Randomized Online Algorithms for Minimum Metric Bipartite Matching

Adam Meyerson * Akash Nanavati † Laura Poplawski ‡

Abstract

We present the first poly-logarithmic competitive online algorithm for minimum metric bipartite matching. Via induction and a careful use of potential functions, we show that a simple randomized greedy algorithm is competitive on a hierarchically separated tree. Application of recent results on randomized embedding of metrics into trees yield the poly-logarithmic result for general metrics.

1 Introduction

Matching is one of the most basic problems in algorithms. Natural applications include assignment of resources to jobs or wireless nodes to base stations. In addition, matching is a subroutine in a wide variety of algorithms.

In the online bipartite matching problem, nodes from one side of the matching arrive one at a time, and each must be matched when it arrives. This models resource allocation in a changing environment (for example). As the matching problem has wide applications, this has been the subject of considerable previous study. Unfortunately, the general online matching problem is intractable. A single "incorrect" match can cause some subsequent node to have no feasible matching at all. This was first observed by Karp et al [7] and an online algorithm was given which would match half the nodes. Because of this intractability of the general problem, it is reasonable to consider the restricted case where distances between nodes form a metric (satisfy symmetry and triangle inequality). The goal is now to find a matching of minimum cost (matchings of maximum cost were addressed by [5, 8]). The quality of an online algorithm is measured via the competitive ratio (the worst-case ratio of the cost of the matching found by our algorithm to the cost of the best possible matching given the metric and points to be matched).

The *Permutation* algorithm of Khuller et al [8] (also discovered independently by Kalyanasundaram and Pruhs [5, 6]) gives a competitive ratio of $2k - 1$ for the problem, where k is the number of nodes to be matched. No deterministic algorithm can perform better than this for general metrics.

At this point, previous work focused on improving the competitive ratio for specified simple metrics. For example, it was conjectured that the Work Function Algorithm of Koutsoupias and Papadimitriou [10] would obtain better performance on the line. Constant-competitive performance was later disproved by Koutsoupias and Nanavati [9], and a lower bound of 9.001 on the line was given by Fuchs et al [4]. While several previous papers mentioned the randomized case as an open problem, to our knowledge no randomized algorithms with sub-linear (expected) competitive ratios were previously known for general metrics.

In this paper, we consider a simple randomized greedy algorithm. As each node arrives to be matched, it will be paired with its closest available mate. If there is a tie (several possibilities at the same distance) then one will be selected at random. It is straightforward that this algorithm will not be competitive on general metrics. However, by making use of a carefully chosen potential function, we shall prove that the randomized greedy is poly-logarithmic (expected) competitive on hierarchically separated trees with a sufficiently large separation factor between the lengths of edges.

A series of recent papers by Yair Bartal [1, 2], culminating in the result of Fakcharoenphol et al [3] attain bounds on the expected distortion created by a randomized embedding of an arbitrary metric into a hierarchically separated tree. By combining this result with our analysis of randomized greedy, we can obtain a randomized algorithm with (expected) poly-logarithmic competitive ratio for general metrics.

We leave open the possibility of improving our competitive ratio to $\Theta(\log k)$. However, we observe that the techniques of this paper (combining tree embedding with an algorithm for hierarchically separated trees) will

*Department of Computer Science, University of California, Los Angeles. Email: awm@cs.ucla.edu.

†Google Inc, Mountain View CA 94043. Email: akash@google.com. This work was done while the author was at UCLA, supported in part by NSF.

‡Department of Computer Science, University of California, Los Angeles. Email: laurap@cs.ucla.edu.

not attain such an improvement – this follows from the fact that there is a logarithmic competitive lower bound even on a tree.

The most compelling open problem arising from this result is a randomized algorithm for the k-server problem. Previous authors have observed a connection between online matching and k-server (in particular the work function algorithm has been applied to both). The only lower bounds for k-server in a randomized setting come from paging, and these bounds are poly-logarithmic. A randomized online algorithm with sub-linear expected competitive ratio would be quite valuable.

