Spliddit: Unleashing Fair Division Algorithms

JONATHAN GOLDMAN
and
ARIEL D. PROCACCIA
Carnegie Mellon University

Spliddit is a first-of-its-kind fair division website, which offers provably fair solutions for the division of rent, goods, and credit. In this note, we discuss Spliddit's goals, methods, and implementation.

Categories and Subject Descriptors: J.4.a [**Social and Behavioral Sciences**]: Economics

General Terms: Algorithms; Design; Human Factors; Economics

Additional Key Words and Phrases: Fair Division

1. OVERVIEW

The origins of the mathematically rigorous study of fair division can be traced back to the work of Hugo Steinhaus during World War II [Steinhaus 1948]. Over the decades, fair division theory has become a major field of study in mathematics, economics, computer science, and political science [Brams and Taylor 1996; Robertson and Webb 1998; Moulin 2003; Procaccia 2013]. Nowadays the literature encompasses *provably fair* solutions for a wide variety of problems — many of them relevant to society at large. But, to date, very few fair division methods have been made publicly available. Exceptions that prove the rule include the Adjusted Winner Website[1], which provides access to a (patented) method for dividing indivisible goods between two players, due to Brams and Taylor [1996]; and Francis Su's Fair Division Calculator[2], which implements methods for splitting rent, divisible goods, and chores from his beautifully written paper [Su 1999]. (Su's rent division calculator was recently updated by the New York Times[3].)

Enter *Spliddit* (`www.spliddit.org`), a new fair division website. Quoting from the website:

> *Spliddit is a not-for-profit academic endeavor. Its mission is twofold:*
> —*To provide easy access to carefully designed fair division methods, thereby making the world a bit fairer.*
> —*To communicate to the public the beauty and value of theoretical research in computer science, mathematics, and economics, from an unusual perspective.*

[1] `http://www.nyu.edu/projects/adjustedwinner`
[2] `https://www.math.hmc.edu/~su/fairdivision/calc`
[3] `http://www.nytimes.com/interactive/2014/science/rent-division-calculator.html`

Authors' addresses: Computer Science Department, Carnegie Mellon University {jagoldma,arielpro}@andrew.cmu.edu

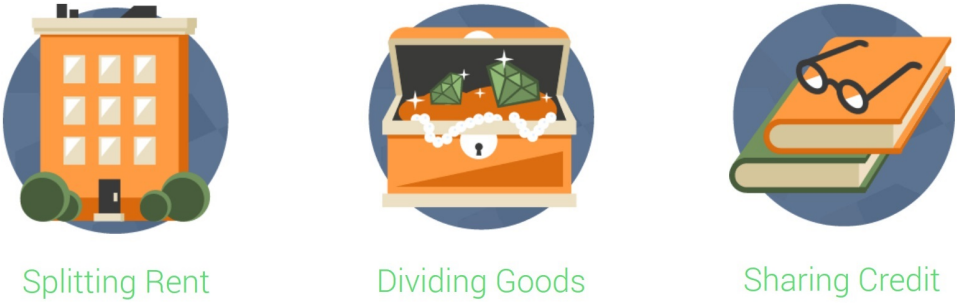

Fig. 1. Spliddit's three applications.

Spliddit was officially launched on November 4, 2014. Building on early press coverage in popular technology and science websites such as Gizmodo, Lifehacker, and Slashdot, Spliddit has attracted almost 20,000 visitors in the first week after launch. We expect to serve hundreds of thousands of users in the coming months.

2. THE THREE APPLICATIONS

Spliddit currently includes three applications (see Figure 1), which were selected to maximize broad appeal and usability. A single algorithm is provided for each application, even in cases where several incomparable approaches are present in the fair division literature. This nontrivial design choice is partly driven by usability considerations, and partly by educational considerations: it allows us to focus on giving an accessible explanation (enhanced by animations) of the guaranteed fairness properties in the context of each of the three applications.

In the remainder of this section we describe Spliddit's applications, as well as their corresponding algorithms and fairness guarantees.

