Extensions of bipartite matching

Szilvia Keszthelyi

Mathematics BSc

Thesis

Supervisor:

Kristóf Bérczi

ELTE Institute of Mathematics,

Department of Operations Research

Eötvös Loránd University

Faculty of Science

Budapest, 2019

Contents

Introduction

The aim of this thesis is to give an overview of bipartite matching problems and the related results focusing on the online input models, and to study their generalizations with matroid constraints.

The well-known theorems of Kőnig and Hall characterize the maximum cardinality of a matching in a bipartite graph. In the past decades, numerous extensions of the bipartite matching problem have been studied, many of them are motivated by real world problems. Such a model was introduced by Karp, Vazirani, and Vazirani [13]. In their model, called online bipartite matching, one side of the bipartition is known in advance and the vertices of the other side arrive in an online fashion. Upon the arrival of a vertex, its incident edges are revealed and an immediate and irrevocable matching decision should be made, possibly leaving the vertex unmatched. The goal is to maximize the size of the obtained matching. One of the real-world applications is advertising, where one side corresponds to the set of items and the other side to the set of buyers. Upon the arrival of a buyer, his preferred items are revealed and a matching decision should be made.

Since its first appearance, online bipartite matching has received considerable attention, therefore Chapter 1 provides a perspective on the allocation problems and the related input models. Although both the offline and online versions of the problems are described, the emphasis is on the latter one.

In Chapter 2, results on the online bipartite matching problem are described including the early theorems and conjectures of Karp, Vazirani and Vazirani and the latest work of Feige.

Lastly, in Chapter 3, two matroid generalizations of bipartite matching are considered. After a brief overview of the offline problems, greedy algorithms, as possible optimal online algorithms, are studied with the aim of giving bounds for the online problems.

Chapter 1

Landscape of the problems

There are numerous generalizations of bipartite matching, which are studied in different input models. This chapter focuses on results corresponding to the *online* variants.

1.1 Input models and arrival orders

There are several input models for bipartite allocation problems. These models differ from each other in the information available on the arriving vertices.

In the *offline model*, the algorithm knows the whole graph $G = (S, T; E)$ ahead of time. In the *online models*, in contrast, one side of the bipartiton is known in advance, but the vertices of the other partition and the edges arrive in an online fashion.

Online problems are studied under *adversarial order* in most of the cases, since it is the most general setting. In this model, no preliminary knowledge of the arriving vertices and edges is assumed. The algorithm knows only the vertices of S that have already arrived, edges incident to them and vertices of T. In the case of randomized algorithms, the adversary knows the code of the algorithm and it can create the worst possible input, but does not know the random choices.

In the *random arrival model*, the adversary selects the graph, but not the arrival order – the vertices of S arrive in a random order.

1.2 Online bipartite matching

In online bipartite matching, there is a bipartite graph $G = (S, T; E)$ where one side T, called offline vertices, is known in advance and the vertices of the other side S arrives online, one vertex at a time. When a vertex $s \in S$ arrives, edges incident to s are revealed. The arriving vertex can be matched to one of its unmatched neighbors. The goal is to maximize the size of the matching. This problem was introduced by Karp, Vazirani and Vazirani in 1990 [13].

Karp, Vazirani and Vazirani gave a randomized greedy algorithm, called Ranking (see Algorithm 3). An algorithm for online bipartite matching is called *greedy* if the only vertices of S that it leaves unmatched are those that upon their arrival do not have an unmatched neighbor. Ranking chooses a permutation π over the vertices of T uniformly at random, π is not known by the adversary, and when an online vertex s arrives, it matches s to the eligible neighbor t with the smallest rank $\pi(t)$ or leaves it unmatched if all the neighbors of s are unavailable. For bipartite graphs with maximum matching of size n, the *competitive ratio* of Ranking is at least $1 - (1 - \frac{1}{n+1})^n$, which tends to $1 - \frac{1}{e}$ as n goes to infinity. In 2018, Feige [8] showed that, when the input is chosen from a certain distribution D_n, no online algorithm matches more than $(1 - \frac{1}{e})n + 1 - \frac{2}{e} + \mathcal{O}(\frac{1}{n!})$ edges in expectation, thus the bound is essentially tight.

1.3 Competitive ratio

The performance of an online matching algorithm A is quantified by the competitive ratio of A, defined as the ratio of the expected size of matching obtained by A and the size of the maximum matching. The competitive ratio is determined by graphs that have a perfect matching [2, 7], in other words, when the input graphs have a perfect matching, the algorithms' competitive ratio is at most as much as that of the methods when the input graphs are arbitrary, see the proof in Section 2.2. Hence, for every n, consider the following class $\mathcal{G}_n$ of bipartite graphs. For every $G = (S, T; E) \in \mathcal{G}_n$, $|S| = |T| = n$ and G has a perfect matching. The following notations extend those introduced by Feige [8] and enable the generalization of the competitive ratio which is an algorithm-independent bound.

- $\rho(A, G)$ is the (expected) cardinality of matching produced by A when the input graph is G
- $\rho(\mathcal{A}, \mathcal{G})$ is the (expected) cardinality when A is chosen from a class of algorithms $\mathcal{A}$ and graph $G \in \mathcal{G}$ and $\rho(\mathcal{A}, \mathcal{G}) = \max_{A\in\mathcal{A}} \min_{G\in\mathcal{G}} \rho(A, G)$
- $\hat{\rho}(A) = \min_{G \text{ bipartite graph}} \frac{\rho(A,G)}{m}$ where m is the size of the maximum matching i.e. $\hat{\rho}(A)$ is the competitive ratio of A
- $\rho(-, \mathcal{G}) = \rho(\mathcal{G})$ is the maximum of $\rho(A, \mathcal{G})$ over all A
- $\hat{\rho} = \inf_n \frac{\rho(\mathcal{G}_n)}{n}$ is the largest constant such that $\hat{\rho} n \le \rho(\mathcal{G}_n)$ for all n

Competitive ratio is used for measuring the performance not only of the online bipartite matching, but of its generalizations. In the latter case, $\rho(A, G)$ denotes the value of the objective function attained by the algorithm A. If A is randomized then $\rho(A, G)$ is the expected value, and $OPT(G)$ is the optimal solution of the offline problem [16].

$$\hat{\rho}(A) = min_{G \text{ bipartite graph}} \frac{\rho(A, G)}{OPT(G)}$$

In general, let $\hat{\rho}$ be the largest constant such that $\hat{\rho} \le \min_G \frac{\rho(G)}{OPT(G)}$ where $\rho(G)$ is the maximum over all A of $\rho(A, G)$, as above. The main online allocation problems and their best known ratio $\hat{\rho}$ are summarized in Table 1.2.

1.4 Online vertex-weighted bipartite matching

The simplest generalization of online bipartite matching is when the offline vertices have weights. This was introduced by Aggarwal in [1]. In this version of bipartite matching, each vertex $t \in T$ has a non-negative weight w_t, which are known in advance. The goal is to maximize the sum of the weights of the matched vertices of T.

