Online hypergraph matching: hiring teams of secretaries

Rafael M. Frongillo
Advisor: Robert Kleinberg

May 29, 2008

1 Introduction

The goal of this paper is to find a competitive algorithm for the following problem. We are given a hypergraph $G = (X, E)$, $|E| = n$, with weighted edges and maximum edge size k. We wish to maximize the weight of a matching $M \subseteq E$ of G, given that edges are revealed in a random order, and that when an edge e is revealed the algorithm must decide either $e \in M$ or $e \notin M$.

Note that this is a generalization of the classic secretary problem; given a set $\{w_i\}$ of weights for the candidates in the secretary problem, we create a hypergraph with one vertex v and n edges, where the ith edge $e_i = \{v\}$ has cardinality 1 and weight w_i. Clearly a matching can be only one edge e_i, corresponding to choosing candidate i. In fact, when $k = 1$, the hypergraph matching problem is just a collection of independent secretary problems, one for each vertex.

2 Algorithm

The algorithm we present is very simple. For some $a \in [0, 1]$ to be determined, the algorithm does nothing while the first $a \cdot n$ edges are revealed. After that, a new edge is selected if it is in the maximum matching among all edges seen so far, and does not intersect any edge already selected. Formally, this is the

procedure given as Algorithm 1, where $\mathsf{maxmat}(E')$ is the maximum weight matching of any edge set E'.

Algorithm 1 The algorithm.

$\tilde{E} \leftarrow \emptyset$
$M \leftarrow \emptyset$
for $t = 1$ to an **do**
  pick e from $E \setminus \tilde{E}$ uniformly at random
  $\tilde{E} \leftarrow \tilde{E} \cup \{e\}$
end for
for $t = an + 1$ to n **do**
  pick e from $E \setminus \tilde{E}$ uniformly at random
  $\tilde{E} \leftarrow \tilde{E} \cup \{e\}$
  if $e \in \mathsf{maxmat}(\tilde{E})$ **and** $\Gamma(e) \cap M = \emptyset$ **then**
    $M \leftarrow M \cup \{e\}$
  end if
end for

3 Analysis

3.1 Definitions

Let E_{t_0} be a random variable representing the value of $\tilde{E}$ in the algorithm when $t = t_0$; in other words, E_t is the first t edges revealed. Similarly, e_t is the edge revealed at time t. We will denote the weight of a matching M as $w(M)$. Finally, using the definition of maxmat from the previous section, define $M_t = \mathsf{maxmat}(E_t)$ as simply the max matching the algorithm sees at time t. Note that M_n is the maximum matching overall; we therefore are striving to come within a constant factor of $w(M_n)$.

Some graph-theoretic shorthand will be useful; we denote the edges neighboring of v in G by $\Gamma(v) = \{e \in E | v \in e\}$. Similarly, the neighbors of an edge $\Gamma(e) = \bigcup_{v \in e} \Gamma(v)$ are all edges that share an endpoint with (i.e. intersect) e. We now define some more complicated random variables that describe the behavior of the algorithm.

Definition 1. *We say t is a* critical time *for v if v is on the edge revealed at time t, and that edge is in the matching M_t, which is the max matching*

of all edges seen so far. Let $\mathsf{crit}(v,t)$ *be the event that* t *is critical for* v*; then formally,* $\mathsf{crit}(v,t) := \{e_t \in M_t, v \in e_t\}$*. Similarly,* $\mathsf{crit}(e,t) := \bigcup_{v \in e} \mathsf{crit}(v,t)$ *is event that* t *is critical for* e*. Finally,* $\mathsf{nocrit}(e,T)$ *is the event that no* $t \in T$ *is critical for* e*.*

Definition 2. *Let* $\mathsf{avail}(e,t)$ *be the event that edge* e *is available to be selected by the algorithm at time* t*; by the definition of the algorithm, this means that* e *has not been seen before and no edge adjacent to the endpoints of* e *has been selected.*

Let $\mathsf{sel}(e,t)$ *be the event that the algorithm selects edge* e *at time* t*. Similarly,* $\mathsf{sel}(e,T) = \bigcup_{t \in T} \mathsf{sel}(e,t)$ *is the event that* e *is picked in a set* T *of times.*

3.2 Proof

To understand the behavior of the algorithm, we first ask how often a node v changes its 'assignement' from one edge to another. Note that for our reduction from the original secretary problem, the unconditional version of this bound is exactly the bound for how often a better candidate appears.

