A Stronger Impossibility for Fully Online Matching[*]

Alexander Eckl[1], Anja Kirschbaum[1], Marilena Leichter[1], Kevin Schewior[2]

[1] Advanced Optimization in a Networked Economy (AdONE), Technical University of Munich, Germany
[2] Department of Mathematics and Computer Science, University of Cologne, Germany

Abstract

We revisit the fully online matching model (Huang et al., J. ACM, 2020), an extension of the classic online matching model due to Karp, Vazirani, and Vazirani (STOC 1990), which has recently received a lot of attention (Huang et al., SODA 2019 and FOCS 2020), partly due to applications in ride-sharing platforms. It has been shown that the fully online version is harder than the classic version for which the achievable competitive ratio is at most 0.6317, rather than precisely $1 - 1/e \approx 0.6321$. We introduce two new ideas to the construction. By optimizing the parameters of the modified construction numerically, we obtain an improved impossibility result of 0.6297. Like the previous bound, the new bound even holds for fractional (rather than randomized) algorithms on bipartite graphs.

1 Introduction

In the *fully online matching* model due to Huang et al. [2], the vertices of an undirected graph $G = (V, E)$ arrive over time. Every vertex has a deadline at which it *departs*. At any time, two adjacent vertices can be matched if both of them have arrived, neither of them has departed, and neither of them has been previously matched. An *online* algorithm may base its decisions only on the subgraph of G induced by the vertices that have already arrived, the deadlines of these vertices, and, in the case of a randomized algorithm, random bits.

The classic model due to Karp, Vazirani, and Vazirani [6] is a special case. Here, additionally, G is bipartite, initially all vertices from one side arrive, and then the vertices from the other side arrive. The vertices from the former side only depart at the very end. The vertices from the latter side depart even before the next vertex from that side arrives, or, equivalently, they must be matched either upon arrival or never. The classic model and its variants have been extensively studied prior to the work of Huang et al. [2] and have many applications, e.g., in online advertising [7]. However, a scenario not addressed by these models is, e.g., that of ride-sharing platforms in which customers and compatible drivers have to be matched, but both customers and drivers enter and leave the system at arbitrary times. This aspect is addressed by the fully online model.

In both models, the performance of an online algorithm is measured through standard competitive analysis with respect to the total number of matches performed by the algorithm over the entire time horizon: A randomized online algorithm is called α-competitive if, for all instances, the expected number of matches it performs is at least an α fraction of the number of matches an omniscient algorithm could have performed. It is well known [6, 1] that for the classic model the largest competitive ratio achievable by randomized algorithms is $1 - 1/e \approx 0.6321$, ignoring $o(1)$ terms as $|V| \to \infty$. It can be obtained through the Ranking algorithm. The same holds true for

---

[*] Funding: This work was supported in part by the Alexander von Humboldt Foundation with funds from the German Federal Ministry of Education and Research (BMBF) and by the Deutsche Forschungsgemeinschaft (DFG), GRK 2201.

Email addresses: `alexander.eckl@tum.de` (Alexander Eckl), `anja.kirschbaum@tum.de` (Anja Kirschbaum), `marilena.leichter@tum.de` (Marilena Leichter), `kschewior@gmail.com` (Kevin Schewior)

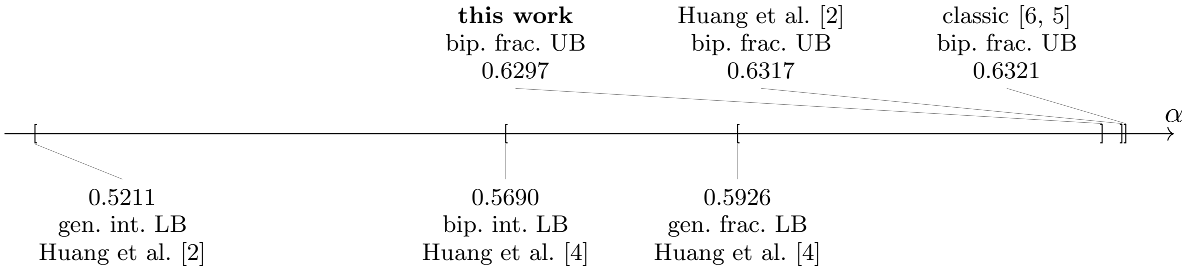

Figure 1: Known bounds on the best-possible competitive ratio α for fully online matching. We distinguish between bipartite and general graphs as well as fractional and integral algorithms.

the *fractional* relaxation of the model in which a (w.l.o.g. deterministic) algorithm may fractionally match vertices, thus obtaining a fractional matching [5].

In their seminal work on fully online matching, Huang et al. [2] prove that a generalization of the Ranking algorithm to general graphs is 0.5211-competitive, beating the trivial baseline of 0.5, and that no fractional algorithm can achieve a guarantee better than 0.6317 even on bipartite graphs, also beating the baseline of $1 - 1/e$. In follow-up work, Huang et al. [3, 4] revisit the fractional and bipartite cases. The state of the art is a 0.5926-competitive fractional algorithm on general graphs [4] and a 0.5690-competitive integral (as opposed to fractional) algorithm on bipartite graphs [4]. In particular, the impossibility of 0.6317 is still the state of the art, even for integral algorithms on general graphs. For further related work, we refer to the literature review by Huang et al. [2] and the slightly older (but in-depth) survey by Metha [7].

1.1 Our Contribution

In this work, we give an improvement of the impossibility from 0.6317 by Huang et al. [2] to 0.6297. Again, this bound also holds for fractional algorithms and on bipartite graphs. While the construction of Huang et al. is not explicitly analyzed for fractional algorithms, the analysis can be easily adapted. In contrast, we explicitly analyze fractional algorithms. We do so without using Yao's principle, avoiding any difficulties that would arise from an infinite strategy space. We show the evolution of this impossibility as well as the state-of-the-art competitive ratios in Figure 1.

