Randomized Primal-Dual Analysis of RANKING for Online Bipartite Matching

Nikhil R. Devanur* Kamal Jain† Robert D. Kleinberg‡

Abstract

We give a simple proof that the RANKING algorithm of Karp, Vazirani and Vazirani [KVV90] is 1-1/e competitive for the online bipartite matching problem. The proof is via a randomized primal-dual argument. Primal-dual algorithms have been successfully used for many online algorithm problems, but the dual constraints are always satisfied deterministically. This is the first instance of a non-trivial randomized primal-dual algorithm in which the dual constraints only hold in expectation. The approach also generalizes easily to the vertex-weighted version considered by Agarwal et al. [AGKM11]. Further we show that the proof is very similar to the deterministic primal-dual argument for the online budgeted allocation problem with small bids (also called the AdWords problem) of Mehta et al. [MSVV05].

1 Introduction

The online bipartite matching problem is as follows. An instance of the problem is a bipartite graph $G = (L, R, E)$ and the objective is to find a matching of greatest cardinality in the graph. The "online" nature of the problem is that the graph is revealed over time: in each step a vertex j in R *arrives* and all the edges incident to it are revealed. The algorithm has to make decisions online as well; that is, the algorithm must decide the neighbor that j is matched to (if any) before the next vertex in R arrives. Matches once made cannot be revoked.

An obvious greedy algorithm for this problem matches each vertex with an arbitrary unmatched neighbor whenever such a choice is possible. This algorithm always succeeds in choosing a matching which is set-wise maximal, and therefore has at least half as many edges as the maximum matching. In a seminal paper in 1990, Karp, Vazirani, and Vazirani [KVV90] presented a randomized algorithm known as RANKING that improves this guarantee, ensuring that in expectation the algorithm chooses at least $1 - \frac{1}{e}$ fraction of the edges in the maximum matching. An easy lower bound construction in [KVV90] shows that no randomized online algorithm can achieve a better worst-case approximation ratio.

The RANKING algorithm is extremely simple — it simply selects a random total ordering of the elements of L, and when matching a newly arrived vertex $j \in R$ to an unmatched neighbor, it selects the one that occurs earliest in this ordering — but the analysis in the original paper by Karp, Vazirani, and Vazirani was surprisingly complicated. Subsequent papers by Goel and Mehta [GM08] and Birnbaum and Mathieu [BM08] simplified the analysis considerably.

In this short paper, we provide what we believe to be the simplest analysis yet of the RANKING algorithm, interpreting it as a randomized online primal-dual algorithm. By laying bare the primal-dual foundations of the RANKING algorithm, our analysis unifies and synthesizes two fundamental strands of research on online matching algorithms, one originating from the analysis of the integral version of the problem and the other from the online fractional matching problem.

1.1 Overview of online matching algorithms

Essentially two kinds of online matching problems studied in the prior literature are within the scope[1] of this paper. The first kind is online *integral* matching and the second kind is online *fractional* matching. The algorithm of Karp, Vazirani, and Vazirani [KVV90] is designed for *unweighted* online integral matching. Here we already know that any deterministic algorithm cannot be better than $\frac{1}{2}$-competitive in the worst case. Simple examples show that for the integral version of the problem, randomization is essential for any competitive ratio above $\frac{1}{2}$. The randomized algorithm of [KVV90],

*Microsoft Research. nikdev@microsoft.com.

†Ebay Research. kamaljain@gmail.com. Part of the work done while the author was at Microsoft Research.

‡Cornell University. rdk@cs.cornell.edu. Research supported in part by NSF awards CCF-0643934, IIS-0905467, and AF-0910940, AFOSR grant FA9550-09-1-0100, a Microsoft Research New Faculty Fellowship, and a Google Research Grant. Part of the work done while the author was visiting Microsoft Research.

[1] For example, online matching with stochastic assumptions is widely studied but lies outside our paper's scope.

also known as RANKING, is essentially the only known randomized approach attaining optimal worst-case performance. The same approach RANKING is also used in [AGKM11] for online vertex-weighted integral matching; that paper essentially summarizes all progress to date on the integral version of the problem, since it encompasses [KVV90] as a special case. The algorithm we analyze in this paper is also exactly the same RANKING algorithm when applied to both the unweighted and vertex-weighted integral matching problems. Thus, the novelty of our approach lies in the analysis alone, not in the algorithm or performance guarantee.