In this case, a greedy algorithm means that it matches the arriving vertex to the available neighbor with the highest weight. Since Ranking ignores the weights, it does not achieve the ratio of $1 - \frac{1}{e}$ if the weights are highly skewed. One can construct examples in which the ratio of Ranking goes to 0. However, the aforementioned greedy algorithm works well in this case, but has a competitive ratio of $\frac{1}{2}$ when the weights are equal. Algorithm 1, called Perturbed greedy, is a generalization of Ranking, but based on the other greedy algorithm as well. It runs greedy on perturbed weights, i.e. on $\tilde{w}_t = w_t(1 - e^{r_t - 1})$, where r_t is chosen from $U[0, 1]$. Instead of choosing a random permutation, it selects a permutation from a distribution that depends on the weights of the vertices such that vertices with higher weights have higher probability of being matched. In the case of equal weights, the algorithm is the same as Ranking since selecting r_t and choosing the highest $1 - e^{r_t - 1}$ is equivalent to choosing a random permutation and picking the highest ranked vertex. Perturbed greedy has competitive ratio of $1 - \frac{1}{e}$ for any set of weights. For further details, see Aggarwal [1].

Algorithm 1 PERTURBED GREEDY

```
1: initialize
2:     For every $t \in T$:
3:     Pick $r_t$ IID from $U[0, 1]$.
4:     Define its perturbed weight as $\tilde{w}_t := w_t(1 - e^{r_t - 1})$.
5: when a vertex $s \in S$ arrives
6:     If $s$ has no available neighbors, continue.
7:     Match $s$ to the available neighbor $t$ with the maximum value of $\tilde{w}_t$.
```

1.5 Adwords

The Adwords problem, introduced by Mehta [17], generalizes the online bipartite matching problem as follows. Each offline vertex $t \in T$ has a budget B_t, and edges $(s, t) \in E$ have bids bid_{st}. When a vertex $s \in S$ arrives, it may be matched to a neighbor t that has not spent all its budget. If s is matched to t, t depletes bid_{st} amount of its budget. Once a vertex depletes its entire budget, it becomes unavailable. The goal is to maximize the total budget spent.

The main motivation of Adwords problem is online advertising [16]. Sponsored search is an example for this, in which set T corresponds to advertisers who are in contract with a given search engine. Each advertiser puts in a bid value for each of the given keywords that are relevant to their ad, and they set a budget B_t serving as the limit of their total expenses. The search engine gets the ad-requests S sequentially as users search during the day. These are also called queries and ad-slots. Each ad-request can be allocated to one advertiser, who is charged its bid when the allocation is made. Once an advertiser runs out of its budget, it cannot be allocated to any more ad-slots. The objective in this model is to maximize the total amount of money spent by the advertisers. In some models, if an advertiser's remaining budget B'_t is less than its bid $bid_{s,t}$, then the search engine runs with B'_t as a new bid.

Small bid assumption

Adwords problem with small bid assumption is a special case of the above problem, motivated by realistic situations. For each edge (s, t), bid_{st} is small compered to B_t. The Adwords problem is mostly studied under the small bid assumption. When the bids are small, the algorithms can ignore the above correction of the bids, since overspending is not significant.

Algorithms

The best known competitive ratio differs between algorithms for Adwords with general bids and with small bids. *Greedy*, which allocates each s to that available t which has the maximum value of bid_{st}, achieves a ratio of $\frac{1}{2}$. The known $(1 - \frac{1}{e})$-competitive method works only in the case of small bids. It scales the bids with a function similar to the one used by Perturbed greedy, and then runs greedy on the scaled bids.

Adwords problem generalizes both online bipartite matching and online vertex-weighted matching problems. The special case when $B_t = w_t$ for all t and $bid_{st} = w_t$ for all $(s, t) \in E$ corresponds to the vertex-weighted version. Online bipartite matching is equivalent to Adwords when all budgets and bids are equal to 1. However, neither problem is a special case under the small-bids assumption.

1.6 Display Ads

Display Ads problem is an edge-weighted and capacitated generalization of online bipartite matching motivated by the display ads applications [9, 16]. In this problem, each edge $(s, t) \in E$ has a weight w_{st} and each vertex $t \in T$ has a capacity c_t which is an upper bound on the number of vertices that can be matched to t. When a vertex $s \in S$ arrives, it may be assigned to an available neighbor t. The goal is to maximize the total weight of the selected edges.

Practical examples fulfill the so-called *large capacity assumption*, that is, the capacities of the vertices are significantly larger than the weights of the edges. This special case was considered by Mehta [9, 16].

Relevant input models

If there is only one offline vertex and its capacity is $c_t = 1$, then the problem is identical to picking the maximum from a stream of numbers which arrive online. Therefore it is not possible to obtain any non-trivial competitive ratio for this model, since as soon as an algorithm chooses a number, the adversary can generate a much larger number subsequently. The random order input model and the *free-disposal model* are two motivated models, in which a better competitive ratio can be achieved. In the free-disposal model, edges can be removed later from the matching, but this choice is irrevocable.

1.7 Online submodular welfare maximization

Online submodular welfare maximization is a generalization of all the above problems (see Table 1.1). Each offline vertex $t \in T$ has a non-negative monotone submodular valuation function $f_t \colon 2^S \to \mathbb{R}^+$. When a vertex $s \in S$ arrives, it has to be allocated to a neighbor t. The allocation produces a partition of S, let S_t be the vertices of S allocated to t. The goal is to maximize $\sum_{t \in T} f_t(S_t)$.

A set function $f \colon 2^S \to \mathbb{R}^+$ is called *monotone* if $f(Y) \leq f(X)$ for every two sets $Y \subseteq X \subseteq S$, and *submodular* if for any two sets X and Y, $f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y)$. An equivalent form of submodularity is $f(Y \cup x) - f(Y) \geq f(X \cup x) - f(X)$ for every two sets $Y \subseteq X \subseteq S$ and an element $x \in S \setminus X$.

Algorithm

For online submodular welfare maximization problem, Algorithm 2, which is a greedy method, achieves a competitive ratio of $\frac{1}{2}$ [16]. Kapralov [12] showed that this ratio is optimal when vertices of S arrive in an adversarial order and the allocations are irrevocable. However, in the setting when vertices arrive in a uniformly random order, Buchbinder [4] proved that the same algorithm achieves a strictly larger competitive ratio. For submodular maximization, a greedy algorithm means that it allocates each $s \in S$ to $t \in T$ with the highest marginal gain.

Algorithm 2 GREEDY FOR ONLINE SUBMODULAR MAXIMIZATION

```
1: when a vertex $s \in S$ arrives
2:     For each $t \in T$, let $S_t$ denote the set of vertices in $S$ already allocated to $t$.
3:     Allocate $s$ to that $t$ which maximizes $f_t(S_t \cup s) - f_t(S_t)$.
```

Table 1.1: The aforementioned problems are special cases of the online submodular welfare maximization with the valuation functions below.

