Adjusting Adjusted Winner

Robert Bredereck[1], Eyal Briman[2], Bin Sun[1] and Nimrod Talmon[2]

[1]Institut für Informatik, TU Clausthal
[2]Ben-Gurion University of the Negev

Abstract. The *Adjusted Winner* method (AW) is frequently used to fairly divide goods between two agents. In some of its runs, goods may be split or transferred between the agents, a phenomenon that may lead to conflicts between the participating agents. To address this issue, we aim to identify situations in which such conflicts do not arise; specifically, we describe an approach to avoid these conflicts – which consists of the removal of a minimum number of goods as a preprocessing phase. We thoroughly examine the computational complexity of the corresponding combinatorial problem and show that, while the problem is generally intractable, there are several ways to cope with its computational hardness. We complement our theoretical analysis with computer-based simulations.

1 Introduction

The *Adjusted Winner* (AW) method is a widely recognized fair division technique used for allocating goods between two agents. Renowned for its simplicity and effectiveness in achieving fairness and efficiency, it has gained popularity in practical scenarios involving resource allocation and has garnered considerable attention in scholarly discussions. Despite its merits, the AW method occasionally produces outcomes that lead to conflicts. These conflicts arise when certain goods are divided in the allocation, resulting in splits that trigger disagreements among participants due to differing preferences and valuations of goods. Our focus is on addressing the issue of conflicts stemming from splits that emerge during the application of the AW method. Our primary objectives encompass two aspects: firstly, we seek to pinpoint the structural properties of input scenarios that are prone to producing splits in allocations. Secondly, we introduce a novel combinatorial problem that involves the pre-selection of a subset of available goods before running the AW method. This selection process is strategically designed to ensure that applying the method to this pre-elected subset will eliminate the possibility of any splits, thus presenting a preemptive resolution to the conflict issue.

Additionally, we delve into an examination of the computational complexity associated with the proposed combinatorial problem. We show that the problem is inherently computationally intractable. Subsequently, we explore various strategies to manage and mitigate this computational complexity, thereby offering practical approaches to address the challenge. In summary, this paper contributes to a deeper understanding of conflicts arising from splits in the Adjusted Winner method and proposes novel approaches to preemptively resolve such conflicts through careful good selection. By investigating the computational aspects, we pave the way for practical implementations that enhance the utility of the Adjusted Winner method in real-world allocation scenarios.

Motivation. The basic motivation is to avoid dividing indivisible goods. The AW method can be particularly problematic when it suggests splitting an indivisible item. Consider these two examples:

- In the case of a divorcing couple, the AW method might suggest splitting a truly indivisible item like a dog or a house. If the house holds significant value for one partner, neither sharing it nor selling it and dividing the money would be acceptable, especially if one partner perceives a higher ideological value.
- Imagine two children dividing items from a shared room. The AW method might require numerous transfers. Our model reduces transfers by removing some items from division, assigning remaining ones to the child who values them most. This approach makes "no transfer" sensible, as items can be assigned to the one with higher value, with some items shared publicly or donated.

Removing goods from the division process would indeed sacrifice Pareto-optimality. However, whether the division based on a big difference in valuations between players is necessarily beneficial to the players remains uncertain. Additionally, aside from Pareto-optimality, equitability is rarely achieved. The only axiom that would be restored is envy-freeness. It's important to note that splitting an item is quite different from deleting it. Splitting an item per the AW method requires clear ratios, while deleting an item can mean making it publicly available for the couple or selling it. However, when selling, the price does not translate directly to points or tokens. By restructuring the process, we aim to address these concerns while acknowledging the trade-offs involved, such as sacrificing Pareto-optimality for improved fairness and envy-freeness.

Related Work

We discuss related work on the adjusted winner method, on control in elections, and on conflict resolution in fair division.

The Adjusted Winner (AW) method. The Adjusted Winner (AW) procedure is a well-known approach that aims to yield allocations satisfying multiple fairness criteria and has been tested and evaluated in real-world scenarios and simulations [2, 17, 15, 16, 9, 7]. Specifically, AW allocations exhibit envy-freeness, equity, Pareto optimality, and minimally fractional divisions. Notably, it has been observed that the Adjusted Winner with continuous strategies may not guarantee pure Nash equilibria. However, it has been established that each instance of the Adjusted Winner procedure has an ϵ-Nash equilibrium for every $\epsilon > 0$ [8, 3].

Control in elections. In our paper, we allow for the deletion of certain goods from an instance of AW, so that some properties can be achieved. In this context, we mention work on control in elections [4, 10], which is concerned with external agents that can alter the structure of a given election to achieve some goal (usually, in elections, the goal is to ensure the winner).

Conflict resolution. From a conflict resolution point of view, one lesson from the biblical story about the judgement of Solomon (I Kings 3) is that, while splitting some element (in the case of Solomon, a baby) may be fair in some sense, it is not always desired. In our paper, we also aim to avoid the necessity of splitting a good between the agents. Indeed, the topic of conflict resolution in the fair division is very vast [8, 6]. In this study, we explore the adjusted algorithm as a conflict resolution procedure for two players in the context of fair division with additive valuations of goods and thus we also mention the work of [13].