The other type of algorithm, which is more suitable for online fractional matching, was introduced by Kalyanasundaram and Pruhs [KP00]. They framed their problem as the online b-matching problem, in which each node can be matched up to b times, where b is typically large. As $b \to \infty$, their scenario becomes equivalent to online fractional matching. Since the fractional version of the problem can only be easier than the integral version, far more progress has been made for this set of problems. The primary approach for this set of problems, known as WATERLEVEL, is a deterministic approach; for fractional matching problems, randomization does not offer any additional power. The WATERLEVEL algorithms maintain a level (or potential, or time, the choice of terminology is immaterial) for each node on the offline side, L. When a new node on the online side, R, arrives, it is brought into the algorithm in small pieces (either as an assumption or within the flexibility of fractional assignment). The WATERLEVEL algorithms then apply an appropriate function of two inputs on each node of the offline side. One input is the existing level of the node and the other input is the weight of the edge between the node and the arriving node. The arriving node is assigned to whichever neighbor maximizes the value of this function. The essential difficulty in this set of problems thus lies in constructing the functions that combine the level of the node and the weight of the edge.

The algorithms of [MSVV05] and [BJN07] follow the same pattern, as does the algorithm of [KP00]. When the algorithms of [MSVV05] and [BJN07] are restricted to the unweighted version, they become equivalent to [KP00] (and not [KVV90]). The algorithm of [BJN07] showed how to systematically obtain the requisite two-variable function using the primal-dual schema. This systematic approach then turned out to be useful in the algorithm of [DJ12], which is currently the state of the art for online fractional matching problems.

As discussed, there are two streams of problems, integral and fractional. There are likewise two sets of approaches: RANKING for the integral version and WATERLEVEL (= PRIMAL-DUAL) for the fractional version. So far these two problems and the two approaches have been mostly disconnected, despite their similarity and the intriguing coincidence that the optimal competitive ratio for both problems is $1 - \frac{1}{e}$. A big open problem in the area which generalizes both [KVV90] and [MSVV05] is the online (integral) budgeted allocation problem. We make concrete progress in this paper by unifying the two approaches, which we believe will be useful in attacking the general problem.

1.2 Our contribution in a nutshell We provide a reinterpretation of the RANKING algorithm as a randomized Primal-Dual algorithm. Under this interpretation, the algorithm combines the Primal-Dual fractional matching schema of [BJN07, DJ12] with a novel dual-based online randomized rounding step.[2] The technique for constructing primal and dual solutions bears a strong resemblance to the fractional matching algorithms in [BJN07, DJ12], with a crucial twist: our randomized algorithm does not construct a feasible dual solution; instead it outputs a random dual vector that is feasible *in expectation*. Relaxing dual feasibility to hold only in expectation is vital to the goal of constructing the integer primal and fractional dual solutions jointly in an online fashion.

To describe the algorithm in a bit more detail, it is necessary to recall the basic Primal-Dual fractional matching framework of [BJN07]. That algorithm keeps track, for each vertex $i \in L$ on the offline side, of the total fractional weight y_i that has been assigned to i. One can visualize y_i as a "water level" associated to i, initially set to zero. The fractional allocation of vertex j is computed by continuously "filling water" into the neighboring vertices with the lowest water level, until either one unit has been allocated in total or the water level in every neighboring vertex reaches 1. While computing this fractional primal solution, the algorithm also associates a dual value to each vertex which depends on its own water level (if the vertex belongs to the offline side) or the water level of the neighboring vertices (if it belongs to the online side). The function g that relates water levels to dual values is obtained by solving an integral equation that is carefully chosen to ensure dual feasibility.