Problem	Valuation function
Bipartite matching	$\min\{1, \lvert S_t \rvert\}$
Vertex-weighted matching	$w_t \min\{1, \lvert S_t \rvert\}$
Adwords	$\min\{B_t, \sum_{s \in S_t} bid_{st}\}$
Display ads	$\max_{X \subseteq S_t, \lvert X \rvert \leq c_t}\{\sum_{s \in X} w_{st}\}$

1.8 Online bipartite matching under matroid constraints

The above problems can be generalized by introducing matroid constraints. One of these generalizations is when the objective is to maximize the size of the matching while the set of matched offline vertices are independent in a given matroid. In the case of online bipartite matching, offline vertices T and a matroid on T are known in advance and vertices of S arrive online, one by one along with their incident edges. Wang and Wong gave an $1 - \frac{1}{e}$-competitive algorithm in the random arrival model [20].

Adwords problem can be generalized this way as well and a greedy algorithm presented by Wang [20] achieves a ratio of $1 - \frac{1}{e}$ with the small bid assumption. This is a motivated problem, if an advertiser manages multiple ads with a fixed total budget, it corresponds to the matroid generalization of Adwords, with a partition matroid on T.

1.9 Online submodular maximization with matroid constraints

A matroid constrained generalization of submodular maximization was studied by Chan [6]. Each offline vertex $t \in T$ has a non-negative monotone submodular function $f_t \colon 2^S \to \mathbb{R}^+$ and a matroid $H = (S, \mathcal{I})$ is given on S. As a vertex $s \in S$ arrives, it may be assigned to a vertex $t \in T$ with respect to the matroid constraint, that is, the vertices allocated to t have to form an independent set $S_t \in \mathcal{I}$ for each $t \in T$. The objective is to maximize $\sum_{t \in T} f_t(S_t)$, just as in Section 1.7. Online bipartite matching is a special case corresponding to 1-uniform matroid constraints.

In the *free-disposal model*, one may remove some of the matched vertices in order to ensure the independence of a set S_t. In [6], authors gave a $\frac{1}{1+\alpha}$-competitive algorithm with free-disposal, where α is the root of $\alpha = e^{\alpha-2}$.

1.10 Offline versions

In the offline problem, the entire graph is known in advance. The well-known theorems of Hall and Kőnig give conditions on the existence of a matching covering one side of the bipartition and on the maximum cardinality of a matching in a bipartite graph [10, 15]. The offline version of vertex-weighted bipartite matching and display ads problems can be solved in polynomial time as well. However, the Adwords problem (without the small bid assumption) is NP-hard to approximate to any factor better than $\frac{15}{16}$ and the best known offline algorithm achieves $\frac{3}{4}$-approximation [5, 16, 19]. Adwords problem with small bid assumption can be approximated within a factor of $(1-4\varepsilon)$ if the bid to budget ratio $\frac{bid_{st}}{B_t}$ is at most ε for all s, t [5, 16]. The Submodular Welfare

Maximization problem is NP-hard to approximate better than $1 - \frac{1}{e}$ [14, 16]. Rado's theorem [18] generalizes Hall's theorem for the offline matroidal bipartite matching problem, see the details in Section 3.1.

Table 1.2: Summary of the best known approximation ratios for online matching problems.

Problem	$\hat{\rho}$
Bipartite matching	$1 - \frac{1}{e}$
Vertex-weighted matching	$1 - \frac{1}{e}$
Adwords	$\geq \frac{1}{2}$
• with small bids	$1 - \frac{1}{e}$
Display ads	0
• in random order	$\frac{1}{e}$
• with free disposal	$\geq \frac{1}{2}$
• with free disposal and large bids	$1 - \frac{1}{e}$
Submodular welfare maximization	$\frac{1}{2}$
Matroidal bipartite matching (in random order)	$1 - \frac{1}{e}$

Chapter 2

Online bipartite matching

In the online bipartite matching problem, the input is a bipartite graph $G = (S, T; E)$ where the offline vertices T are known in advance and the vertices of S arrive online. Edges incident to $s \in S$ are revealed when s arrives and s may be matched to an available adjacent vertex $t \in T$. The goal is to maximize the size of the matching.

2.1 Algorithms for online bipartite matching

When a vertex s arrives, an algorithm may match s to one of its unmatched neighbors or leave it unmatched irrevocably.

An algorithm for the online bipartite matching is called *greedy* if the only vertices of S that are left unmatched are those that do not have an available neighbor upon their arrival. Every non-greedy algorithm A can be replaced by a greedy method A' that produces at least as many edges as A does. Indeed, let s be a vertex that is left unmatched upon arriving, although it has an available neighbor. If s still has an unmatched neighbor t at the end of the algorithm, the matching produced by A can be extended with edge (s, t) to get a better solution. If algorithm A'' matches s to t and does the same as A otherwise, it has strictly better performance. In the other case, when A matches all the neighbors of s, let A'' differ from A in matching s to a neighbor t, which is available at the arrival of s. Thus, all the vertices of T that are matched by A are matched by A'' as well, hence A'' matches at least as many vertices as A does. By applying this to each non-greedy choice of A, one gets a greedy algorithm A'.

Every greedy algorithm produces a maximal matching and the size of a maximal matching is at least half of that of a maximum matching, since an edge can block at most two edges from the maximum matching.

For every deterministic algorithm, the adversary can select an input on which the algorithm produces only half the size of the maximum matching. For example, let the first $\frac{|S|}{2}$ arriving vertices be incident to all the vertices of T and let the remaining vertices of S be neighbors only of vertices that the algorithm matches to the first $\frac{|S|}{2}$ vertices.

2.2 Ranking

To improve the size of the matching beyond $\frac{n}{2}$, Karp, Vazirani and Vazirani [13] considered randomized methods. Algorithm 3 is the randomized greedy method of Karp, Vazirani and Vazirani [13], called Ranking. For bipartite graphs with maximum matching of size n, Ranking finds at least $(1-\frac{1}{e})n$ edges in expectation. They showed that this bound is essentially tight. When the input is chosen from a certain distribution $\mathcal{D}_n$, no online algorithm matches more than $(1-\frac{1}{e})n+\mathcal{O}(1)$ edges in expectation. In 2018, Feige [8] gave a tighter $(1-\frac{1}{e})n+1-\frac{2}{e}+\mathcal{O}(\frac{1}{n!})$ upper bound on the performance ratio of any online matching algorithm.

Algorithm 3 Ranking

```
1: initialize
2:     Choose a random permutation $\pi$ of $T$.
3: when a vertex $s \in S$ arrives
4:     If $s$ has no available neighbors, leave it unmatched.
5:     Else, match $s$ to a neighbor $t$ with the lowest value of $\pi(t)$.
```

The aforementioned distribution $\mathcal{D}_n$ is defined as follows. For each graph from $\mathcal{D}_n$, let the cardinality of the two sides be n i.e. $|S| = |T| = n$. Select a permutation τ over T. For every $i = 1, \dots, n$, $(s_j, t_i) \in E$ iff $j = \tau(k)$ for some $k \geq i$. $\mathcal{D}_n$ satisfies the following three conditions, suggesting that any graph from $\mathcal{D}_n$ is a worst input against Ranking.