2 Algorithms and Lower Bounds on the Uniform Metric

In this section, we will formally define the problem and the competitive ratio measurement. We will develop various upper and lower bounds for online matching. In most cases these results were previously known, however they form the base case of our induction and are illustrative of the functioning of our greedy algorithms, so we will include them for completeness. We formally state the online matching problem as follows:

PROBLEM STATEMENT 1. *We are given a metric V, d (where V is a possibly-infinite set of vertices and d is a function from $V \times V$ to $\Re^+$ such that $d(x,x) = 0$, $d(x,y) = d(y,x)$, and $d(x,y) + d(y,z) \geq d(x,z)$ for all $x, y, z \in V$). In addition, we are given a set of distinguished "right-hand" nodes $R \subseteq V$ with $|R| = k$. Nodes designate themselves as members of the set of "left-hand" nodes $L \subseteq V$ one at a time. We will permit R and L to be non-disjoint. As each node $x \in L$ is designated, we must immediately match it to some node $\mu(x) \in R$. This matching is permanent, in that the value of $\mu(x)$, once assigned, can never be changed. We must maintain that $\mu(x) \neq \mu(y)$ for any $x, y \in L$ (i.e. μ is a matching). Our goal is to minimize $\Sigma_{x \in L} d(x, \mu(x))$.*

In designing an algorithm for this problem, our goal will be to minimize the competitive ratio. This measure is standard in the online algorithms literature, and can be defined as follows:

DEFINITION 1. *The competitive ratio of an algorithm for the online matching problem is the maximum, over all possible inputs V, d, R, L (where V, d is a metric, R is a subset of V, and L is an ordered subset of V) of the ratio $cost_{US}/cost_{OPT}$. Here $cost_{US}$ is the cost of the matching produced by the algorithm, and $cost_{OPT}$ is the cost of the best matching of R to L. In general this competitive ratio might grow large as the sizes of sets V, R, L increase, so our goal is to bound it by a function of $k = |R|$ (hopefully independent of $n = |V|$, the number of points in the metric space).*

Restricting ourselves to the uniform metric, we will first consider deterministic online algorithms for the problem. The greedy algorithm assigns $\mu(x)$ to be the closest unmatched $y \in R$. Somewhat unsurprisingly, this is best-possible for the uniform metric, as we prove below.

THEOREM 2.1. *The greedy algorithm attains a competitive ratio of k on the uniform metric. No deterministic algorithm can attain a competitive ratio better than k.*

Proof. If the optimum offline matching has a cost of zero, then the optimum matching must be $\mu^*(x) = x$ for all $x \in L$. The greedy algorithm will always assign $\mu(x) = x$ when possible, so it will also have a cost of zero. On the other hand, if optimum offline has a nonzero cost, then the cost must be at least one (by the nature of the uniform metric). The greedy algorithm, at worst, has $d(x, \mu(x)) = 1$ for all $x \in L$, which would give a cost of k. Thus the competitive ratio of greedy is at most k. On the other hand, consider an adversary which first designates $x_1 \in L$ for some x_1 which is not in R. The greedy algorithm matches this to $\mu(x_1) = y_1$. The adversary then continues by designating $x_i \in R$ where $x_i = y_{i-1}$. Notice that the greedy can never match x_i to itself, since at each step x_i was already matched (in the previous iteration). Thus greedy has $d(x_i, y_i) = 1$ and pays a total of k. On the other hand, the offline optimum could match $\mu^*(x_i) = y_{i-1}$ for all $i > 1$, and match x_1 to y_k (the last remaining point), yielding a total cost of one. This proves that the competitive ratio of greedy is *exactly* k. We observe that the adversary described above works against any deterministic online algorithm, giving a general lower bound of k on the competitive ratio.

For a general metric, the lower bound on the competitive ratio is $2k - 1$. This bound comes from the same example in the theorem above, except that the first node (x_1) will be at distance $\frac{1}{2}$ from all nodes in R. Thus the optimum pays only $\frac{1}{2}$ whereas any deterministic algorithm pays at least $\frac{1}{2} + (k - 1)$. The *Permutation* algorithm obtains this best-possible competitive bound [8].

Since our results deal with greedy algorithms, we observe that greedy does not obtain a particularly good competitive ratio on general metrics.

THEOREM 2.2. *There exists an instance of the online matching problem on a line metric, such that the greedy algorithm has a competitive ratio of $\Omega(2^k)$.*

Proof. We place the nodes of R on the line at $y_0 = 0$ and $y_i = 2^i$ for each $1 \leq i < k$. The nodes of L arrive one at a time, with $x_0 = 1$ and $x_i = 2^i$ for each $1 \leq i < k$. The optimum offline matching matches $\mu^*(x_i) = y_i$ for a cost of 1. However, greedy can match $\mu(x_i) = y_{i+1}$ for each $0 \leq i < k-1$ and then match $\mu(x_{k-1}) = y_0$. In each case the greedy had a choice of two equidistant points to select from (we can force this behavior from greedy by adding small epsilons to the distance between the points.) This yields a cost of $1+2+4+...+2^{k-1} = 2^k - 1$ for greedy.