2 Preliminaries

2.1 A Formal Model of Fair Division

An instance of a standard model of fair division includes:

- A finite set of participants: $P = \{1, 2, \ldots, n\}$.
- A finite set of indivisible goods: $W = \{w_1, w_2, \ldots, w_m\}$.
- Utility function for each participant i, denoted u_i, where $u_i(W')$ represents the value that participant i assigns to a subset $W' \subseteq W$ of goods. This utility function is referred to as *additive* since the utility U_i of participant i for their allocated goods W'_i equals the sum of the values assigned to the goods:

$$U_i(W'_i) = \sum_{w \in W'_i} u_i(w) . \quad (1)$$

- This allocation is denoted by $A = (W_1, W_2, \ldots, W_n)$, where $W_i \subseteq G$ represents the allocation of goods to participant i.

Example 1. *Consider the following example: two artists, A and B, due to certain special circumstances, must part ways and divide the seven paintings they had collaboratively created. Among these seven paintings $\{w_1, \ldots, w_7\}$, w_1 is exceptionally valuable and extraordinarily successful, while w_7 is rather less successful. The evaluations of the seven paintings by the two artists are as follows:*

Artist	w_1	w_2	w_3	w_4	w_5	w_6	w_7
A	41	12	12	12	12	12	1
B	60	8	8	8	8	8	2

Table 1. Artist valuations of different paintings

Using the AW method would necessitate cutting the most valuable painting w_1 ($\frac{2}{1} \geq \frac{3}{2} \geq \frac{60}{41}$), which is unacceptable for such a masterpiece. Therefore, a more viable solution is proposed: donate the least desirable painting w_7.

Under this solution, A would receive five paintings of moderate value, while B would receive the most valuable painting. This solution ensures that the highly valuable painting is preserved intact and eliminates the need for swapping paintings between the artists, while still ensuring that both artists receive equal overall value. Consequently, exploring the AW method and considering the possibility of discarding a resource proves to be both valuable and fair. This approach strikes a balance by preserving the integrity of the artwork and maintaining fairness in value distribution.

The *Adjusted Winner* Method (AW). The *Adjusted Winner* method (AW) was developed by Brams and Taylor for fairly dividing k divisible goods between two agents. Let $W = \{w_1, w_2, \ldots, w_m\}$ be the set of goods or issues under consideration. The utilities of agent 1 and agent 2 for the good w_j are denoted by $u_1(w_j) \in \mathbb{N}$ and $u_2(w_j) \in \mathbb{N}$, respectively. The sums of their utilities, (i.e the total number of tokens available for each agent) are such that $\sum_{j=1}^{m} u_1(w_j) = \sum_{j=1}^{m} u_2(w_j) = 100$. The bundles allocated to agent 1 and agent 2 are denoted by W'_1 and W'_2, respectively, and their utilities are computed by $U_i(W'_i) = \sum_{w \in W'_i} u_i(w)$ for $i \in \{1, 2\}$.

Minimizing Conflicts in AW. Our primary motivation in this work is to prevent conflicts that may arise in the output of the adjusted winner procedure, by removing minimal number of goods, necessary for the AW to work without conflicts.

Our point of view is that a good that is divided by the algorithm and also a good that is transferred by the algorithm may cause personal conflicts between the agents, compared to runs of the algorithm in which such operations do not occur. Specifically, we aim to ensure that no good is divided between the two agents, or that no goods are transferred between them. Therefore, in the following sections, we provide some definitions to establish the core methodology and framework for this study; this allows us to be able to have a clear, formal model that corresponds to such conflict minimization.

Note that we study two problems: One, the No Transfer Problem, in which we aim to reach a setting in which AW does not do any transfer (and thus, of course, also no splits); and another, the No Split Problem, in which we aim to reach a setting in which we do not mind if AW does some transfers, but we do not wish it to make any splits (i.e., we wish to keep *all* goods indivisible). In both problems, we wish to reach the corresponding situation by **removing** a small number of original goods.

A Generalization of Adjusted Winner (GAW). In order to formally define the two problems discussed in this work we introduce the following, generalized version of the *Adjusted Winner* algorithm termed *Generalized Adjusted Winner*. It is structured into three steps. This generalized adaptation involves initializing a threshold d, which serves as the parameter representing the maximum allowable difference between the utilities. This threshold is defined as $d = |U_1(W'_1) - U_2(W'_2)|$. (In the original problem, d is set to be 0.) Additionally, we modify the number of tokens available for each agent to be a parameter z, rather than a constant 100. Hereafter, when referring to the Adjusted Winner algorithm, we denote the Generalized Adjusted Winner as AW.

3 Avoiding-Transfers Problem (AWNT)

We introduce the AWNT problem, which seeks to determine whether the deletion of at most k goods, resulting in a subset of $m - k$ goods, exists to impede the transfer of goods between the agents during the execution of the AW procedure. In essence, this problem poses a decision-based inquiry wherein, given an instance, the task is to ascertain if, when applying the GAW algorithm to it, only Step 1 is executed (and not Step 2 and 3).

Definition 1 (AWNT). *Given an instance W of m goods, n utility functions of n agents, and a threshold d, the objective is to identify if there exists a subset $W' \subseteq W$ of at most $k \leq m$ goods that can be removed, ensuring no transfers of goods occur when executing the AW method on subset W' with a threshold d.*

It is useful to consider an equivalent formulation of AWNT.