- It has a unique perfect matching, which is the set of edges $(s_j, t_{\tau(j)})$ for $1 \leq j \leq n$.
- If s_j cannot be matched, neither can any of the vertices afterwards since all neighbors of $s_j + 1$ are also neighbors of s_j. Hence all the vertices in a prefix of S are matched and then no vertices in the resulting suffix are matched.
- An offline vertex $t_{\tau(j)}$ can be matched only in the first j rounds.

The expected size of the matching produced by Ranking is the same on every graph from $\mathcal{D}_n$ since Ranking generates a random permutation over T, hence graphs that only differ by a permutation over T are considered to be equal. Thus, consider one representative graph, called MonotoneG [8], which is shown on Figure 2.1. Permutation τ of MonotoneG is the identity permutation i.e. the set of the edges are $E = \{(s_i, t_j) | 1 \leq i \leq n, i \leq j \leq n\}$. The unique perfect matching is $M = \{(s_i, t_i) | 1 \leq i \leq n\}$.

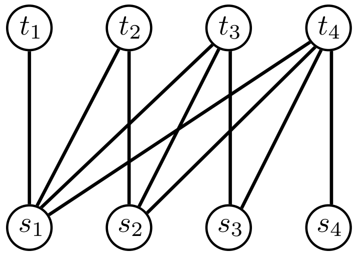

Figure 2.1: A representative graph of $\mathcal{D}_4$.

It is known that Ranking is optimal against $\mathcal{D}_n$, but it is an open question whether $\mathcal{D}_n$ is optimal against Ranking.

Conjecture 2.2.1. *$\rho(\mathcal{G}_n) = \rho(Ranking, \mathcal{D}_n)$ for all n.*

The above conjecture of Karp, Vazirani and Vazirani is still open, but the bounds are essentially tight. The theorems below summarize the known bounds on $\hat{\rho}$. Mehta [17] gave an upper bound.

Theorem 2.2.2. *For $\mathcal{D}_n$ defined as above, $\rho(\mathcal{D}_n) \leq (1 - \frac{1}{e})n + \mathcal{O}(1)$.*

Feige [8] proved that there exists a tighter bound on $\rho(\mathcal{D}_n)$.

Theorem 2.2.3. *Let the function $a(n)$ be such that $\rho(Ranking, \mathcal{D}_n) = \frac{a(n)}{n!}$ for all n. Then $a(n) = (n+1)! - d(n+1) - d(n)$, where $d(n)$ is the number of derangements (permutations with no fixed points) on the numbers $1, \ldots, n$. Consequently, $\rho(\mathcal{D}_n) = (1 - \frac{1}{e})n + 1 - \frac{2}{e} + \mathcal{O}(\frac{1}{n!})$, and this is an upper bound on $\rho(\mathcal{G}_n)$.*

Karp, Vazirani ang Vazirani [13] showed a lower bound on $\rho(\mathcal{G}_n)$, but the presented proof had gaps. Later, Goel and Mehta [11, 17] gave alternative proofs motivated by the Adwords problem.

Theorem 2.2.4. *For each bipartite graph $G \in \mathcal{G}_n$, the competitive ratio of Ranking algorithm is at least $1 - (1 - \frac{1}{n+1})^n$, which tends to $1 - \frac{1}{e}$ as n goes to infinity. Hence $\rho(Ranking) \geq (1 - \frac{1}{e})n$ and $\hat{\rho} \geq 1 - \frac{1}{e}$.*

Proof of Theorem 2.2.4

The proof below is based on the one presented by Birnbaum and Mathieu [2]. Let $Ranking(G, \sigma, \pi)$ denote algorithm Ranking on input graph G when the arrival order is permutation σ over S and the ranking is π over T. When G and σ are fixed, $Ranking(\pi)$ is the simplification of notation $Ranking(G, \sigma, \pi)$. Let $M_{\sigma,\pi}$ and M_π be the matchings produced by algorithms $Ranking(G, \sigma, \pi)$ and $Ranking(\pi)$, respectively. Let $M(v)$ be the neighbor of vertex v in a given matching M.

Lemma 2.2.5. *Let $v \in S \cup T$, $G' = G \setminus v$, and σ' and π' be permutations over S' and T' induced by σ and π, respectively. If the matchings produced by*

Ranking(G, σ, π) and Ranking(G', σ', π') are not identical, then they differ in a single alternating path starting at vertex v.

Proof. In set $M_{\sigma,\pi} \cup M_{\sigma',\pi'}$, the degree of each vertex is at most 2, therefore it is a disjoint union of paths and cycles. If $M_{\sigma,\pi} \cup M_{\sigma',\pi'}$ is a matching, then the two matchings are identical or $M_{\sigma,\pi}(v)$ cannot be matched after removing v. One has to show that if there is a path of length at least 2, then it is unique and starts at vertex v.

Suppose that there is a path P of length k where $k > 1$, which does not include v. Consider the first arriving $s \in S$ of P. If s is matched to t in the case of one input graph, but not for the other input graph, then in the second case, t must be matched to an earlier vertex, which is a contradiction. Thus such path does not exist. Cycles are not contained by $M_{\sigma,\pi} \cup M_{\sigma',\pi'}$ since a cycle means that Ranking matches the same vertices in different order in the two cases, which contradicts the construction of σ' and π'. $\square$

Claim 2.2.6. *The competitive ratio is determined by graphs that have a perfect matching.*

Proof. Let G be an arbitrary bipartite graph and fix σ and π. Let v be a vertex, $G' = G \setminus v$ and σ' and π' be the permutations over S' and T', just as in Lemma 2.2.5. If the matchings $M_{\sigma,\pi}$ and $M_{\sigma',\pi'}$ are not identical, then they differ in a single alternating path starting at vertex v, see Lemma 2.2.5. Thus, for every σ and π, $M_{\sigma,\pi}$ has at least as many edges as $M_{\sigma',\pi'}$. For all σ and π, $|M_{\tilde{\sigma},\tilde{\pi}}| \leq |M_{\sigma,\pi}|$ when $\tilde{G} \in \mathcal{G}_n$ for any n is the graph constructed from G by removing all vertices which are not in the maximum matching, $\tilde{\sigma}$ and $\tilde{\pi}$ are the permutations over $\tilde{S}$ and $\tilde{T}$. Therefore the size of the maximum matching is the same in both graphs, hence the competitive ratio does not increase. $\square$

Hence assume that G has a perfect matching. Fix graph G and arrival order σ. Let M^* be a perfect matching, and $n = |S|$. The following two lemmas are structural observations focusing on the ranks of the matched vertices.