Our randomized Primal-Dual interpretation of the RANKING algorithm can also be visualized as a water-filling process, but with a different stopping condition. Instead of stopping the process deterministically when one unit has been assigned, each vertex i has a uniformly

[2] It is worth mentioning that, unlike many other randomized rounding procedures, ours does not output a solution whose expected value is a scaled copy of the given fractional solution.

random threshold Y_i and the water-filling process for node j stops the first time the water level reaches the threshold of an unmatched neighbor i. At that time, the edge (i, j) is added to the matching and the dual variables corresponding to vertices i and j are set according to the water level at i, using the same function g as in the fractional matching algorithm. The dual variables of the other unmatched neighbors of j remain at zero, potentially leading to dual infeasibility. We are able to prove, however, that the expected values of the dual variables constitute a feasible dual solution, by appealing to the same integral equation that underpins the Primal-Dual analysis of the online fractional matching algorithm.

It is worth pointing out that in our Primal-Dual interpretation of the RANKING algorithm, the randomized procedure for computing the edge-set of the matching is integrated into the online Primal-Dual algorithm that computes the fractional primal and dual solutions. This is in contrast to simpler approaches that treat the fractional matching algorithm as a black box and apply online randomized rounding to the output of the fractional algorithm. Since the online fractional matching algorithms of [KP00, BJN07, DJ12] are $(1 - \frac{1}{e})$-competitive, and since the fractional matching that they output can be represented as a convex combination of integral matchings, it is tempting to hope that there is an online randomized rounding procedure that constructs an integral matching whose expected size equals the weight of the fractional matching computed by the WATERLEVEL algorithm. In fact, there are simple examples[3] that demonstrate that this is impossible, for instance an 8-cycle in which the fractional solution places a value of $\frac{1}{2}$ on each edge and the first two arriving vertices are diametrically opposite one another.

Our equivalent formulation of RANKING opens itself up to potential generalizations, e.g., the generalization to the vertex-weighted matching algorithm of [AGKM11] is quite easy and natural. Our Dominance and Monotonicity Lemmas — Lemmas 2.2 and 2.3 — are stated within the framework of duals, but the essence of these lemmas is also present in [KVV90] (though not the lemmas themselves as stated).[4] Much closer versions of these lemmas are also present in [GM08] and [BM08], both of which give alternate proofs of [KVV90]. Finally, we note that a blog post [Mat11] by Claire Mathieu, who is also the second author of [BM08], gives an (informal but complete) explanation of our framework.

[3]The earliest such example that was presented to us is due to Nick Harvey [Har06].

[4]The [KVV90] analysis was based on showing that the upper triangular graph is the worst case graph.

2 Algorithm and Analysis

We begin with a reinterpretation of the RANKING algorithm, in a way that is conducive to our analysis. Instead of picking a random total ordering of the vertices in L, each vertex in L picks a random number in $[0, 1]$ and a vertex $j \in R$, upon its arrival, is assigned to the unmatched neighbor who picked the lowest number. The algorithm is presented as Algorithm 1 below.

Algorithm 1: The RANKING algorithm.

```
foreach i ∈ L do
    Pick Y_i ∈ [0,1] uniformly at random
foreach j ∈ R do
    When j arrives, let N(j) denote the set of
    unmatched neighbors of j;
    if N(j) = ∅ then
        j remains unmatched
    else
        Match j to arg min{Y_i : i ∈ N(j)}
```

To analyze the algorithm, we note the standard LP relaxation for matching and its dual.

$$\begin{aligned}
&\text{maximize} \sum_{(i,j)\in E} x_{ij} \text{ s.t.} \\
&\forall\ i \in V, \sum_{j:(i,j)\in E} x_{ij} \leq 1. \\
&\forall\ (i,j) \in E, x_{ij} \geq 0.
\end{aligned}$$

$$\begin{aligned}
&\text{minimize} \sum_{i\in L} \alpha_i + \sum_{j\in R} \beta_j \\
&\text{s.t. } \forall\ (i,j) \in E, \alpha_i + \beta_j \geq 1. \\
&\forall\ i, j, \alpha_i, \beta_j \geq 0.
\end{aligned}$$

Our analysis constructs a dual solution which is also randomized. The duals we construct may not always be feasible. The competitive ratio of $F \in [0, 1]$ will follow from the fact that the value of the dual solution is always a factor $1/F$ of the size of the matching found, and that the expectation of the duals is feasible. This is summarized in the following lemma.

