On the Power of Advice and Randomization for Online Bipartite Matching*

Christoph Dürr[1], Christian Konrad[2], and Marc Renault[3]

1 **Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6, Paris, France**
Christoph.Durr@lip6.fr
2 **Reykjavik University, Reykjavik, Iceland**
christiank@ru.is
3 **IRIF, CNRS, Université Paris Diderot, Paris, France**
mrenault@liafa.univ-paris-diderot.fr

Abstract

While randomized online algorithms have access to a sequence of uniform random bits, deterministic online algorithms with advice have access to a sequence of *advice bits*, i.e., bits that are set by an all-powerful oracle prior to the processing of the request sequence. Advice bits are at least as helpful as random bits, but how helpful are they? In this work, we investigate the power of advice bits and random bits for online maximum bipartite matching (MBM).

The well-known Karp-Vazirani-Vazirani algorithm [24] is an optimal randomized $(1 - \frac{1}{e})$-competitive algorithm for MBM that requires access to $\Theta(n \log n)$ uniform random bits. We show that $\Omega(\log(\frac{1}{\epsilon})n)$ advice bits are necessary and $O(\frac{1}{\epsilon^5}n)$ sufficient in order to obtain a $(1-\epsilon)$-competitive deterministic advice algorithm. Furthermore, for a large natural class of deterministic advice algorithms, we prove that $\Omega(\log\log\log n)$ advice bits are required in order to improve on the $\frac{1}{2}$-competitiveness of the best deterministic online algorithm, while it is known that $O(\log n)$ bits are sufficient [9].

Last, we give a randomized online algorithm that uses cn random bits, for integers $c \geq 1$, and a competitive ratio that approaches $1 - \frac{1}{e}$ very quickly as c is increasing. For example if $c = 10$, then the difference between $1 - \frac{1}{e}$ and the achieved competitive ratio is less than 0.0002.

1998 ACM Subject Classification F.1.2 Online computation, G.2.2 Graph algorithms

Keywords and phrases On-line algorithms, Bipartite matching, Randomization

Digital Object Identifier 10.4230/LIPIcs.ESA.2016.37

1 Introduction

Online Bipartite Matching. The maximum bipartite matching problem (MBM) is a well-studied problem in the area of online algorithms [24, 5, 12]. Let $G = (A, B, E)$ be a bipartite graph with $A = [n] := \{1, \ldots, n\}$ and $B = [m]$, for some integers n, m. We assume $m = \Theta(n)$ allowing bounds to be stated as simple functions of n rather than of n and m. The A-vertices together with their incident edges arrive online, one at a time, in some adversarial chosen order $\pi : [n] \to [n]$. Upon arrival of a vertex $a \in A$, the online algorithm has to irrevocably decide to which of its incident (and yet unmatched) B-vertices it should be matched. The considered quality measure is the well-established *competitive ratio* [32], where the performance of an

* Research supported in part by the ANR projects ANR-11-BS02-0015, ANR-15-CE40-0015, ANR-12-BS02-005, by the Icelandic Research Fund grants-of-excellence no. 120032011 and 152679-051 and by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme no. 648032.

online algorithm is compared to the performance of the best offline algorithm: A randomized online algorithm **A** for MBM is c-competitive if the matching M output by **A** is such that $\mathbb{E}|M| \geq c \cdot |M^*|$, where the expectation is taken over the random coin flips, and M^* is a maximum matching.

In 1990, Karp, Vazirani and Vazirani [24] initiated research on online MBM and presented a $(1 - \frac{1}{e})$-competitive randomized algorithm denoted KVV. It chooses a permutation $\sigma : [m] \to [m]$ of the B-vertices uniformly at random and then runs the algorithm RANKING(σ), which matches each incoming A-vertex a to the free incident B-vertex b of minimum rank (i.e., $\sigma(b) < \sigma(c)$ for all free incident vertices $c \neq b$). If there is no free B-vertex, then a remains unmatched. They showed that no online algorithm has a better competitive ratio than $1 - \frac{1}{e}$, implying that KVV is optimal. For deterministic online algorithms, it is well-known that the GREEDY matching algorithm, which can be seen as running RANKING(σ) using a fixed arbitrary σ, is $\frac{1}{2}$-competitive, and is optimal for the class of deterministic online algorithms.

Improving on $1 - \frac{1}{e}$. Additional assumptions are needed in order to improve on the competitive ratio $1 - \frac{1}{e}$. For example, Feldman et al. [17] introduced the online stochastic matching problem, where a bipartite graph $G' = (A', B', E')$ and a probability distribution $\mathcal{D}$ is given to the algorithm. The request sequence then consists of vertices of A' that are drawn according to $\mathcal{D}$. Feldman et al. showed that the additional knowledge can be used to improve the competitive ratio to 0.67, which has subsequently been further improved [3, 27]. Another example is a work by Mahdian and Yan [26], who considered the classical online bipartite matching problem with a random arrival order of vertices. They analysed the KVV algorithm for this situation and proved that it is 0.696-competitive.

Online Algorithms with Advice. It is a common theme in online algorithms to equip an algorithm with additional knowledge that allows it to narrow down the set of potential future requests and, thus, design algorithms that have better competitive ratios as compared to algorithms that have no knowledge about the future. Additional knowledge can be provided in many different ways, e.g. access to lookahead [22, 19], probability distributions about future requests [17, 26], or even by giving an isomorphic copy of the input graph to the algorithm beforehand [21]. Dobrev et al. [13] and later Emek et al. [15] first quantified the amount of additional knowledge (advice) given to an online algorithm in an information theoretic sense. They showed that a specific problem requires at least $b(n)$ bits of advice, for some function b, in order to achieve optimality [13] or in order to achieve a particular competitive ratio [15]. Advice lower bounds are meaningful in practice as they apply to any potential type of additional information that could be given to an algorithm.

In the advice model, a computationally all-powerful oracle is given the entire request sequence and computes an advice string that is provided to the algorithm. Algorithms with advice are not usually designed with practical considerations in mind but to show a theoretical limit on what can be done. As such, the algorithms are often impractical due to the nature of the advice or the complexity in calculating the advice. However, from a theoretical perspective, advice algorithms are necessary to determine the exact advice complexity of online problems (how many advice bits are necessary and sufficient) and thus provide limits on the achievable and more practically relevant lower bounds.

Our Objective and Previous Results. Our objectives are to determine the advice complexity of MBM and to investigate the power of random and advice bits for this problem.

A starting point is a result of Böckenhauer et al. [9], who gave a method that allows the transformation of a randomized online algorithm into a deterministic one with advice with a similar approximation ratio. More precisely, given a randomized online algorithm **A** for a minimization problem $\mathcal{P}$ with approximation factor c and possible inputs $\mathcal{I}(n)$ of length n, Böckenhauer et al. showed that a $(1+\epsilon)c$-competitive deterministic online algorithm **B** with $\log n + 2\log\log n + \log \frac{\log|\mathcal{I}(n)|}{\log(1+\epsilon)}$ bits[1] of advice can be deduced from **A**, for any $\epsilon > 0$, where log is the binary logarithm in this paper. The calculation of the advice and the computations executed by **B** require exponential time, since **A** has to be simulated on all potential inputs $\mathcal{I}(n)$ on all potential random coin flips.

