Online Fair Division under Stochastic Inputs

Christodoulos Karavasilis

April 2021

1 Introduction

We consider an online fair division problem, with T indivisible items arriving one by one and having to be allocated to n (offline) agents. In each time step $t \in [T]$, a new item arrives, and it must be irrevocably assigned to an agent. Each agent i has a valuation function v_i that assigns a value to every item. We consider additive valuation functions where an agent's valuation of the bundle of items assigned to them is the sum of the utility assigned to each item. The online setting for this classical fair division problem is motivated by applications such as food banks [Ale+15], or a company's allocation of computing resources amongst its employees. More information on the different variations of online fair division can be found in a survey by Aleksandrov and Walsh [AW20]. The quality of a solution to this problem is measured in terms of efficiency and fairness. Some algorithms sacrifice one for the other, while ideally we would satisfy both. The trade-off between the two has been investigated both for the offline version of the problem where the entire utility profile is known in advance [Car+19], and more recently in the online setting as well [ZP20]. As is often the case with theoretical work, worst-case performance against powerful adversaries is what is usually considered. Still, the performance of algorithms on more realistic inputs is worth investigating. This is sometimes achieved by relaxing the power of the adversary or assuming some structure on the input. This is an experimental study comparing various well-known algorithms using various fairness and efficiency metrics, on inputs generated from artificial distributions and real data. We also try and motivate the use of algorithms that take advantage of a known input structure.

2 Model

The classical online setting has the algorithm work against an adversary that chooses both the input items and their order of arrival (usually picked in advance, without knowing the algorithm's random coin flips). In this problem, the adversary can be thought of as having to pick the utility profile of each item. Motivated by the literature on online bipartite matching [Meh13; Fel+09], we consider a model where input items are drawn independently from the same distribution. The most relevant work is by Zeng and Psomas [ZP20], who consider this model exactly (and some additional ones). Another way to think of this input model (instead of thinking about utility profiles being drawn at each step) is the following: Consider all the m different item types, a type being specified by a unique utility profile, that can arrive online. The algorithm has access to all the different types, and their arrival rates, namely the amount of times we expect to see a type arrive online after T steps. We will consider integral arrival rates, and w.l.o.g.[1] we assume

[1] We can multiply a type as many times as its arrival rate, while giving it a new name.

each type has arrival rate 1. In each step, a new item is drawn i.i.d. from this known distribution. This terminology is often used in the matching literature and it was also used by Zeng and Psomas [ZP20]. Bogomolnaia et al [BMS21] considers a similar setting and extend [ZP20]. They point out the similarity with the stochastic models considered in the matching literature, and explicitly define a *competitive ratio* for their setting. Another somewhat related paper that aims to provide more practical results is by Sinclaire et al [Sin+20].

To the best of our knowledge, [ZP20] were the first to consider an algorithm for this model (of course algorithms from the purely adversarial setting transfer here, and so do their guarantees). There's a striking resemblance between their algorithm and algorithms considered in the matching literature [HMZ11], and it lies in line with the idea of solving an offline problem on the "expected" instance, and using that solution to guide the online allocation.

3 Metrics

The performance of different algorithms is evaluated with respect to the following efficiency and fairness metrics:

Social Welfare: This is a purely efficiency metric, adding the total utility of all agents.

Envy: This a fairness metric, computing the maximum pair-wise envy amongst all pairs of agents. We say that an agent i with bundle of items A_i envies agent j with bundle of items A_j, if $u_i(A_j) > u_i(A_i)$.

Nash Social Welfare: A metric offering a balance between fairness and efficiency, is given by the geometric mean of the utility valuations of the agents under an allocation $(\prod_{i\in[n](u_i(A_i))^{\frac{1}{n}}})$.

Maxmin: This is another fairness metric, denoting the minimum utility of an agent.

Using these metrics we see how each algorithm fares in terms of efficiency and fairness, and we can compare them. It is not always possible to compare these empirical results with theoretical bounds, because an algorithm might have only been studied for a different variation of the problem (divisible items, normalized valuations, etc) and a direct comparison might not be very informative. Another reason such comparisons are hard, is because computing an optimal solution to the offline instance is (sometimes) computationally hard, and we're unable to get an empirical competitive ratio to measure against theoretical bounds.

4 Algorithms

The algorithms used are some of the more commonly cited and studied. As in most online problems, these online algorithms run in polynomial time. Most of the following algorithms have the additional feature that they are conceptually simple, and very attractive from an implementation point of view.

Uniform Random. This algorithm allocates all items amongst the agents uniformly at random, without taking any utilities under consideration. This algorithm was shown to have the *vanishing envy* property (Envy/T goes to zero as T goes to infinity) [Ben+18] in the classical online setting.