Since the deterministic problem has essentially been resolved, both for the uniform metric and for general metrics, we consider the case of randomized algorithms. The matching returned by a randomized algorithm depends not only upon the inputs, but also upon the random bits generated and used by that algorithm. While an "adversary" is assumed to generate worst-case inputs, this adversary is not aware of the random bits; this is the *oblivious adversary* model which is standard in analysis of randomized online algorithms. We can measure performance of a randomized algorithm for online matching via the expected competitive ratio:

DEFINITION 2. *The expected competitive ratio of an algorithm is the maximum, over all inputs V, d, R, L, of the expected value $E[cost_{US}/cost_{OPT}]$. This expectation is taken over the random bits generated by the algorithm. Since the inputs themselves are worst case (and not random), the cost of the best matching does not depend upon any random bits. It follows that $E[cost_{US}/cost_{OPT}] = E[cost_{US}]/cost_{OPT}$. Once again $cost_{US}$ and $cost_{OPT}$ represent the cost of the matching generated by our algorithm, and the best matching of R to L respectively.*

We will make use of the following randomized greedy algorithm.

ALGORITHM 1. **Randomized Greedy**: *When point $x \in L$ arrives, we select the unmatched point $y \in R$ which minimizes $d(x, y)$. If there are multiple points in R which satisfy this condition, then we select one of them uniformly at random. We set $\mu(x) = y$.*

Of course, the randomized greedy is as bad as deterministic greedy on general metrics. We can force "wrong" choices by applying small epsilons to the distances, thus ensuring that there are no ties and the randomized choice never occurs. On the other hand, the randomized greedy performs quite well on the uniform metric.

THEOREM 2.3. *Randomized greedy has an expected competitive ratio of $H_k = \Theta(\log k)$ on the uniform metric. This is the best possible for any randomized algorithm.*

Proof. If each node of L is collocated with a node of R, $cost_{US} = cost_{OPT} = 0$. Therefore, assume at least one left-hand node is not collocated with a right-hand node. Let m = the number of left-hand nodes that are not collocated with right-hand nodes ($m > 0$). This gives $cost_{OPT} = m$.

Now consider our algorithm. When a node arrives that is not collocated, we will pay 1, for a total over all non-collocated nodes of m. When a node arrives that is collocated, our cost is 1 times the probability that another node is already matched to the collocated right-hand node.

Look at the i^{th} collocated left-hand node to arrive, call it x_i. Call its collocated right-hand node y_i. Let $m(i) \leq m$ be the number of non-collocated left-hand nodes that have already arrived. We know that $i-1$ of the previous left-hand nodes to arrive have been matched to the $i-1$ previous collocated right-hand nodes (since, for each such right-hand node, either the collocated left-hand node was matched to it or else the collocated left-hand node was already matched). Therefore, the total number of left-hand nodes that could have been matched to y_i is $m(i)$, and the total number of right-hand nodes to which each of these could have been matched is $k-(i-1)$. So, the probability that y_i has already been matched is $\frac{m(i)}{k-(i-1)}$.

Summing over all left-hand nodes, we get

$$\begin{aligned} cost_{US} &= m + \sum_{i=1}^{k-m} \frac{m(i)}{k-i+1} \\ &\leq m + \sum_{i=1}^{k-m} \frac{m}{k-i+1} \\ &\leq m + m \sum_{i=m+1}^{k} \frac{1}{i} \\ &\leq m(1 + H_k - H_m) \\ &\leq cost_{OPT} H_k \end{aligned}$$

For the lower bound, the adversary first designates some node x_0 which is not in R. It then repeatedly designates the node in R which is not yet in L and which is most likely to be already matched. The offline optimum can pay 1. The online algorithm pays at least 1 plus the sum of probabilities that each requested node was already matched. The probability that x_i was already matched must be at least $\frac{1}{k-i+1}$ since at least one of the remaining nodes in R is matched. This gives a cost of at least H_k for any randomized algorithm, matching the upper bound for randomized greedy.