The construction of Huang et al. [2] starts off with presenting a tree level by level starting from the root. The root has degree $\lambda + 1$, and exactly one of its children is a leaf. Which of the children the designated leaf is, however, remains unknown until the root has departed. While the optimum can match the leaf vertex to the root, the online algorithm cannot do better than matching the first-level vertices to the root with identical fractional value, leaving some fraction of the leaf vertex permanently unmatched. The construction is then repeated with the λ non-leaf children playing the role of the root, for a total of h levels. Finally, level h of the tree is augmented with the later-arriving side of the "triangle" construction of Karp, Vazirani, and Vazirani [6]. It can be shown that the extra tree puts the algorithm in a worse position and that choosing $\lambda = 7$ and $h \to \infty$ yields the bound of 0.6317.

The first ingredient to our result is the observation that this result can be reproduced with a slightly simpler construction: Rather than constructing a tree, one can also construct a sequence of bicliques. Specifically, we start with k vertices (corresponding to the root) on the first level and $(\lambda + 1)k$ vertices on the second level, connected by a biclique. After the vertices on the previous level depart, it is revealed which of the λk vertices form a biclique with $(\lambda + 1)\lambda k$ new vertices on the next level. Again, this process is repeated for a total of h steps and the "triangle" construction is added at the end. Note that now we are also allowed to choose any rational value for λ as long as we make k large enough. In fact, choosing $\lambda \approx 7.2336$ already slightly improves the bound by about $9.5 \cdot 10^{-7}$.

The observation leading to our second idea is that not all levels play exchangeable roles in the construction. This is clear for the last level, which is part of the triangle construction. Further, we will see the following for an optimal algorithm: Increasing the number of children in any level i changes the average matching value upon departure in any level j with $i < j < h$. Indeed, this matching value decreases if $j - i$ is odd and increases otherwise. This suggests that choosing a uniform λ for all levels is unlikely to yield a tight impossibility.

In general, we replace λ with different factors $\gamma_1, \ldots, \gamma_\ell$ for ℓ levels between the first h levels and the "triangle" construction. After taking the limit $h \to \infty$, we express the precise resulting competitive ratio as a function of $\lambda, \gamma_1, \ldots, \gamma_\ell$. Unfortunately, this function is already non-convex for $\ell = 0$, and the explicit formula gets quite complicated even for small values of ℓ. We therefore have to resort to numerical optimization. Specifically, we obtain the bound of 0.6297 through numerical optimization with Matlab. In this way, although we cannot prove that 0.6297 is the strongest impossibility achievable with our approach, the results of the numerical optimization do hint at that, and the claimed impossibility is provably correct.

We note that the numerical values we find for $\lambda, \gamma_1, \ldots, \gamma_\ell$ back our intuition that the optimal value heavily depends on whether the corresponding level is the last one, and otherwise its parity.

2 Description of the Construction

In this section, we formally describe the construction that is used to show the improved impossibility. The construction is parameterized by integers $h, \ell \geq 0$ and $k \geq 1$ as well as fractional numbers $\lambda > 1$ and $\gamma_j > 1$ for all $j \in [\ell] := \{1, \ldots, \ell\}$. We first define the bipartite graph G, visualized in Figure 2, underlying the construction and then specify the arrival and departure times of the vertices.

The vertex set of G can be partitioned into $2 \cdot (h+\ell+1)$ disjoint sets U_i, V_i for all $i \in \{0, \ldots, h+\ell\}$. We have $|U_i| = |V_i|$ for each such i. The set U_0 consists of k vertices. With increasing index, the number of vertices increases for the first h steps by a factor of λ and in the j-th subsequent step by a factor of γ_j. Formally,

$$\begin{aligned} |U_i| &= \lambda^i k && \forall i \in \{0, \ldots, h\}, \\ |U_{h+j}| &= \gamma_j |U_{h+j-1}| && \forall j \in \{1, \ldots, \ell\}. \end{aligned}$$

We note that by the continuous nature of the variables γ_j and λ, some of the sizes $|U_i| = |V_i|$ might take non-integer values. We avoid this by choosing the parameter k in dependence of h large enough such that all of these numbers are integer.

We introduce edges such that $(U_i, V_i \cup U_{i+1})$ is a complete bipartite graph for all $i \in \{0, \ldots, h+\ell-1\}$. The edges between $U_{h+\ell}$ and $V_{h+\ell}$ form an exception. Due to this exception, we define $A := U_{h+\ell}$ and $B := V_{h+\ell}$. In Section 3, we describe how the adversary chooses an ordering of $A = \{a_1, \ldots, a_{|A|}\}$ and $B = \{b_1, \ldots, b_{|B|}\}$. Then a_i and b_j are connected by an edge if and only if $i \geq j$. The subgraph of G induced by $A \cup B$ has been previously used to show impossibilities for online matching problems [6, 5]. This completes the description of G.

To complete the description of our *online* construction, we specify the arrival and departure times of the vertices in the graph G. At the beginning, all vertices in U_0 arrive simultaneously. Let $i \in \{0, \ldots, h+\ell-1\}$ and assume all vertices in U_j for $j \leq i$ and in V_j for $j \leq i - 1$ have already arrived, and all of these vertices not in U_i have departed again. Then, all vertices in U_{i+1} and V_i (by definition, along with their edges to U_i) arrive simultaneously. Next, the vertices in U_i and V_{i-1} (if $i \geq 1$) depart simultaneously. At this point, it is impossible for an algorithm to differentiate

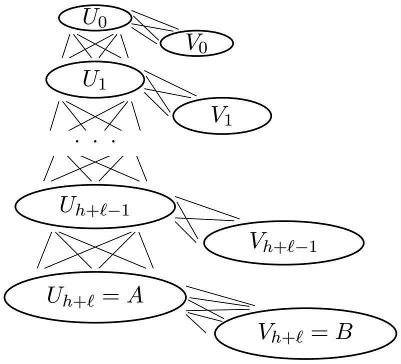

Figure 2: Visualization of the bipartite graph G.

between the neighbors of U_i, in particular identifying which of them belong to U_{i+1} and which to V_i. In Section 3, we describe how the adversary can assign these vertices to U_{i+1} and V_i based on the matching decisions of the algorithm. If $i \leq h+\ell-2$, the initial assumption is reestablished for the next-larger index, and the arrivals and departures happen as described. Otherwise, $A = U_{h+\ell}$ has now arrived, and all other previously arrived vertices have departed. Then the vertices in B arrive in order of $b_1, \ldots, b_{|B|}$, and every vertex departs again immediately after its arrival. Finally, the vertices A and $V_{h+\ell-1}$ depart simultaneously. Note that all edges of the graph respect release and departure times in the sense that, for all of them, there is a time when both their endpoints have arrived but not departed.