Utility Greedy. This greedy algorithm tries to maximize the social welfare by allocating each item to the agent with the highest utility. It's easy to see this maximizies the social welfare, but may starve some agents, especially if utilites aren't normalized.

Proportional Random. This algorithm was inspired by the proportional algorithm for divisible goods in the classical adversarial setting, that allocates each agent a fraction of the item, proportional to their utility (for agent i and item t, a fraction $v_{it}/\sum_{j\in[n]} v_{jt}$). We consider a randomized variant that assigns the item to an agent with probability proportional to their utility. Gorokh et al [Gor+20] showed that for the divisible case, proportional allocation pareto dominates uniform allocation, and it achieves a competitive ratio of $\sqrt{n}/2$ with respect to the NSW of an optimal allocation.

NSW Greedy. This greedy algorithms maximizes the Nash Social Welfare at each step. For the divisible case, [Gor+20] also showed that this algorithm achieves a competitive ratio $\Omega(n)$.

The above algorithms are all blind with respect to the distributional information given to the algorithm. The algorithm by Zeng and Psomas [ZP20] works by computing an allocation maximizing the NSW for the fractional version of the problem on the expected (distributional) instance, and then using that fractional allocation as a guide to assign items online. When a new item t arrives, assign it to an agent i with probability proportional to the fractional allocation x_{it}. The solution to the fractional (divisible) version of the problem is computed though convex programming, specifically by computing a Fisher market equilibrium using the Eisenberg-Gale convex program. In the offline setting, for the special case of binary utilities and indivisible goods, Barman et al [BKV18], give a simple combinatorial algorithm for computing an allocation that maximizes NSW. Their algorithm takes as input any sub-optimal allocation, and then through a series of greedy (chain)swaps of items between agents, incrementally improves the solution. One could test an algorithm that finds a solution maximizing NSW on the distributional (binary) instance, and use that as an online guide to (deterministically) assign each incoming item to the agent its type was allocated to in the distributional instance. We leave this as future work. Lastly, motivated by the idea of *promised utilities* [Gor+20], another algorithm worth considering is using the aformentioned optimal offline allocation maximizing NSW, and having each agent's utility in that allocation used to showcase *potentional*, with agents having higher remaining potential getting priority.

5 Experimental Results

Setup: We consider instances of $n = 50$ agents with $T = 200$ item types, and T items arriving online. Each instance was created by drawing T items uniformly at random from a distribution. We consider three different distributions, or utility profiles: **Uniform** distribution, where the utility of an agent for each item is a uniform number in the range of $[1, 20]$, **Gaussian** distribution, where the corresponding utilities follow the normal distribution with $\mu = 10, \sigma = 2$, and a distribution modeled by the real dataset Jester [Gol+01]. This real dataset contains users' ratings of jokes on a scale $[-10, 10]$. This dataset was used in another experimental study on online fair division [GKP21]. That paper also made the case for sampling from uniform distributions. The metrics reported were averaged over 1000 trials. Tables 1, 2, and 3, present the experimental results for the uniform, Gaussian, and the real-data distributions respectively. Each column highlights the best (worst) performing algorithm in green (red).

Algorithm	SW	Envy	NSW	Maxmin
Uniform Random	2100.68	117.32	33.22	1.66
Utility Greedy	3987.19	224.99	28.47	0.0
Proportional Random	2728.65	112.97	44.51	3.34
NSW Greedy	3820.88	14.58	76.16	58.36

Table 1: Results for **uniform** weight distribution. Columns correspond to the following efficiency and fairness metrics: Social Welfare (SW), maximum pairwise envy (Envy), Nash Social Welfare (NSW), minimum utility of any agent (Maxmin).

Algorithm	SW	Envy	NSW	Maxmin
Uniform Random	1890.04	87.77	31.54	2.82
Utility Greedy	2772.81	137.79	29.13	0.0
Proportional Random	1975.71	88.14	32.95	3.19
NSW Greedy	2583.07	12.24	51.44	39.74

Table 2: Results for **Gaussian** weight distribution.

Algorithm	SW	Envy	NSW	Maxmin
Uniform Random	1758.94	104.27	26.51	0.52
Utility Greedy	3116.04	339.65	8.9	0.0
Proportional Random	1947.44	101.97	28.92	0.49
NSW Greedy	2564.91	18.77	49.91	33.13

Table 3: Results on the **real dataset** Jester.