Definition 2 (AWNT, alternative definition). *Here, we are given:*

- *A matrix $A \in \mathbb{N}_0^{2 \times m}$ where each cell $A[i][j] \in \mathbb{N}_0$ represents the utility $u_i(w_j)$ that agent $i \in \{1, 2\}$ assigns to good $w_j \in W$. Additionally, $\sum_{j=1}^{m} A[1][j] = \sum_{j=1}^{m} A[2][j] = z$.*

Algorithm 1 Generalized Adjusted Winner Method (GAW)

Given an input (utilities matrix) u_{ij}
Given an input (utility difference threshold) d
Initialize W_1' and W_2' as empty bundles ($W_1' = W_2' = ()$)
Step 1 (Initial allocation):
for each good j **do**
  if agent 1 values j higher than agent 2 **then**
    Allocate j to agent 1 (add j to W_1')
  else
    Allocate j to agent 2 (add j to W_2')
Step 2 (Possible transfers):
if $U_1(W_1') > U_2(W_2')$ **then**
  Sort, increasingly, the goods allocated to agent 1 by $\frac{u_1(w_j)}{u_2(w_j)}$
  while $|U_1(W_1') - U_2(W_2')| > d$ **do**
    Transfer the next good in W_1' to W_2', unless doing so results in $U_2(W_2') > U_1(W_1')$
  Sort, increasingly, the goods allocated to agent 2 by $\frac{u_2(w_j)}{u_1(w_j)}$
  while $|U_2(W_2') - U_1(W_1')| > d$ **do**
    Transfer the next good in W_2' to W_1', unless doing so results in $U_1(W_1') > U_2(W_2')$
Step 3 (Possible split):
if $|U_1(W_1') - U_2(W_2')| > d$ **then**
  if $U_1(W_1') > U_2(W_2')$ **then**
    Transfer a fraction $p \in (0, 1]$ from the next good in W_1' to W_2' so that $|U_1(W_1') - U_2(W_2')| \le d$
  else
    Transfer a fraction $p \in (0, 1]$ from the next good in W_2' to W_1' so that $|U_2(W_2') - U_1(W_1')| \le d$

- *An integer $k \le m$ – the number of columns to be deleted.*
- *A threshold d – the difference between the agents' utilities.*

The problem is to find $m-k$ columns – $1, \dots, m-k$ – that satisfy:

$$|\sum_{j=1}^{m-k} I(A[1][j], A[2][j]) - \sum_{j=1}^{m-k} I(A[2][j], A[1][j])| \le d \qquad (2)$$

where

$$I(x, y) = \begin{cases} x, & \text{if } x > y \\ 0, & \text{otherwise} \end{cases}$$

Observation 1. *The AWNT matrix variation problem is equivalent to the original AWNT problem, as the solution for the AWNT matrix variation is the same as the solution for the original AWNT problem.*

3.1 Computational Intractability (of AWNT)

Our first interest is in the computational complexity of AWNT.

Theorem 1. *AWNT is weakly NP-hard.*

Proof. We provide a reduction from a restriction of the Subset Sum problem known as weakly NP-hard [14, 11] in which we have numbers $S = \{a_1, \dots, a_m\}$, a bound B, (this is the restriction) m and each $a_i \in S$ divides B, and the task is to decide whether there is a subset of S that sums to B (NP-hardness proof for this Subset Sum variant is deferred to the full version). We reduce as follows:

- Initialize the matrix A (as described in Definition 2).
- Append an additional row to matrix X to represent the target sum B: set the last element of this row to B so that $A = \begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}$.

Parameter	Complexity class
m	FPT
d	Para NP-Hard
$m-k$	W-h (using [1])
k	W-h
z	FPT

Table 2. Parameters and Complexity: z number of tokens to be distributed among the items according to agent preferences, m- number of goods, k-*number* of goods to delete, d- threshold.

- Scale the entire matrix by a factor of m.
- Modify A (to ensure equal row sums): compute z – the sum of elements in the first row of A (excluding the last element); compute S, the sum of elements in the second row of A; adjust $A[2][m]$, the last element of the second row, to match the sum of the first row: $A[2][m] = A[2][m] + (z - S)$.
- Modify A (to ensure $A[2][j] > A[1][j]$ for all $j \in \{1, \cdots, m\}$ and $A[2][m] < A[1][j]$): for each element $A[2][j]$ in the second row (except the last) – while $A[2][j]$ is greater than $A[1][j]$: decrease $A[2][j]$ by 1 until it is less than $A[1][j]$. If $A[2][j] - 1 < 0$, continue to the next value without decreasing. Then, subtract $m - 1$ tokens from $A[2][1]$; and, introduce an additional token to each cell $A[2][j]$, where $j \in \{2, \cdots, m\}$.

Correctness follows as column $m + 1$ could not be deleted; then, a solution to AWNT corresponds to a Subset Sum solution as all non-deleted goods are assigned to Agent 1. □

3.2 Parameterized Complexity (of AWNT)

We analyze the parameterized complexity of AWNT. The detailed proofs will be presented in the full version of this paper in the future and the results are given in Table 2.

3.3 Approximation Algorithms (of AWNT)

We consider approximating AWNT; first, we wish to minimize the difference between the utilities d.