Lemma 2.2.7. *Fix $s \in S$ and let $t = M^*(s)$. If t is not matched by Ranking(π), then s is matched to a vertex t' whose rank $\pi(t')$ is less than $\pi(t)$.*

Proof. If t is not matched by *Ranking*(π), then when s arrives, it has some available neighbors since t is one of them. Thus, s is matched to the available neighbor t' with the minimum rank, which is less than $\pi(t)$. $\square$

Lemma 2.2.8. *Let $s \in S$ and $t = M^*(s)$. Let π' be a permutation and let π_i be the permutation obtained from π' by changing the rank of vertex t to i, in other words, removing t from π' and putting it back in so its rank is i. If t is not matched by Ranking(π'), then for every i, s is matched by Ranking(π_i) to a vertex t_i with rank $\pi_i(t_i)$ at most $\pi'(t)$.*

Proof. By Lemma 2.2.7, s is matched to a vertex t' in $M_{\pi'}$ and $\pi'(t') < \pi'(t)$. If matchings $M_{\pi'}$ and M_{π_i} are not identical, then they differ along a single alternating path starting at vertex t with an edge in M_{π_i}, since t is unmatched by *Ranking*(π'). Moreover, the vertices of T traversed by the path have increasing rank in π_i. Thus, s is matched to vertex t_i by *Ranking*(π_i), whose rank is $\pi_i(t_i) \leq \pi_i(t')$. By definition of π_i, $\pi_i(t') \leq \pi'(t') + 1 \leq \pi'(t)$. $\square$

Lemma 2.2.9. *Let x_k denote the probability over π that the vertex of T of rank k is matched by the algorithm. Then $1 - x_k \leq \frac{1}{n} \sum_{1 \leq j \leq k} x_j$.*

Proof. Let $R_k \subseteq S$ denote the set of vertices that are matched by the algorithm to the vertices of T of rank at most k. Given a random permutation π over T, define a new random permutation π' by choosing a vertex $t \in T$ uniformly at random, taking it out of π and moving it back in so that its rank is k. Consider the matching produced by *Ranking*(π'). Let s be such that $t = M^*(s)$. The probability that t is not matched by *Ranking*(π') is $1 - x_k$. By Lemma 2.2.8, applied for i such that $\pi_i = \pi$, if t is not matched by *Ranking*(π'), then s is matched by *Ranking*(π) to a vertex $\tilde{t}$ such that $\pi(\tilde{t}) \leq k$, in other words, $s \in R_k$. The choice of t is random, so s is a vertex chosen uniformly at random as well and Lemma 2.2.8 holds for all i, thus s

is independent of π, hence independent of R_k. In other words, the event that $s \in R_k$ does not depend of π. Thus, conditional on π, the event that $s \in R_k$ has probability $\frac{|R_k|}{n}$. Taking expectations over π, one gets $\frac{1}{n}\sum_{1\le j\le k} x_j$. $\square$

With Lemma 2.2.9, the proof of Theorem 2.2.4 can be completed. Since the considered graphs have perfect matchings, the competitive ratio is the infimum of $\frac{1}{n}\sum_{1\le j\le n} x_j$. Let $s_k = \sum_{1\le j\le k} x_j$. Rewrite x_k as $s_k - s_{k-1}$ to get an equivalent form of Lemma 2.2.9 which is $1 + s_{k-1} \le s_k(1+\frac{1}{n})$. The infimum occurs when all inequalities are tight equalities. This yields a recursive formula $s_k = \frac{n}{n+1} + \frac{ns_{k-1}}{n+1}$. It follows that $s_k = \sum_{1\le j\le k}(1-\frac{1}{n+1})^j$. Hence, the competitive ratio is at least $\frac{1}{n}s_n = \frac{1}{n}\sum_{1\le j\le n}(1-\frac{1}{n+1})^j = 1-(1-\frac{1}{n+1})^n$, which tends to $1-\frac{1}{e}$ as n goes to infinity.

Sketch of the proof of Theorem 2.2.3

In what follows, the main steps of the proof of Feige [8] are presented.

Theorem 2.2.10. *Let function $a(n)$ be s.t. $\rho(Ranking, MonotoneG) = \frac{a(n)}{n!}$ for all n. Then $a(n) = (n+1)! - d(n+1) - d(n)$, where $d(n)$ is the number of derangements (permutations with no fixed points) of the numbers $1, \dots, n$.*

Proof. Fix n and $MonotoneG \in \mathcal{D}_n$ as the input. Let Π_n denote the set of all permutations over T. Ranking picks a permutation $\pi \in \Pi_n$ uniformly at random. Let $\pi(t_i)$ denote the rank of t_i under π and π_i be the vertex of rank i in π, that is $\pi_i = \pi^{-1}(t_i)$.

For $1 \le i \le n$, let $d(n,i)$ be the number of permutations over $1, \dots, n$ in which all fixed points (if any) are among the first i items.

Lemma 2.2.11. *The function $d(n,i)$ satisfies the following properties.*

- *$d(n,n) = n!$ for every $n \ge 1$,*
- *$d(n,i+1) = d(n,i) + d(n-1,i)$ for all $1 \le i < n$.*

Proof. The first statement holds, since $d(n, n)$ denotes the number of permutations on $1, \ldots, n$ with no restrictions, hence $d(n, n) = n!$. The combinatorial interpretation of the second statement is that $d(n, i + 1)$ is the sum of the number of permutations over $1, \ldots, n$ where $i+1$ is not a fixed point and the number of permutations in which $i + 1$ is a fixed point. $\square$

By a similar argument, $d(n + 1, 1) = d(n + 1) + d(n)$ follows. Therefore, it is sufficient to prove that $d(n + 1, 1) = (n + 1)! - a(n)$.

For $1 \leq i \leq n$, let $a(n, i)$ denote the number of permutations $\pi \in \Pi_n$ in which π_i is matched by $\mathit{Ranking}(\pi)$. (Notation $\mathit{Ranking}(\pi)$ is used as in the proof of Theorem 2.2.4.) Lemma 2.2.12 motivates the study of $a(n, i)$.

Lemma 2.2.12. *For $a(n)$ and $a(n, i)$ as defined above, it holds that $a(n) = \sum_{i=1}^{n} a(n, i)$.*

Proof. For a permutation $\pi \in \Pi_n$, let M_π denote the greedy matching produced by Ranking when it uses permutation π and the input graph is MonotoneG. Then by definition,

$$a(n) = \sum_{\pi \in \Pi_n} |M_\pi|.$$

By changing the order of summation, one gets that

$$\sum_{\pi \in \Pi_n} |M_\pi| = \sum_{i=1}^{n} a(n, i).$$

$\square$

Lemma 2.2.13. *Function $a(n, i)$ satisfies the following equations.*

- *$a(n, 1) = n!$ for every $n \geq 1$,*
- *$a(n, i) = a(n, i + 1) + a(n - 1, i)$ for all $1 \leq i < n$.*

Proof (sketch). The first equation holds, since s_1 is matched to π_i in every permutation π. For the second statement, define a bijection $B_i \colon \Pi_n \to \Pi_n$ for