The technique of Böckenhauer et al. [9] can also be applied to maximization problems such as MBM[2]. Applied to the KVV algorithm, we obtain:

▶ **Theorem 1.** *There is a deterministic online algorithm with* $O(\log n)$ *bits of advice for* MBM *with competitive ratio* $(1-\epsilon)(1-1/e)$*, for any* $\epsilon > 0$*.*

This result is complemented by a recent result of Mikkelsen [28], who showed that for *repeatable problems* (see [28] for details) such as MBM, no deterministic online algorithm with advice sub-linear in n has a substantially better competitive ratio than any randomized algorithm without advice. Thus, using $O(\log n)$ advice bits, a $(1-\epsilon)(1-\frac{1}{e})$-competitive deterministic algorithm can be obtained, and no algorithm using $o(n)$ advice bits can substantially improve on this result. Furthermore, Miyazaki [29] showed that $\Theta(\log(n!)) = \Theta(n\log n)$ advice bits are necessary and sufficient in order to compute a maximum matching.

Our Results on Online Algorithms with Advice. Consider a deterministic online algorithm with $f(n)$ bits of advice for MBM. Our previous exposition of related works shows that the ranges $f(n) \in \Omega(\log n) \cap o(n)$ and $f(n) \in \Theta(n\log n)$ are well understood. In this work, we thus focus on the ranges $f(n) \in o(\log n)$ and $f(n) \in \Omega(n) \cap o(n\log n)$. Our first set of results concerns $(1-\epsilon)$-competitive deterministic advice algorithms. We show:

1. There is a deterministic $(1-\epsilon)$-competitive online algorithm, using $O(\frac{1}{\epsilon^5}n)$ advice bits for MBM.
2. Every deterministic $(1-\epsilon)$-competitive online algorithm for MBM uses $\Omega(\log(\frac{1}{\epsilon})n)$ bits of advice.

Our lower bound result is obtained by a reduction from the *string guessing game* of Böckenhauer et al. [6], a problem that is difficult even in the presence of a large number of advice bits. This technique has repeatedly been applied for obtaining advice lower bounds, e.g. [1, 20, 10, 2, 11, 4]. Our algorithm simulates an augmenting-paths-based algorithm by Eggert et al. [14], that has originally been designed for the data streaming model, with the help of advice bits. It is fundamentally different to the KVV algorithm, however, inspired by the simplicity of KVV, we are particularly interested in the following class of algorithms:

▶ **Definition 2** (Ranking-algorithm)**.** An online algorithm **A** for MBM is called Ranking-algorithm if it follows the steps: (1) Determine a ranking σ; (2) Return Ranking(σ).

The KVV algorithm is a Ranking-algorithm, where in step (1), the permutation σ is chosen uniformly at random. The algorithm described in Theorem 1 is a deterministic Ranking-algorithm with $O(\log n)$ bits of advice that computes the permutation σ from the

[1] Throughout the paper, logarithms, where the base is omitted, are implicitly binary logarithms.
[2] It is straightforward to adapt the proof of Theorem 5 of [9] accordlingly. For completeness, a proof is given in the full version of this paper.

available advice bits. While we cannot answer the question how many advice bits are needed for deterministic online algorithms in order to obtain a competitive ratio strictly larger than $\frac{1}{2}$ (and thus to improve on GREEDY), we make progress concerning RANKING algorithms:

3. Every RANKING-algorithm that chooses σ from a set of at most $C \log\log n$ permutations, for a small constant C, has approximation factor at most $(\frac{1}{2} + \delta)$, for any $\delta > 0$.

The previous result implies that every $(\frac{1}{2} + \delta)$-competitive deterministic online RANKING-algorithm requires $\Omega(\log\log\log n)$ advice bits.

Next, since the computation of the advice and the algorithm of Theorem 1 are not efficient, we are interested in fast and simple RANKING algorithms. We identify a subclass of RANKING algorithms, denoted CATEGORY algorithms, that leads to interesting results, both as deterministic algorithms with advice and randomized algorithms without advice.

▶ **Definition 3** (CATEGORY-algorithm)**.** A RANKING-algorithm **A** is called a CATEGORY-algorithm if it follows the steps:

- Determine a category function $c : B \to \{1, 2, 3, \ldots, 2^k\}$ for some integer $k \geq 1$ with $2^k < m$;
- Let $\sigma_c : [m] \to [m]$ be the unique permutation of the B-vertices such that for two vertices $b_1, b_2 \in B : \sigma_c(b_1) < \sigma_c(b_2)$ if and only if $c(b_1) < c(b_2)$ or $(c(b_1) = c(b_2)$ and $b_1 < b_2)$.
- Return RANKING(σ_c).

Categories can be seen as coarsened versions of rankings, where multiple items with adjacent ranks are grouped into the same category and within a category, the natural ordering by vertex identifier is used. We prove the following:

4. There is a deterministic $\frac{3}{5}$-competitive online CATEGORY-algorithm, using m bits of advice (and thus two categories).

The oracle determines the categories depending on whether a B-vertex would be matched by a run of GREEDY. We believe that this type of advice is particularly interesting since it does not require the oracle to compute an optimal solution.

Our Results on Randomized Algorithms. Last, we consider randomized algorithms with limited access to random bits. The KVV-algorithm selects a permutation σ uniformly at random, and, since there are $m!$ potential permutations, $\log(m!) = \Theta(m \log m)$ random bits are required in order to obtain a uniform choice. We are interested in randomized algorithms that employ fewer random bits. We consider the class of randomized CATEGORY-algorithms, where the categories of the B-vertices are chosen uniformly at random. We show:

5. There is a randomized CATEGORY-algorithm using km random bits with approximation factor $1 - \left(\frac{2^k}{2^k+1}\right)^{2k}$, for any integer $k \geq 1$.

For $k = 1$, the competitive ratio evaluates to $5/9$. It approaches $1 - 1/e$ very quickly, for example, for $k = 10$ the absolute difference between the competitive ratio and $1 - 1/e$ is less than 0.0002. Our analysis is based on the analysis of the KVV algorithm by Birnbaum and Mathieu [5] and uses a result by Konrad et al. [25] concerning the performance of the GREEDY algorithm on a randomly sampled subgraph which was originally developed in the context of streaming algorithms.

The results as described above are summarized in Table 1.

Models for Online Algorithms with Advice. The two main models for online computation with advice are the per-request model of Emek et at. [15] and the tape model of Böckenhauer et al. [7]. Both models were inspired by the original model proposed by Dobrev et al. [13]. In the model of Emek et at. [15], a bit string of a fixed length is received by the algorithm with

Table 1 Overview of our results, sorted with decreasing competitiveness.

Deterministic ratio	# of advice bits	Description and Authors
1	$\Theta(n \log n)$	(Miyazaki [29])
$1-\epsilon$	$O(\frac{1}{\epsilon^5} n)$	Application of Eggert et al. [14] (here)
$1-\epsilon$	$\Omega(\log(\frac{1}{\epsilon})n)$	LB holds for any online algorithm (here)
$1-\frac{1}{e}+\epsilon$	$\Omega(n)$	LB holds for any online algorithm (Mikkelsen [28])
$1-\frac{1}{e}$	$O(\log n)$	Exp. time RANKING-alg. (Böckenhauer et al. [9])
$\frac{3}{5}$	m	CATEGORY-algorithm using two categories (here)
$\frac{1}{2}+\epsilon$	$\Omega(\log\log\log n)$	LB holds for RANKING-algorithms (here)
Randomized ratio	**# of random bits**	**Description and Authors**
$1-\frac{1}{e}$	$m \log m$	KVV algorithm (Karp, Vazirani, Vazirani [24])
$1-\left(\frac{2^k}{2^k+1}\right)^{2k}$	km	CATEGORY-algorithm using 2^k categories (here)

each request for a total amount of advice that is at least linear in the size of the input. For this work, we use the tape model of Böckenhauer et al. [7], where the algorithm has access to an infinite advice string that it can access at any time (see Section 2 for a formal definition), allowing for advice that is sub-linear in the size of the input. Many online problems have been studied in the setting of online algorithms with advice (e.g. metrical task system [15], k-server problem [15, 9, 30, 20], paging [13, 7], bin packing problem [31, 11, 2], knapsack problem [8], reordering buffer management problem [1], list update problem [10], minimum spanning tree problem [4] and others). Interestingly, a variant of the algorithm with advice for list update problem of [10] was used to gain significant improvements in the compression rates for Burrows-Wheeler transform compression schemes [23]. The information-theoretic lower bound techniques for online algorithms with advice proposed by Emek et al. [15] applies to randomized algorithms and uses a reduction to a matching pennies game (essentially equivalent to the string guessing game). The reduction technique using the string guessing game of Böckenhauer et al. [6] is a refinement specifically for deterministic algorithms of the techniques of Emek et al.