LEMMA 2.1. *Suppose that a randomized primal-dual algorithm has a primal feasible solution with value P (which is a random variable) and a dual solution which is not necessarily feasible, with value D (which is also a random variable) such that*

1. *for some universal constant F, $P \geq FD$, always, and*

2. *the expectations of the randomized dual variables form a feasible dual solution.*

The expectation of P is then at least $F \cdot \mathrm{OPT}$ where OPT *is the value of the optimum solution.*

Proof. Since $P \geq FD$ always, taking expectations, $\mathbf{E}[P] \geq F \cdot \mathbf{E}[D]$. The cost of the dual solution obtained by taking the expectations of the randomized dual variables is $\mathbf{E}[D]$ and they form a feasible dual solution, therefore $\mathbf{E}[D] \geq \mathrm{OPT}$. Hence $\mathbf{E}[P] \geq F \cdot \mathrm{OPT}$.

Our construction of the duals depends on a monotone non-decreasing function $g : [0,1] \to [0,1]$. We later identify other properties of g that we need in order to prove a competitive ratio of F. Whenever i is matched to j, let

$$\alpha_i = g(Y_i)/F, \qquad \beta_j = (1 - g(Y_i))/F.$$

For all unmatched i and j, set $\alpha_i = \beta_j = 0$. It will be useful to interpret the algorithm as follows: on matching i to j, we generate a value of 1 for the primal, which translates to a value of $1/F$ for the dual. Each unmatched vertex $i \in L$ that is a neighbor of j offers $(1 - g(Y_i))/F$ of this value to j (to be assigned to β_j), while keeping the rest to itself (to be assigned to α_i). Then j is matched to the vertex that makes the highest offer.

Before we show that the expectation of the duals is feasible, we need certain properties of the algorithm specified by the following two lemmas. These properties are well-known and form the basis of all the earlier proofs. Let $(i,j) \in E$ be any edge in the graph. Consider an instance of the algorithm on $G \setminus \{i\}$, with the same choice of $Y_{i'}$ for all other $i' \in L$. Let y^c be the value of $Y_{i'}$ for the i' that is matched to j. Define y^c to be 1 if j is not matched. Let β_j^c be the value of β_j in this run. We further impose that $g(1) = 1$, which implies $\beta_j^c = (1 - g(y^c))/F$.

Lemma 2.2. (Dominance Lemma) *Given $Y_{i'}$ for all other $i' \in L$, i gets matched if $Y_i < y^c$.*

Proof. Suppose i is not matched when j arrives. This means that the run of the algorithm until then is identical to the run without i. From the definition of y^c, in the run without i, j is matched to i' such that $Y_{i'} = y^c$. Since $Y_i < y^c$, j is matched to i.

Lemma 2.3. (Monotonicity Lemma) *Given $Y_{i'}$ for all other $i' \in L$, for all choices of Y_i, $\beta_j \geq \beta_j^c$.*

Proof. Consider executing the algorithm on graphs G and $G \setminus \{i\}$ in parallel. At the start of every step of the two parallel executions, the unmatched vertices in L for the G execution constitute a superset of the unmatched vertices in L for the $G \setminus \{i\}$ execution. This statement is easily proven by induction: given that it holds at the start of one step, the only way it could be violated at the start of the next step is if the G execution chooses a vertex $i' \in L$ that is also unmatched, but is not chosen, in the $G \setminus \{i\}$ execution. Instead the $G \setminus \{i\}$ execution must choose some other vertex i'' such that $Y_{i''} < Y_{i'}$. By our induction hypothesis i'' was also unmatched in the G execution, contradicting the fact that the algorithm chose i' instead.

When node j arrives, its unmatched neighbors in the G execution form a superset of its unmatched neighbors in the $G \setminus \{i\}$ execution, so in both the executions j has an unmatched neighbor whose Y-value is y^c. If the algorithm instead chooses another neighbor of j, its Y-value can be at most y^c and hence, by the monotonicity of g, we have $\beta_j \geq \beta_j^c$.

We now show that for any g that satisfies a certain integral equation, the above properties imply a competitive ratio of F for RANKING. This integral equation is also at the heart of the deterministic primal-dual analysis for the fractional matching problem. We will later give a short proof of this as well, in Section 3.2.

Lemma 2.4. *If g and F are such that*

$$\forall\ \theta \in [0,1] \int_0^\theta g(y)\,dy + 1 - g(\theta) \geq F, \tag{2.1}$$

then the duals constructed are feasible in expectation.

