Fast & Fair: A Collaborative Platform for Fair Division Applications

Jiatong Han and Warut Suksompong

National University of Singapore
jiatong.han@u.nus.edu, warut@comp.nus.edu.sg

Abstract

Fair division, the study of how to fairly allocate resources among agents, has received substantial interest in the areas of artificial intelligence and multiagent systems. While there is an extensive theoretical literature on fair division by now, the developed algorithms are still mostly confined to research papers and inaccessible to the public. We attempt to bridge this gap by developing Fast & Fair, an open-source web application that hosts a number of fair allocation algorithms with user-friendly interfaces and explainable outcomes. In contrast to existing implementations, Fast & Fair is a collaborative platform that is open to community contributions and thereby facilitates the deployment of additional algorithms.

Introduction

Allocating resources fairly among interested agents can be a complex task. Sources of difficulty include the facts that the resource may consist of indivisible items (such as books, stationery, or human resources), the valuations over items may vary across agents, and agents may have different entitlements to the resource. The research area of *fair division*, which addresses this problem through precise mathematical models, has attracted significant attention in economics and computer science, including in artificial intelligence and multiagent systems (Aziz 2020; Walsh 2020; Suksompong 2021; Amanatidis et al. 2023). For instance, a basic fairness criterion called *envy-freeness* stipulates that no agent should envy another agent based on the allocated resources.

Although fair division has given rise to a rich and beautiful literature with many elegant solutions, research in the area has remained mostly theoretical, with the developed algorithms primarily confined to research papers. Two notable exceptions are *Spliddit* (Goldman and Procaccia 2014), a platform that implements a few applications including sharing apartment rent, splitting taxi fare, and dividing credit in a group project, and *Kajibuntan* (Igarashi and Yokoyama 2023), a website focusing on the fair allocation of household chores between couples. While these platforms certainly help bring fair division research closer to the public, they still implement only a handful of algorithms from the plethora of fair division algorithms that have been devised in the literature, which means that the vast majority of algorithms remain inaccessible to the wider community.

In light of this, we present *Fast & Fair*, a new collaborative platform for fair division applications. Currently, Fast & Fair provides implementations of three algorithms, each of which addresses a practical fair allocation scenario. It offers intuitive interfaces and explains the fairness of suggested outcomes to users via established properties. Moreover, unlike existing platforms, Fast & Fair is open to community contributions—this means that developers are welcome to contribute code for their desired applications, and the resulting implementations can then be incorporated into our platform with standardized graphical widgets and interfaces.

Fast & Fair is available at *http://fair-alloc.streamlit.app*.

Current Applications

We first discuss the three fair division applications that we have implemented on Fast & Fair thus far.

1. **Weighted picking sequence.** This application concerns the allocation of indivisible items among agents who may have different entitlements represented by weights. A weighted picking sequence proposed by Chakraborty, Segal-Halevi, and Suksompong (2022) guarantees a version of envy-freeness called *WEF*$(x, 1-x)$. Intuitively, the procedure simulates the agents taking turns picking their favorite items, where agents with larger entitlements are allowed to pick more frequently than those with smaller entitlements. Part of the user interface for this application is shown in Figure 1.
2. **Fair two-sided matching.** The fair division literature typically assumes that agents have preferences over the items to be allocated, but not vice versa. However, in certain scenarios, the "items" may correspond to human resources, who have preferences on where they are allocated to. Our application implements an algorithm by Igarashi et al. (2023), which computes an allocation that is fair among the agents (with respect to a version of envy-freeness called *EF[1,1]*), balanced (distributes roughly the same number of items to each agent), and swap stable (there is no swap that would benefit all parties involved). The algorithm ensures a desirable allocation by taking into account the preferences of both sides.

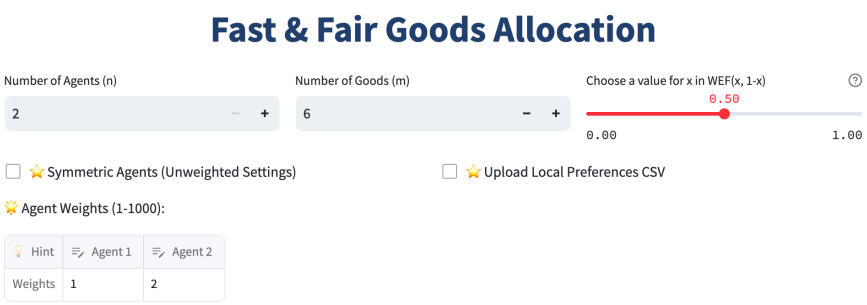

Figure 1: User input interface for fair goods allocation

3. **Envy-free house assignment.** House allocation refers to the problem of allocating m houses to n agents (where $n \leq m$) so that every agent is assigned to exactly one house. The model can also be applied to the allocation of students to dormitory rooms, company workers to offices, and clients to web servers. Our implementation of an algorithm by Gan, Suksompong, and Voudouris (2019) outputs an envy-free house allocation whenever one exists, thereby ensuring that agents will not have an incentive to swap houses.