Outline. Preliminaries are discussed in Section 2. Our $(1-\epsilon)$-competitive algorithm and a related advice lower bound are presented in Section 3. Then, in Section 4, we give the advice lower bound for $(\frac{1}{2}+\epsilon)$-competitive RANKING-algorithms. Last, in Section 5, we consider our randomized CATEGORY algorithm and our $\frac{3}{5}$-competitive advice CATEGORY algorithm.

2 Preliminaries

Unless stated otherwise, we consider a bipartite input graph $G=(A,B,E)$ with $A=[n]$ and $B=[m]$, for integers m,n such that $m=\Theta(n)$. The neighbourhood of a vertex v in graph G is denoted by $\Gamma_G(v)$. Let M be a matching in G. We denote the set of vertices matched in M by $V(M)$. For a vertex $v \in V(M)$, $M(v)$ denotes the vertex that is matched to v in M. Generally, we write M^* to denote a maximum matching, i.e., a matching of largest cardinality. For $A' \subseteq A, B' \subseteq B$, $opt(A',B')$ denotes the size of a maximum matching in $G[A' \cup B']$, the subgraph induced by $A' \cup B'$.

The Ranking Algorithm. Given permutations $\pi : [n] \to [n]$ and $\sigma : [m] \to [m]$, we write RANKING(G,π,σ) to denote the output matching of the RANKING algorithm when the A-vertices arrive in the order given by π, and the B-vertices are ranked according to σ. We may write RANKING(σ) to denote RANKING(G,π,σ) if π and G are clear from the context.

The Greedy Matching Algorithm. GREEDY processes the edges of a graph in arbitrary order and inserts the current edge e into an initially empty matching M if $M \cup \{e\}$ is a matching. It computes a maximal matching which is of size at least $\frac{1}{2}|M^*|$.

Category Algorithms. For an integer k, let $c : [m] \to \{1, \ldots, 2^k\}$ be an assignment of categories to the B-vertices. Then let $\sigma_c : [m] \to [m]$ be the unique permutation of the B-vertices such that for two vertices $b_1, b_2 \in B : \sigma_c(b_1) < \sigma_c(b_2)$ if and only if $c(b_1) < c(b_2)$ or $(c(b_1) = c(b_2)$ and $b_1 < b_2)$. The previous definition of σ_c is based on the natural ordering of the B-vertices. This gives a certain stability to the resulting permutation, since changing the category of a single vertex b does not affect the relative order of the vertices $B \setminus \{b\}$.

The Tape Advice Model. For a given request sequence I of length n for a maximization problem, an *online algorithm with advice* in the *tape advice model* computes the output sequence $\text{ALG}(I, \Phi) = \langle y_1, y_2, \ldots, y_n \rangle$, where y_i is a function of the requests from 1 to i of I and the infinite binary advice string Φ. Algorithm ALG has an advice complexity of $b(n)$ if, for all n and any input sequence of length n, ALG reads no more than $b(n)$ bits from Φ.

3 Deterministic $(1 - \epsilon)$-competitive Advice Algorithms

3.1 Algorithm With $O(\frac{1}{\epsilon^5}n)$ Bits of Advice

The main idea of our online algorithm is the simulation of an augmenting-paths-based algorithm with the help of advice bits. We employ the deterministic algorithm of Eggert et al. [14] that has been designed for the data streaming model. It computes a $(1 - \epsilon)$-approximate matching, using $O(\frac{1}{\epsilon^5})$ passes over the edges of the input graph, where each pass i is used to compute a matching M_i in a subgraph $G_i = G[A_i \cup B_i]$, for some subsets $A_i \subseteq A$ and $B_i \subseteq B$, using the GREEDY matching algorithm. In the first pass, M_1 is computed in G and thus constitutes a $\frac{1}{2}$-approximation. Let $M = M_1$. Then, $O(\frac{1}{\epsilon^2})$ phases follow, where in each phase, a set of disjoint augmenting paths is computed using $O(\frac{1}{\epsilon^3})$ applications of the GREEDY matching algorithm (and thus $O(\frac{1}{\epsilon^3})$ passes per phase). At the end of a phase, M is augmented using the augmenting-paths found in this phase. Upon termination of the algorithm, M constitutes a $(1 - \epsilon)$-approximation (see [14] for the analysis).

The important property that allows us to translate this algorithm into an online algorithm with advice is the simple observation that the computed matching M is a subset of $\bigcup_i M_i$. For every i, we encode the vertices $A_i \subseteq A$ and $B_i \subseteq B$ that constitute the vertices of G_i using $n+m$ advice bits. Furthermore, for every vertex $a \in A$, we also encode the index $j(a)$ of the matching $M_{j(a)}$ that contains the edge that is incident to a in the final matching M (if a is not matched in M, then we set $j(a) = 0$). Last, using $O(\log n)$ bits, we encode the integers n and m, using a self-delimited encoding. Parameters n, m are required in order to determine the word size that allows the storage of the indices $j(a)$, and to determine the subgraphs G_i. The total number of advice bits is hence $O(\frac{1}{\epsilon^5}(n + m) + \log(\frac{1}{\epsilon^5})(m) + \log(n)) = O(\frac{1}{\epsilon^5}n)$.

After having read the advice bits, our online algorithm computes the $O(\frac{1}{\epsilon^5})$ GREEDY matchings M_i simultaneously in the background while receiving the requests. Upon arrival of an $a \in A$, we match it to the $b \in B$ such that $ab \in M_{j(a)}$ incident to a if $j(a) \geq 1$, and we leave it unmatched if $j(a) = 0$. We thus obtain the following theorem:

▶ **Theorem 4.** *For every $\epsilon > 0$, there is a $(1 - \epsilon)$-competitive deterministic online algorithm for* MBM *that uses $O(\frac{1}{\epsilon^5}n)$ bits of advice.*

3.2 $\Omega(\log(\frac{1}{\epsilon})n)$ Advice Lower Bound

We complement the advice algorithm of the previous section with an $\Omega(\log(\frac{1}{\epsilon})n)$ advice lower bound for $(1-\epsilon)$-competitive deterministic advice algorithms. To show this, we make use of the lower bound techniques of [6] using the *string guessing game*, which is defined as follows.

▶ **Definition 5** (q-SGKH [6])**.** The *string guessing problem with known history* over an alphabet Σ of size $q \geq 2$ (q-SGKH) is an online minimization problem. The input consists of n and a request sequence $\sigma = r_1, \ldots, r_n$ of the characters, in order, of an n length string. An online algorithm A outputs a sequence $a_1, \ldots, a_n$ such that $a_i = f_i(n, r_1, \ldots, r_{i-1}) \in \Sigma$ for some computable function f_i. An important aspect of this problem is that the algorithm needs to produce its output character *before* the corresponding request: request r_i is revealed immediately after the algorithm outputs a_i. The cost of A is the Hamming distance between $a_1, \ldots, a_n$ and $r_1, \ldots, r_n$.