3 Randomized Greedy on a HST

We will now extend the analysis of randomized greedy to a hierarchically separated tree. Because of various results on metric embedding, this will be sufficient to construct an algorithm for general metrics. An HST is defined as follows:

DEFINITION 3. *An α-Hierarchically Separated Tree (α-HST) metric is defined by a rooted tree $T = (V, E)$ along with a distance function on the edges $d : E \rightarrow \Re^+$. The HST has the following additional properties on the distance metric:*

1. *For any $v \in V$, if $c_1(v)$ and $c_2(v)$ are children of v in the rooted tree, $d(v, c_1(v)) = d(v, c_2(v))$.*
2. *For any $v \in V$, let $p(v)$ be the parent of v and $c(v)$ be a child of v. Then $d(v, p(v)) = \alpha d(v, c(v))$.*
3. *All leaves are at the same level of the tree. Thus if $v \in V$ and $\lambda_1(v), \lambda_2(v)$ are leaves which are descendants of v, then $d(v, \lambda_1(v)) = d(v, \lambda_2(v))$.*

We will assume that only leaf nodes are members of sets L and R. Our proof of the competitive ratio will proceed by induction on the number of levels in the tree. Since the proof is somewhat notation-heavy, we will define our notation in the following table.

δ	the distance from the root to each of its children
$(\beta+1)\delta$	the distance from the root to each leaf
x	the number of children of the root
S_i	the subtree rooted at the ith child of the root
r_i	the number of right-hand nodes in S_i
l_i	the number of left-hand nodes in S_i
m_i	$\lvert r_i - l_i \rvert$
M	$\frac{\sum_{i=1}^{x} m_i}{2}$, or the number of left-hand nodes that the optimal solution must match to different subtrees.
M_i^*	the set of left-hand nodes not in S_i that randomized greedy matches to right hand nodes in S_i
m_i^*	$\lvert M_i^* \rvert$
$cost_{US}(S_i)$	the cost of our algorithm to match nodes within S_i
$cost_{OPT}(S_i)$	the cost for the best possible matching within S_i

We will first prove a useful lemma, which bounds the expected number of nodes which randomized greedy matches on paths through the root.

LEMMA 3.1.

$$E\left[\sum_{i=1}^{x} m_i^*\right] \leq 1 + 2M \ln k$$

Proof. By definition of m_i^*, the sum of the m_i^*s is the expected number of left-hand nodes that our algorithm will match to right-hand nodes in different subtrees. Label the left-hand nodes in the reverse of the order in which they arrive, $\{l_k, l_{k-1}, ..., l_2, l_1\}$. We consider an offline algorithm OFF which matches the points so as to minimize the number matched between subtrees (via the root). Define a potential function Φ as follows:

$$\Phi_t = \sum_{i=1}^{t-1} \begin{cases} 2 \ln i & \text{if } \mu_{OFF}(l_i) \text{ outside } S_i \\ 0 & \text{otherwise} \end{cases}$$

Also define $\rho_t =$ the number of left-hand nodes that randomized greedy has matched outside their subtrees after l_t is matched.

Claim: $\forall t > 1, E[\Phi_t + \rho_t] \leq 2M \ln k$

Base case: $t = k + 1$ (the first left-hand node has not arrived yet). OFF will match exactly M left-hand nodes outside their subtrees. Since $k \geq i$ for all possible values of i, we have

$$\Phi_{k+1} \leq \sum_{i=1}^{M} 2 \ln k \leq 2M \ln k$$

Of course $\rho_{k+1} = 0$, since the algorithm has not yet matched any nodes. Combining these two yields the base case of the claim.

Inductively assume the claim is true for all nodes arriving before l_t (ie, for all $t' > t$). Now suppose l_t arrives. If there are unmatched right-nodes in this subtree, then we will match to one of these. We can modify the solution of OFF to also match to this right-node, bumping whichever left-node matched there before to wherever OFF matched l_t. This can only decrease the potential value, since the only possible change to the set of nodes matched outside their subtree is replacing l_t with a later node (thus decreasing potential). Thus the only interesting case is where no unmatched right-nodes exist in this subtree. Both the randomized greedy and OFF must match this node to some other subtree; the difficulty exists because the greedy might pick the "wrong" subtree to match to.

In this case, we know $\rho_t = \rho_{t+1} + 1$. We need to bound the change in the potential function. The

randomized greedy is equally likely to match to the right-hand node OFF intended for each of the left-hand nodes $\{l_t, l_{t-1}, ..., l_2, l_1\}$.

