CS 466/666: Algorithm Design and Analysis **University of Waterloo: Fall 2024**

Lecture 12

October 22, 2024

Instructor: Sepehr Assadi

Topics of this Lecture

1 The Online Matching Problem

We will study another application of the LP duality in the analysis of the algorithms, this time to the **online (bipartite) matching** problem. In this problem, we have a bipartite graph $G = (L, R, E)$ with $|L| = |R| = n$, where vertices in L are considered 'online' and vertices in R are 'offline'. At each time step, a vertex $u \in L$ arrives and only then we get to see all edges of u to R at the same time. Moreover, we need to decide, *irrevocably*, which edge incident on u, if any (or if possible at all), to add to our matching M (while ensuring that M remains a matching, i.e., u cannot be matched to an already matched vertex in R). The objective of the algorithm is to maximize the size of the matching M it returns.

The online matching problem belongs to the large family of **online algorithms** which concerns problems in which information about the input is revealed over a sequence of time steps and the algorithm must make decisions (often irrevocably) in each time step, without knowing about the information in future steps. We often analyze the performance of online algorithms via the notion of **competitive ratio**, namely, the ratio of the quality of the solution returned by the online algorithm, versus the *offline* optimal algorithm that is aware of the the entire input. In the context of the online matching problem, we are interested in maximizing

$$\frac{|M|}{opt(G)},$$

where $opt(G)$ denotes the maximum matching size in G.

1.1 Deterministic Algorithms?

There is a quite simple deterministic algorithm for online matching already: whenever u arrives, match u to any of its unmatched neighbors if available (chosen arbitrarily), and otherwise skip matching u. We call this the *greedy* algorithm.

The greedy algorithm outputs a *maximal matching*, namely, a matching which is not a proper subset of any other matchings. As we proved in the last lecture, this means that $|M| \geqslant (1/2) \cdot opt(G)$, thus, this is a $(1/2)$-competitive algorithm.

Can we do better? *No* (at least, as long as we want a deterministic algorithm). See the example below:

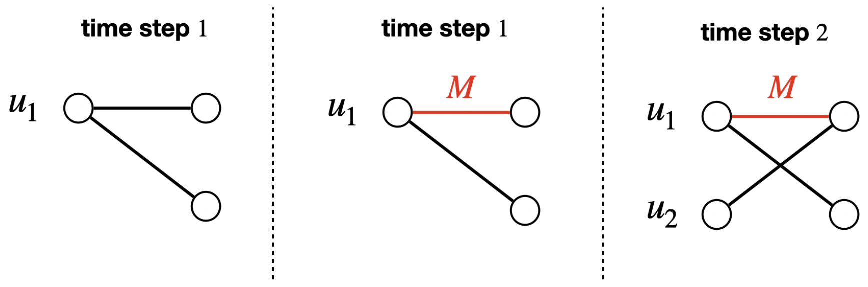

Figure 1: Vertex u_1 arrives first and at this point it is oblivious to which of the two edges it should pick. Whichever vertex v the algorithm chooses to match u_1 with is going to also be the sole neighbor of the vertex u_2 that arrives next. Thus, the algorithm can only get a matching of size 1 while the optimum matching is of size 2. Hence, the competitive ratio of the deterministic algorithms cannot be better than half.

1.2 Randomized Algorithms?

The example we saw above only works against deterministic algorithms. This is because, when talking about randomized algorithms, we need to fix the input sequence (even though we are not revealing it to the algorithm), and then let the algorithm picks its randomness independent of the input[1]. In this case, we will be interested in the **expected competitive ratio** of the algorithm, namely,

$$\frac{\mathbb{E}\,|M|}{opt(G)}.$$

We can think of a very simple randomized strategy to "break" the above example: when a vertex $u \in L$ arrives, match u to one of its unmatched neighbors chosen uniformly at random. Let us call this the *randomized greedy* algorithm. The randomized greedy algorithm now works as follows in the above example:

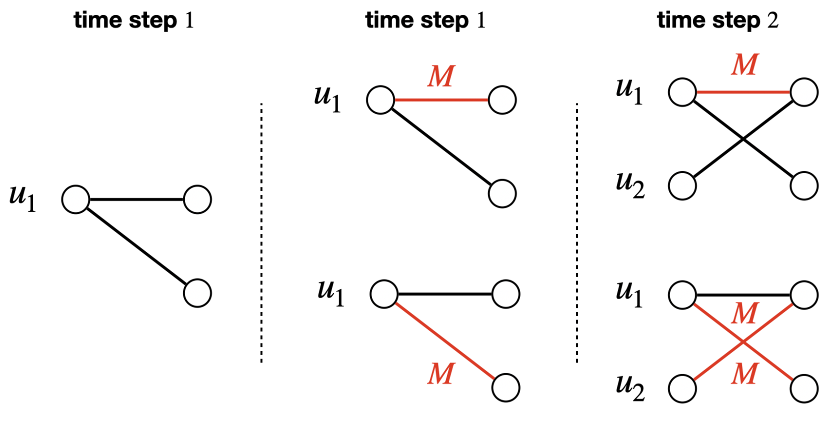

Figure 2: The final graph is now fixed even though the algorithm is oblivious to it. Each of the middle cases happen with probability 1/2, so $\mathbb{E}\,|M| = 1/2 \cdot 1 + 1/2 \cdot 2 = 3/2$, while $opt(G) = 2$. Hence, the expected competitive ratio of randomized greedy on this particular input is 3/4.

Nevertheless, randomized greedy still cannot get better than a $1/2 + o(1)$ competitive ratio. One example is the following:

[1]This is called an *oblivious adversary* assumption; there are other types of adversaries but that is a topic for another time...

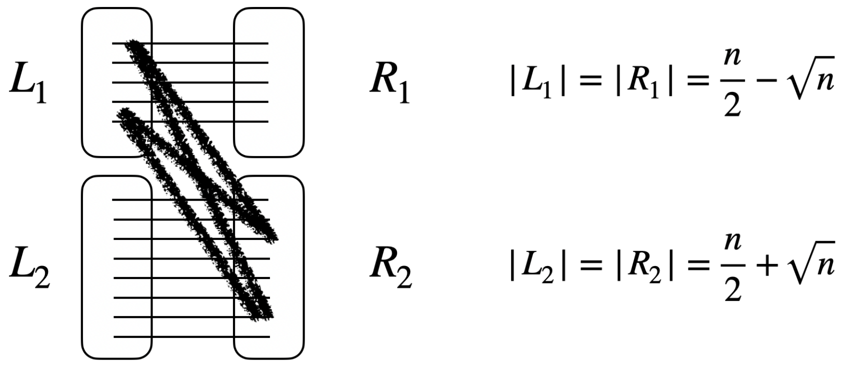
L_1
R_1
$|L_1| = |R_1| = \frac{n}{2} - \sqrt{n}$
L_2
R_2
$|L_2| = |R_2| = \frac{n}{2} + \sqrt{n}$

Figure 3: The thin (normal) edges form a perfect matching between the two components, while the cross thick (hashed) edges is a bi-clique between the two components. Here, L_1 arrives before L_2. Even by the time the last vertex in L_1 arrives and even if all of previous vertices were matched to R_2, still, the probability that this vertex picks the edge to R_1 is at most $1/(\sqrt{n}+1) < 1/\sqrt{n}$. This means that the expected number of vertices that are matched from L_1 to R_1 is only $O(\sqrt{n})$. Hence, the largest matching found by the randomized greedy algorithm in expectation only has size $n/2 + O(\sqrt{n})$, while this graph has a matching of size n. Thus, the expected competitive ratio is only $1/2 + o(1)$.

Can we do better? *Yes* indeed. This is done by *correlating* the random choices of online vertices in L, so that even though *marginally* each one is still picking a random unmatched neighbor, overall, they collectively "value" some vertices in R over the others. We formalize this in the next subsection.

1.3 The Ranking Algorithm

A better strategy than the randomized greedy algorithm is the so-called Ranking algorithm, introduced by Karp, Vazirani, and Vazirani in 1990 [KVV90], who also defined the online matching problem itself (this algorithm is also often called the *KVV-algorithm*). The algorithm is as follows. We give two clearly equivalent interpretation of the algorithm; the first one is easier to implement algorithmically, while the second one is easier to work with analytically.