Proof. We show that for all $(i,j) \in E$,

$$\mathbf{E}_{Y_i}[\alpha_i + \beta_j] \geq 1$$

for all choices of $Y_{i'}$ for all $i' \neq i \in L$. By the Dominance Lemma (Lemma 2.2) i is matched whenever $Y_i \leq y^c$. Hence

$$\mathbf{E}_{Y_i}[\alpha_i] \geq \int_0^{y^c} g(y)\,dy/F.$$

By the Monotonicity Lemma (Lemma 2.3), $\beta_j \geq \beta_j^c = (1 - g(y^c))/F$ for all choices of Y_i. The lemma now follows from condition (2.1) in the hypothesis.

It is easy to solve the integral equation (2.1) along with the boundary condition $g(1) = 1$ to get an explicit function g. (2.1) does not have a solution satisfying $g(1) = 1$ for all values of F; one can also calculate the largest value of F for which it does have a solution, which turns out to be $1 - 1/e$.

Theorem 2.1. RANKING *is $1 - 1/e$ competitive.*

Proof. Whenever i is matched to j, $\alpha_i + \beta_j = 1/F$. Therefore the ratio of the primal solution to the dual is always F. One may verify that the function $g(y) = e^{y-1}$ and $F = 1 - \frac{1}{e}$ satisfy the condition in Lemma 2.4, therefore the lemma implies that the duals are feasible in expectation. The hypothesis in Lemma 2.1 is hence satisfied with $F = 1 - \frac{1}{e}$ and the theorem follows.

3 Extensions

In this section we show how our approach easily generalizes to the vertex-weighted version of the problem. We also give an analysis of the deterministic primal-dual algorithm for the fractional matching problem (and its generalization, the online budgeted allocation problem) that highlights the similarity to the foregoing analysis of the RANKING algorithm.

3.1 The Vertex-Weighted Case In the vertex-weighted version of the problem, each vertex $i \in L$ has a weight v_i and the objective function is the sum of weights of matched vertices in L. Agarwal et al. [AGKM11] gave a $1 - \frac{1}{e}$ competitive algorithm for this problem, generalizing the RANKING algorithm. The analysis presented in the previous section extends easily to their algorithm as well, as we shall see in this section.

Algorithm 2: Vertex-weighted version of the RANKING algorithm.

```
foreach i ∈ L do
    Pick Y_i ∈ [0,1] uniformly at random
foreach j ∈ R do
    When j arrives, let N(j) denote the set of
    unmatched neighbors of j;
    if N(j) = ∅ then
        j remains unmatched
    else
        Match j to
        arg max{v_i(1 − g(Y_i)) : i ∈ N(j)}
```

Algorithm 2 presents the modification of the RANKING algorithm for the vertex-weighted case. The only change is that we now select the neighbor of j that maximizes $v_i(1 - g(Y_i))$, rather than minimizing Y_i. The function g is defined by $g(y) = e^{y-1}$ as in the previous section. (When the weights v_i are identical, maximizing $v_i(1 - g(Y_i))$ is identical to minimizing Y_i since g is monotone increasing.) As before, we analyze the algorithm using the LP relaxation for vertex-weighted matching and its dual, which are respectively as follows.

$$\begin{aligned}
&\text{maximize} \sum_{(i,j)\in E} v_i x_{ij} \\
&\text{s.t. } \forall\, i \in V, \sum_{j:(i,j)\in E} x_{ij} \le 1. \\
&\forall\, (i,j) \in E,\ x_{ij} \ge 0.
\end{aligned}$$

$$\begin{aligned}
&\text{minimize} \sum_{i\in L} \alpha_i + \sum_{j\in R} \beta_j \\
&\text{s.t. } \forall\, (i,j) \in E, \alpha_i + \beta_j \ge v_i. \\
&\forall\, i,j,\ \alpha_i, \beta_j \ge 0.
\end{aligned}$$

Our random dual solution is constructed as follows. For a vertex $i \in L$ define functions

$$a_i(y) = v_i g(y)/F, \qquad b_i(y) = v_i(1 - g(y))/F.$$

Whenever i is matched to j, let $\alpha_i = a_i(Y_i)$, $\beta_j = b_i(Y_i)$. For all unmatched i and j, set $\alpha_i = \beta_j = 0$. As before, one can interpret the duals by envisioning that i makes an offer of $b_i(Y_i)$ to j, and j accepts the highest offer.