The graph has a perfect matching which is the union of the following perfect matchings: From the complete bipartite subgraphs induced by $U_i \cup V_i$ we choose an arbitrary perfect matching. For A and B we choose the perfect matching containing (a_i, b_i) for all $i \in [|A|]$. We also remark that $U := \bigcup_{i=0}^{h+\ell} U_i$ and $V := \bigcup_{i=0}^{h+\ell} V_i$, however, do *not* correspond to the two sides of the bipartite graph.

3 Derivation of the Upper Bound

Using the construction from the previous section, we show our result in this section.

Theorem 3.1. *The competitive ratio of any fractional algorithm for fully online matching is at most* 0.6297*, even on bipartite graphs.*

To prove this result, we have to bound both the value of the offline optimum and that of any online algorithm. Clearly, since G has a perfect matching and its vertex set is $U \cup V$ where $|U| = |V|$, we have $\mathrm{OPT} = |U| = |V|$.

Now let ALG be any fractional online algorithm. For all vertices in our graph, we determine the fraction to which they are matched. For simplicity, we make two assumptions that are without loss of generality. First, we assume that the algorithm only performs matches along an edge whenever one of its endpoints departs. Furthermore, when a vertex departs, we assume that the algorithm fully matches it unless all of its neighbors are fully matched. It can be seen by a simple exchange argument that the latter assumption is indeed without loss of generality (see also [2]).

Note that we only need to consider the matching value distributed at the departure times of vertices in $U_0 \cup \cdots \cup U_{h+\ell-1} \cup B$. We do so in order of departure of the respective vertices, so we start with U_{i-1} for $i \in \{1, \ldots, h+\ell\}$. Recall that, when these vertices reach their deadlines, their *not departed* neighbors $N^+(U_{i-1}) := U_i \cup V_{i-1}$ are indistinguishable for the algorithm. For simplicity, we would like to analyze the water-filling algorithm [5] $\mathrm{ALG}_{\mathrm{wf}}$ which also treats these vertices equally, i.e., matches them all to the same fraction. It may, however, conceivably help the algorithm to match vertices to *different fractions*. In the following, we first show that this is (essentially) not the case.

We will define an adversary that assigns the vertices in $N^+(U_{i-1})$ to U_i and V_{i-1} (respecting $|U_{i-1}| = |V_{i-1}|$) based on the matching decisions of the algorithm. A first approach may be to just assign those vertices to V_{i-1} that have been matched the least. We will, however, later need a lower *and an upper* bound on the value to which U_i has been matched. The latter approach only gives us the lower bound.

We define p_{i-1} such that the total matching value of the vertices in U_{i-1} just after the departure of U_{i-2} is $p_{i-1} \cdot |U_{i-1}|$. In other words, p_{i-1} is the average matching value across all (again, conceivably different) matching values of vertices in U_{i-1} at that point. We define $p_0 = 0$.

Note that, upon departure of U_{i-1}, $\mathrm{ALG}_{\mathrm{wf}}$ would assign a total fractional matching value of

$$d \coloneqq (1 - p_{i-1}) \cdot |U_{i-1}| \, \frac{|U_i|}{|N^+(U_{i-1})|} \tag{1}$$

to any set of $|U_i|$ vertices in $N^+(U_{i-1})$. Our adversary simply chooses an assignment of $N^+(U_{i-1})$ to U_i and V_{i-1} such that $p_i \cdot |U_i|$ is as close as possible to d. The following lemma shows that the error becomes vanishingly small.

Lemma 3.2. *Let $i \in [h+\ell]$ and define $n_i \coloneqq |U_i|$. For any given distribution of matching value to $N^+(U_{i-1})$ by the algorithm, there is an adversary that chooses sets U_i and V_{i-1} such that*

$$\begin{aligned} p_i &= \frac{1 - p_{i-1}}{\lambda + 1} + \varepsilon_1(i), && \text{for } i \le h \quad \text{and} \\ p_i &= \frac{1 - p_{i-1}}{\gamma_j + 1} + \varepsilon_1(h+j), && \text{for } i = h+j, j \in [\ell], \end{aligned}$$

where the error term $\varepsilon_1(i)$ fulfills $|\varepsilon_1(i)| \le \frac{1}{n_i}$.

Proof. For now, let $i \in [h]$ and let a distribution of the matching value from U_{i-1} to $N^+(U_{i-1})$ be given. Let $m(U_i)$ and $m(V_{i-1})$ be the total matching value received by U_i and V_{i-1}, respectively. The adversary decides on a partition $U_i \,\dot\cup\, V_{i-1}$ of $N^+(U_{i-1})$. We let the adversary partition the vertices such that $m(U_i)$ is as close as possible to d as in Equation (1). Note that here

$$d = (1 - p_{i-1})|U_{i-1}| \, \frac{|U_i|}{|N(U_{i-1})|} = |U_i| \, \frac{1 - p_{i-1}}{\lambda + 1}.$$

Note that this value does not depend on the explicit partition since $|U_i|$ has a prescribed size which is not chosen by the adversary. It is clear that d corresponds to the total matching value assigned to U_i if all vertices in $N^+(U_{i-1})$ have received the same amount of matching value. It turns out that, independently of the assignment by the algorithm, it is always possible to assign sets U_i and V_{i-1} with the correct cardinalities such that

$$d - 1 \le m(U_i) \le d + 1. \tag{2}$$

To verify this claim, assume for contradiction that $\hat{U}_i$ is the set with minimal distance $\left| m(\hat{U}_i) - d \right| > 1$. Additionally, let $\hat{V}_{i-1} = N^+(U_{i-1}) \setminus \hat{U}_i$ be the complementary set in the partition.