$i < n$, where $B_i(\pi)$ flips the order between indices i and $i{+}1$, i.e. $B_i(\pi)_i = \pi_{i+1}$ and $B_i(\pi)_{i+1} = \pi_i$. There are three possibility when $t = \pi_i = B_i(\pi)_{i+1}$ is matched by any of $\mathit{Ranking}(\pi)$ and $\mathit{Ranking}(B_i(\pi))$ or neither of them. Using these events, Lemma 2.2.13 can be proved, but the rather long and technical proof is omitted here. See [8] for details. □

Lemma 2.2.14. *For $a(n,i)$ and $a(n)$ as defined above, $a(n) = (n+1)! - a(n+1, n+1)$.*

Proof. Applying Lemma 2.2.12 and the second statement of Lemma 2.2.13, one gets that $a(n) = \sum_{i=1}^{n} a(n,i) = \sum_{i=1}^{n}(a(n+1,i) - a(n+1,i+1)) = a(n{+}1,1) - a(n{+}1,n{+}1)$. Lemma 2.2.13 also states that $a(n{+}1,1) = (n{+}1)!$, proving the lemma. □

Applying both Lemma 2.2.14 and Lemma 2.2.11, Theorem 2.2.10 is equivalent to equation $d(n+1,1) = a(n+1,n+1)$, which holds by Lemma 2.2.15.

Lemma 2.2.15. *For every $1 \le i \le n$, it holds that $a(n,i) = d(n, n+1-i)$.*

Proof. The proof is by induction on n and i.

For the base case $n = 1$, necessarily $i = 1$ and $a(1,1) = 1 = d(1,1)$ holds by definition. For given $n > 1$, the base case is $i = 1$, and indeed $a(n,1) = n! = d(n,n)$ follows. For the inductive step, consider $a(n,i)$ where $1 < i \le n$, and assume that the inductive hypothesis holds for $n' < n$ and for n with $i' < i$. One easily gets that

$$\begin{aligned} a(n,i) &= a(n,i-1) - a(n-1,i-1) \\ &= d(n, n+2-i) - d(n-1, n+1-i) \\ &= d(n, n+1-i). \end{aligned}$$

The first and third equalities holds by Lemma 2.2.13 and Lemma 2.2.11, while the inductive hypothesis implies the second one. □

Claim 2.2.16 restates Theorem 2.2.3.

Claim 2.2.16. *For every n and $MonotoneG \in \mathcal{D}_n$,*

$$\rho(Ranking, MonotoneG) = (1 + \frac{1}{e})n + (1 - \frac{2}{e}) + \nu(n),$$

where $|\nu(n)| < \frac{1}{n!}$.

Proof. Theorem 2.2.10 shows that $a(n) = (n+1)! - d(n+1) - d(n)$. It is well-known that $d(n) = \frac{n!}{e}$ rounded to the nearest integer. Hence $|d(n) - \frac{n!}{e}| < \frac{1}{2}$ and $|a(n) - (1 - \frac{1}{e})(n+1)! - \frac{n!}{e}| < 1$. Dividing the latter inequality by $n!$ and replacing $(1 - \frac{1}{e})(n+1)$ by $(1 - \frac{1}{e})n + 1 - \frac{1}{e}$, it follows that

$$|\rho(Ranking, MonotoneG) - (1 - \frac{1}{e})n + 1 - \frac{1}{e} - \frac{1}{e}| < \frac{1}{n!},$$

proving the claim. $\square$

Chapter 3

Matroidal bipartite matching

In this section, two matroidal bipartite matching problems are studied. The first one is when there is a matroid on one side of the bipartition. In the online problem, this side is the offline class T. This online version and the related results were briefly mentioned in Section 1.8. The second one, when there is a matroid on the other side as well, has more open questions in the online case.

Matroids generalize linear independence in vector spaces, the notation was introduced by Whitney [21]. A matroid H is a pair $(S, \mathcal{I})$ where S is the *ground set* of the matroid and $\mathcal{I} \subseteq 2^S$ is a family of subsets of S that satisfy the following properties, called *independence axioms.*

(I1) $\emptyset \in \mathcal{I}$,

(I2) $X \subseteq Y \in \mathcal{I} \Rightarrow X \in \mathcal{I}$,

(I3) For every subset $X \subseteq S$, the maximal subsets of X which are in $\mathcal{I}$ have the same cardinality.

The members of $\mathcal{I}$ are called *independent* sets, all other subsets of S are *dependent.* Axiom (I3) means that the maximal independent subsets of each subset of S are equicardinal. This maximum number is called the *rank* of X and is denoted by $r(X)$, where r is the *rank function* of the matroid.

The rank of the ground set, $r(S)$, is the *rank of the matroid.* For a ground set S, the matroid in which every subset of S is independent is called the *free matroid.* As other examples of matroids, the *uniform* and the *partition matroids* are defined as follows. Let $\{S_1, \dots, S_k\}$ be a partition of S and let $g_1, \dots, g_k$ be non-negative integers. Declare a subset $I \subseteq S$ independent if $|I \cap S_i| \le g_i$ holds for each i. The matroid obtained this way is called the partition matroid. One might prove that its rank function r is given by $r(X) = \sum_i (\min\{g_i, |X \cap S_i|\})$. The uniform matroid is a partition matroid with $k = 1$. For a survey on matroid theory, see Frank [10].

3.1 H-independent matchings

Let $G = (S, T; E)$ be a simple bipartite graph and H a matroid on T. A matching is called *H-independent* if the vertices of T covered by the matching form an independent set of H. Note that H_E-independence is an other approach to obtain the same problem, since H determines a matroid on ground set E in which a subset F is independent if $d_F(t) \le 1$ for every $t \in T$ and the subset of T covered by F is independent in H – this way one gets a matroid H_E.

Rado [18] gave a matroid generalization of Hall's Theorem. For a subset $X \subseteq S$, let $\Gamma(X)$ denote the set of neighbors, i.e. $\Gamma(X) = \{t \in T : \text{there is an edge } (s,t) \in E \text{ with } s \in S\}$ [10].

Theorem 3.1.1 (Rado)**.** *Let $G = (S, T; E)$ be a bipartite graph and let H be a matroid on T. G has an H-independent matching covering S if and only if the following* Rado condition *holds,*

$$r(\Gamma(X)) \ge |X|$$

for every subset $X \subseteq S$.

The defect form of Rado's theorem [10] gives the maximum cardinality of the H-independent matching when the entire S cannot be covered, and it depends only on the extent of the violation of the previous condition.

Theorem 3.1.2 (Rado, defect form). *Let $G = (S, T; E)$ be a bipartite graph and H a matroid on T. The maximum cardinality of an H-independent matching is equal to*

$$\mu := \min_{X \subseteq S} \{r(\Gamma(X)) + |S - X|\}.$$

As the above theorems show, the conditions for the offline optimality are known.

In the online version of matroidal bipartite matching, the offline vertices T and a matroid H on T are known in advance and the vertices of S arrive online, one by one along with their incident edges, just as in Section 1.8. The objective is to maximize the size of an H-independent matching.