In every iteration of the algorithm, the change in the dual objective is always $1/F$ times the change in the primal objective. To finish the analysis of the algorithm, we must prove that the expectation of the dual solution is feasible, i.e. that for all $(i,j) \in E$, $\mathbf{E}[\alpha_i + \beta_j] \ge v_i$.

For an edge (i,j) and for a fixed choice of $(Y_{i'} : i' \in L \setminus \{i\})$, we may ask: for what values y is it the case that running the algorithm with $Y_i = y$ results in matching i either to j or to an earlier vertex? As before, the answer is that there is a *critical value* $y^c \le 1$ such that this happens if $y < y^c$ and not if $y > y^c$. (At $y = y^c$ the answer may depend on tie-breaking.) In the vertex-weighted case, the value of y^c may be determined by running the algorithm on the graph $G \setminus \{i\}$ with the same values of $Y_{i'}$ $(i' \ne i)$. Denoting by i' the vertex that is matched to j in this execution, y^c is the unique value in $[0,1]$ such that $b_i(y^c) = b_{i'}(Y_{i'})$, if such a value exists. If the equation $b_i(y) = b_{i'}(Y_{i'})$ has no solution in $[0,1]$, we let $y^c = 0$. If j remains unmatched when the algorithm is run on $G \setminus \{i\}$, we let $y^c = 1$.

Under this definition of y^c, the Dominance and Monotonicity Lemmas (Lemmas 2.2 and 2.3) remain true, with only the following modifications to the proofs. Anywhere that a relation such as $Y_{i'} = y^c$ or $Y_{i''} < Y_{i'}$ appears, it should instead be rewritten by applying the relevant functions $b_i, b_{i'}$, etc., and reversing the sign of the inequality. Thus, for example, $Y_{i''} < Y_{i'}$ becomes $b_{i''}(Y_{i''}) > b_{i'}(Y_{i'})$, and $Y_{i'} = y^c$ becomes $b_{i'}(Y_{i'}) = b_i(y^c)$. When reasoning about the case when

$y^c = 0$ because $b_i(y) = b_{i'}(Y_{i'})$ has no solution in $[0,1]$ — equivalently, because $b_i(0) < b_{i'}(Y_{i'})$ — the argument is also slightly different. Lemma 2.2 now holds vacuously because the hypothesis $Y_i < y^c$ is never satisfied. Lemma 2.3 holds because in that case $\beta_j^c = b_i(0) < b_{i'}(Y_{i'})$, and the proof of the Lemma shows that $\beta_j \geq b_{i'}(Y_{i'})$.

By the Dominance Lemma, i is matched whenever $Y_i \leq y^c$, hence

$$\mathbf{E}[\alpha_i] \geq \int_0^{y^c} v_i g(y)\, dy/F.$$

By the Monotonicity Lemma, $\beta_j \geq \beta_j^c = v_i(1-g(y^c))/F$ for all choices of Y_i. Hence, making use of integral equation (2.1),

$$\mathbf{E}[\alpha_i + \beta_j] \geq \frac{v_i}{F}\left[\int_0^{y^c} g(y)\, dy + 1 - g(y^c)\right] \geq v_i,$$

which verifies dual feasibility in expectation.

3.2 Fractional Matching and Online Budgeted Allocation (AdWords) The online budgeted allocation problem with small bids (also called the AdWords problem) of Mehta et. al. [MSVV05] is as follows. The weights are on the edges, edge (i,j) has weight v_{ij}. The vertices in L have budget constraints instead of matching constraints, that is, for every $i \in L$, the sum of the weights of the edges matched to it cannot exceed B_i, which is given in the beginning. Notice that this problem generalizes vertex-weighted fractional matching, because the special case of online budgeted allocation in which all edges incident to $i \in L$ have weight v_i, and the budget B_i equals Bv_i for some large integer B, approximates fractional vertex-weighted matching in the limit as $B \to \infty$.