In [6], the following lower bound on the number of advice bits is shown for q-SGKH.

▶ **Theorem 6** ([6])**.** *Consider an input string of length n for q-SGKH. The minimum number of advice bits for any deterministic online algorithm that is correct for more than αn characters, for $\frac{1}{q} \leq \alpha < 1$, is $((1 - H_q(1-\alpha)) \log_2 q)n$, where $H_q(p) = p \log_q(q-1) - p \log_q p - (1-p) \log_q(1-p)$ is the q-ary entropy function.*

First, we define a sub-graph that is used in the construction of the lower bound sequence.

▶ **Definition 7.** A bipartite graph is *c-semi complete*, if it is isomorphic to $G = (A, B, E)$ with $A = \{a_1, \ldots, a_c\}, B = \{b_1, \ldots, b_c\}$, and $E = \{a_i, b_j : j \geq i\}$.

The following lemma presents the reduction from q-SGKH to MBM.

▶ **Lemma 8.** *For an integer $c \geq 3$, suppose that there is a deterministic ρ-competitive online algorithm for* MBM*, using bn bits of advice, where $1 - \frac{1}{c} + \frac{1}{c!} \leq \rho < 1$. Then, there exists a deterministic algorithm for $c!$-SGKH, using cbn bits of advice, that is correct for at least $(1 - (1-\rho)c)n$ characters of the n-length string.*

Proof. Let ALG_{MAT} be a deterministic ρ-competitive online algorithm for MBM, using bn bits of advice, with $1 - \frac{1}{c} + \frac{1}{c!} \leq \rho$, for an integer $c \geq 3$. We will present an algorithm $\text{ALG}_{c!\text{-SGKH}}$ that, in an online manner, will generate a request sequence I_{MAT} based on its input, I (of length n), that can be processed by ALG_{MAT}. Further, the advice received by $\text{ALG}_{c!\text{-SGKH}}$ will be the advice that ALG_{MAT} requires for I_{MAT}. As shown below, the length of I_{MAT} is cn, hence $\text{ALG}_{c!\text{-SGKH}}$ requires cbn bits of advice. The solution produced by ALG_{MAT} on I_{MAT} will define the output produced by $\text{ALG}_{c!\text{-SGKH}}$.

Suppose first that the entire input sequence I is known in advance (we will argue later how to get around this assumption). Let Π be an enumeration of all the permutations of length c, and let $g : \Sigma \to \{1, \ldots, c!\}$ be a bijection between Σ, the alphabet of the $c!$-SGKH problem, and an index of a permutation in Π. The request sequence I_{MAT} has a length of cn, consisting of n distinct c-semi-complete graphs, where each graph is based on a request of I. That is, for each request r_i in I, we append c requests to I_{MAT} that correspond to the A-vertices of a c-semi-complete graph, where the indices of the B-vertices are permuted according to the permutation $\Pi[g(r_i)]$.

Since I is not known in advance, we must construct I_{MAT} in an online manner while predicting the requests r_j. For each request r_j, the procedure is as follows:

Let I_{MAT}^{j-1} be the $c(j-1)$-length prefix of I_{MAT}. Note that when predicting request r_j, requests $r_1, \ldots, r_{j-1}$ have already been revealed, and I_{MAT}^{j-1} can thus be constructed. The

algorithm $\text{ALG}_{c!\text{-SGKH}}$ simulates ALG_{MAT} on I_{MAT}^{j-1} followed by another c-semi-complete graph $G_j = (A_j, B_j, E_j)$ such that, for $1 \leq k \leq c$, when vertex $a_k \in A_j$ is revealed, the B-vertices incident to a_k correspond exactly to the unmatched B-vertices of B_j in the current matching of ALG_{MAT}. By construction, ALG_{MAT} computes a perfect matching in G_j. The computed perfect matching corresponds to a permutation π at some index z of Π, and algorithm $\text{ALG}_{c!\text{-SGKH}}$ outputs $g^{-1}(z)$ as a prediction for r_j.

Consider a run of ALG_{MAT} on I_{MAT}. If ALG_{MAT} computes a perfect matching on the jth semi-complete graph, then our algorithm predicted r_j correctly. Similarly, if this matching is not perfect, then our algorithm failed to predict r_j. Let ν be the total number of imperfect matchings, let $\text{ALG}_{\text{MAT}}(I_{\text{MAT}})$ denote the matching computed by ALG_{MAT} on I_{MAT}, and let $\text{OPT}(I_{\text{MAT}})$ denote a perfect matching in the graph given by I_{MAT}. Then:

$$|\text{ALG}_{\text{MAT}}(I_{\text{MAT}})| \leq |\text{OPT}(I_{\text{MAT}})| - \nu \iff \nu \leq |\text{OPT}(I_{\text{MAT}})| - \rho \cdot |\text{OPT}(I_{\text{MAT}})| = (1-\rho)cn\,. \blacktriangleleft$$

We prove now the main lower bound result of this section.

▶ **Theorem 9.** *For an integer $c \geq 3$, any deterministic online algorithm with advice for* MBM *requires at least $\left(\frac{(1-H_q(1-\alpha))}{2}\log c\right) n$ bits of advice to be ρ-competitive for $1 - \frac{1}{c} + \frac{1}{c!} \leq \rho < 1$, where H_q is the q-ary entropy function and $\alpha = 1 - (1-\rho)c$.*

Proof. For $1 - \frac{1}{c} + \frac{1}{c!} \leq \rho < 1$, let ALG_{MAT} be a deterministic ρ-competitive online algorithm for MBM, using bn bits of advice. By Lemma 8, there exists an algorithm for $c!$-SGKH that uses cbn bits of advice and is correct for at least αn characters of the n-length input string. The bounds on ρ and c imply $1/(c!) \leq \alpha \leq 1$. Thus, Theorem 6 implies $cbn \geq ((1 - H_q(1-\alpha))\log(c!))n$ and, hence,

$$b \geq \frac{(1-H_q(1-\alpha))}{c}\log(c!) \geq \frac{(1-H_q(1-\alpha))}{2}\log c, \text{ as } c! \geq c^{c/2}. \blacktriangleleft$$

Setting $\varepsilon = 1/(2c) < 1/c - 1/(c!)$ for all $c \geq 3$, we get the following corollary. Note that, as ρ approaches 1 from below, α also approaches 1 from below and $H_q(1-\alpha)$ approaches 0.

▶ **Corollary 10.** *For any $0 < \varepsilon \leq 1/6$, any $(1-\varepsilon)$-competitive deterministic online algorithm with advice for* MBM *requires $O(\log(\frac{1}{\epsilon})n)$ bits of advice.*

4 Advice Lower Bound for Ranking Algorithms

Let $\sigma_1, \ldots, \sigma_k : [n] \to [n]$ be rankings. We will show that there is a $2n$-vertex graph $G = (A, B, E)$ and an arrival order π such that $|\text{RANKING}(G, \pi, \sigma_i)| \leq n(\frac{1}{2} + \epsilon) + o(n)$, for every σ_i and every constant $\epsilon > 0$, while G contains a perfect matching. Furthermore, the construction is such that $k \in \Omega(\log \log n)$.