3.1.1 Algorithms

In general, a *greedy algorithm* for matroidal online bipartite matching is similar to the one given in Section 2.1 for online bipartite matching. It leaves $s \in S$ unmatched only if upon its arrival all of its neighbors are already matched or it has no unmatched neighbors $t \in T$ such that both $C_S \cup \{s\}$ and $C_T \cup \{t\}$ are independent sets in the appropriate matroid, where C_S and C_T denote the previously covered vertices in S and T, respectively. In this problem, the only matroid is on ground set T, therefore consider the above definition of the greedy algorithm with the free matroid on S.

Claim 3.1.3. *Greedy algorithms are optimal for the online bipartite H-independent matching problem.*

Proof. One has to show that for every non-greedy algorithm A, there exists a greedy A' that produces at least as many edges as A does. The construction goes as follows for a given graph $G = (S, T; E)$ and a matroid H on T. Consider the matching produced at the end of algorithm A and the event that a vertex $s \in S$ is left unmatched by algorithm A even though it had available neighbors at its arrival. There are three possible cases,

- s has a neighbor t which is matched by A to an $s' \in S$ arriving later than s,
- t is unmatched and the union of $\{t\}$ and the matched vertices of T is independent in the matroid,
- t is unmatched but the matched vertices and t are dependent all together.

In the first two cases, the argument in Section 2.1 can be applied. Otherwise, there exists a circuit containing t and some later-arriving vertices from which any vertex can be removed, so that the remaining set is independent. Thus, a vertex t' which is matched to a vertex s' that arrived later than s can be left out. So, if algorithm A' matches s to t (and leaves out edge (s', t')) for each $s \in S$ that A left unmatched by a non-greedy choice, then A' matches at least as many vertices as A does. □

Definition 3.1.4. An algorithm is an α-*approximation* for a problem if it finds a solution within a factor α of the optimum solution for every instance of the problem.

If the problem is a maximization, then $\alpha < 1$ and the approximate solution is at least α times the optimum solution by the definition. In the case of a minimization problem, $\alpha > 1$ and the approximate solution found by the algorithm is at most α times the optimum.

Claim 3.1.5. *Greedy is $\frac{1}{2}$-approximation algorithm for the H-independent matching problem.*

Proof. Let $G = (S, T; E)$ be a bipartite graph and H a matroid on T. Consider the following two matchings of G, the maximum H-independent matching produced by the greedy algorithm and a fixed optimal H-independent matching. Let P denote the vertices of T that are matched by the greedy algorithm and let R consists of those vertices in T that are covered by the optimal matching. Furthermore, let $P_1 = P \cap R$, $P_2 = P \setminus P_1$ and $R_2 = R \setminus (R_1 \cup P_1)$, where $R_1 \subseteq T$ is such that $t \in R_1$ if and only if t has a neighbor s for which edge (s,t) is in the maximum matching and s is matched by the greedy algorithm too, see Figure 3.1.

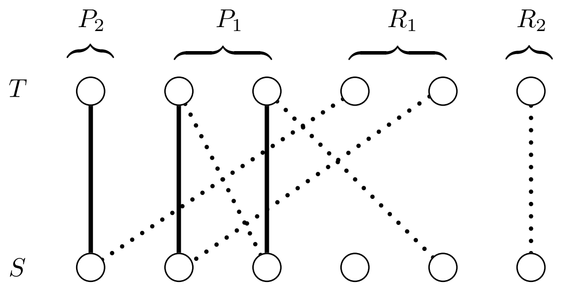

Figure 3.1: Subsets of T. Dotted edges correspond to the optimal matching and the solid ones are selected by the greedy algorithm.

The claim is equivalent to the following inequality.

$$|R| \leq 2|P|$$

Using the notations defined above, this can be reformulated as

$$|P_1| + |R_1| + |R_2| \leq 2(|P_1| + |P_2|)$$

which, in turn, is equivalent to

$$|R_1| + |R_2| \leq |P_1| + 2|P_2|.$$

By the definition, $|R_1| \leq |P_1| + |P_2|$. Thus, to conclude the proof, one has to show inequality $|R_2| \leq |P_2|$. Sets $P_1 \cup P_2$ and $P_1 \cup R_2$ are independent, since both are covered by one of the H-independent matchings, hence $r(P_1 \cup P_2) = |P_1| + |P_2|$ and $r(P_1 \cup R_2) = |P_1| + |R_2|$ holds. For all $t \in R_2$, $P_1 \cup P_2 \cup \{t\}$ is dependent by the definition of R_2. When some $s \in \Gamma(t)$ arrived, t is an available neighbor, yet both s and t are unmatched. This scenario occurs only if t is dependent with the already matched vertices, since the algorithm is greedy. This implies $r(P_1 \cup P_2 \cup R_2) = r(P_1 \cup P_2)$. However, $P_1 \cup R_2 \cup \{p\}$

might be independent for some $p \in P_2$, hence $r(P_1 \cup R_2) \leq r(P_1 \cup R_2 \cup \{p\}) \leq r(P_1 \cup R_2 \cup P_2)$ follows. Thus, one gets

$$|P_1| + |R_2| = r(P_1 \cup R_2) \leq r(P_1 \cup R_2 \cup P_2) = r(P_1 \cup P_2) = |P_1| + |P_2|.$$

This implies $|R_2| \leq |P_2|$ as required, finishing the proof. $\square$

3.2 Strongly independent matchings

Let $G = (S, T; E)$ be a bipartite graph and H_1 and H_2 be matroids on ground sets S and T, respectively. A matching is called *strongly independent* if it covers an independent set of vertices in both matroids. Brualdi [3, 10] gave a formulation which includes the theorems of Kőnig and Rado as special cases.

Theorem 3.2.1 (Brualdi)**.** *Let $G = (S, T; E)$ be a bipartite graph with matroids H_1 and H_2 on ground sets S and T, respectively. The largest cardinality of a strongly independent matching of G is equal to*

$$\min\{r_1(X) + r_2(Y) : X \subseteq S, Y \subseteq T, X \cup Y \text{ covers each edge of } G\}.$$

3.2.1 Algorithm for the online problem

In the online version, only the offline vertices T and a matroid H_2 on T are known in advance, and the vertices of S together with H_1 arrive online. The objective is to maximize the size of a strongly independent matching.

The greedy algorithm, defined above, is optimal if there is no matroid given on the online vertices, however, the above arguments cannot be extended to the case of two matroids. In the following instance, one cannot guarantee that the greedy matching has at least as many edges as the non-greedy one. Let s_1 be a vertex which has available neighbor t_1 at its arrival, and assume that the non-greedy algorithm leaves s_1 unmatched. If a later-arriving vertex s_2 is matched to t_1 and s_3 to t_2 and the union of $\{s_1, s_3\}$ and some of the previously matched vertices is dependent, then greedy – that matches s_1 – may produce strictly less edges, since it is possible that both s_2 and s_3 remain unmatched. Yet, greedy has the same approximation factor regardless whether a matroid is given on S or not. Note that one obtains Claim 3.1.5 as a special case of Claim 3.2.2 when H_1 is the free matroid. However, the arguments are entirely different, hence both are presented.

