

Matching Algorithms with advices, and multi-agent problems

The simplest version of online matching problem is the following [24, 35]. Given a bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E})$, where $\mathcal{U}$ and $\mathcal{V}$ are two distinct sets of vertices (the supply and the demand) and $\mathcal{E} \subset \mathcal{U} \times \mathcal{V}$ is a set of edges, the objective is to construct a **matching** $\mathcal{M}$, i.e., a subset of $\mathcal{E}$ such that any vertex $u \in \mathcal{U}$ or $v \in \mathcal{V}$ belongs to at most one edge of $\mathcal{M}$ [19, 43, 30, 4]. The “quality” of a matching is simply its cardinality $|\mathcal{M}|$, and we shall denote (with a slight abuse of notations as it depends on $\mathcal{G}$) by $\mathcal{M}^* \subset \mathcal{E}$ any optimal matching with maximal cardinality.

In the **online** variant of the problem [24], the matching $\mathcal{M}$ is constructed sequentially. Although the set $\mathcal{U}$ is known at the beginning, elements of $\mathcal{V}$ unveil one at a time: first $v_1 \in \mathcal{V}$ is observed, along with $\mathcal{E}_1 := \{(u, v_1) \in \mathcal{E}\}$. A decision-maker/platform/algorithm decides to pair v_1 with some u_1 so that $(u_1, v_1) \in \mathcal{E}_1$; this forms the matching $\mathcal{M}_1$. Then $v_2 \in \mathcal{V}$ is observed, along with $\mathcal{E}_2 := \{(u, v_2) \in \mathcal{E}\}$. The decision-maker/platform/algorithm decides, if possible, to pair v_2 with some u_2 so that $e_2 := (u_2, v_2) \in \mathcal{E}_2$ and $\mathcal{M}_2 := \mathcal{M}_1 \cup \{e_2\}$ is still a matching. This process is repeated until all elements of $\mathcal{V}$ are observed (and potentially paired) and this construct an online matching $\mathcal{M}$. The performance of an online algorithm [27, 5, 25] is called the **competitive ratio** [16] (for this unknown graph $\mathcal{G}$), which is simply $|\mathcal{M}|/|\mathcal{M}^*|$. The overall objective is then to control the minimum of this ratio over all possible graphs $\mathcal{G}$ (or over an admissible class of graphs).

To give an idea of typical results, it is quite possible to show that any deterministic algorithm cannot achieve a worst-case competitive ratio smaller than $1/2$, and so does the greedy one (that picks at each iteration uniformly at random an admissible edge). On the other hand, a slight variation, that selects first a ranking over $\mathcal{U}$ at random, and then picks the admissible vertex $u_k \in \mathcal{U}$ with the smallest rank achieves a competitive ratio of $1 - \frac{1}{e}$, which is optimal in the generic adversarial case [24, 13, 2]. Those competitive ratios can be improved with additional assumptions: if the vertex arrival order is chosen at random and/or the underlying bipartite graph is itself chosen at random [20, 32], etc.

The problem of online matching has many variants, and generalizations (or simplifications actually, such as prophet inequalities [31, 12], that can also be studied in the fixed order, random order, with iid random variables or not, etc. [34, 22, 8, 15]) and they all fall in the category of “online computations” [5]; maybe one of the most famous one is called the k -server problems [33]. There are strong connections with stochastic multi-armed bandits [10, 26] (yet they have only been very scarcely investigated in the literature) where the objective is to sequentially learn **which action is optimal** in a given set. Unlike these online learning problems (bandits and reinforcement), the target of online algorithms problem rather consists in finding out **when to use this action**.

Typical proof techniques rely on reductions to some optimization problems solved sequentially (by constructing a “factor revealing” program [23, 32, 2], by controlling simultaneously the primal and its dual [11] or by trying to track the optimal current algorithms [5]) and/or on careful - sometimes lengthy - computations.

The standard online matching problem described above is interesting but quite limited; few of the motivating examples can be appropriately modelled by it. This explains why many variants have already been investigating. Probably one of the more general one (called “adwords” as it was used to model this specific problem) consists in assuming that vertices $u \in \mathcal{U}$ have a budget B_u and edges (u, v) a cost $b_{u,v}$. Then the global objective consists in sequentially constructing a subset of edges $\mathcal{M} \subset \mathcal{E}$ that maximizes $\sum_{(u,v) \in \mathcal{M}} b_{u,v}$ such that the budget constraints are satisfied, i.e., $\sum_{v: (u,v) \in \mathcal{M}} b_{u,v} \leq B_u$. As before, a greedy algorithm would have a competitive ratio of only $1/2$, contrary to variants (with similar proof techniques), while adapted version of rankings [36] still get a ratio of $1 - \frac{1}{e}$.