Firstly, assume that $m(\hat{U}_i)$ is larger than d. Let $u \in \hat{U}_i$ be the element of $\hat{U}_i$ with maximum matching value and let $v \in \hat{V}_{i-1}$ be the element in $\hat{V}_{i-1}$ with minimum matching value. It must hold that $1 \geq m(u) > m(v) \geq 0$, otherwise the matching $m(\hat{U}_i)$ cannot exceed the weighted average d. This implies $0 < m(u) - m(v) \leq 1$. Hence when we exchange u and v in the sets $\hat{U}_i$ and $\hat{V}_{i-1}$, we have $m(\hat{U}_i \setminus \{u\} \cup \{v\}) - d = m(\hat{U}_i) - m(u) + m(v) - d$ and $0 \leq m(\hat{U}_i) - d - (m(u) - m(v)) < m(\hat{U}_i) - d$, so $\left| m(\hat{U}_i \setminus \{u\} \cup \{v\}) - d \right| < \left| m(\hat{U}_i) - d \right|$, a contradiction to the minimality of the distance.

Secondly, when $m(\hat{U}_i)$ is smaller than d, let $u \in \hat{U}_i$ be the element of $\hat{U}_i$ with minimum matching value and let $v \in \hat{V}_{i-1}$ be the element in $\hat{V}_{i-1}$ with maximum matching value. It holds $0 \leq m(u) < m(v) \leq 1$, and hence we can again switch u and v to receive $\left| m(\hat{U}_i \setminus \{u\} \cup \{v\}) - d \right| < \left| m(\hat{U}_i) - d \right|$. Therefore, we have proven Equation (2) by contradiction. We divide it by $|U_i|$ to receive

$$\frac{1 - p_{i-1}}{\lambda + 1} - \frac{1}{|U_i|} \leq \frac{m(U_i)}{|U_i|} \leq \frac{1 - p_{i-1}}{\lambda + 1} + \frac{1}{|U_i|}.$$

By the definition of p_i as $m(U_i)/|U_i|$, we finally have

$$p_i = \frac{1 - p_{i-1}}{\lambda + 1} + \varepsilon_1(i)$$

for some error term with $|\varepsilon_1(i)| \leq \frac{1}{n_i}$.

We repeat the above arguments for $j \in [\ell]$ with the corresponding growth parameters to obtain

$$p_{h+j} = \frac{1 - p_{h+j-1}}{\gamma_j + 1} + \varepsilon_1(h{+}j)$$

with $|\varepsilon_1(h{+}j)| \leq \frac{1}{n_{h+j}}$. □

Via induction, we obtain a closed-form description of p_i (see appendix):

$$p_i = \frac{1}{\lambda + 2} \left(1 - \left(\frac{-1}{\lambda + 1} \right)^i \right) + \varepsilon_2(i), \qquad \forall i \in [h], \tag{3}$$

with $|\varepsilon_2(i)| \leq \frac{i}{n_i}$. Computing explicit formulas for $p_j, j > h$ is quite cumbersome. We detail the algebraic transformations for $\ell = 3$ in the appendix.

We are interested in the matching value the algorithm gives to vertices in V. Let us define q_i as the total matched fraction of the vertices in V_i for all $i \in \{0, \ldots, h{+}\ell{-}1\}$. Note that $B = V_{h+\ell}$ is matched differently, something we will consider at a later point.

Lemma 3.3. *Let $i \in \{0, \ldots, h{+}\ell{-}1\}$. Then it holds that*

$$q_i = \overline{p}_{i+1} + \varepsilon_3(i),$$

where $\overline{p}_{i+1} \coloneqq p_{i+1} - \varepsilon_2(i{+}1)$ and $|\varepsilon_3(i)| \leq \frac{i+3}{n_i}$.

Proof. Again, we only prove the cases $i \in \{0, \ldots, h{-}1\}$; the cases of larger i are analogous with adapted growth parameters. Let a distribution of the matching value from U_i to their neighbors be given. Since the total matching value in U_i is $(1 - p_i)n_i$, it holds that

$$(1 - p_i)n_i = p_{i+1}n_{i+1} + q_i n_i.$$

Rearranging this equation, we use the formula from Lemma 3.2 to write

$$\begin{aligned} q_i n_i &= (1 - p_i)n_i - p_{i+1}n_{i+1} \\ &= (1 - p_i)n_i - \left(\frac{1 - p_i}{\lambda + 1} + \varepsilon_1(i{+}1) \right) \lambda n_i \\ &= n_i \left(\frac{1 - p_i}{\lambda + 1} - \lambda \varepsilon_1(i{+}1) \right). \end{aligned}$$

With $n_i = |U_i| = |V_i|$ we can simplify to $q_i = \frac{1-p_i}{\lambda+1} - \lambda\varepsilon_1(i{+}1)$. Inserting Lemma 3.2 gives

$$\begin{aligned} q_i &= p_{i+1} - (\lambda+1)\varepsilon_1(i{+}1) \\ &= (p_{i+1} - \varepsilon_2(i{+}1)) + (\varepsilon_2(i{+}1) - (\lambda+1)\varepsilon_1(i{+}1)) \\ &= \overline{p}_{i+1} + \varepsilon_3(i), \end{aligned}$$

where it is easy to see that $|\varepsilon_3(i)| \leq \frac{i+3}{n_i}$. □

In other words, we express the q_i in terms of p_{i+1} without the error terms $\varepsilon_2(i{+}1)$ for all $i \in \{0, \ldots, h{+}\ell{-}1\}$.

We can write the amount of matching value the vertex sets V_i (except B) contribute to the algorithmic value as

$$\sum_{i=0}^{h+\ell-1} q_i |V_i|.$$

Finally, we consider the vertices in A. Recall that their total matching value is $p_{h+\ell} \cdot |A|$ before any vertex in B has arrived. For simplicity, we define $p_A := p_{h+\ell}$. We also define ρ such that the additional matching value that A receives (that is, at the departure times of vertices in B) is $\rho \cdot |A|$. In the following, we will give an upper bound on ρ.