Theorem 2. *Unless P = NP, for any approximation ratio, there is no polynomial-time algorithm that provides a multiplicative approximation guarantee to minimize the threshold d (AWNT).*

Proof. Towards a contradiction, assume such an approximation algorithm A with an approximation ratio r exists. For the whole set of instances of AWNT, we can divide them into two sets: $\mathcal{I}_0$ and $\mathcal{I}_+$. $\mathcal{I}_0$ stands for the instances where $d = 0$ is feasible and $\mathcal{I}_+$ stands for the instances where $d = 0$ is not feasible.

For any instance $I \in \mathcal{I}_0$, the output of the approximation algorithm A must be 0. Otherwise, the approximation ratio is infinity. For any instance $I \in \mathcal{I}_+$, the output of the approximation algorithm A must not be 0. Otherwise, the algorithm is not correct. Thus, the algorithm A is a polynomial algorithm that could decide whether $d = 0$ is feasible for I. However, we already know that AWNT with $d = 0$ and $k < n$ is weakly NP-Hard from Theorem 1. Contradiction. □

Next, we present a new optimization problem called AWNT-k:

Definition 3 (AWNT-k)**.** *Let the minimal k Adjusted Winner No transfer (AWNT-k) be an optimization version of the AWNT problem that minimizes k, the number of goods to delete while maintaining $|U_1(W_1') - U_1(W_2')| \le d$.*

We introduce the following two problems:

- $\frac{1}{2}$-Balanced Subset Sum: Given a set of integers $S = \{a_1, a_2, \cdots, a_n\}$ and a target sum B, determine if there exists a subset A of integers from S that sums up to B and $|A| = \lfloor \frac{1}{2}|S| \rfloor$.
- $\frac{1}{k}$-Balanced Partition: Given a set of integers $S = \{a_1, a_2, \cdots, a_n\}$, determine whether there exists a subset A of integers from S that sums up to $\frac{1}{2}\sum_{s\in S} s$ and $|A| = \lfloor \frac{1}{k}|S| \rfloor$.

Claim 1. *$\frac{1}{2}$-Balanced Partition is weakly NP-Hard, so $\frac{1}{2}$-Balanced Subset Sum is also weakly NP-Hard.*

Theorem 3. *For any k such that $2 \leq k < n$, $\frac{1}{k}$-Balanced Partition is weakly NP-Hard.*

The proof is deferred to the full version of the paper.

Theorem 4. *Unless P = NP, for any approximation ratio, there is no polynomial-time algorithm with a multiplicative approximation guarantee to minimize the number of goods k to delete (AWNT-k).*

Proof. We reduce $\frac{1}{k}$-Balanced Partition to AWNT-k. For an instance of $\frac{1}{k}$-Balanced Partition $\mathcal{S}$, we create an instance $\mathcal{A}$ of AWNT-k:

$$\mathcal{A} = \begin{bmatrix} m^* & a_1+m & a_2+m & \cdots & a_o+m & \cdots & a_n+m & B+m^*-s & B \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 & B+lm+m^* & B+om+m^* \end{bmatrix}$$

with $l = n - \lfloor \frac{n}{k} \rfloor$, $o = \lfloor \frac{n}{k} \rfloor$, m is a large enough number, $m^* = mn$, $s = \sum_{i=1}^{n} a_i$, and $B = \frac{1}{2}s$. Wlog., we assume that is $\frac{n}{k}$ is an integer, s.t. $l = (k-1)o$. Now we will have further observations.

- Exactly one good of the last two goods should be removed. Otherwise, agent 2 gets either 0 or a too large number.
- The first good cannot be removed; this can ensure that agent 1 gets at least m^*.
- If the last but one good is removed, then exactly l more goods have to be removed; if the last one good is removed, then exactly o more goods have to be removed. The latter is the optimum solution.

We show that $\mathcal{S}$ is a YES-instance iff $\mathcal{A}$ is a YES-instance.

($\Longrightarrow$) Since S is a YES-instance, there is a subset A with $\sum_{s\in A} s = \frac{1}{2}\sum_{s\in S} s$ and $|A| = \lfloor \frac{1}{k}|S| \rfloor = o$. Then, we get two solutions for $\mathcal{A}$: (1) Deleting the last good and exactly o other goods corresponding to the goods in A. In this case, either of the two agents gets $B + lm + m^*$. (2) Deleting the last but one good and exactly l other goods corresponding to the goods in $S \backslash A$. In this case, either of the two agents gets $B + om + m^*$.

($\Longleftarrow$) Since $\mathcal{A}$ is a YES-instance, there are at least two solutions: (a) Deleting the last good and exactly o other goods. (b) Deleting the last but one good and exactly l other goods. The solution (a) induces an optimum solution for $\mathcal{S}$.

Now, for contradiction, we assume that such an approximation algorithm exists for an approximation ratio k'. For any constant k', we could create an instance of AWNT-k where $\frac{l+1}{o+1} \leq k'$ is satisfied. Since the optimum solution is to delete exactly $o+1$ goods, the algorithm can find an approximation solution which deletes $l+1$ goods in polynomial time, since $l+1 \leq (o+1)k'$. This contradicts the fact that $\frac{1}{k}$ Partition is NP-Complete unless $P = NP$. □

Since AWNT is not approximable for both k and d, we define a similar problem called AWNT-r and prove that it admits an FPTAS.