Algorithm 1. The Ranking Algorithm of [KVV90]

1. Sample a random permutation π of vertices in R

   Alternate way: sample a real number $p_v \in [0,1]$ for every $v \in R$.

2. For each arriving vertex $u \in L$, let $UN(u)$ denote the set of unmatched neighbors of u. If $UN(u) = \emptyset$ skip to the next vertex, otherwise, let $v = \arg\min_{w \in UN(u)} \pi(w)$. Add (u, v) to the matching M.

   Alternate way: let $v = \arg\min_{w \in UN(u)} p_w$.

The following theorem of [KVV90] establishes the expected competitive ratio of Ranking.

Theorem 1 ([KVV90]). *For any online matching instance $G = (L, R, E)$, the matching M output by* Ranking(G) *satisfies* $\mathbb{E}|M| \geqslant (1 - 1/e) \cdot opt(G)$.

Moreover, despite its simplicity, the $(1 - 1/e)$-expected competitive ratio of Ranking is also *optimal* among *all* online algorithms for bipartite matching! In other words, one cannot hope to achieve an algorithm with a better competitive ratio (even regardless of the runtime or simplicity of the algorithm)! While this fact has an easy proof, we will not cover that for now and instead skip directly to the analysis of Ranking and the proof of Theorem 1.

2 A Primal-Dual Analysis of the Ranking Algorithm

The original proof of Theorem 1 in [KVV90] is quite complicated. We will instead cover a different proof for this theorem due to Devanur, Jain, and Kleinberg [DJK13] that uses a primal-dual approach (not quite unlike what we did in the last lecture also) and is considerably simpler.

A quick reminder that the primal and dual LPs for fractional matchings and fractional vertex covers are:

Primal LP (P): Fractional Matching

$$\begin{aligned} \max_{x \in \mathbb{R}^E} \quad & \sum_{e \in E} x_e \\ \text{subject to} \quad & \sum_{e \ni u} x_e \leqslant 1 \qquad \forall u \in L \\ & \sum_{e \ni v} x_e \leqslant 1 \qquad \forall v \in R \\ & x_e \geqslant 0 \qquad \forall e \in E. \end{aligned}$$

Dual LP (D): Fractional Vertex Cover

$$\begin{aligned} \min_{y \in \mathbb{R}^L, z \in \mathbb{R}^R} \quad & \sum_{u \in L} y_u + \sum_{v \in R} z_v \\ \text{subject to} \quad & y_u + z_v \geqslant 1 \qquad \forall (u,v) \in E \\ & y_u, z_v \geqslant 0 \qquad \forall u \in L, v \in R. \end{aligned}$$

Let us define our dual variables. To do so, we first need to pick a function $g : [0,1] \to [0,1]$ with several properties that we will have to fix throughout the proof. At the end, we show how to pick a choice of g that satisfies these properties and is "best" for our purpose[2]. The dual variables are as follows:

- If a vertex $u \in L$ or $v \in R$ is unmatched, we set $y_u = 0$ or $z_v = 0$.
- If a vertex $u \in L$ is matched to a vertex $v \in R$ by M, we set

$$y_u := (1 - g(p_v)) \qquad \text{and} \qquad z_v := g(p_v).$$

Given that our interest is in $\mathbb{E}|M|$ (and not each M specifically), we will also define "expected dual variables"

$$\tilde{y} := \mathbb{E}[y] \qquad \tilde{z} := \mathbb{E}[z],$$

where specifically, we mean that $\tilde{y} \in \mathbb{R}^L$ is such that $\tilde{y}_u := \mathbb{E}[y_u]$ for all $u \in L$ and similarly $\tilde{z} \in \mathbb{R}^R$ is such that $\tilde{z}_v := \mathbb{E}[z_v]$ for all $v \in R$.

The first (easy) lemma establishes the competitive ratio of the algorithm as a function of the dual variables via a simple primal-dual analysis.