Again, an analysis of $\mathrm{ALG}_{\mathrm{wf}}$ would be simpler but some carefulness is required because the matching value $p_A \cdot |A|$ is not necessarily distributed uniformly across A. Fortunately, a simple adversary for choosing the ordering of A and B suffices here.

Lemma 3.4. *The adversary can choose the ordering of A and B such that*

$$\rho \leq 1 - \exp(-(1 - p_A)) + \frac{2}{|A|}.$$

Proof. As described in Section 2, the vertices in B arrive, and immediately depart again, sequentially. The vertices are labeled $b_1, \ldots, b_{|B|}$ in this order, and any vertex b_i is adjacent to the vertices $a_i, \ldots, a_{|A|}$. Note that the algorithm only learns about the identity of vertex a_i after the departure of b_i. Here, the adversary simply chooses a_i in every round such that it has the minimum current matching value out of all remaining unlabeled vertices in A.

We now bound $\rho \cdot |A|$, the fractional matching value placed by the algorithm on edges between A and B, by the matching value $\omega \cdot |A|$ placed by $\mathrm{ALG}_{\mathrm{wf}}$ on the same instance: Assuming that the matching value in A is equally distributed before B arrives, i.e., all vertices have *exactly* p_A matching value, this algorithm simply matches b_i equally among all vertices $a_i, \ldots, a_{|A|}$. We claim that $\rho \leq \omega$. In the following, we denote the matching value placed on $a \in A$ by our algorithm and $\mathrm{ALG}_{\mathrm{wf}}$ at the time of the departure of b_i by $m_i(a)$ and $\omega_i(a)$, respectively. We omit the index i if we refer to the final value.

Let $\eta \in [|B|]$ the final index for which $\mathrm{ALG}_{\mathrm{wf}}$ is able to assign matching value to A. Note that every time i appears before b_η, $\omega_i(A) \geq m_i(A)$, as $\mathrm{ALG}_{\mathrm{wf}}$ always matches the current vertex b_i fully if possible. Also, for $i \geq \eta$ it holds that $1 = \omega(a_i) \geq m(a_i)$, so $\omega(\{a_i : i \geq \eta\}) \geq m(\{a_i : i \geq \eta\})$. Now let $i' < \eta$ be the maximal index i so that $\omega(a_i) < m(a_i)$. By the adversary's choice of $a_{i'}$, it holds that

$$\begin{aligned} m_{i'}(\{a_i : i > i'\}) &\geq m(a_{i'})|\{a_i : i > i'\}| \\ &> \omega(a_{i'})|\{a_i : i > i'\}| = \omega_{i'}(\{a_i : i > i'\}). \end{aligned}$$

Using

$$\begin{aligned}\omega(\{a_i : i \leq i'\}) + \omega_{i'}(\{a_i : i > i'\}) &= \omega_{i'}(A) \\ \geq m_{i'}(A) &= m(\{a_i : i \leq i'\}) + m_{i'}(\{a_i : i > i'\}),\end{aligned}$$

we have $\omega(\{a_i : i \leq i'\}) \geq m(\{a_i : i \leq i'\})$ and as $i' < \eta$ is the last vertex with $\omega(a_i) < m(a_i)$, we know that for all remaining vertices $\mathrm{ALG}_{\mathrm{wf}}$ assigns more matching value. Therefore, $\omega(\{a_i : i' < i < \eta\}) \geq m(\{a_i : i' < i < \eta\})$ and thus $\omega(A) \geq m(A)$. Finally, we note that if i' does not exist, $\omega(\{a_i : i < \eta\}) \geq m(\{a_i : i < \eta\})$ holds trivially.

All that remains to prove the lemma is to show that $\omega \leq 1 - \exp(-(1-p_A))$. Let again $\eta \in [|B|]$ be the final index for which $\mathrm{ALG}_{\mathrm{wf}}$ is able to assign value to A. At the departure of all vertices b_i, $i < \eta$ in B, $\mathrm{ALG}_{\mathrm{wf}}$ assigns $1/(|A|-i+1)$ of matching value each to $a_i, \ldots, a_{|A|}$, while at the departure of b_i, $i > \eta$ no value is assigned. So the average fractional matching value ω fulfills $\omega|A| \leq \eta$, which holds with equality exactly if the last active vertex b_η is fully matched.

Let us again refer to the matching value placed by $\mathrm{ALG}_{\mathrm{wf}}$ on a vertex a_i by $\omega(a_i)$. After b_η has departed, all remaining vertices $a_\eta, \ldots, a_{|A|}$ are already fully matched. Hence, for all $i \geq \eta$, we have $\omega(a_i) = 1$. At the same time, $\omega(a_i), i \geq \eta$ is composed of the initial value p_A plus the entire value assigned by the vertices $b_1, \ldots, b_\eta$. For all $i \geq \eta$ we can obtain:

$$\begin{aligned}1 = \omega(a_i) &\geq p_A + \sum_{j=1}^{\eta-1} \frac{1}{(|A| - j + 1)} \\ &= p_A + H_{|A|} - H_{|A|-\eta+1},\end{aligned}$$

where H_n is the n-th harmonic number which we estimate by $\ln(n) \leq H_n \leq \ln(n+1)$. Hence, we have

$$1 \geq p_A + \ln(|A|) - \ln(|A| - \eta + 2) \geq p_A + \ln\left(\frac{|A|}{|A|(1-\omega)+2}\right),$$

where we used $\omega|A| \leq \eta$. By further rearranging,

$$\exp(-(1-p_A)) \leq 1 - \exp(-(1-p_A)) + \frac{2}{|A|}.$$

This completes the proof. □

We have now computed all necessary values for our formula. We double-count the matching value by counting the fractional value to which each *vertex* is matched and establish for the entire value of ALG:

$$2\,\mathrm{ALG} = \sum_{i=0}^{h+2} q_i|V_i| + |U \setminus A| + p_A|A| + 2\rho|A|.$$

For $\ell = 0$, by inserting our formulas into $\mathrm{ALG}/\mathrm{OPT}$ and taking the limit $h \to \infty$, we obtain in congruence with Huang et al. [2]

$$\frac{\lambda - 1}{\lambda} \cdot \left(1 - \exp\left(-\frac{\lambda+1}{\lambda+2}\right)\right) + \frac{\lambda+1}{\lambda \cdot (\lambda+2)}$$

as an upper bound on the competitive ratio. Interestingly, this function is non-convex as its derivative has a local maximum at $\lambda \approx 10.0266$.