Definition 4 (AWNT-r). *Given an instance W of m goods, the objective is to identify if there exists a subset $W' \subseteq W$ of at most $k \leq m$ goods that can be removed, ensuring no transfers of goods occur when executing the AW method on subset W' and minimizing $\frac{u_1}{u_2}$ (with $u_1 > u_2$), where u_1 is the value of the bundle assigned to agent 1 and u_2 is the value of the bundle assigned to agent 2.*

First, we design a dynamic programming algorithm for AWNT and AWNT-r. We have a table $T[i, k, v_1, v_2]$ where the first entry refers to the ith good, the second entry refers to the k goods to be removed, and the third and fourth entries refer to the values agent 1 and agent 2 get respectively. Both algorithms have a time complexity of $\mathcal{O}(n^2(B + w_{\max})^2)$ (Here, $w_{\max}$ is the maximum w_{ij} in the matrix A except the number B). Here, A_1 is the set of goods which are originally assigned to agent 1 and A_2 is the set of goods which are originally assigned to agent 2. Here is the base of the algorithm:

- $T[a, b, c, d] = 0$ if any of a, b, c, d is negative or $a < b$,
- $T[0, 0, 0, 0] = [\emptyset, \emptyset]$,
- $T[0, b, c, d] = 0$ for all $b, c, d \geq 0$ if $b + c + d > 0$.

The induction part is then:

$$T[i, k, v_1, v_2] = \begin{cases} [S_1, S_2], & \text{if } T[i-1, k-1, v_1, v_2] = [S_1, S_2], \\ [S_1 \cup \{e_i\}, S_2], & \text{if } T[i-1, k, v_1 - w_{i1}, v_2] = [S_1, S_2] \\ & \text{and } e_i \in A_1, \\ [S_1, S_2 \cup \{e_i\}], & \text{if } T[i-1, k, v_1, v_2 - w_{i2}] = [S_1, S_2] \\ & \text{and } e_i \in A_2, \\ 0, & \text{otherwise.} \end{cases}$$

In this table, if $T[a, b, c, d] = 0$, it means that this combination is infeasible, and if $T[a, b, c, d] = [S_1, S_2]$, it means that this combination is feasible by choosing S_1 and S_2 as the remaining goods for agent 1 and agent 2 respectively. This algorithm is a preparation for an FPTAS; next, we define the problem AWNT-r*.

Definition 5 (AWNT-r*). *Given an instance W of m goods and two indices j_1 and j_2, the objective is to identify if there exists a subset $W' \subseteq W$ of at most $k \leq m$ goods that can be removed, ensuring no transfers of goods occur when executing the AW method on subset W' and minimizing $\frac{u_1}{u_2}$ (with $u_1 > u_2$), where u_1 is the value of the bundle assigned to agent 1 and u_2 is the value of the bundle assigned to agent 2, and no good assigned to a_1 can have a greater value than $w_{j_1 1}$ and no good assigned to a_2 can have a greater value than $w_{j_2 2}$.*

Remark. *For each instance I' of AWNT-r, we could create an instance I for it AWNT-r*. In the first step, W remains unchanged. There must be an optimum solution for I to minimize the ratio $r = \frac{u_1}{u_2}$ with solution $S = S_1 \cup S_2$, where $S_1(S_2)$ is the set of assigned goods for agent 1 (2) in S. If such a solution must imply $u_2 > u_1$, we just need to swap the name of agent 1 and agent 2; therefore, $u_1 \geq u_2$ is guaranteed. In addition, there must be a good with index j_1 for S_1 such that $w_{j_1 1} = \max\{w_{i1} \in S_1\}$ and a good with index j_2 for S_2 such that $w_{j_2 2} = \max\{w_{i2} \in S_2\}$.*

For $I = (\{w_{11}, w_{21}, \cdots, w_{n1}\}, \{w_{12}, w_{22}, \cdots, w_{n2}\}, j_1, j_2)$, we rescale it and get a new instance $\hat{I}$ in the following way:

(1) For each $i \in \{1, 2, \cdots, n\}$: if $w_{i1} \leq w_{j_1 1}$, then $\hat{w_{i1}} \leftarrow \lceil \frac{w_{i1} \cdot n}{\epsilon \cdot w_{j_1,1}} \rceil$; otherwise, then $\hat{w_{i1}} \leftarrow \infty$;
(2) For each $i \in \{1, 2, \cdots, n\}$: if $w_{i2} \leq w_{j_2 2}$, then $\hat{w_{i2}} \leftarrow \lfloor \frac{w_{i2} \cdot n}{\epsilon \cdot w_{j_2,2}} \rfloor$; otherwise, then $\hat{w_{i1}} \leftarrow -\infty$.

(3) And we get $\hat{I} = (\{\hat{w_{11}}, \cdots, \hat{w_{n1}}\}, \{\hat{w_{12}}, \cdots, \hat{w_{n2}}\}, j_1, j_2)$.