Splitting rent. The rent division problem involves n players (housemates) and rooms.[4] The total rent, say R, is also given. The goal is to fairly assign the rooms to players and divide the rent.

We assume that each player i has value V_{ij} for room j, so that for all i, $\sum_j V_{ij} = R$. Suppose that price of room j is p_j, and $\sum_j p_j = R$. The utility functions of the players are *quasi-linear* in prices, that is, the utility of player i for room j is $V_{ij} - p_j$. Therefore, the only information Spliddit needs to elicit from each player is her value V_{ij} for each room j (see Figure 2).

Spliddit's rent division scheme is an implementation of an algorithm developed by Abdulkadiroğlu et al. [2004]. In a nutshell, it is a market-based algorithm which iteratively increases the prices of overdemanded rooms (at the same rate) and decreases the prices of the other rooms (at the same rate). The allocation is guaranteed to be *envy-free*: If player i is assigned room j for price p_j, then for any room j', $V_{ij} - p_j \geq V_{ij'} - p_{j'}$. In this domain, every envy-free allocation is also *Pareto efficient*, that is, there is no other allocation for which all players have weakly greater utility, and at least one player has strictly greater utility.

[4]Spliddit deals with rooms that are shared by multiple roommates by creating several copies of the room, one for each roommate.

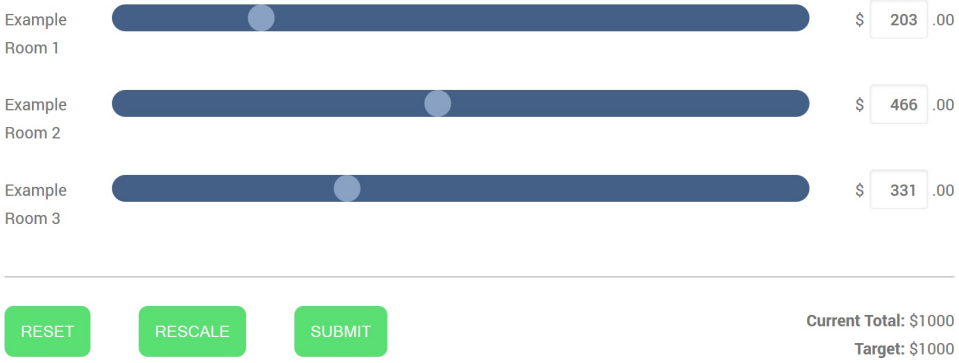

Fig. 2. Bidding interface for rent division.

Interestingly, Brams and Kilgour [2001] observe that there are situations where some prices are negative at every envy-free allocation — some players are paid to live in the house! But the algorithm of Abdulkadiroğlu et al. [2004] guarantees that prices will be nonnegative if an envy-free allocation with nonnegative prices exists. In practice, we believe it is highly unlikely that the algorithm will output negative prices for inputs that reflect real-life values.

Dividing goods. Spliddit's second application allocates a set of goods G to a set of n players. The goods can be divisible or indivisible, but the problem is nontrivial only when some of the goods are indivisible. Inheritance is the paradigmatic use case, e.g., dividing an art or jewelry collection (consisting of indivisible goods) among three or more heirs. Each player i has value V_{ij} for good j. We assume that valuations are *additive*, that is, the value of player i for a bundle of goods X is $V_i(X) \triangleq \sum_{j \in X} V_{ij}$. Additivity underlies our fairness guarantees, and, perhaps even more importantly, it facilitates easy elicitation of preferences. On Spliddit, each player simply distributes a pool of 1000 points between the goods.

Let us denote an allocation of the goods by $A_1, \ldots, A_n$, where A_i is the bundle of goods allocated to player i. Spliddit's algorithm considers three increasingly weaker levels of fairness:

(1) *Envy-freeness:* $V_i(A_i) \geq V_i(A_{i'})$ for every pair of players i, i'. It is clear that envy-freeness is not always feasible.

