e^x(e^x - (1+x))}{(e^x - 1)^2} \geq 0, \text{ since } e^x \geq 1 + x.$$

Therefore we may assume that $r_1 = 2s(\ln x_1 + 1)$ and $r_2 = 2s(\ln x_2 + 1)$. Without loss of generality we can assume that $x_1 \geq x_2$ and let $y \geq 2$ be such that $x_1 = (x_1 + x_2)(1 - 1/y)$. By substitution we get

$$r_1 - r_2 = 2s \ln(y - 1)$$

and

$$f(s, r_1, r_2) = r_1 + \frac{r_1 - r_2}{e^{(r_1 - r_2)/s} - 1} =$$

$$= 2s\Big(\ln(x_1 + x_2) + 1 + \ln(y-1) - \ln y + \frac{\ln(y-1)}{(y-1)^2 - 1}\Big)$$

$$\leq 2s\Big(\ln(x_1 + x_2) + 1 - \frac{1}{y} + \frac{\ln(y-1)}{(y-1)^2 - 1}\Big).$$

We now prove that for $y \geq 2$, $-\frac{1}{y} + \frac{\ln(y-1)}{(y-1)^2-1} \leq 0$. When y approaches 2, the limit of the expression is zero. For $y > 2$, we multiply the left side by $(y-1)^2 - 1$, and get $g(y) = -(y-2) + \ln(y-1)$. Since $g(2) = 0$ and $g'(y) = -1 + 1/(y-1) < 0$ for $y > 2$, we are done. □

CLAIM 15. *For $a \geq b > 0$ and $z \geq 1$,*

$$a^z + (2^z - 2)b^z \leq (a + b)^z.$$

PROOF.

$$a^z + (2^z - 2)b^z$$

$$\leq (\max\{a, 2b\})^{\{z\}}\left(a^{\lfloor z \rfloor} + (2^{\lfloor z \rfloor} - 1)b^{\lfloor z \rfloor}\right)$$

$$\leq (a + b)^{\{z\}}(a + b)^{\lfloor z \rfloor} \leq (a + b)^z$$

The second inequality follows since $\max\{a, 2b\} \leq a + b$ and from the sum of the binomial coefficients. □