- We consider an optimum solution S for I. To this end, let $r^* = \frac{u_1}{u_2}$ be the minimum ratio, where $u_1 \geq u_2$. Moreover, let $S_1 \subseteq S$, $S_2 \subseteq S$, $S_1 \cap S_2 = \emptyset$ s.t. $u_1 = \sum_{i \in S_1} w_{i1}$, $u_2 = \sum_{i \in S_2} w_{i2}$, $w_{\max,1} = \max\{w_{i1} \in S_1\} = w_{j_1 1}$ and $w_{\max,2} = \max\{w_{i2} \in S_2\} = w_{j_2 2}$.
- We now fix some value $\bar{u}$ in order to assume that the value of the bundle assigned to agent 1 is at least $\bar{u}$ and the value of the bundle that is assigned to agent 2 is at most $\bar{u}$. We call this $\bar{u}$-*consistent*. We consider the best $\bar{u}$-*consistent* solution $\hat{S}$ for $\hat{I}$ which can be found in the DP. Formally: Let $\hat{r} = \frac{\hat{u_1}}{\hat{u_2}}$ be the minimum ratio, where $\hat{u_1} \geq \bar{u} \geq \hat{u_2}$, where $\bar{u} = \frac{u_1 + u_2}{2}$; and let $\hat{S_1} \subseteq \hat{S}$, $\hat{S_2} \subseteq \hat{S}$, $\hat{S_1} \cap \hat{S_2} = \emptyset$ s.t. $\hat{u_1} = \sum_{i \in \hat{S_1}} \hat{w_{i1}}$, $\hat{u_2} = \sum_{i \in \hat{S_2}} \hat{w_{i2}}$.

Theorem 5. *The optimal value $\hat{r}$ found in the DP-table restricted to* $\bar{u}$-consistent *entries is at most $(1+\epsilon) \cdot r^*$ when $\bar{u} = \frac{u_1+u_2}{2}$.*

Proof. First, we prove that $\hat{u_1} \leq u_1(1+\epsilon')$ and $\hat{u_2} \geq u_2(1-\epsilon')$:

(a) $\hat{u_1} \leq u_1(1+\epsilon')$:
For instance $\hat{I}$, we have $\hat{S_1} = \arg\min_{\hat{S_1'}}\{\sum_{i \in \hat{S_1'}} \hat{w_{i1}} \geq \bar{u}\}$
$\implies \hat{u_1} \leq \sum_{i \in \hat{S_1}} \hat{w_{i1}} \leq \sum_{i \in S_1} \hat{w_{i1}} \leq \sum_{i \in S_1} \lceil \frac{w_{i1} \cdot n}{\epsilon' \cdot w_{\max,1}} \rceil$
$\leq \sum_{i \in S_1} \frac{w_{i1} \cdot n}{\epsilon' \cdot w_{\max,1}} + n \leq \frac{n}{\epsilon' \cdot w_{\max,1}} \sum_{i \in S_1} w_{i1} + n$
$\implies \sum_{i \in \hat{S_1}} w_{i1} \leq \frac{\epsilon' \cdot w_{\max,1}}{n} \sum_{i \in \hat{S_1}} \hat{w_{i1}}$
$\leq \frac{\epsilon' \cdot w_{\max,1}}{n} (\frac{n}{\epsilon' \cdot w_{\max,1}} \sum_{i \in S_1} w_{i1} + n) = \sum_{i \in S_1} w_{i1} + \epsilon' \cdot w_{\max,1}$
$\leq \sum_{i \in S_1} w_{i1}(1+\epsilon')$

(b) $\hat{u_2} \geq u_2(1-\epsilon')$:
For instance $\hat{I}$, we have $\hat{S_2} = \arg\max_{\hat{S_2'}}\{\sum_{i \in \hat{S_2'}} \hat{w_{i2}} \leq \bar{u}\}$
$\implies \sum_{i \in \hat{S_2}} \hat{w_{i2}} \geq \sum_{i \in S_2} \hat{w_{i2}} \geq \sum_{i \in S_2} \lfloor \frac{w_{i2} \cdot n}{\epsilon' \cdot w_{\max,2}} \rfloor$
$\geq \sum_{i \in S_2} \frac{w_{i2} \cdot n}{\epsilon' \cdot w_{\max,2}} - n \geq \frac{n}{\epsilon' \cdot w_{\max,2}} \sum_{i \in S_2} w_{i2} - n$
$\implies \sum_{i \in \hat{S_2}} w_{i2} \geq \frac{\epsilon' \cdot w_{\max,2}}{n} \sum_{i \in \hat{S_2}} \hat{w_{i2}}$
$\geq \frac{\epsilon' \cdot w_{\max,2}}{n} (\frac{n}{\epsilon' \cdot w_{\max,2}} \sum_{i \in S_2} w_{i2} - n) = \sum_{i \in S_2} w_{i2} - \epsilon' \cdot w_{\max,2}$
$\geq \sum_{i \in S_2} w_{i2}(1-\epsilon')$

In the second step, we set $\epsilon = \frac{2\epsilon'}{1-\epsilon'}$, so that $\epsilon' = \frac{\epsilon}{\epsilon+2}$. Thus,

$$\hat{r} \leq (\frac{1+\epsilon'}{1-\epsilon'}) r^* = (1+\epsilon) r^* .$$

This finishes the proof. □

Theorem 6. *There is an FPTAS for AWNT-r.*

Proof. For each instance of I of AWNT-r, we try out all possible pairs of j_1 and j_2. Here, we create n^2 instances of AWNT-r* for I. For each instance of I^* AWNT-r*, we can rescale it into $\hat{I}^*$. Then for each cell in the table of the DP algorithms, it is represented as $[i, k, \hat{u_1}, \hat{u_2}]$. We know $\hat{u_1} = \sum_{i \in \hat{S_1}} \hat{w_{i1}} = \sum_{i \in \hat{S_1}} \lceil \frac{w_{i1} \cdot n}{\epsilon \cdot w_{\max,1}} \rceil \leq \frac{n^2}{\epsilon} + n$ and $\hat{u_2} = \sum_{i \in \hat{S_2}} \hat{w_{i2}} \leq \sum_{i \in \hat{S_2}} \lfloor \frac{w_{i2} \cdot n}{\epsilon \cdot w_{\max,2}} \rfloor \leq \frac{n^2}{\epsilon}$. Here, all the numbers are polynomially bounded by the input length and $\frac{1}{\epsilon}$. Combining with Theorem 5, we obtain an FPTAS for AWNT-r. □