The key property required for our lower bound is the fact that we can partition the set of B-vertices into disjoint subsets $B_1, \ldots, B_q$, each of large enough size, such that for every B_i with $B_i = \{b_1, \ldots, b_p\}$ and $b_1 < b_2 < \cdots < b_p$, the sequence $(\sigma_j(b_i))_i$ is monotonic, for every $1 \leq j \leq k$. In other words, the ranks of the nodes $b_1, \ldots, b_p$ appear in the rankings σ_i in either increasing or decreasing order. For each set B_i, we will construct a vertex-disjoint subgraph G_i on which RANKING computes a matching that is close to a $\frac{1}{2}$-approximation. The subgraphs G_i are based on graph H_z that we define next.

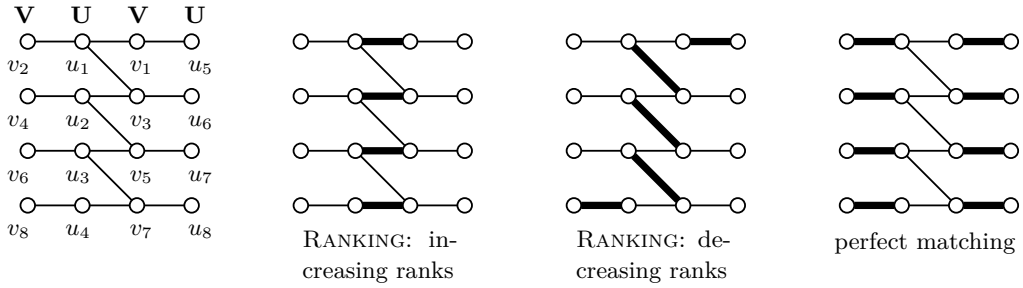

Figure 1 Left: U-vertices arrive in order $u_1, u_2, \ldots, u_8$. 'RANKING: increasing ranks' shows the resulting matching when $\sigma(v_1) < \sigma(v_2) < \cdots < \sigma(v_8)$. 'RANKING: decreasing ranks' shows the resulting matching when $\sigma(v_1) > \sigma(v_2) > \cdots > \sigma(v_8)$. Right: Perfect matching.

Construction of H_z. We construct now graph $H_z = (U, V, F)$ with $U = V = [z]$, for some even integer z, on which RANKING computes a matching that is close to a $\frac{1}{2}$-approximation, provided that the V vertices are ranked in either increasing or decreasing order.

Let $U = \{u_1, \ldots u_z\}$ be so that u_i arrives before u_{i+1} in π. Let $V = \{v_1, \ldots v_z\}$ be so that $v_i < v_{i+1}$ (which implies $v_i = i$). Then, for $1 \leq i \leq z/2$ we define $\Gamma_{H_z}(u_i) = \{v_{2i-1}, v_{2i}, v_{2i+1}\}$, and for $z/2 < i \leq z$ we define $\Gamma_{H_z}(u_i) = \{v_{2i-z-1}\}$. The graph H_8 is illustrated in Figure 1. It has the following properties:

1. If the sequence $(\sigma_i(b_j))_j$ is increasing, then $|\text{RANKING}(H_z, \pi, \sigma_i)| = z/2$.
2. If the sequence $(\sigma_i(b_j))_j$ is decreasing, then $|\text{RANKING}(H_z, \pi, \sigma_i)| = z/2 + 1$.
3. H_z has a perfect matching (of size z).

Lower Bound Proof. We prove first that we can appropriately partition the B-vertices that allow us to define the graphs G_i. Our prove relies on the well-known Erdős-Szekeres theorem [16] that we state in the form we need first.

▶ **Theorem 11** (Erdős-Szekeres [16]). *Every sequence of distinct integers of length n contains a monotonic (either increasing or decreasing) subsequence of length $\lceil \sqrt{n} \rceil$.*

▶ **Lemma 12.** *Let $\epsilon > 0$ be an arbitrary small constant. Then for any k permutations $\sigma_1, \ldots, \sigma_k : [n] \to [n]$ with $k \leq \log\log n - \log\log \frac{1}{\epsilon} - 2$, there is a partition of $B = [n]$ into subsets $C, B_1, B_2, \ldots$ such that:*

1. *$|B_i| \geq 1/\epsilon$ for every i,*
2. *$|C| \leq \sqrt{n}$,*
3. *For every $B_i = \{b_1, \ldots, b_p\}$ with $b_1 < b_2 < \cdots < b_p$, and every σ_j, the sequence $(\sigma_j(b_l))_l$ is monotonic.*

Proof. Let $S = B$. We iteratively remove subsets B_i from S until $|S| \leq \sqrt{n}$. The remaining elements then define set C. Thus, by construction, Item 2 is fulfilled.

Suppose that we have already defined sets $B_1, \ldots, B_i$. We show how to obtain set B_{i+1}. Let $S = B \setminus \bigcup_{j=1}^{i} B_j$ ($S = B$ if $i = 0$). Note that $|S| \geq \sqrt{n}$. By Theorem 11, there is a subset $B'_1 = \{b^1_1, \ldots, b^1_{\lceil n^{1/4} \rceil}\} \subseteq S$ with $b^1_1 < b^1_2 < \cdots < b^1_{\lceil n^{1/4} \rceil}$ such that the sequence $(\sigma_1(b_i))_{b_i \in B'_1}$ is monotonic. Then, again by Theorem 11, there is a subset $B'_2 = \{b^2_1, \ldots, b^2_{\lceil n^{1/8} \rceil}\} \subseteq B'_1$ with $b^2_1 < b^2_2 < \cdots < b^2_{\lceil n^{1/8} \rceil}$ such that the sequences $(\sigma_j(b_i))_{b_i \in B'_2}$ are monotonic, for every $j \in \{1, 2\}$. Similarly, we obtain that there is a subset $B'_w = \{b^w_1, \ldots, b^w_{\lceil n^{(1/2)^{w+1}} \rceil}\} \subseteq B'_{w-1}$ with $b^w_1 < b^w_2 < \cdots < b_{\lceil n^{(1/2)^{w+1}} \rceil}$ such that the sequences $(\sigma_j(b_i))_{b_i \in B'_w}$ are monotonic, for every $j \in \{1, \ldots, w\}$.

In order to guarantee Item 1, we solve the inequality $n^{(\frac{1}{2})^{w+1}} \geq \frac{1}{\epsilon}$ for w, and we obtain $w \leq \log\log n - \log\log\frac{1}{\epsilon} - 2$. This completes the proof. ◀

Equipped with the previous lemma, we are ready to prove our lower bound result.

▶ **Theorem 13.** *Let $\epsilon > 0$ be an arbitrary constant. For any k permutations $\sigma_1, \ldots, \sigma_k : [n] \to [n]$ with $k \leq \log\log n - \log\log\frac{2}{\epsilon} - 2$ and arrival order $\pi : [n] \to [n]$, there is a graph $G = (A, B, E)$ such that for every σ_i:*

$$|\textsc{Ranking}(G, \pi, \sigma_i)| \leq (\frac{1}{2} + \epsilon)n + o(n),$$

while G contains a perfect matching.