Oracle-Guided Algorithms There is an interesting reason to get better, or even case-dependent, competitive ratios. The global idea is that algorithms might have, sometimes, good a-priori about the solution of a problem [1, 14]. For instance, finding a word in a dictionary of size n takes $\log_2(n)$ comparison in the worst-case; yet in practice, an agent knows the “global” position of a word and used it to lower the number of search (for instance, it’s clear that baobab is at the beginning while yoghourt at the end). This idea has been integrated to the standard primal/dual approach, where the algorithm gets some advices from some “Oracle”, to get improved ratios [37, 21, 28]. However, those advices are usually exogenous, and not constructed or learned from the data. **Learning the Oracle**, using **online learning** approaches for instance and then plugging it into **online**

algorithms should provide improve competitive ratios. Finally, let us mention that state of the art usually assume that predictions are perfect, and an algorithm should then be “consistent”, or very bad and then the criteria to achieve is the “robustness”. The dichotomy between those two objectives is quite arbitrary and not really fulfilling [42]. We will aim at defining **intermediate and more relevant regimes**.

Oracle guided Speed Scaling The first simple setup that we are investigating to combine Oracle guided algorithms and temporal dependencies is called “speed scaling” (a sub-task of scheduling). In this setting n tasks arrive sequentially at time τ_i and takes c_i computation units to be completer. But there is only one machine (for the moment) whose time can be split to solve those tasks. Denote by f_i the final completion time of task i . The objective is then to minimize $\sum_{i=1}^n [f_i - \tau_i]$. When c_i is known in advance ”Shortest Remaining Processing Time” is optimal [40]. Assume now that c_i is unknown but an Oracle gives a prediction $\hat{c}_i$; very positive recent results prove that the variant of SRPT (run with respect to $\hat{c}_i$ instead of c_i) can be combined to the naïve Round Robin algorithm to improve competitive ratios if predictions are “good enough”. Although we wonder whether it should not be actually combined with another algorithm, agnostic to completion time called ”Shortest Elapsed Time First ” [39], it will be more interesting to construct the Oracle while the algorithm is running. For instance, assume that the task i is parameterized by i.i.d. feature vector $X_i \in \mathbb{R}^d$ and that the computation time follows a linear model $c_i = X_i^\top \beta + \varepsilon_i$. Then the Oracle will be the output of a learning algorithm of β , based on delayed-feedback bandit techniques (tasks must be completed before getting relevant information).

Multi-agents Algorithms. A variant of the matching problem is to assume that vertices $u \in \mathcal{U}$ are agent that learn themselves, in a decentralized way, some strategies. A typical approach would be to assume that they follow independently some online learning algorithm (as data are gathered and treated on the fly), a.k.a, bandits algorithms (or online convex optimization) [9]. The bandit literature is now almost exhaustive, with thousands of variants of the basic model introduced, so one might think that some off-the-shelf algorithm could be directly used. This is actually incorrect because there are not just one agent that is learning, but several of them. Hence the problem falls into the realm of **multi-player multi-armed bandit**, which recently gains a lot of interest [6, 29]. The motivating reason is the emergence of 5G networks, where users will be able to choose bandwidths to transmit data (they are of different quality, and shall be understood as a set of arms; the bandwidth is the reward of an arm). If two users select at the same time the same bandwidth, they interfere, lost packets and their rewards is set to 0 instead. The objective is then to devise decentralized protocol that achieves (hopefully) the same asymptotic performances than the centralized one. This setting is quite interesting as it incorporates several agents are learning at the same time, and that decisions of one of them impact the others. We started investigating multi-agents online problems beyond the bandit realm, into online queuing system [17, 18, 41]. We were able to devise new protocols, leveraging **ideas coming from multi-player multi-armed bandits** (to achieve the same results as in the totally centralized case) **combined to standard offline matching algorithms**. This approach is therefore quite promising for more general multi-agent online computations problems.

Interactions between agents can be more subtle than mere interference; for instance, in **strategic experiments** [3, 38] agents observe the behavior/decisions/algorithms of the other and they can use this as extra (and free) information. This yielded the question of finding not only decentralized learning algorithms, but more importantly, **equilibrium of algorithms** (with typical questions such as: does observing other agents exploration policy increase, or decrease, the global exploration over all agents ?). We successfully answered this question in a restricted setting [7] using **repeated game ideas** (Grim Trigger strategies to be precise), and this paves the way to exciting new challenges.

Multi-Agent Prophet Inequalities and Other Online Problems Introducing multi-agent decision in classical prophet inequalities settings would be quite interesting. For instance, similarly to the multi-armed bandit motivations, agents take decisions independently (their stopping times) and get as reward the random variable when they stopped, on the condition that there are only one agent stopping at that time. Otherwise, there is a collision and the reward shrinks to 0.

The similarity to multi-player bandit [6] or queuing systems [41] we investigated will be very valuable.