4 Avoiding-Split Problem (AWNS)

We introduce the AWNS problem, which aims to identify whether deleting at most k goods, resulting in a subset of $m-k$ goods, exists to prevent the splitting of goods between the agents when performing the AW procedure. In essence, this problem poses a decision-based inquiry wherein, given an instance, the task is to ascertain if, when applying the AW algorithm to it, the execution halts before reaching the third step. (That is, only Step 1 and Step 2 are to be executed.)

Definition 6. *Given an instance W of m goods and threshold d, the objective is to identify if there exists a subset $W' \subseteq W$ of at most $k \leq m$ goods that can be removed, ensuring no splits of goods occur when executing the AW method on subset W' with threshold d.*

The following observation connects AWNT and AWNS:

Observation 2. *A yes-instance of AWNT is also a yes-instance to AWNS, as no split occurs when there are no transfers. However, a yes-instance to AWNS does not have to be a yes-instance to AWNT. A no-instance of AWNS is also a no-instance to AWNT, since splitting a good necessitates a transfer. However, a no-instance of AWNT does not have to be a no-instance of AWNS.*

Definition 7. *(AWNS alternative definition) we introduce the AWNS matrix variation problem.*

- *A matrix $A \in \mathbb{N}_0^{2 \times m}$ where each cell $A[i][j] \in \mathbb{N}_0$ is the utility $u_i(w_j)$ that agent $i \in \{1, 2\}$ assigns to good $w_j \in W$, and also $\sum_{j=1}^{m} A[1][j] = \sum_{j=1}^{m} A[2][j] = z$. We also denote $a_{i \in [2]}$ as a group containing the i-th row of the matrix.*
- *An integer $k \leq m$ – the number of columns to be deleted.*
- *A threshold d – the difference between the agents' utilities.*

We formulate the problem as determining if k columns exist to be deleted such that after executing AW, we obtain two sets of numbers, b_1 and b_2, representing the utilities for all allocated goods j belonging to each agent's bundle. This deletion ensures that:

$$|\sum_{t \in b_1} t - \sum_{t \in b_2} t| \leq d \tag{3}$$

and

$$b_1 \cup b_2 = a_1 \cup a_2 .$$

Observation 3. *The AWNS matrix variation problem is equivalent to the original AWNS problem, as the solution for the AWNS matrix variation is the same as the solution for the original AWNS problem.*

4.1 Computational Intractability (of AWNS)

We are interested in the computational complexity of AWNS.

Theorem 7. *AWNS is weak NP-hard.*

Proof. Given an instance of the Subset Sum problem with m numbers and a target number B (where, without loss of generality, we assume that B is a multiple to m^2), we create an instance of AWNS in its matrix variation, as follows:

- Initialize the matrix A as in the definition of AWNS.
- Append an additional row to the matrix A to represent the target sum B: set the last element of this row to B; let $A = \begin{bmatrix} X & 0 \\ 0 & B \end{bmatrix}$.
- Scale the entire matrix by another factor of m.

- To ensure equal row sums: compute z, the sum of elements in the first row of A (excluding the last element); compute S, the sum of elements in the second row of A; adjust $A[2][m]$, the last element of the second row, to match the sum of the first row: $A[2][m] = A[2][m] + (z - S)$.
- To ensure that $A[2][j] > A[1][j]$ for all $j \in \{1, \cdots, m\}$ and $A[2][m] < A[1][j]$: for each element $A[2][j]$ in the second row (except the last): while $A[2][j]$ is greater than $A[1][j]$: $A[2][j]$ by a multiple of m until it is less than $A[1][j]$; if $A[2][j] - m < 0$, continue to the next value without decreasing.
- Subtract $m - 1$ tokens from $A[2][1]$.
- Introduce an additional token to each cell $A[2][j]$, where $j \in \{2, \cdots, m\}$.

This finishes the description of the reduction. For correctness, intuitively, the matrix is crafted to ensure that the summation of any subset of numbers in the second row, taken modulo m, will never result in zero. This critical adjustment guarantees that any potential solution, if it exists for the subset sum problem (represented by the first row of the matrix), simultaneously fulfills the criteria for a no-split solution. Specifically, the sum of any subset of numbers from the second row of the matrix, excluding the last value in the $m + 1$ position, modulo m, will always yield a non-zero result as long as $|subset(a2)| < m$. More formally, we show that an instance is a yes instance for AWNT iff it is a yes instance for the constructed AWNS instance. To this end, note that if the Subset Sum problem has a solution, i.e., there exists a subset of k numbers (where $k < m$) whose sum is equal to the target number B', we can select the columns for agent 1 that correspond to these numbers. The resulting matrix A' will have $m - k + 1$ columns. When the Adjusted Winner method is applied to A', no good will be split between the two agents. Thus, the No Split problem has a solution. If the Subset Sum problem has no solution, the construction determines that the corresponding AWNS problem also has no solution. This is because for each subset of cells in the second row, the modulus of the total sum of that subset will always be different from zero, resulting in a split.