Lemma 2. *Suppose there exists a $\beta \in (0,1)$ such that for every edge $(u,v) \in E$, $\tilde{y}_u + \tilde{z}_v \geqslant \beta$. Then,*

$$\mathbb{E}|M| \geqslant \beta \cdot opt(G).$$

Proof. For any fixed choice of y, u, we have,

$$\begin{aligned} |M| &= \sum_{(u,v) \in M} 1 = \sum_{(u,v) \in M} y_u + z_v && \text{(as } y_u + z_v = (1 - g(p_v)) + g(p_v) = 1\text{)} \\ &= \sum_{u \in L} y_u + \sum_{v \in R} z_v. && \text{(as } y_u = 0 \text{ and } z_v = 0 \text{ for every unmatched } u \text{ and } v\text{)} \end{aligned}$$

This implies that, by linearity of expectation,

$$\mathbb{E}|M| = \sum_{u \in L} \tilde{y}_u + \sum_{v \in R} \tilde{z}_v.$$

[2]Well, if you cannot (or do not want to) wait, the function is $g(x) = e^{x-1}$, but why we pick this random-looking function can only become clear after going through the proof.

On the other hand, by the assumption in the lemma statement, we have that for every edge $(u, v) \in E$,

$$\frac{1}{\beta} \cdot (\tilde{y}_u + \tilde{z}_v) \geqslant 1.$$

This means that the vectors $\frac{1}{\beta} \cdot \tilde{y}$ and $\frac{1}{\beta} \cdot \tilde{z}$ together form a feasible dual solution. Since the dual is a minimization LP, we have,

$$\frac{1}{\beta}\left(\sum_{u \in L} \tilde{y}_u + \sum_{v \in R} \tilde{z}_v\right) \geqslant \mathit{opt}_D \geqslant \mathit{opt}_P = \mathit{opt}(G),$$

where opt_D is the optimal dual value, $\mathit{opt}_P = \mathit{opt}(G)$ is the optimal primal value, and the inequality is by the weak duality. Putting all these together implies that

$$\mathbb{E}\,|M| \geqslant \beta \cdot \mathit{opt}(G),$$

as desired. □

The main part of the analysis is then to show that there exists a "large enough" β such that $\tilde{y}, \tilde{z}$ is a β-approximate dual solution. I.e.,

Lemma 3. *For every edge $e = (u, v) \in E$, we have,*

$$\tilde{y}_u + \tilde{z}_v \geqslant (1 - 1/e)\,.$$

Before getting to the proof of this (not-so-easy) lemma, we note that combining this lemma with Lemma 2 immediately proves Theorem 1. The main technical part of the proof is to establish Lemma 3. For the rest of the proof, fix any edge $(u, v) \in E$.

The key idea of the proof is to separate the randomness of p_v from $p^{-v} := \{p_w \mid w \in R \setminus \{v\}\}$ and further "couple" the performance of the algorithm on (G, p) with that of (G^{-v}, p^{-v}) where $G^{-v} := G \setminus \{v\}$. We now formalize this. Define:

- Ranking(G, p): as the *deterministic* algorithm obtained by running Ranking on the graph G with p-values fixed by p.
- $M_{<u}$: as the matching M computed by Ranking(G, p) before the vertex u arrives.
- Ranking(G^{-v}, p^{-v}): as the *deterministic* algorithm obtained by running Ranking on the graph G^{-v} with p-values fixed by p^{-v}.
- $M^{-v}_{<u}$: as the matching M computed by Ranking(G^{-v}, p^{-v}) before the vertex u arrives.

For any fixed choice of p^{-v}, consider running Ranking(G^{-v}, p^{-v}). Let w to be the matched pair of u in this run (w might be $\emptyset$ also). We refer to w as the *hypothetical match* of u and denote it by $hM(u)$. If $w = \emptyset$, then we define $p_{hM(u)} = 1$. This also means we need to fix one property of g:

◇ **Property 1:** $g(1) = 1$.

This is so that $(1 - g(p_{hM(u)}))$, namely, the "hypothetical dual" value becomes 0 in the case when $w = \emptyset$.

We use these definitions to prove the following two claims. The first one lower bounds $\tilde{z}_v$ and the next one lower bounds $\tilde{y}_u$. Recall that we have fixed the edge (u, v) for all of this proof.