ℓ	λ	$\cdots$	$\gamma_{\ell-4}$	$\gamma_{\ell-3}$	$\gamma_{\ell-2}$	$\gamma_{\ell-1}$	γ_{ℓ}	impossibility
0	7.233629	$\cdots$	–	–	–	–	–	0.631744
1	2.581174	$\cdots$	–	–	–	–	8.053197	0.629748
2	3.148324	$\cdots$	–	–	–	2.390115	7.874599	0.629678
3	2.875859	$\cdots$	–	–	3.249854	2.403421	7.864072	0.629674
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
10	2.94419	$\cdots$	2.986001	2.843101	3.241640	2.404098	7.863523	0.629674

Table 1: The results of the numerical optimization. All numbers are rounded to the sixth decimal digit.

Computing explicit formulas for all q_j when $j > h$ is quite cumbersome, and the same holds for the explicit formula of the resulting lower bound on the competitive ratio. We showcase the result of this computation for $\ell = 3$. We refer to the appendix for all details on the computation. The error terms in the formula vanish as we take the limit $h \to \infty$. The final formula for our upper bound on the competitive ratio depends only on $\gamma_1, \gamma_2, \gamma_3$ and λ:

$$\begin{aligned}
&\frac{\lambda + \gamma_1(\gamma_2 + 1)(\lambda - 1)}{2(\lambda + \bar{\gamma}(\lambda - 1))} \\
&+ \frac{\lambda^2 + \gamma_1}{2(\lambda + 2)(\gamma_1 + 1)(\lambda + \bar{\gamma}(\lambda - 1))} \\
&+ \frac{\gamma_1(\lambda - 1)}{2(\lambda + \bar{\gamma}(\lambda - 1))} \cdot \frac{\gamma_1(\lambda + 2) + 1}{(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 2)} \\
&+ \frac{\gamma_1\gamma_2(\lambda - 1)}{2(\lambda + \bar{\gamma}(\lambda - 1))} \cdot \frac{\gamma_2(\gamma_1 + 1)(\lambda + 2) + (\lambda + 1)}{(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 2)} \\
&+ \frac{\gamma_1\gamma_2\gamma_3(\lambda - 1)}{(\lambda + \bar{\gamma}(\lambda - 1))} \cdot \\
&\qquad \left[1 - \exp\left(-\frac{1}{\gamma_3 + 1}\left(\gamma_3 + \frac{\gamma_1(\lambda + 2) + 1}{(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 2)}\right)\right)\right],
\end{aligned}$$

using the abbreviation $\bar{\gamma} := \sum_{i=1}^{3} \prod_{j=1}^{i} \gamma_j$.
We use the numerical computing software Matlab for the numerical optimization. Using a trusted-region algorithm for unconstrained multivariate minimization we receive the values shown in Table 3. In particular, we obtain an upper bound smaller than 0.6297. This completes the proof of Theorem 3.1.

References

[1] B. E. Birnbaum and C. Mathieu. On-line bipartite matching made simple. *SIGACT News*, 39(1):80–87, 2008.

[2] Z. Huang, N. Kang, Z. G. Tang, X. Wu, Y. Zhang, and X. Zhu. Fully online matching. *J. ACM*, 67(3):17:1–17:25, 2020.

[3] Z. Huang, B. Peng, Z. G. Tang, R. Tao, X. Wu, and Y. Zhang. Tight competitive ratios of classic matching algorithms in the fully online model. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2875–2886, 2019.

[4] Z. Huang, Z. G. Tang, X. Wu, and Y. Zhang. Fully online matching II: Beating ranking and water-filling. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1380–1391, 2020.

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

[6] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *ACM Symposium on Theory of Computing (STOC)*, pages 352–358, 1990.

[7] A. Mehta. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4):265–368, 2013.

A Computing closed forms for p_i

We use induction to obtain the closed form

$$p_i = \frac{1}{\lambda+2}\left(1-\left(\frac{-1}{\lambda+1}\right)^i\right) + \varepsilon_2(i), \qquad \forall i \in [h],$$

with an error term fulfilling $|\varepsilon_2(i)| \leq \frac{i}{n_i}$. Since we set $p_0 = 0 = \frac{1}{\lambda+2}\left(1-({}^{-1}/_{\lambda+1})^0\right)$, the induction base holds. For the induction hypothesis we use Lemma 3.2.

$$\begin{aligned}
p_i &= \frac{1-p_{i-1}}{\lambda+1} + \varepsilon_1(i) \\
&= \frac{1}{\lambda+1} - \frac{1}{\lambda+1}\left(\frac{1}{\lambda+2}\left(1-\left(\frac{-1}{\lambda+1}\right)^{i-1}\right) + \varepsilon_2(i-1)\right) + \varepsilon_1(i) \\
&= \frac{1}{\lambda+1} - \frac{1}{\lambda+2}\left(\frac{1}{\lambda+1} + \left(\frac{-1}{\lambda+1}\right)^i\right) + \frac{\varepsilon_2(i-1)}{\lambda+1} + \varepsilon_1(i) \\
&= \frac{1}{\lambda+1} - \frac{1}{(\lambda+2)(\lambda+1)} - \frac{1}{\lambda+2}\left(\frac{-1}{\lambda+1}\right)^i + \underbrace{\frac{\varepsilon_2(i-1)}{\lambda+1} + \varepsilon_1(i)}_{=:\varepsilon_2(i)} \\
&= \frac{1}{\lambda+2}\left(1-\left(\frac{-1}{\lambda+1}\right)^i\right) + \varepsilon_2(i),
\end{aligned}$$

with

$$|\varepsilon_2(i)| \leq \frac{|\varepsilon_2(i-1)|}{\lambda+1} + |\varepsilon_1(i)| \leq \frac{i-1}{\lambda n_{i-1}} + \frac{1}{n_i} = \frac{i}{n_i},$$