Say we match l_t to the right-hand node intended for l_i. At worst, OFF can match l_i with the right-hand nodes intended for l_t. In this case the potential will decrease by $2\ln t$ (since the matching of l_t is no longer part of the potential) and increase by $2\ln i$ (unless we select $i = t$ in which case the potential does not increase at all).

This gives the expected value of the potential:

$$\begin{aligned} E[\Phi_t] &\leq \Phi_{t+1} - 2\ln t + \frac{1}{t}\sum_{i=1}^{t-1} 2\ln i \\ &\leq \Phi_{t+1} - 1 \end{aligned}$$

when $t > 1$, using $\sum_{i=1}^{t-1} 2\ln i < 2\int_1^t (\ln i)di \leq (2t\ln t) - (2t-2) \leq (2t\ln t) - t$ for $t > 1$. So,

$$\begin{aligned} E[\Phi_t + \rho_t] &\leq \Phi_{t+1} - 1 + \rho_{t+1} + 1 \\ &\leq \Phi_{t+1} + \rho_{t+1} \\ &\leq 2M\ln k \text{ by inductive assumption} \end{aligned}$$

When the last node arrives, the potential reduces to zero. Since at most this single node was matched via the root, we have $\rho_1 \leq \rho_2 + 1$. It follows that $E[\rho_1] \leq 1 + E[\rho_2] \leq 1 + E[\Phi_2 + \rho_2] \leq 1 + 2M\ln k$, proving the lemma.

THEOREM 3.1. *The competitive ratio is $R = O(\log k)$ provided that $\alpha \geq 1 + 2\ln k$.*

Proof. The proof is by induction on the number of levels in the tree. The base case is just a star: this is the same as the uniform metric and we apply theorem 2.3. We now consider the inductive step.

By definition of $cost_{OPT}(S_i)$ and definition of M, $cost_{OPT}(T) \geq \sum_{i=1}^{x} cost_{OPT}(S_i) + 2(\beta+1)\delta M$

Define S_i^* as $S_i \cup M_i^*$,where the distance from each left-hand node in M_i^* to the root of S_i is replaced with 0. That is, S_i^* includes all left-hand nodes that we match to right-hand nodes in S_i at the root of the subtree. Since the distance from each left-hand node in M_i^* to the root of S_i is bounded by $(\beta+1)\delta$, we can write:

$$cost_{US}(T) \leq \sum_{i=1}^{x} cost_{US}(S_i^*) + \sum_{i=1}^{x} (2\delta + \beta\delta)\, m_i^*$$

By the inductive assumption, $cost_{US}(S_i^*) \leq R * cost_{OPT}(S_i^*)$. Since OPT could use the solution of S_i to solve S_i^*, matching the other m_i^* nodes in M_i^* to whichever right-hand nodes remain unmatched, $cost_{OPT}(S_i^*) \leq cost_{OPT}(S_i) + \beta\delta m_i^*$. So, $cost_{US}(S_i^*) \leq R(cost_{OPT}(S_i) + \beta\delta m_i^*)$. Now we have $cost_{US}(T)$ is bounded by:

$$(3.1)\quad R\sum_{i=1}^{x} cost_{OPT}(S_i) + (R\beta\delta + 2\delta + \beta\delta)\sum_{i=1}^{x} m_i^*$$

In order for the induction to hold, we need a guarantee that the quantity in equation 3.1 is no larger than R times the optimum cost. This means we require:

$$cost_{US}(T) \leq R\sum_{i=1}^{x} cost_{OPT}(S_i) + 2R(\beta+1)M\delta$$

Simplifying this expression and applying lemma 3.1 yields:

$$(R\beta + 2 + \beta)(1 + 2M\ln k) \leq 2(\beta+1)MR$$

Suppose that we set $R = 12\ln k$. We observe that $\alpha \geq 1 + 2\ln k$ implies that $\beta \leq \frac{1}{2\ln k}$. We can then guarantee:

$$\begin{aligned} (R\beta + 2 + \beta)(1 + 2M\ln k) &\leq 24M\ln k + 1.5M \\ &\leq 2(\beta+1)MR \end{aligned}$$

This completes the induction.

This analysis of randomized greedy on a hierarchically separated tree is essentially tight. Consider an adversary which designates only one right-hand node which is not a left-hand node, then designates collocated nodes one subtree at a time, starting with the subtree containing the non-collocated node. Such an adversary could pay as little as $\alpha^{\lambda-1}$ in the offline setting, where λ is the number of levels. On the other hand, by repeating the lower bound proof from the star, randomized greedy will pay an expected $(\log k)^{\lambda}$. It follows that we must require $\alpha = \Omega(\log k)$ to obtain any competitive ratio which is independent of the number of levels.