Claim 4. *For any choice of p^{-v},*

$$\mathbb{E}\left[z_v \mid p^{-v}\right] \geqslant \int_0^{p_{hM(u)}} g(x)\, dx.$$

Proof. Let $w := hM(u)$ in the run of $\textsc{Ranking}(G^{-v}, p^{-v})$. Firstly, we claim that if we choose $p_v < p_w$, then v will be matched in $\textsc{Ranking}(G, p)$. Consider running $\textsc{Ranking}(G, p)$ up until the vertex u arrives. If v is already matched, we are done. Otherwise, if v is unmatched, then even if v was not part of the graph, we will be computing the same exact matching, i.e., $M_{<u} = M^{-v}_{<u}$. Now, in $\textsc{Ranking}(G^{-v}, p^{-v})$, we have decided to match u to w which means the smallest unmatched p-value in the neighborhood of u under $M^{-v}_{<u}$ is p_w. But now that $M_{<u} = M^{-v}_{<u}$ and $p_v < p_w$, we have that the smallest unmatched p-value in the neighborhood of u under $M_{<u}$ is p_v, meaning that in $\textsc{Ranking}(G, p)$, the algorithm picks the edge (u, v) to be included in M, thus matching v.

For any $x \in [0, 1]$, let $\Pr(p_v = x)$ denote the probability density function of p_v chosen randomly from $[0, 1]$ at the point x. The rest of the proof now simply follows because,

$$
\begin{aligned}
\mathbb{E}\left[z_v \mid p^{-v}\right] &\geqslant \int_0^1 \Pr\left(p_v = x \ \wedge \ v \in M \mid p^{-v}\right) g(x)\, dx \\
&\qquad\qquad \text{(because if } p_v = x \text{ and } v \text{ gets matched, we set } z_v = g(p_v) = g(x)) \\
&\geqslant \int_0^{p_w} \Pr\left(p_v = x \ \wedge \ v \in M \mid p^{-v}\right) g(x)\, dx \qquad \text{(by non-negativity of } g) \\
&= \int_0^{p_w} \Pr\left(p_v = x \mid p^{-v}\right) \cdot g(x)\, dx \qquad \text{(as } v \text{ will be matched whenever } p_v < p_w) \\
&= \int_0^{p_w} g(x)\, dx \qquad \text{(as we pick } p_v \text{ randomly from } [0,1] \text{ even conditioned on } p^{-v})
\end{aligned}
$$

This implies the claim. ☐

Claim 5. *For any choice of p^{-v}, deterministically,*

$$y_u \geqslant 1 - g(p_{hM(u)}).$$

Proof. We define $U(M_{<u}) \subseteq R$ be the set of R-vertices *unmatched* by $M_{<u}$ in $\textsc{Ranking}(G, p)$ and similarly, $U(M^{-v}_{<u}) \subseteq R \setminus \{v\}$ be the set of R-vertices unmatched by $M^{-v}_{<u}$ in $\textsc{Ranking}(G^{-v}, p^{-v})$. We claim that

$$U(M_{<u}) \supseteq U(M^{-v}_{<u}).$$

We prove this inductively (see Figure 4 for illustration of this proof). This is true before any vertex arrives as both sets are equal to R. Now suppose this is true up until the arrival of some vertex u'. Let v' be the vertex matched by u' in $M_{<u}$. We know that v' has the smallest p-value in $N(u) \cap U(M_{<u})$. If v' is not in $U(M^{-v}_{<u})$, then, removing v' from $U(M_{<u})$ does not violate the condition. On the other hand, if v' is in $U(M^{-v}_{<u})$, then, since by induction we have $U(M_{<u}) \supseteq U(M^{-v}_{<u})$ until the arrival of u', we have that v' also has the smallest p-value in $U(M^{-v}_{<u})$. Thus, v' will also be removed from $U(M^{-v}_{<u})$, hence, the condition remains true. This proves the induction hypothesis.

We now have that when u arrives and picks an unmatched vertex with the smallest p-value, since $U(M_{<u}) \supseteq U(M^{-v}_{<u})$, the p-value of $M(u)$ will be at most as large as p-value of $M^{-v}(u)$, the latter being $p_{hM(u)}$. Thus, we have,

$$p_{M(u)} \leqslant p_{hM(u)}.$$

We now want a second property from g:

- ◇ **Property 2:** g is monotonically increasing between $[0, 1]$.