All tables paint a similar picture. *Utility Greedy* always achieves the highest utility, as expected, while doing very poorly in terms of fairness and performing the worst across all fairness metrics. *Uniform Random* achieves the lowest social welfare, not surprising given how utility-unaware it is. *Uniform Random* and *Proportional Random* perform very similar in all fairness metrics. Lastly, *NSW Greedy* (NSWG), outperforms every other algorithm in terms of fairness, while achieving near-optimal social welfare. Looking at the Maxmin metric, it is noteworthy that all algorithms except NSWG allow for agents that get next to nothing, a fact that might not be acceptable in certain applications. Looking at the Envy metric for NSWG, we see that the average maximum fairness amongst agents is roughly the same as the expected utility of an item. This seems to suggest that NSWG approximately satisfies the EF1 fairness requirement. Overall, for non-adversarial instances, it looks like a simple greedy algorithm maximizing NSW can offer great performance, maintaining fairness while achieving near-optimal efficiency.

6 Summary & Open Directions

We presented an experimental study for the problem of online fair division of indivisible items in a relaxed online model that doesn't generate the most pessimistic adversarial examples, but aims at modeling real inputs more accurately, assuming we have access to a distribution from which items are drawn. This isn't an unrealistic assumption as past data can be used to predict the future, and this model has also been used in the context of other problems. A very common assumption that we didn't extensively discuss is the idea of having normalized utilities. That assumption has

been used to break previous barriers [GPT20], and its overall necessity and practical justification has been argued for [Gor+20]. Therefore it makes sense to extend such an experimental study to the normalized case. It's easy to imagine many additional directions, such as items arriving from different distributions (similar to the prophet inequality setting), performance as the number of items increases, both agents and items arriving online, restricted deletion of items, and of course, designing algorithms that take advantage of the known distribution. Lastly, we think another promising direction is using algorithms with advice [MV20]. These would be algorithms equipped with error-prone predictions about future input. Similar to how [Gor+20] uses *promised utilities* from uniform allocation in the context of divisible items, promised utilities could also be used assuming we have predictions about different metrics in some optimal allocation.

References

[Gol+01] Ken Goldberg et al. "Eigentaste: A constant time collaborative filtering algorithm". In: *information retrieval* 4.2 (2001), pp. 133–151.

[Fel+09] Jon Feldman et al. "Online stochastic matching: Beating 1-1/e". In: *2009 50th Annual IEEE Symposium on Foundations of Computer Science*. IEEE. 2009, pp. 117–126.

[HMZ11] Bernhard Haeupler, Vahab S Mirrokni, and Morteza Zadimoghaddam. "Online stochastic weighted matching: Improved approximation algorithms". In: *International workshop on internet and network economics*. Springer. 2011, pp. 170–181.

[Meh13] Aranyak Mehta. "Online matching and ad allocation". In: (2013).

[Ale+15] Martin Aleksandrov et al. "Online fair division: Analysing a food bank problem". In: *arXiv preprint arXiv:1502.07571* (2015).

[BKV18] Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. "Greedy algorithms for maximizing Nash social welfare". In: *arXiv preprint arXiv:1801.09046* (2018).

[Ben+18] Gerdus Benade et al. "How to make envy vanish over time". In: *Proceedings of the 2018 ACM Conference on Economics and Computation*. 2018, pp. 593–610.

[Car+19] Ioannis Caragiannis et al. "The unreasonable fairness of maximum Nash welfare". In: *ACM Transactions on Economics and Computation (TEAC)* 7.3 (2019), pp. 1–32.

[AW20] Martin Aleksandrov and Toby Walsh. "Online fair division: A survey". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 09. 2020, pp. 13557–13562.

[GPT20] Vasilis Gkatzelis, Alexandros Psomas, and Xizhi Tan. "Fair and Efficient Online Allocations with Normalized Valuations". In: *arXiv preprint arXiv:2009.12405* (2020).

[Gor+20] Artur Gorokh et al. "Online Nash Social Welfare via Promised Utilities". In: *arXiv preprint arXiv:2008.03564* (2020).

[MV20] Michael Mitzenmacher and Sergei Vassilvitskii. "Algorithms with predictions". In: *arXiv preprint arXiv:2006.09123* (2020).

[Sin+20] Sean R Sinclair et al. "Sequential Fair Allocation of Limited Resources under Stochastic Demands". In: *arXiv preprint arXiv:2011.14382* (2020).

[ZP20] David Zeng and Alexandros Psomas. "Fairness-efficiency tradeoffs in dynamic fair division". In: *Proceedings of the 21st ACM Conference on Economics and Computation*. 2020, pp. 911–912.

[BMS21] Anna Bogomolnaia, Hervé Moulin, and Fedor Sandomirskiy. "On the fair division of a random object". In: *Management Science* (2021).

[GKP21] Yuan Gao, Christian Kroer, and Alex Peysakhovich. "Online Market Equilibrium with Application to Fair Division". In: *arXiv preprint arXiv:2103.12936* (2021).