4 Extension to General Metrics

In order to extend our result to general metrics, we first embed the metric randomly into a tree. Once this computation has been performed, we can use randomized greedy on the (randomly generated) tree to obtain the competitive ratio. We first summarize (without proof) the result of Fakcharoenphol et al [3].

THEOREM 4.1. *Given any metric (V, d), we can randomly produce an α-HST metric $T = (V_T, d_T)$ such that the leaves of the tree correspond to the nodes of V. For any two nodes $u, v \in V$ we can guarantee that $d_T(u, v) \geq d(u, v)$ and also that $E[d_T(u, v)] \leq O(\alpha \log |V|)d(u, v)$. Here the expectation is over randomly selected HST metrics.*

Our randomized algorithm for online matching proceeds by first creating such a tree and then executing randomized greedy. The following result is immediate:

THEOREM 4.2. *There is a randomized algorithm for online matching with expected competitive ratio $O(\log n \log^2 k)$. Here n is the number of points in the metric given, and k is the number of left-hand nodes.*

Proof. The optimum matching has expected cost of $\alpha \log n$ times its original cost when transferred to the new HST metric. Thus randomized greedy will produce a matching of expected cost at most $R\alpha \log n$. Using the bounds on R and α from the previous section gives the required bound.

This result is not particularly useful, because of the dependence on the number of nodes in the metric (n). In particular, our initial metric might be infinite. We can improve this result (and eliminate dependence on n) by restricting the metric to only the right-hand nodes given. Our final algorithm functions as follows:

1. Construct a $(2 \ln k + 1)$-HST randomly on the metric formed by the k right-hand nodes given, using the method of [3].
2. Whenever a left-hand node l_i arrives, find its nearest right-hand node in the original graph: ν_i.
3. Find the nearest unmatched right-hand node to ν_i in the HST. If there is a tie, then select one of the tied nodes uniformly at random. Call the selected node r_i.
4. Match $\mu(l_i) = r_i$ and continue.

THEOREM 4.3. *The algorithm has an (expected) competitive ratio bounded by $O(\log^3 k)$.*

Proof. If each arriving left-hand node had $l_i = \nu_i$ the proof would be immediate (we could consider the entire metric to be finite and consist only of the nodes in R). We observe that $\sum_i d(l_i, \nu_i)$ is a lower bound on the optimum cost. Let OPT' be the cost to match the nodes ν_i to the r_i; the triangle inequality implies that $OPT' \leq 2OPT$ (match each node ν_i first to l_i and then to the appropriate r_i; this cost must exceed OPT', but cannot exceed $2OPT$). Thus the expected total distance between the ν_i and $\mu(\nu_i)$ constructed by our algorithm is bounded by $O(\log^3 k)$ times optimum. Again using the bound on $d(l_i, \nu_i)$ (and applying triangle inequality again) gives the required bound.

References

[1] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. *37th IEEE Symposium on Foundations of Computer Science*, 1996.

[2] Y. Bartal. On approximating arbitrary metrics by tree metrics. *30th ACM Symposium on Theory of Computing*, 1998.

[3] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *ACM Symposium on Theory of Computing*, 2003.

[4] B. Fuchs, W. Hochstattler, and W. Kern. Online matching on a line. *Theoretical Computer Science 332(1-3)*, 2005.

[5] B. Kalyanasundaram and K. Pruhs. Online weighted matching. *Journal of Algorithms 14(3) (also SODA 1991)*, 1993.

[6] B. Kalyanasundaram and K. Pruhs. Online transportation problem. *Journal of Discrete Mathematics 13(3) (also ESA 1995)*, 2000.

[7] R. Karp, U. Vazirani, and V. Vazirani. An optimal algorithm for online bipartite matching. *22nd ACM Symposium on Theory of Computing*, 1990.

[8] S. Khuller, S. Mitchell, and V. Vazirani. On-line algorithms for weighted bipartite matching and stable marriages. *Theory of Computer Science 127(2)*, 1994.

[9] E. Koutsoupias and A. Nanavati. The online matching problem on a line. *Workshop on Approximation and Online Algorithms*, 2003.

[10] E. Koutsoupias and C. Papadimitriou. On the k-server conjecture. *Journal of the ACM 42(5)*, 1995.