With this property, we will then have,

$$y_u = 1 - g(p_{M(u)}) \geqslant 1 - g(p_{hM(u)}),$$

which proves the claim. ☐

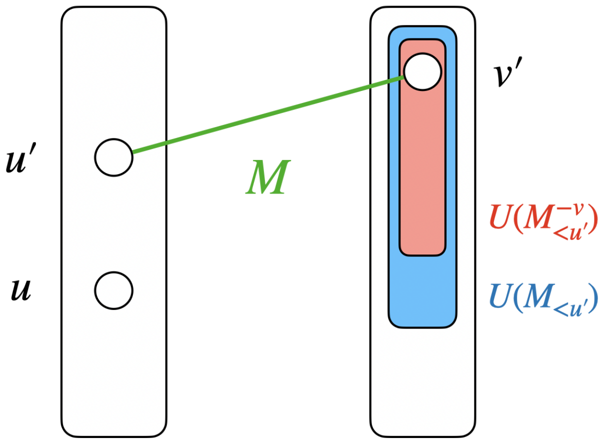

Figure 4: The only time we may violate the induction hypothesis is if upon arrival of a vertex u', we remove a vertex $v' \in U(M_{<u'}) \cap U(M^{-v}_{<u'})$ only from $U(M^{<u})$ and not the other one. But this cannot happen because $U(M_{<u'}) \supseteq U(M^{-v}_{<u'})$ by induction and thus if v' has the smallest p-value in the neighborhood of u' in $\textsc{Ranking}(G,p)$, it also has the smallest p-value in the same neighborhood in $\textsc{Ranking}(G^{-v},p^{-v})$ also and thus will be matched to u' here as well.

We are now ready to wrap up the proof of Lemma 3. So far, we established that for every edge $(u,v) \in E$ and every choice of p^{-v},

$$\mathbb{E}\left[y_u \mid p^{-v}\right] + \mathbb{E}\left[z_v \mid p^{-v}\right] \geqslant 1 - g(p_{hM(u)}) + \int_0^{p_{hM(u)}} g(x)\,dx.$$

This implies that

$$\begin{aligned}
\tilde{y}_u + \tilde{z}_v = \mathbb{E}\left[y_u + z_v\right] &= \mathop{\mathbb{E}}_{p^{-v}}\left[\mathop{\mathbb{E}}_{p_v}\left[y_u + z_v \mid p^{-v}\right]\right] && \text{(by the law of conditional expectation)}\\
&\geqslant \mathop{\mathbb{E}}_{p^{-v}}\left[1 - g(p_{hM(u)}) + \int_0^{p_{hM(u)}} g(x)\,dx\right] && \text{(by the equation above)}\\
&\geqslant \min_{\theta\in[0,1]}\left(1 - g(\theta) + \int_0^{\theta} g(x)\,dx\right).
\end{aligned}$$

(as we are replacing $p_{hM(u)}$ with the choice θ that minimizes the RHS)

In order for us to be able to work with this g (and in fact even earlier we needed a property that made sure g is integrable for our purpose), we will require yet another property:

- ◇ **Property 3:** g is convex over $[0,1]$ and is differentiable.

Now that we have g is convex, we can compute the min-term by differentiating and setting it equal to 0:

$$\frac{\partial\left(1 - g(\theta) + \int_0^{\theta} g(x)\,dx\right)}{\partial\theta} = -g'(\theta) + g(\theta) = 0,$$

which implies that $g(\theta) = g'(\theta)$ at the minimum choice for θ. While there are still many choices for g at this point, we can ask for even a stronger requirement, namely, that $g(x) = g'(x)$ for all $x \in [0,1]$ (and not only for the minimizer θ). This now forces $g(x) = e^{x+c}$ for some constant c. Note that this satisfies property 2 from earlier also. Finally, to satisfy property 1, we will get $g(1) = e^{1+c} = 1$ which means $c = -1$. This implies that if we pick

$$g(x) = e^{x-1},$$

we satisfy all the properties we wanted.

But then we will have,

$$\tilde{y}_u + \tilde{z}_v \geqslant \min_{\theta \in [0,1]} 1 - e^{\theta-1} + e^{\theta-1} - e^{-1} = (1 - 1/e)\,.$$

This proves Lemma 3, and concludes the whole proof.

References

[DJK13] Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized primal-dual analysis of RANKING for online bipartite matching. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 101–107. SIAM, 2013. 4

[KVV90] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 352–358. ACM, 1990. 3, 4