Proof. Let $\epsilon' = \epsilon/2$. Let $G = (A, B, E)$ denote the hard instance graph. Let $C, B_1, B_2, \ldots$ denote the partition of B according to Lemma 12 with respect to value ϵ'. Then, partition A into sets $A_0, A_1, \ldots$ such that $|A_0| = |C|$ and for $i \geq 1$, $|A_i| = |B_i|$. Graph G is the disjoint union of subgraphs $G_0 = (A_0, C, E_0)$ and $G_i = (A_i, B_i, E_i)$, for $i \geq 1$. Subgraph G_0 is an arbitrary graph that contains a perfect matching. If $|B_i|$ is even, then G_i is an isomorphic copy of H_i. If $|B_i|$ is odd, then G_i is the disjoint union of an isomorphic copy of H_{i-1} and one edge. Then,

$$|\textsc{Ranking}(G, \pi, \sigma_i)| \leq \sum_{B_i}(|B_i|/2 + 2) + |C| \leq n/2 + 2\epsilon' n + \sqrt{n}.$$ ◀

5 Category Algorithms

5.1 Randomized Category Algorithm

In this section, we analyse the following randomized Ranking-algorithm:

Algorithm 1 Randomized Category Algorithm

Require: $G = (A, B, E)$, integer parameter $k \geq 1$
For every $b \in B : c(b) \leftarrow$ random number in $\{1, 2, 3, \ldots, 2^k\}$
$\sigma_c \leftarrow$ permutation on $[m]$ such that $\sigma_c(b_1) < \sigma_c(b_2)$ iff $(c(b_1) < c(b_2))$ or $(c(b_1) = c(b_2)$ and $b_1 < b_2)$, for every $b_1, b_2 \in B$
return $\textsc{Ranking}(\sigma_c)$

Considering Graphs with Perfect Matchings. First, similar to [5], we argue that the worst-case performance ratio of Algorithm 1 is obtained if the input graph contains a perfect matching. It requires the following observation:

▶ **Theorem 14** (Monotonicity [18, 24])**.** *Consider a fixed arrival order π and ranking σ for an input graph $G = (A, B, E)$. Let $H = G \setminus \{v\}$ for some vertex $v \in A \cup B$. Let π', σ' be the arrival order/ranking when restricted to vertices $A \cup B \setminus \{v\}$. Then, $\textsc{Ranking}(G, \pi, \sigma)$ and $\textsc{Ranking}(H, \pi', \sigma')$ are either identical or differ by a single alternating path starting at v.*

The previous theorem shows that the size of the matching produced by Algorithm 1 is monotonic with respect to vertex removals. Hence, if H is the graph obtained from G by removing all vertices that are not matched by a maximum matching in G, then the performance ratio of Ranking on H cannot be better than on G. We can thus assume that the input graph G has a perfect matching and $|A| = |B| = n$.

Analysis: General Idea. Let $B_i = \{b \in B : c(b) = i\}$, and denote the matching computed by the algorithm by M. The important quantities to consider for the analysis of Algorithm 1 are the probabilities:

$$x_i = \Pr_{b \in B}\left[b \in V(M) \,|\, b \in B_i\right],$$

i.e., the probability that a randomly chosen B-vertex of category i is matched by the algorithm. Determining lower bounds for the quantities x_i is enough in order to bound the expected matching size, since

$$\begin{aligned}\mathbb{E}|M| &= \sum_{b\in B} \Pr\left[b \in V(M)\right] = \sum_{b\in B}\sum_{i=1}^{2^k} \Pr\left[b \in B_i\right] \cdot \Pr\left[b \in V(M) \,|\, b \in B_i\right] \\ &= \frac{1}{2^k}\sum_{b\in B}\sum_{i=1}^{2^k} \Pr\left[b \in V(M) \,|\, b \in B_i\right] = \frac{n}{2^k}\sum_{i=1}^{2^k} x_i. \qquad (1)\end{aligned}$$

We will first prove a bound on x_1 using a previous result of Konrad et al. [25]. Then, using similar ideas as Birnbaum and Mathieu [5], we will prove inequalities of the form $x_{i+1} \geq f(x_i, \ldots, x_1)$, for some function f which allow us to bound the probabilities $(x_i)_{i\geq 2}$.

Bounding x_1. Let $H = (U, V, F)$ be an arbitrary bipartite graph and let $U' \subseteq U$ be a uniform and random sample of U such that a node $u \in U$ is in U' with probability p. Konrad et al. showed in [25] that when running Greedy on the subgraph induced by vertices $U' \cup \Gamma_G(U')$, a relatively large fraction of the U'-vertices will be matched, for any order in which the edges of the input graph are processed that is independent of the choice of U'. More precisely, they prove the following theorem (Greedy(H', ω) denotes the output of Greedy on subgraph H' if edges of H' are considered in the order given by ω):

▶ **Theorem 15** ([25]). *Let $H = (U, V, F)$ be a bipartite graph, M^* a maximum matching, and let $U' \subseteq U$ be a uniform and independent random sample of U such that every vertex belongs to U' with probability p, $0 < p \leq 1$. Then for any edge arrival order ω,*

$$\mathbb{E}|\textsc{Greedy}(H[U' \cup \Gamma_H(U')], \omega)| \geq \frac{p}{1+p}|M^*|.$$

In Ranking, the vertices B_1 are always preferred over vertices $B \setminus B_1$. Thus, the matching $M_1 = \{ab \in M \,|\, b \in B_1\}$ is identical to the matching obtained when running Ranking on the subgraph induced by $A \cup B_1$. Since the previous theorem holds for any edge arrival order (that is independent from the choice of B'), we can apply the theorem (setting $B' = B_1, p = \frac{1}{2^k}$) and we obtain:

$$\mathbb{E}|B_1 \cap V(M)| \geq \frac{\frac{1}{2^k}}{1+\frac{1}{2^k}} n = \frac{1}{2^k+1} n.$$

Since $\mathbb{E}|B_1 \cap V(M)| = \sum_{b\in B} \Pr\left[b \in B_1\right] \cdot \Pr\left[b \in V(M) \,|\, b \in B_1\right] = \frac{n}{2^k} x_1$, we obtain $x_1 \geq 1 - \frac{1}{2^k+1}$.

Bounding $(x_i)_{i\geq 2}$. The key idea of the analysis of Birnbaum and Mathieu for the KVV-algorithm is the observation that, if a B-vertex of rank i is not matched by the algorithm, then its partner in an optimal matching is matched to a vertex of rank smaller than i.

Applied to our algorithm, if a B-vertex of category i is not matched, then its optimal partner $M^*(b)$ is matched to a B-vertex that belongs to a category $j \leq i$. Thus:

$$1 - x_i = \Pr_{b \in B}[b \notin V(M) \,|\, b \in B_i] = \Pr_{b \in B}[b \notin V(M) \text{ and } M^*(b) \text{ matched in } M \text{ to a } b' \text{ with } c(b') \leq i \,|\, b \in B_i]. \quad (2)$$

The following lemma is similar to a clever argument by Birnbaum and Mathieu [5].

▶ **Lemma 16.**

$$\Pr_{b \in B}[b \notin V(M) \textit{ and } M^*(b) \textit{ matched in } M \textit{ to a } b' \textit{ with } c(b') \leq i \,|\, b \in B_i] \leq \Pr_{b \in B}[M^*(b) \textit{ matched in } M \textit{ to a } b' \textit{ with } c(b') \leq i]. \quad (3)$$

Proof. Let c be uniformly distributed and let σ_c be the respective ranking. Pick now a random $\tilde{b} \in B$ and create new categories c' such that $c'(\tilde{b}) = i$ and for all $b \neq \tilde{b} : c'(b) = c(b)$. Let $\sigma_{c'}$ be the ranking given by c'.

Let $\tilde{a} = M^*(\tilde{b})$. Suppose that in a run of $\textsc{Ranking}(\sigma_{c'})$, $\tilde{a}$ is matched to a vertex d' with $c'(d') \leq i$ and $\tilde{b}$ remains unmatched. Then, we will show that in the run of $\textsc{Ranking}(\sigma_c)$, $\tilde{a}$ is matched to a vertex d with $c(d) \leq i$. This implies our result.