where we used $n_i = \lambda n_{i-1}$. Analogously, applying Lemma 3.2 to the closed-form of p_h, we obtain

$$\begin{aligned} p_{h+1} &= \frac{1}{\gamma_1+1}\left(\frac{\lambda+1}{\lambda+2}\left(1-\left(\frac{-1}{\lambda+1}\right)^{h+1}\right)\right) \\ &\quad + \varepsilon_2(h{+}1), && (4) \\ p_{h+2} &= \frac{1}{(\gamma_2+1)(\gamma_1+1)(\lambda+2)}\left(\gamma_1(\lambda+2)+1-\left(\frac{-1}{\lambda+1}\right)^{h}\right) \\ &\quad + \varepsilon_2(h{+}2), && (5) \\ p_{h+3} &= \frac{(\gamma_2\gamma_1+\gamma_2+1)(\lambda+2)-1+\left(\frac{-1}{\lambda+1}\right)^{h}}{(\gamma_3+1)(\gamma_2+1)(\gamma_1+1)(\lambda+2)} \\ &\quad + \varepsilon_2(h{+}3). && (6) \end{aligned}$$

B Detailed algebraic computations

In this section, we present the detailed computations to receive our upper bound formula which is optimized at the end of Section 3. We recall the value of the optimal solution of our instance:

$$\begin{aligned} 2\,\mathrm{OPT} &= |U|+|V| = 2|U| = 2\sum_{i=0}^{h+3}|U_i| \\ &= 2\left(k\left(\frac{\lambda^{h+1}-1}{\lambda-1}\right)+k\lambda^h\underbrace{(\gamma_1+\gamma_1\gamma_2+\gamma_1\gamma_2\gamma_3)}_{=:\bar{\gamma}}\right) \\ &= \frac{2k\lambda^h}{\lambda-1}\left(\lambda-{}^1\!/\!_{\lambda^h}+\bar{\gamma}(\lambda-1)\right). && (7) \end{aligned}$$

The algorithmic solution value is given by

$$\begin{aligned} 2\,\mathrm{ALG} &= \sum_{i=0}^{h+2} q_i|V_i| + |U\setminus A| + p_A|A| + 2\rho|A| \\ &= 2\rho|A| + \sum_{i=0}^{h+2}|U_i| + \sum_{i=0}^{h-1} q_i|V_i| \\ &\quad + q_h|V_h| + q_{h+1}|V_{h+1}| + q_{h+2}|V_{h+2}| + p_A|A|. \end{aligned}$$

We insert the values of p_i, q_i and the cardinalities n_i:

$$\begin{aligned} 2\,\mathrm{ALG} &= 2\rho|A| && (8) \\ &\quad + \frac{k\lambda^h}{\lambda-1}\left(\lambda-{}^1\!/\!_{\lambda^h}+\gamma_1(\gamma_2+1)(\lambda-1)\right) && (9) \\ &\quad + \sum_{i=0}^{h-1} n_i(p_{i+1}-\varepsilon_2(i+1)) + \sum_{i=0}^{h-1} n_i\varepsilon_3(i) && (10) \\ &\quad + n_h(p_{h+1}-\varepsilon_2(h{+}1)) + n_h\varepsilon_3(h) && (11) \\ &\quad + n_{h+1}(p_{h+2}-\varepsilon_2(h{+}2)) + n_{h+1}\varepsilon_3(h{+}1) && (12) \\ &\quad + n_{h+2}(p_A-\varepsilon_2(h{+}3)) + n_{h+2}\varepsilon_3(h{+}2) && (13) \\ &\quad + n_A(p_A-\varepsilon_2(h{+}3)) + n_A\varepsilon_2(h{+}3). && (14) \end{aligned}$$

By equations (3), (4)-(6), the terms $p_i - \varepsilon_2(i)$ do not contain any error terms (they are directly subtracted). Hence all remaining error terms in the above formula sum up to:

$$\begin{aligned}\varepsilon &:= \left|\sum_{i=0}^{h+2} n_i \varepsilon_3(i) + n_A \varepsilon_2(h{+}3)\right| \\ &\leq \sum_{i=0}^{h+2}(i{+}3) + (h{+}3) \leq \frac{(h{+}5)^2 + h{+}5}{2} + h{+}3 \in \Omega(h^2).\end{aligned}$$

As $2\,\mathrm{OPT}$ is dominated by λ^h and $\lambda > 1$, $\frac{\varepsilon}{2\,\mathrm{OPT}}$ vanishes for $h \to \infty$. We insert (4)-(6) into lines (11)-(14). Then, we divide all lines (8)-(14) without error terms by $2\,\mathrm{OPT}$ and take limits $h \to \infty$:

$$\begin{aligned}\frac{(9)}{2\,\mathrm{OPT}} &= \frac{k\lambda^h}{\lambda - 1}\frac{\lambda - 1}{k\lambda^h}\frac{(\lambda - {}^{1}\!/_{\lambda^h} + \gamma_1(\gamma_2 + 1)(\lambda - 1))}{2\,(\lambda - {}^{1}\!/_{\lambda^h} + \bar{\gamma}(\lambda - 1))} \\ &\to \frac{(\lambda + \gamma_1(\gamma_2 + 1)(\lambda - 1))}{2\,(\lambda + \bar{\gamma}(\lambda - 1))} = (15),\end{aligned}$$

$$\begin{aligned}\frac{(10)}{2\,\mathrm{OPT}} &= \frac{k\lambda^h}{\lambda + 2}\frac{\lambda - 1}{k\lambda^h}\frac{\left(\frac{1 - {}^{1}\!/_{\lambda^h}}{\lambda - 1} - \frac{({}^{-1}\!/_{\lambda+1})^h - {}^{1}\!/_{\lambda^h}}{2\lambda + 1}\right)}{2\,(\lambda - {}^{1}\!/_{\lambda^h} + \bar{\gamma}(\lambda - 1))} \\ &\to \frac{\lambda - 1}{\lambda + 2}\frac{\left(\frac{1}{\lambda - 1}\right)}{2\,(\lambda + \bar{\gamma}(\lambda - 1))} \\ &= \frac{1}{2(\lambda + 2)\,(\lambda + \bar{\gamma}(\lambda - 1))} = \text{part of } (16),\end{aligned}$$