Lemma 3. *Let* e *be given and let* $v \in e$*. Then for all* $s < t$*,*

$$\Pr[\mathsf{crit}(v,s) | e \in M_t \cap e_t = e] \leq \frac{1}{s},$$

Proof. Let $e(F,v) = \mathsf{maxmat}(F) \cap \Gamma(v)$ be the (possibly empty) set containing the edge adjacent to v in the max matching of the edge set F. Note that the event $e \in M_t \cap e_t = e$ can be determined soley from E_{t-1} and e. Thus, for some edge set F let $q(F,e)$ be the predicate which is true when $e \notin F$ and $e \in \mathsf{maxmat}(F \cup e)$, and let $Q(F,e)$ be the even that $q(F,e)$ is true and $e_t = e$. Then by this construction $Q(E_{t-1}, e) = e \in M_t \cap e_t = e$. Now let F_s and F be edge sets with $|F_s| = s$ and $|F| = t-1$. Using this notation, we see

$$\begin{aligned}
&\Pr[\mathsf{crit}(v,s) \mid Q(F,e) \cap E_s = F_s] \\
&\quad = \quad \Pr[e_s \in e(F_s, v) \mid E_{t-1} = F \cap e_t = e \cap E_s = F_s] \\
&\quad = \quad \begin{cases} 0 & \text{if } e(F_s,v) = \emptyset \\ \frac{1}{t} & \text{otherwise} \end{cases} \quad \leq \quad \frac{1}{s},
\end{aligned}$$

since even under the conditions on E_{t-1} and e_t, all orderings of the edges $e_1, \ldots, e_s$ are equally likely. Hence, we have

$$\begin{aligned}
&\Pr[\mathsf{crit}(v,s)|e \in M_t \cap e_t = e] \\
&\quad = \sum_{\substack{F: \\ q(F,e)}} \Pr[\mathsf{crit}(v,s) \mid Q(F,e)] \\
&\quad = \sum_{\substack{F: \\ q(F,e)}} \sum_{F_s \subseteq F} \Pr[\mathsf{crit}(v,s) \mid Q(F,e) \cap E_s = F_s] \Pr[E_s = F_s \mid Q(F,e)] \\
&\quad \leq \sum_{\substack{F: \\ q(F,e)}} \sum_{F_s \subseteq F} \frac{1}{s} \Pr[E_s = F_s \mid Q(F,e)] \\
&\quad = \frac{1}{s} \sum_{F_s:\, e \notin F_s} \Pr[E_s = F_s \mid e_t = e] \;=\; \frac{1}{s}.
\end{aligned}$$

□

Next, we find a constant which bounds the probability that an edge is 'untouched' in a time interval T.

Lemma 4. *If $T = [t_1 + 1, t_2]$ is some time interval and $t \in T$, then*

$$\Pr[\mathsf{avail}(e,t)|e \in M_t \cap e_t = e] \geq \left(1 + k \ln \tfrac{t_1}{t_2}\right).$$

Proof. Recall that k is the maximum size of a hyperedge. First, from the definition of avail, we have

$$\begin{aligned}
&\Pr[\mathsf{avail}(e,t)|e \in M_t \cap e_t = e] \\
&\quad = \Pr[e \notin E_{t_1} \;\cap\; \mathsf{nocrit}(e, [t_1 + 1, t - 1])|e \in M_t \cap e_t = e] \\
&\quad = \Pr[\mathsf{nocrit}(e, [t_1 + 1, t - 1])|e \in M_t \cap e_t = e]\,. \qquad (1)
\end{aligned}$$

By Lemma 3 we have

$$\begin{aligned}
&\Pr[\mathsf{nocrit}(e, T')|e \in M_t \cap e_t = e] \\
&\quad \geq 1 - \sum_{v \in e} \sum_{s=t_1+1}^{t-1} \Pr[\mathsf{crit}(v,s)|e \in M_t \cap e_t = e] \\
&\quad \geq 1 - \sum_{v \in e} \sum_{s=t_1+1}^{t-1} \frac{1}{s} \;\geq\; 1 - k \sum_{s=t_1+1}^{t_2} \frac{1}{s} \qquad (2)
\end{aligned}$$

Using Riemann sums to bound the harmonic series, we see that

$$\sum_{s=t_1+1}^{t_2} \frac{1}{s} \leq \ln t_2 - \ln t_1 = -\ln \frac{t_1}{t_2} \tag{3}$$

Combining (1), (2), and (3), we obtain the desired bound. □

Using the bound from Lemma 4, we can bound how likely the algorithm is to select a given edge in a given time interval. This is expressed in terms of the probability that the revealed edge is in the current max matching M_t, which in turn is bounded in Lemma 6.