First, suppose that $\tilde{b}$ remains unmatched in $\textsc{Ranking}(\sigma_c)$. Then, $\textsc{Ranking}(\sigma_c) = \textsc{Ranking}(\sigma_{c'})$ and the claim is trivially true. Suppose now that $\tilde{b}$ is matched in $\textsc{Ranking}(\sigma_c)$. Then, similar to the argument of [5], it can be seen that $\textsc{Ranking}(\sigma_c)$ and $\textsc{Ranking}(\sigma_{c'})$ differ only by one alternating path $b_0, a_1, b_1, a_2, b_2, \ldots$ starting at $b_0 = \tilde{b}$ such that for all i, (1) $a_{i+1}b_i \in \textsc{Ranking}(\sigma_c)$, (2) $a_i b_i \in \textsc{Ranking}(\sigma_{c'})$, and (3) $\sigma_c(b_i) > \sigma_c(b_{i+1})$. Property (3) implies $c(b_i) \leq c(b_{i+1})$. Thus if the category $\sigma_{c'}$ of the node that a_i is matched to in $\textsc{Ranking}(\sigma_{c'})$ is k, then the category c of the node that a_i is matched to in $\textsc{Ranking}(\sigma_c)$ is also at most k. ◀

The right side of Inequality 3 can be computed explicitly as follows:

$$\begin{aligned} \Pr_{b \in B}[M^*(b) \text{ matched in } M \text{ to a } b' \text{ with } c(b') \leq i] &= \Pr_{b \in B}[c(b) \leq i \text{ and } b \in V(M)] \\ &= \frac{1}{2^k} \sum_{j=1}^{i} x_j. \end{aligned}$$

This, together with Inequalities 2 and 3, yields $1 - x_i \leq \frac{1}{2^k} \sum_{j=1}^{i} x_j$. We obtain:

▶ **Theorem 17.** *Let $k \geq 1$ be an integer. Then Algorithm 1 is a randomized online algorithm for* MBM *with competitive ratio $1 - \left(\frac{2^k}{2^k+1}\right)^{2k}$ that uses $k \cdot m$ random bits.*

Proof. Following [5], the inequality $1 - x_i \leq \frac{1}{2^k} \sum_{j=1}^{i} x_j$ yields $S_i(1 + \frac{1}{2^k}) \geq 1 + S_{i-1}$, where $S_i = \sum_{j=1}^{i} x_i$ and $S_1 = x_1 \geq 1 - \frac{1}{2^k+1}$. According to Equality 1, we need to bound S_{2^k} from below. Quantity S_{2^k} is minimized if $S_i(1 + \frac{1}{2^k}) = 1 + S_{i-1}$, for all $i \geq 2$, which yields

$$S_i = \sum_{j=1}^{i} (1 - \frac{1}{2^k+1})^j = 2^k \cdot \left(1 - \left(\frac{2^k}{2^k+1}\right)^i\right).$$

The result follows by plugging S_{2^k} into Equality 1. ◀

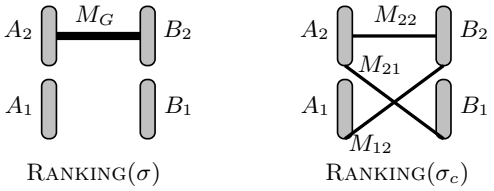

Figure 2 Quantities employed in the analysis of Algorithm 2.

5.2 Advice Category Algorithm

Let $\sigma : [m] \to [m]$ be the identity function, and let $M = \text{RANKING}(\sigma)$. It is well-known that M might be as poor as a $\frac{1}{2}$-approximation. Intuitively, B-vertices that are not matched in M are ranked too high in σ and have therefore no chance of being matched. We therefore assign category 1 to B-vertices that are not matched in M, and category 2 to all other nodes, see Algorithm 2. We will prove that this strategy gives a $\frac{3}{5}$-approximation algorithm.

Algorithm 2 Category-Advice Algorithm

Computation of advice bits

$\sigma \leftarrow$ permutation such that $\sigma(b) = b$, $M_G \leftarrow \text{RANKING}(\sigma)$, $M^* \leftarrow$ maximum matching

$$\forall b \in B : c(b) \leftarrow \begin{cases} 1, & \text{if } b \notin V(M), \\ 2, & \text{otherwise.} \end{cases}$$

Online Algorithm with Advice {Function c is provided using m advice bits}

$\sigma_c \leftarrow$ permutation on $[m]$ such that $\sigma_c(b_1) < \sigma_c(b_2)$ iff $(c(b_1) < c(b_2))$ or $(c(b_1) = c(b_2)$ and $b_1 < b_2)$, for every $b_1, b_2 \in B$

return RANKING(σ_c)

Our analysis requires a property of RANKING that has been previously used, e.g., in [5].

▶ **Lemma 18** (Upgrading unmatched vertices, Lemma 4 of [5]). *Let σ be a ranking and let $M = \text{RANKING}(\sigma)$. Let $b \in B$ be a vertex that is not matched in M. Let σ' be the ranking obtained from σ by changing the rank of b to any rank that is smaller than $\sigma(b)$ (and shifting the ranks of other vertices accordingly), and let $M' = \text{RANKING}(\sigma')$. Then, every vertex $a \in A$ matched in M to a vertex $b \in B$ is matched in M' to a vertex $b' \in B$ with $\sigma(b') \leq \sigma(b)$.*

▶ **Theorem 19.** *Alg. 2 is a $\frac{3}{5}$-competitive online algorithm for* MBM *using m advice bits.*

Proof. Let M denote the matching computed by the algorithm. Let $A_2 \subseteq A$, $B_2 \subseteq B$ be the subsets of vertices that are matched in M_G. Further, let $A_1 = A \setminus A_2$ and $B_1 = B \setminus B_2$ (the vertices not matched in M_G). See Figure 2 for an illustration of these quantities.

Then, for $i \in \{1, 2\}$, let $B_i^* = B_i \cap V(M^*)$. Let $M_{ij} = \{ab \in M \mid a \in A_i \text{ and } b \in B_j\}$. Then, $M = M_{21} \cup M_{12} \cup M_{22}$ since $M_{11} = \varnothing$ (the input graph does not contain any edges between A_1 and B_1 since otherwise some of them would also be contained in M_G). This setting is illustrated in Figure 2 in the appendix. We will bound now the sizes of M_{21}, M_{12} and M_{22} separately:

- *Bounding* $|M_{21}|$. Since B_1-vertices are preferred over B_2-vertices in RANKING(σ_c) and since there are no edges between A_1 and B_1, M_{21} is a maximal matching between A_2 and B_1. Since $opt(A_2, B_1) = |B_1^*|$, we have $|M_{21}| \geq \frac{1}{2}|B_1^*|$.
- *Bounding* $|M_{22}|$. By Lemma 18, all A_2-vertices are matched in M. Thus, $|M_{22}| = |A_2| - |M_{21}|$.

- *Bounding* $|M_{12}|$. The algorithm finds a maximal matching between A_1 and $B_2 \setminus B(M_{22})$. Since $opt(A_1, B_2) \geq |A_1^*|$, we have $opt(A_1, B_2 \setminus B(M_{22})) \geq |A_1^*| - |M_{22}|$, and thus $|M_{12}| \geq \frac{1}{2}(|A_1^*| - |M_{22}|)$.