[BJN07] gave a deterministic algorithm with primal-dual analysis proving a competitive ratio approaching $1 - 1/e$ for this problem, as the ratio of edge weights to budgets tends to zero. We now give a short version of this proof, which highlights its similarity to the analysis of the RANKING algorithm in the previous section. The primal and dual linear programs are as follows.

$$\begin{aligned}
&\text{maximize} \sum_{(i,j)\in E} v_{ij}x_{ij} \\
&\text{s.t. } \forall\ i \in L,\ \sum_{j:(i,j)\in E} v_{ij}x_{ij} \leq B_i. \\
&\forall\ j \in R,\ \sum_{i:(i,j)\in E} x_{ij} \leq 1. \\
&\forall\ (i,j) \in E,\ x_{ij} \geq 0.
\end{aligned}$$

$$\begin{aligned}
&\text{minimize} \sum_{i\in L} \alpha_i B_i + \sum_{j\in R} \beta_j \\
&\text{s.t. } \forall\ (i,j) \in E,\ v_{ij}\alpha_i + \beta_j \geq v_{ij}. \\
&\forall\ i,j,\ \alpha_i, \beta_j \geq 0.
\end{aligned}$$

The algorithm is as before, except that a vertex $i \in L$ now offers $v_{ij}(1 - g(y_i))/F$ where y_i is the fraction of i's budget consumed at that point. Vertex j is matched to the highest bidder, β_j is set to $v_{ij}(1 - g(y_i))/F$, and α_i is *incremented* by $v_{ij}g(y_i)/(B_iF)$. Thus when j is matched to i the total increment in the primal is v_{ij} and that in the dual is v_{ij}/F. We only need that the dual is feasible. Since the bid to budget ratio is small, v_{ij}/B_i can be thought of as an infinitesimal increase in y_i, and α_i can be approximated as $\int_0^{y_i} g(y)\, dy/F$. It is now easy to see that dual feasibility follows from equation (2.1) in Lemma 2.4. (We need $g(1) = 1$ to ensure that we don't match to a vertex whose budget is already exhausted.)

Finally, it has been brought to our notice by Aman Dhesi [Dhe12] that the competitive analysis of the greedy algorithm for the (fractional) online budgeted allocation problem when the vertices in R arrive in a random order (giving a competitive ratio of $1 - 1/e$) presented in [GM08] can also be simplified and analyzed in our framework. The difference is that it is the vertices in R that draw a number uniformly at random from $[0,1]$. These numbers are then used to set the duals in a similar way, such that the ratio of primal to dual is always F. The proof is completed by showing that once again the duals are feasible in expectation.

4 Acknowledgements

The second author would like to thank Nick Harvey and Mohit Singh, with whom the second author first started the work of interpreting the algorithm of [KVV90] as an online rouding of the algorithm of [BJN07] in 2006. We would also like to thank Monika Henzinger for pointing out a small technical error in an earlier version of this paper.

References

[AGKM11] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *SODA*, pages 1253–1264, 2011.

[BJN07] Niv Buchbinder, Kamal Jain, and Joseph Seffi Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *ESA'07: Proceedings of the 15th annual European conference on Algorithms*, pages 253–264, Berlin, Heidelberg, 2007. Springer-Verlag.

[BM08] Benjamin E. Birnbaum and Claire Mathieu. Online bipartite matching made simple. *SIGACT News*, 39(1):80–87, 2008.

[Dhe12] Aman Dhesi. Personal communication, 2012.

[DJ12] Nikhil Devanur and Kamal Jain. Online matching with concave returns. In *STOC*, pages 137–144, 2012.

[GM08] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 982–991, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[Har06] Nicholas J. A. Harvey. Personal communication, 2006.

[KP00] Bala Kalyanasundaram and Kirk R. Pruhs. An optimal deterministic algorithm for online b-matching. *Theor. Comput. Sci.*, 233(1-2):319–325, 2000.

[KVV90] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *STOC '90: Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 352–358, 1990.

[Mat11] Claire Mathieu. A primal-dual analysis of the ranking algorithm, June 2011. *A CS Professor's blog*, http://teachingintrotocs.blogspot.com/2011/06/primal-dual-analysis-of-ranking.html.

[MSVV05] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized on-line matching. In *In FOCS 05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 264–273. IEEE Computer Society, 2005.