$$\begin{aligned}\frac{(11)}{2\,\mathrm{OPT}} &= \frac{(\lambda + 1)\left(1 - ({}^{-1}\!/_{\lambda+1})^{h+1}\right)}{(\lambda + 2)(\gamma_1 + 1)} \cdot \lambda^h k \frac{(\lambda - 1)}{2\lambda^h k\,(\lambda - {}^{1}\!/_{\lambda^h} + \bar{\gamma}(\lambda - 1))} \\ &\to \frac{(\lambda + 1)(\lambda - 1)}{2(\lambda + 2)(\gamma_1 + 1)\lambda + \bar{\gamma}(\lambda - 1)} = \text{part of } (16),\end{aligned}$$

$$\begin{aligned}\frac{(12)}{2\,\mathrm{OPT}} &= \frac{\gamma_1(\lambda + 2) + 1 - ({}^{-1}\!/_{\lambda+1})^h}{(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 2)} \cdot \gamma_1\lambda^h k \frac{(\lambda - 1)}{2\lambda^h k\,(\lambda - {}^{1}\!/_{\lambda^h} + \bar{\gamma}(\lambda - 1))} \\ &\to \frac{\gamma_1(\lambda + 2) + 1}{(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 2)}\frac{\gamma_1(\lambda - 1)}{2\,(\lambda + \bar{\gamma}(\lambda - 1))} = (17),\end{aligned}$$

$$\begin{aligned}\frac{(13) + (14)}{2\,\mathrm{OPT}} &= \frac{(\lambda - 1)(\lambda^h k\gamma_1\gamma_2 + \lambda^h k\gamma_1\gamma_2\gamma_3)}{2\lambda^h k\,(\lambda - {}^{1}\!/_{\lambda^h} + \bar{\gamma}(\lambda - 1))} \cdot \\ &\qquad \frac{(\gamma_2\gamma_1 + \gamma_2 + 1)(\lambda + 2) - 1 + ({}^{-1}\!/_{\lambda+1})^h}{(\gamma_3 + 1)(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 1)} \\ &= \frac{(\lambda - 1)\gamma_1\gamma_2}{2\,(\lambda - {}^{1}\!/_{\lambda^h} + \bar{\gamma}(\lambda - 1))} \cdot \\ &\qquad \frac{\gamma_2(\gamma_1 + 1)(\lambda + 2) + (\lambda + 1) + ({}^{-1}\!/_{\lambda+1})^h}{(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 1)} \\ &\to \frac{\gamma_1\gamma_2(\lambda - 1)}{2(\lambda + \bar{\gamma}(\lambda - 1))} \cdot \frac{\gamma_2(\gamma_1 + 1)(\lambda + 2) + (\lambda + 1)}{(\gamma_2 + 1)(\gamma_1 + 1)(\lambda + 2)} \\ &= (18),\end{aligned}$$

$$\begin{aligned}
\frac{(8)}{2\,\mathrm{OPT}} &\le \frac{|A|(1-\exp(-(1-p_A)) + {}^{2}\!/\!{}_{|A|})}{\mathrm{OPT}} \\
&= \frac{2}{\mathrm{OPT}} + \frac{k\lambda^h \gamma_1\gamma_2\gamma_3(\lambda-1)}{k\lambda^h\left(\lambda - {}^{1}\!/\!{}_{\lambda^h} + \bar{\gamma}(\lambda-1)\right)} \cdot \Bigg[1-\exp\Bigg(-1+ \\
&\qquad \frac{\gamma_2(\gamma_1+1)(\lambda+2)+(\lambda+1)+({}^{-1}\!/\!{}_{\lambda+1})^h}{(\gamma_3+1)(\gamma_2+1)(\gamma_1+1)(\lambda+2)} + \varepsilon_2(h+3)\Bigg)\Bigg] \\
&\to \frac{\gamma_1\gamma_2\gamma_3(\lambda-1)}{(\lambda+\bar{\gamma}(\lambda-1))} \cdot \\
&\qquad \left[1-\exp\left(-1+\frac{\gamma_2(\gamma_1+1)(\lambda+2)+(\lambda+1)}{(\gamma_3+1)(\gamma_2+1)(\gamma_1+1)(\lambda+2)}\right)\right] \\
&= (19).
\end{aligned}$$

As $h \to \infty$, our upper bound on the competitive ratio of ALG therefore approaches

$$\frac{\lambda+\gamma_1(\gamma_2+1)(\lambda-1)}{2(\lambda+\bar{\gamma}(\lambda-1))} \tag{15}$$

$$+\frac{\lambda^2+\gamma_1}{2(\lambda+2)(\gamma_1+1)(\lambda+\bar{\gamma}(\lambda-1))} \tag{16}$$

$$+\frac{\gamma_1(\lambda-1)}{2(\lambda+\bar{\gamma}(\lambda-1))} \cdot \frac{\gamma_1(\lambda+2)+1}{(\gamma_2+1)(\gamma_1+1)(\lambda+2)} \tag{17}$$

$$+\frac{\gamma_1\gamma_2(\lambda-1)}{2(\lambda+\bar{\gamma}(\lambda-1))} \cdot \frac{\gamma_2(\gamma_1+1)(\lambda+2)+(\lambda+1)}{(\gamma_2+1)(\gamma_1+1)(\lambda+2)} \tag{18}$$

$$\begin{aligned}
&+\frac{\gamma_1\gamma_2\gamma_3(\lambda-1)}{(\lambda+\bar{\gamma}(\lambda-1))} \cdot \Bigg[1-\exp\Bigg(-\frac{1}{\gamma_3+1} \cdot \\
&\qquad \left(\gamma_3+\frac{\gamma_1(\lambda+2)+1}{(\gamma_2+1)(\gamma_1+1)(\lambda+2)}\right)\Bigg)\Bigg].
\end{aligned} \tag{19}$$