We combine the previous bounds and we obtain:

$$|M| = |M_{21}| + |M_{22}| + |M_{12}| \geq |A_2| + \frac{1}{2}(|A_1^*| - |A_2| + |M_{21}|) \geq \frac{1}{2}(|A_1^*| + |A_2| + \frac{1}{2}|B_1^*|).$$

Next, note that $|A_2| \geq |B_1^*|$ and $|A_1^*| + |B_1^*| = |M^*|$. We thus obtain $|M| \geq \frac{1}{2}|M^*| + \frac{1}{4}|B_1^*|$. Since $|B_1^*| \geq |M^*| - |M_G|$, we obtain $|M| \geq \frac{3}{4}|M^*| - \frac{1}{4}|M_G|$. Furthermore, Lemma 18 implies $|M| \geq |M_G|$, and hence $|M| \geq \max\{|M_G|, \frac{3}{4}|M^*| - \frac{1}{4}|M_G|\}$ which is at least $\frac{3}{5}|M^*|$. ◀

References

1. Anna Adamaszek, Marc P. Renault, Adi Rosén, and Rob van Stee. Reordering buffer management with advice. In *11th International Workshop on Approximation and Online Algorithms (WAOA)*, pages 132–143, September 2013.
2. Spyros Angelopoulos, Christoph Dürr, Shahin Kamali, Marc Renault, and Adi Rosén. Online bin packing with advice of small size. In Frank Dehne, Jörg-Rüdiger Sack, and Ulrike Stege, editors, *Proceedings of the 14th International Symposium on Algorithms and Data Structures (WADS)*, pages 40–53. Springer International Publishing, August 2015.
3. Bahman Bahmani and Michael Kapralov. Improved bounds for online stochastic matching. In *Proceedings of the 18th Annual European Conference on Algorithms (ESA)*, pages 170–181. Springer-Verlag, 2010.
4. Maria Paola Bianchi, Hans-Joachim Böckenhauer, Tatjana Brülisauer, Dennis Komm, and Beatrice Palano. Online minimum spanning tree with advice. In *Proceedings of the 42nd International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pages 195–207, January 2016. doi:10.1007/978-3-662-49192-8_16.
5. Benjamin Birnbaum and Claire Mathieu. On-line bipartite matching made simple. *SIGACT News*, 39(1):80–87, March 2008. doi:10.1145/1360443.1360462.
6. Hans-Joachim Böckenhauer, Juraj Hromkovic, Dennis Komm, Sacha Krug, Jasmin Smula, and Andreas Sprock. The string guessing problem as a method to prove lower bounds on the advice complexity. *Theor. Comput. Sci.*, 554:95–108, 2014.
7. Hans-Joachim Böckenhauer, Dennis Komm, Rastislav Královič, Richard Královič, and Tobias Mömke. On the advice complexity of online problems. In Yingfei Dong, Ding-Zhu Du, and Oscar Ibarra, editors, *Proceedings of the 20th International Symposium on Algorithms and Computation (ISAAC)*, pages 331–340. Springer Berlin Heidelberg, December 2009.
8. Hans-Joachim Böckenhauer, Dennis Komm, Richard Královic, and Peter Rossmanith. The online knapsack problem: Advice and randomization. *Theor. Comput. Sci.*, 527:61–72, 2014.
9. Hans-Joachim Böckenhauer, Dennis Komm, Rastislav Královič, and Richard Královič. On the advice complexity of the k-server problem. In *Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 6755 of *Lecture Notes in Computer Science*, pages 207–218. Springer Berlin Heidelberg, July 2011. doi:10.1007/978-3-642-22006-7_18.
10. Joan Boyar, Shahin Kamali, Kim S. Larsen, and Alejandro López-Ortiz. On the list update problem with advice. In *Proceedings of the 8th International Conference on Language and Automata Theory and Applications (LATA)*, pages 210–221, March 2014. doi:10.1007/978-3-319-04921-2_17.
11. Joan Boyar, Shahin Kamali, Kim S. Larsen, and Alejandro López-Ortiz. Online bin packing with advice. *Algorithmica*, 74(1):507–527, 2016. doi:10.1007/s00453-014-9955-8.

12 Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized primal-dual analysis of RANKING for online bipartite matching. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 101–107, January 2013. doi:10.1137/1.9781611973105.7.

13 Stefan Dobrev, Rastislav Královič, and Dana Pardubská. How much information about the future is needed? In *Proceedings of the 34th conference on Current trends in theory and practice of computer science (SOFSEM)*, pages 247–258, Berlin, Heidelberg, 2008. Springer-Verlag.

14 Sebastian Eggert, Lasse Kliemann, Peter Munstermann, and Anand Srivastav. Bipartite matching in the semi-streaming model. *Algorithmica*, 63(1):490–508, 2011.

15 Yuval Emek, Pierre Fraigniaud, Amos Korman, and Adi Rosén. Online computation with advice. *Theor. Comput. Sci.*, 412(24):2642–2656, 2011. doi:10.1016/j.tcs.2010.08.007.

16 Paul Erdös and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

17 Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 117–126, Washington, DC, USA, 2009. IEEE Computer Society. doi:10.1109/FOCS.2009.72.

18 Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 982–991, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1347082.1347189.

19 Edward F. Grove. Online bin packing with lookahead. In *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 430–436, Philadelphia, PA, USA, 1995. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=313651.313781.

20 Sushmita Gupta, Shahin Kamali, and Alejandro López-Ortiz. On advice complexity of the k-server problem under sparse metrics. In *Proceedings of the 20th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 55–67, July 2013. doi:10.1007/978-3-319-03578-9_5.

21 Magnús M. Halldórsson. Online coloring known graphs. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 917–918, Philadelphia, PA, USA, 1999. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=314500.315088.

22 Magnús M. Halldórsson and Mário Szegedy. Lower bounds for on-line graph coloring. In *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 211–216, Philadelphia, PA, USA, 1992. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=139404.139452.

23 Shahin Kamali and Alejandro López-Ortiz. Better compression through better list update algorithms. In *Proceedings of the Data Compression Conference (DCC)*, pages 372–381, March 2014. doi:10.1109/DCC.2014.86.

24 R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing (STOC)*, pages 352–358, New York, NY, USA, 1990. ACM. doi:10.1145/100216.100262.

25 Christian Konrad, Frédéric Magniez, and Claire Mathieu. Maximum matching in semi-streaming with few passes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 7408 of *Lecture Notes in Computer Science*, pages 231–242. Springer Berlin Heidelberg, 2012. doi:10.1007/978-3-642-32512-0_20.

26 Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: An approach based on strongly factor-revealing lps. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing (STOC)*, pages 597–606, New York, NY, USA, 2011. ACM. doi:10.1145/1993636.1993716.

27 Vahideh H. Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. In *Proceedings of the Twenty-second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1285–1294. SIAM, 2011. URL: http://dl.acm.org/citation.cfm?id=2133036.2133134.

28 Jesper W. Mikkelsen. Randomization can be as helpful as a glimpse of the future in online computation. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, July 2016. URL: http://arxiv.org/abs/1511.05886.

29 Shuichi Miyazaki. On the advice complexity of online bipartite matching and online stable marriage. *Inf. Process. Lett.*, 114(12):714–717, December 2014. doi:10.1016/j.ipl.2014.06.013.

30 Marc P. Renault and Adi Rosén. On online algorithms with advice for the k-server problem. *Theory Comput. Syst.*, 56(1):3–21, 2015. doi:10.1007/s00224-012-9434-z.

31 Marc P. Renault, Adi Rosén, and Rob van Stee. Online algorithms with advice for bin packing and scheduling problems. *Theor. Comput. Sci.*, 600:155–170, 2015. doi:10.1016/j.tcs.2015.07.050.

32 Daniel D. Sleator and Robert E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, February 1985. doi:10.1145/2786.2793.