(2) *Proportionality:* $V_i(A_i) \geq V_i(G)/n$. In Spliddit, this means each player assigns at least $1000/n$ points to her bundle. Again, clearly proportionality may not be feasible.

(3) *Maximin share guarantee:* The *maximin share (MMS) guarantee* of player i is

$$\mathrm{MMS}(i) = \max_{X_1,\ldots,X_n} \min_j V_i(X_j),$$

where the max is taken over partitions of the items into n subsets $X_1, \ldots, X_n$. Intuitively, this is the value player i could guarantee if she divided the items into n bundles, but then selected a bundle last. An MMS allocation satisfies $V_i(A_i) \geq \mathrm{MMS}(i)$ for all players i. Although an MMS allocation may not exist [Procaccia and Wang 2014], counterexamples are elaborate and extremely

unlikely to occur in practice. Moreover, an approximate MMS allocation is guaranteed to exist. Specifically, we can always find an allocation such that $V_i(A_i) \geq \frac{2}{3}\text{MMS}(i)$ [Procaccia and Wang 2014].

Spliddit's algorithm works as follows. First, it finds the highest feasible level of fairness. If envy-freeness and proportionality are infeasible, the algorithm computes the maximum $\alpha > 0$ such that each player can achieve an α fraction of her MMS guarantee. Second, the algorithm maximizes social welfare — $\sum_i V_i(A_i)$ — subject to the fairness constraint found in the first phase. Crucially, while we believe that it will always be possible to find an MMS allocation (with $\alpha = 1$), $\alpha \geq 2/3$ is provably feasible even in the worst case [Procaccia and Wang 2014]. This fact enables us to specify an indisputable fairness guarantee, honoring Spliddit's promise (as made by its tagline) to provide *provably fair solutions*.

Sharing credit. The third and final application divides credit for a joint project between n players. Possible use cases include determining scientific credit for a paper, dividing a company bonus based on employees' relative contributions, or sharing credit for a class project. Player i reports how *the rest* of the credit should be distributed among the other players. For example, if player 1 thinks that players 2 and 3 contributed equally, and player 4 contributed twice as much, then player 1 would report the contributions 25%, 25%, and 50%, respectively. A solution divides 100% of the full credit for the project among the players.

Work by de Clippel et al. [2008] provides a family of rules for credit division; Spliddit implements one of them.[5] This solution is guaranteed to satisfy a slew of desirable properties. Most importantly, it is *impartial*: a player's share of the credit is independent of her report. The solution is also *consensual*: if there is a division that agrees with all players' reports, then it is the outcome. While consensuality seems to be quite unrestrictive at first glance, de Clippel et al. [2008] show that if $n = 3$, an impartial and consensual rule cannot be *exact*, that is, it would have to sometimes allocate less than 100% credit overall.[6] Spliddit therefore enforces $n \geq 4$.

It is worth noting that one of the possible use cases of the credit division application — sharing scientific credit for a paper — gives rise to interesting questions. Determining the order of authors is a notorious source of acrimony in scientific fields where contribution-based ordering is the norm (that is, almost all scientific fields). Ordering authors by their (fair) share of the credit is an obvious solution. But doing so would not preserve impartiality! Intuitively, while a player cannot change her own share of the credit, she can decrease another player's share below her own, thereby increasing her position in the author ordering. Very recent work by Berga and Gjorgjiev [2014] establishes impossibility results for impartial ranking under a strong notion of impartiality. Despite this theoretical difficulty, we do believe it is reasonable (albeit not ideal) to use Spliddit's credit division application for the explicit purpose of ordering authors.

[5]Specifically, the formula in Equation (17) of the paper of de Clippel et al. [2008] is used, with arithmetic means as the aggregators ρ and τ.

[6]Note that normalizing the shares would violate impartiality: by changing the sum of shares, a player can change her own normalized share of the credit.