Claim 3.2.2. *Greedy is $\frac{1}{2}$-approximation algorithm for the strongly independent matching problem.*

Proof. Assume that the greedy algorithm selects the edges $e_1, \dots, e_k$ in this order. Let $\mathcal{F}$ denote the set of strongly independent matchings of G. Let

$$\theta_i = \max\{|F| : F \subseteq E, \{e_1, \dots, e_i\} \subseteq F \text{ and } F \in \mathcal{F}\}, \tag{3.1}$$

that is, θ_i is the cardinality of the largest strongly independent matching that contains the first i edges selected by the greedy method. The approximation factor follows from the inequality

$$\theta_i \geq \theta_{i-1} - 1 \text{ for all } i \geq 1. \tag{3.2}$$

Let M_i denote a matching attaining the maximum in equation (3.1) for all $i \in 1, \dots, k$. Consider the cases of how the size of the matching changes when e_i is added to the edges of M_{i-1} if some edges may be removed. Informally, one has to argue that when e_i is added to the edges of M_{i-1}, there exists (at most) two edges $f, f' \in M_{i-1}$ s.t. $(M_{i-1} \setminus \{f, f'\}) \cup \{e_i\}$ is a strongly independent matching.

For a given $1 \leq i \leq k$, if $e_i \in M_{i-1}$, then $\theta_i = \theta_{i-1}$ and inequality (3.2) holds. Otherwise $e_i = (s_i, t_i) \notin M_{i-1}$ and three conditions on s_i and t_i might be violated, namely they might be covered by another edge from M_{i-1} or might form a dependent set with some matched vertices.

Let us call a vertex v *available* if it is neither dependent of some of the vertices covered by M_{i-1} nor covered by it. Note that these cannot occur simultaneously and the roles of s_i and t_i are symmetric.

Either s_i or t_i is available, there exists a matching M'_{i-1} such that $e_i \in M'_{i-1}$ and $|M'_{i-1}| = \theta_{i-1}$, three distinct cases follow.

- Both s_i and t_i cannot be available, since then M_{i-1} would not be maximal.

- If t_i is available and s_i is dependent from a set $S' \subseteq S$ which is covered by M_{i-1}, then there exists a circuit containing s_i and from which removing a vertex s', s_i is independent with the remaining matched vertices, hence e_i can be added with the loss of only one other edge.

- The next case is when t_i is available and s_i is covered by another matched edge. Leaving that edge and adding e_i instead does not change the cardinality of the matching.

In the following three cases, at most two edges are needed to be removed in order to add e_i. Let t_i be in a circuit with some vertices of $T' \subseteq T$, covered by M_{i-1}.

- If s_i is in a circuit with some vertices of S', then there exist vertices s' and t' on the appropriate circuits such that removing them, both s_i and t_i are independent with the remaining matched vertices. This means that the edges incident to s' and t' are removed, but e_i is added, which makes a loss of at most one edge in total.

- If s_i is covered by M_{i-1}, then the removal of the edges incident to s_i and t' defined as above is sufficient to add e_i.

Otherwise,

- vertex t_i is covered by M_{i-1}, then the last remaining case is that M_{i-1} also covers s_i, but $(s_i, t_i) \notin M_{i-1}$. Replacing the two edges incident to s_i and t_i with e_i makes a strongly independent matching of size $\theta_{i-1} - 1$.

The above cases show that the strongly independent matchings derived from M_{i-1} have at least $\theta_{i-1} - 1$ edges and θ_i, by definition, is at least as large as the size of these matchings, hence inequality (3.2) holds for all i. By definition, θ_0 is the cardinality of the maximal strongly independent matching and θ_k is the cardinality of the greedy matching. To complete the proof, observe that

$$k = \theta_k \geq \theta_{k-1} - 1 \geq \ldots \geq \theta_0 - k$$
$$2k \geq \theta_0$$

follows from inequality (3.2) meaning that the size of the solution found by greedy is at least $\frac{1}{2}$ times the cardinality of the optimal matching. $\square$

3.2.2 Bounds for online strongly independent matching

Claim 3.2.2 gives a lower bound for the strongly independent matching problem, since greedy algorithms have a competitive ratio of at least $\frac{1}{2}$. The results on online bipartite matching serve as an upper bound, since it is a special case when the matroids are the free matroid. Hence any matroidal online bipartite matching algorithm has a competitive ratio of at most $1 - \frac{1}{e}$. However, it remains open to find the tight bound.

Acknowledgement

The project was supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

Bibliography

[1] G. Aggarwal, G. Goel, C. Karande, and A. Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1253–1264. SIAM, 2011.

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

[3] R. A. Brualdi. Common transversals and strong exchange systems. *Journal of Combinatorial Theory*, 8(3):307–329, 1970.

[4] N. Buchbinder, M. Feldman, and M. Garg. Online submodular maximization: Beating 1/2 made simple. *arXiv preprint arXiv:1807.05529*, 2018.

[5] D. Chakrabarty and G. Goel. On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and gap. *SIAM Journal on Computing*, 39(6):2189–2211, 2010.

[6] T. H. Chan, Z. Huang, S. H.-C. Jiang, N. Kang, and Z. G. Tang. Online submodular maximization with free disposal: Randomization beats 1/4 for partition matroids. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1204–1223. SIAM, 2017.

[7] A. Eden, U. Feige, and M. Feldman. Max-min greedy matching. *arXiv preprint arXiv:1803.05501*, 2018.

[8] U. Feige. Tighter bounds for online bipartite matching. *arXiv preprint arXiv:1812.11774*, 2018.

[9] J. Feldman, A. Mehta, V. Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 117–126. IEEE, 2009.

[10] A. Frank. Connections in combinatorial optimization. *Discrete Applied Mathematics*, 160(12):1875, 2012.

[11] G. Goel and A. Mehta. Online budgeted matching in random input models with applications to adwords. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 982–991. Society for Industrial and Applied Mathematics, 2008.

[12] M. Kapralov, I. Post, and J. Vondrák. Online submodular welfare maximization: Greedy is optimal. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1216–1225. SIAM, 2013.

[13] 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*, pages 352–358. ACM, 1990.

[14] S. Khot, R. J. Lipton, E. Markakis, and A. Mehta. Inapproximability results for combinatorial auctions with submodular utility functions. *Algorithmica*, 52(1):3–18, 2008.

[15] D. Kőnig. Graphok és matrixok. *Matematikai és Fizikai Lapok*, 38:116–119, 1931.

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

[17] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. Adwords and generalized on-line matching. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 264–273. IEEE, 2005.

[18] R. Rado. A theorem on independence relations. *The Quarterly Journal of Mathematics*, 13(1):83–89, 1942.

[19] A. Srinivasan. Budgeted allocations in the full-information setting. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 247–253. Springer, 2008.

[20] Y. Wang and S. C.-w. Wong. Matroid online bipartite matching and vertex cover. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 437–454. ACM, 2016.

[21] H. Whitney. On the abstract properties of linear dependence. In *Hassler Whitney Collected Papers*, pages 147–171. Springer, 1992.