Lemma 5. *If $T = [t_1 + 1, t_2]$ is a time interval,*

$$\Pr[\mathsf{sel}(e, T)] \geq \frac{1}{n}\left(1 + k \ln \tfrac{t_1}{t_2}\right) \sum_{t \in T} \Pr[e \in M_t | e_t = e]$$

Proof. For an edge e to be selected, it must have been revealed at some time t, at which point e was both available and in the max matching. Formally,

$$\begin{aligned}
\Pr[\mathsf{sel}(e, T)] &= \sum_{t \in T} \Pr[e \in M_t \ \cap \ e_t = e \ \cap \ \mathsf{avail}(e, t)] \\
&= \sum_{t \in T} \Pr[\mathsf{avail}(e, t) | e \in M_t \cap e_t = e] \Pr[e \in M_t | e_t = e] \Pr[e_t = e] \\
&\geq \frac{1}{n} \sum_{t \in T} \left(1 + k \ln \tfrac{t_1}{t_2}\right) \Pr[e \in M_t | e_t = e] \,,
\end{aligned}$$

by Lemma 4 and the fact that $\Pr[e_t = e] = 1/n$ always, since it depends only on the ordering of the edges. The result follows. □

Lemma 6.

$$\sum_{e \in E} \Pr[e \in M_t | e_t = e] \, w_e \geq w(M_n)$$

Proof. Let $\hat{M}_t = M_n \cap E_t$. Clearly $\hat{M}_t$ is a matching, and since $M_t = \mathsf{maxmat}(E_t)$, we must have $w(M_t) \geq w(\hat{M}_t)$. Thus,

$$\mathbb{E}[w(M_t)] \geq \mathbb{E}[w(\hat{M}_t)] = \sum_{e \in M_n} \Pr[e \in E_t] \, w_e = \sum_{e \in M_n} \frac{t}{n} \, w_e = \frac{t}{n} w(M_n). \tag{4}$$

Observe that

$$\begin{aligned}\mathbb{E}[w(\{e_t\} \cap M_t)] &= \sum_{e \in E} \Pr[e \in \{e_t\} \cap M_t]\, w_e \\ &= \sum_{e \in E} \Pr[e \in M_t | e_t = e] \Pr[e_t = e]\, w_e \\ &= \frac{1}{n} \sum_{e \in E} \Pr[e \in M_t | e_t = e]\, w_e. \end{aligned} \tag{5}$$

On the other hand,

$$\mathbb{E}[w(\{e_t\} \cap M_t)] \geq \frac{1}{t}\mathbb{E}[w(M_t)] \geq \frac{1}{n} w(M_n), \tag{6}$$

where the last inequality is by (4). The result follows from combining (5) and (6). □

We can now put all of the lemmas together to obtain the competitive constant for our algorithm.

Theorem 7. *Algorithm 1 is a competitive algorithm for all k, with competitive constant $c(k) = \Omega(1/k)$.*

Proof. Let $T = [an + 1, n]$. Then

$$\mathbb{E}[\text{weight gain in } T] = \sum_{e \in E} \Pr[\mathsf{sel}(e, T)] \cdot w_e \tag{7}$$

$$\geq \sum_{e \in E} \frac{\alpha(T, k)}{n} \sum_{t \in T} \Pr[e \in M_t | e_t = e] \cdot w_e \tag{8}$$

$$= \frac{\left(1 + k \ln \frac{an}{n}\right)}{n} \sum_{t \in T} \sum_{e \in E} \Pr[e \in M_t | e_t = e] \cdot w_e \tag{9}$$

$$\geq \frac{1 + k \ln a}{n} \sum_{t=an+1}^{n} w(M_n) \tag{10}$$

$$= (1 - a)(1 + k \ln a) \cdot w(M_n), \tag{11}$$

where (8) is by Lemma 5 and (10) is by Lemma 6.

To show our constant $c(k) = \max_{a \in [0,1]}\{(1 - a)(1 + k \ln a)\}$ is $\Omega(1/k)$, we first give an upper bound. Since we must have $a < 1$ (otherwise, we select no edges), we have

$$1 + k \ln a < 1. \tag{12}$$

Since we need $c(k) > 0$, and certainly $1 - a \geq 0$, we must have

$$\begin{aligned} 1 + k \ln a > 0 \quad &\Longrightarrow \quad \ln a > -\frac{1}{k} \\ &\Longrightarrow \quad a > e^{-1/k} > 1 - \frac{1}{k} \\ &\Longrightarrow \quad 1 - a < \frac{1}{k}. \end{aligned} \tag{13}$$

From (12) and (13) we conclude $c(k) < 1/k$.

To show a lower bound, we choose a particular value a_k of a; set $a_k = e^{-1/2k}$. Then using the power series for e^x we have

$$1 - a = 1 - \left(1 - \frac{1}{2k} + \frac{1}{8k^2} - \dots\right) = \frac{1}{2k} + O\left(\tfrac{1}{k^2}\right). \tag{14}$$

Since $1 + k \ln a_k = 1 - 1/2 = 1/2$ by construction, from (14) we have

$$c(k) = \frac{1}{2} \cdot \frac{1}{2k} + O\left(\tfrac{1}{k^2}\right) > \frac{1}{5k} \tag{15}$$

for sufficiently large k. Thus, $c(k)$ is $\Omega(1/k)$. □