3. IMPLEMENTATION DETAILS

The implementation of Spliddit is centered around a single web application written in Ruby on Rails. All of Spliddit's *nouns*, including application instances, players, goods, valuations, and allocations, are modeled using Ruby on Rails Active Record, an object-relational mapping system. On the front end, Spliddit uses the jQuery Javscript library to create interactive user interfaces such as the one used for bidding. Spliddit takes advantage of several services offered by Amazon's cloud computing platform: Elastic Compute Cloud (for hosting Spliddit's webservers and running the division algorithms), Elastic Beanstalk (for deploying and automatically scaling Spliddit to handle heavy web traffic), Relational Database Service (for hosting Spliddit's database), and Simple Email Service (for sending email notifications).

The algorithms described in Section 2 are written in Java for improved performance and ease of implementation. The splitting rent application employs a method of Ünver [2007] for efficiently computing the set of overdemanded rooms based on the Gallai-Edmonds Decomposition Lemma for bipartite graphs. The dividing goods application uses IBM's CPLEX Optimizer and IBM's Concert Technology for modeling and solving mixed integer linear programs in Java (*mixed* integer linear programs are required so that users can mark goods as either divisible or indivisible). Each of the three fairness properties described in Section 2 can be expressed as linear constraints; however, for the weakest level of fairness, the algorithm must first solve mixed integer linear programs to compute each player's MMS guarantee. The implementation for the sharing credit application is straightforward in comparison, involving the computation of several summations. Since these algorithms can be time consuming when inputs are not very small, Spliddit uses Delayed Job to run the algorithms in background processes, freeing the webservers to quickly respond to incoming HTTP requests.

4. CLOSING REMARKS: EMPIRICAL FAIR DIVISION RESEARCH

As noted above, Spliddit's two primary goals are providing access to fair division methods, and outreach. However, Spliddit also has the potential to become a revolutionary platform for empirical fair division research. Indeed, as noted by Herreiner and Puppe [2009], fairness properties such as envy-freeness are difficult to study in the lab. A typical experiment in the context of indivisible goods informs each participant of her "value" for each virtual "good", and pays each participant based on her value for her allocated bundle. In this setting envy-freeness is problematic, because a participant does not truly care about which goods were allocated to another participant (as the goods have no real value) — presumably she mainly cares about how much other participants were paid.

In contrast, Spliddit allows us to partition thousands of users (who report their values for real goods) into multiple groups, and employ a different solution for each group. Happiness surveys will then allow us to gauge the relative importance of various criteria in a way that was previously impossible.

REFERENCES

Abdulkadiroğlu, A., Sönmez, T., and Ünver, M. U. 2004. Room assignment-rent division: A market approach. *Social Choice and Welfare 22,* 3, 515–538.

Berga, D. and Gjorgjiev, R. 2014. Impartial social rankings. Manuscript.

Brams, S. J. and Kilgour, D. M. 2001. Competitive fair division. *Journal of Political Economy 109*, 418–443.

Brams, S. J. and Taylor, A. D. 1996. *Fair Division: From Cake-Cutting to Dispute Resolution*. Cambridge University Press.

de Clippel, G., Moulin, H., and Tideman, N. 2008. Impartial division of a dollar. *Journal of Economic Theory 139*, 176–191.

Herreiner, D. K. and Puppe, C. D. 2009. Envy freeness in experimental fair division problems. *Theory and decision 67,* 1, 65–100.

Moulin, H. 2003. *Fair Division and Collective Welfare*. MIT Press.

Procaccia, A. D. 2013. Cake cutting: Not just child's play. *Communications of the ACM 56,* 7, 78–87.

Procaccia, A. D. and Wang, J. 2014. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the 14th ACM Conference on Economics and Computation (EC)*. 675–692.

Robertson, J. M. and Webb, W. A. 1998. *Cake Cutting Algorithms: Be Fair If You Can*. A. K. Peters.

Steinhaus, H. 1948. The problem of fair division. *Econometrica 16*, 101–104.

Su, F. E. 1999. Rental harmony: Sperner's lemma in fair division. *American Mathematical Monthly 106,* 10, 930–942.

Ünver, M. U. 2007. Market mechanisms for fair allocation of indivisible objects and money. Manuscript.