References

- [1] Étienne Bamas, Andreas Maggiori, and Ola Svensson. The primal-dual method for learning augmented algorithms. *Advances in Neural Information Processing Systems*, 33:20083–20094, 2020.
- [2] Benjamin Birnbaum and Claire Mathieu. On-line bipartite matching made simple. *SIGACT News*, 39(1):80–87, March 2008.
- [3] Patrick Bolton and Christopher Harris. Strategic experimentation. *Econometrica*, 67(2):349–374, 1999.
- [4] Charles Bordenave, Marc Lelarge, and Justin Salez. Matchings on infinite graphs. *Probability Theory and Related Fields*, 157(1-2):183–208, 2013.
- [5] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
- [6] Etienne Boursier and Vianney Perchet. Sic-mmab: synchronisation involves communication in multiplayer multi-armed bandits. *Advances in Neural Information Processing Systems*, 32, 2019.
- [7] Etienne Boursier and Vianney Perchet. Selfish robustness and equilibria in multi-player bandits. In *Conference on Learning Theory*, pages 530–581. PMLR, 2020.
- [8] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. Online stochastic matching: New algorithms and bounds, 2019.
- [9] S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Machine Learning*, 5(1):1–122, 2012.
- [10] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Machine Learning*, 5(1):1–122, 2012.
- [11] Niv Buchbinder, Joseph Seffi Naor, et al. The design of competitive online algorithms via a primal–dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009.
- [12] Jose Correa, Patricio Foncea, Ruben Hoeksma, Tim Oosterwijk, and Tjark Vredeveld. Recent developments in prophet inequalities. *ACM SIGecom Exchanges*, 17(1):61–70, 2019.
- [13] Nikhil Devanur, Kamal Jain, and Robert Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. 01 2013.
- [14] Michael Dinitz, Sungjin Im, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Faster matchings via learned duals. *Advances in Neural Information Processing Systems*, 34, 2021.
- [15] Paul Dutting, Michal Feldman, Thomas Kesselheim, and Brendan Lucier. Prophet inequalities made easy: Stochastic optimization by pricing nonstochastic inputs. *SIAM Journal on Computing*, 49(3):540–582, 2020.
- [16] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and Shan Muthukrishnan. Online stochastic matching: Beating $1-1/e$. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 117–126. IEEE, 2009.
- [17] Jason Gaitonde and Éva Tardos. Stability and learning in strategic queuing systems. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 319–347, 2020.
- [18] Jason Gaitonde and Eva Tardos. Virtues of patience in strategic queuing systems. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 520–540, 2021.
- [19] Christopher David Godsil. Matchings and walks in graphs. *Journal of Graph Theory*, 5(3):285–297, 1981.
- [20] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. pages 982–991, 01 2008.
- [21] Sungjin Im, Ravi Kumar, Mahshid Montazer Qaem, and Manish Purohit. Non-clairvoyant scheduling with predictions. In *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 285–294, 2021.
- [22] Patrick Jaillet and Xin Lu. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research*, 39(3):624–646, 2014.
- [23] Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *Journal of the ACM (JACM)*, 50(6):795–824, 2003.
- [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 '90, page 352–358, New York, NY, USA, 1990. Association for Computing Machinery.
- [25] Dennis Komm. *Introduction to Online Computation*. Springer, 2016.
- [26] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

- [27] Harry R Lewis and Christos H Papadimitriou. Elements of the theory of computation. *ACM SIGACT News*, 29(3):62–78, 1998.
- [28] Alexander Lindermayr and Nicole Megow. Non-clairvoyant scheduling with predictions revisited. *arXiv preprint arXiv:2202.10199*, 2022.
- [29] L. Liu, H. Mania, and M. I.—Jordan. Competing bandits in matching markets. In *arXiv:1906.05363*, pages 1–15, 2019.
- [30] László Lovász and Michael D Plummer. *Matching theory*, volume 367. American Mathematical Soc., 2009.
- [31] Brendan Lucier. An economic view of prophet inequalities. *ACM SIGecom Exchanges*, 16(1):24–47, 2017.
- [32] 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*, pages 597–606, 2011.
- [33] Mark S Manasse, Lyle A McGeoch, and Daniel D Sleator. Competitive algorithms for server problems. *Journal of Algorithms*, 11(2):208–230, 1990.
- [34] Vahideh H Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research*, 37(4):559–573, 2012.
- [35] Aranyak Mehta. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4):265–368, 2013.
- [36] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. *Journal of the ACM (JACM)*, 54(5):22–es, 2007.
- [37] Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ml predictions. *Advances in Neural Information Processing Systems*, 31, 2018.
- [38] Dinah Rosenberg, Antoine Salomon, and Nicolas Vieille. On games of strategic experimentation. *Games and Economic Behavior*, 82:31–51, 2013.
- [39] Tim Roughgarden. Beyond worst-case analysis. *Communications of the ACM*, 62(3):88–96, 2019.
- [40] Linus Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research*, 16(3):687–690, 1968.
- [41] Flore Sentenac, Etienne Boursier, and Vianney Perchet. Decentralized learning in online queuing systems. *Advances in Neural Information Processing Systems*, 34, 2021.
- [42] Alexander Wei and Fred Zhang. Optimal robustness-consistency trade-offs for learning-augmented online algorithms. *Advances in Neural Information Processing Systems*, 33:8042–8053, 2020.
- [43] Lenka Zdeborová and Marc Mézard. The number of matchings in random graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(05):P05003, 2006.