In all three applications, not only does our platform produce fair outcomes, but it also does so via *efficient* implementations (hence the name "Fast & Fair"). Moreover, our interface allows users to specify their preferences in an *intuitive* manner, and the outcomes that our applications suggest are *explainable*—our algorithms offer detailed quantitative explanations as to why the outcomes are considered fair according to fairness notions established in the literature.

Implementation Details

We have developed Fast & Fair using the Streamlit front-end library, which provides us with attractive input widgets and front-end elements. For algorithm implementations, we use the Python programming language due to its popularity, simplicity in showing the logic of the algorithms, and capability to accept a broader range of community contributions. As our applications are meant for single-session use and we do not store user states, there is no need to connect to an external database—this eliminates the risk of exposing sensitive information. Should the users want to save states on their own, we have provided functions packaged as buttons with which they can download the input parameters, suggested outcomes, and associated explanations.

Our applications share a similar layout and panel arrangement. A step-by-step user guide resides on the collapsible left panel. The right-hand side contains the input boxes for the relevant parameters of the application (see Figure 1); these boxes enable automatic input format verification. Once the user clicks the "Run Algorithm" button, the outcome is presented in a neat table format that is easy to visualize. In addition to measuring the running time of the algorithm, our tool also provides concrete justifications for the suggested outcomes under "Explanations of Outcomes".

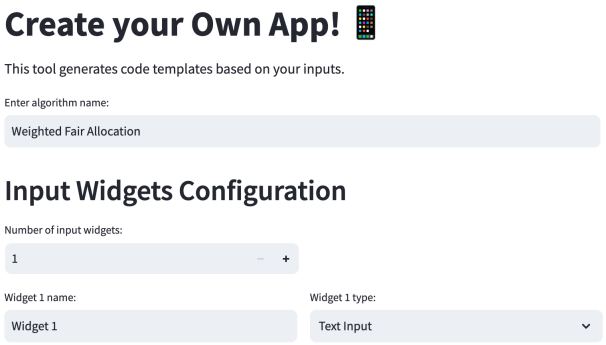

Figure 2: User interface for code template generator

Collaborative Feature and Outlook

While our implementations help expand the range of fair division algorithms accessible beyond researchers, we realize that given the vast number of algorithms in the literature, we cannot implement a significant fraction of them by ourselves. For this reason, not only have we made our code publicly available on GitHub[1], but we have also added a collaborative feature where community contributions can be integrated into our platform. To our knowledge, Fast & Fair is the only fair division platform to offer such a feature.

Specifically, our "Create Your Own App!" page[2] contains an application template generator, which generates code templates for interested users to develop their own algorithm implementations and contribute to the platform. Users can select desired input widgets of several types based on their application scenarios and fill their code into our template. When users finish their implementation, they can commit their code to our GitHub repository, upon which our developers will review it and incorporate the application into our platform. We have made our entire platform open-source and easily extensible so that other developers can also host similar applications on local machines or on cloud.

Given the unique collaborative nature of Fast & Fair, we are optimistic that our platform will inspire the fair division community to bring further solutions to the public and narrow the gap between fair division theory and practice.

[1] http://github.com/JThh/fair-alloc-app-ra/tree/new_main
[2] http://fair-alloc.streamlit.app/Create_Your_Own_App!

Acknowledgments

We are grateful for the support by the Singapore Ministry of Education under grant number MOE-T2EP20221-0001 ("Algorithmic Solutions for Fair Resource Allocation") and by an NUS Start-up Grant, as well as for the encouraging feedback by several of our colleagues.

References

Amanatidis, G.; Aziz, H.; Birmpas, G.; Filos-Ratsikas, A.; Li, B.; Moulin, H.; Voudouris, A. A.; and Wu, X. 2023. Fair division of indivisible goods: Recent progress and open questions. *Artificial Intelligence*, 322: 103965.

Aziz, H. 2020. Developments in multi-agent fair allocation. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*, 13563–13568.

Chakraborty, M.; Segal-Halevi, E.; and Suksompong, W. 2022. Weighted fairness notions for indivisible items revisited. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, 4949–4956.

Gan, J.; Suksompong, W.; and Voudouris, A. A. 2019. Envy-freeness in house allocation problems. *Mathematical Social Sciences*, 101: 104–106.

Goldman, J.; and Procaccia, A. D. 2014. Spliddit: Unleashing fair division algorithms. *ACM SIGecom Exchanges*, 13(2): 41–46.

Igarashi, A.; Kawase, Y.; Suksompong, W.; and Sumita, H. 2023. Fair division with two-sided preferences. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI)*, 2756–2764.

Igarashi, A.; and Yokoyama, T. 2023. Kajibuntan: a house chore division app. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*, 16449–16451.

Suksompong, W. 2021. Constraints in fair division. *ACM SIGecom Exchanges*, 19(2): 46–61.

Walsh, T. 2020. Fair division: the computer scientist's perspective. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, 4966–4972.