Conversely, if the AW-No Split problem has a solution, it implies that there exists a selection of $m - k + 1$ goods from agent 1 such that, after removal, the remaining goods can be allocated using the Adjusted Winner method without any good being split. This selection corresponds to a subset of numbers whose sum is equal to the target number B' in the Subset Sum problem. Therefore, the original Subset Sum problem also has a solution. □

4.2 Parameterized Complexity (of AWNS)

We go on to consider the parameterized complexity of AWNS.

Corollary 1. *The parameterized complexity of AWNS equals that of AWNT – for the results we have for AWNT; this follows by a careful look into the corresponding proofs, which shows that a "yes" instance is achieved without any transfers.*

4.3 Approximation Algorithms (of AWNS)

For AWNS we have analogous inapproximability results as for AWNT. Indeed, the proof of the next theorem is identical to the one for Theorem 2.

Theorem 8. *Unless P = NP, for any approximation ratio, there is no polynomial-time algorithm that provides a multiplicative approximation guarantee to minimize the threshold d (AWNS).*

Analogously, we present the AWNS-k optimization variant:

Definition 8 (AWNS-k)**.** *Let the minimal k Adjusted Winner No Split (AWNS-k) be an optimization version of the AWNS problem that minimizes k, the number of goods to delete while maintaining $|U_1(W_1') - U_1(W_2')| \leq d$.*

Theorem 9. *Unless P = NP, for any approximation ratio, there is no polynomial-time algorithm with a multiplicative approximation guarantee to minimize the number of goods k to delete (AWNS-k).*

The detailed proof will be presented in the full version. One may wonder now whether there also is a FPTAS for AWNS-r. We leave this open and remark that this seems technically very challenging. The main problem is that the scaling approach cannot be easily adapted, since, after scaling, the ratio of the items values changes and the ordering in which items are transferred can be different, with a potentially huge effect on possible solutions.

5 Computer-Based Simulations

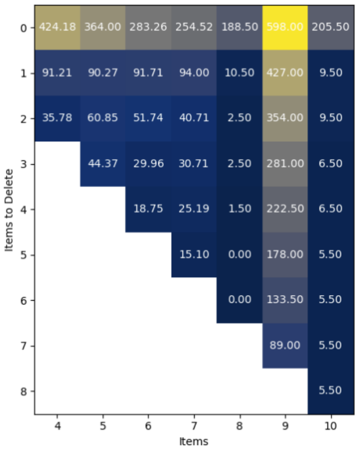
We complement our theoretical analysis of AWNT and AWNS with computer-based simulations. The main aim of these is to better investigate the effectiveness of removing goods and the dependence of such effectiveness on structural properties of different instances. Our experiments were conducted using several datasets – one based on real-world data and the others artificially-generated.

- **Real-world data**: The real-world data set corresponds to instances from the Spliddit, an online fair division platform [12]. We have performed the following processing over the data: we filtered the dataset to include only instances with two agents (as these are the relevant instances for AW), and to instances with between 4 to 10 goods (this was done for convenience, and contains the majority of the available Spliddit instances), resulting in 513 instances. Within this dataset, the total number of tokens (z), representing the sum of each row in the matrix, is standardized to 1000.
- **Artificial data**: The artificially-generated instances contain two agents and 4 to 10 goods. Following the literature on sampling distributions for fair division [5], we have used 3 distributions: Dirichlet resampling, Attributes model, and a Euclidean model. We have generated 252 instances for each of the three distributions, totaling 756 synthetic instances.

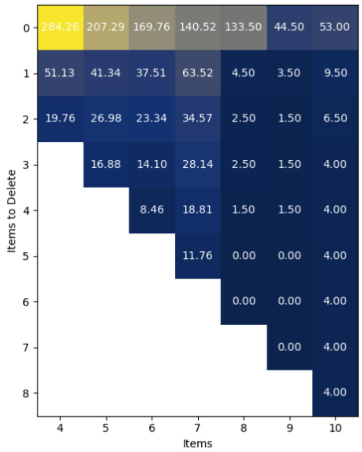
We describe the specific distributions and parameters used for simulations. To standardize the artificial dataset, we adjusted it to ensure that the sum of each row equals 1000 by adding or subtracting tokens to random cells while maintaining each value in a cell to be greater than or equal to 0. For the resampling, which serves as the base for the Dirichlet distribution, we generate a set of approved goods for each agent, over which the agent splits the total utility of 1 equally. The parameters include $p \in \{0.6, 0.4, 0.2\}$ denoting the probability of selecting a good for the central approval set V^*, and $\phi \in \{0.2, 0.8\}$ representing the probability of resampling a good's membership in the agent's approval set V. The procedure involves choosing the central approval set V^* by uniformly drawing $\lfloor p \cdot m \rfloor$ goods and generating each agent's approval set V by copying V^* and altering it according to the given probabilities. For the Dirichlet-Resampling model, we sample each agent's approval set V using the resampling model and set each agent's utility for goods outside of V to zero. The parameter $\alpha \in \{1, 2, 3\}$ denotes the shape parameter for the Dirichlet distribution. The procedure involves sampling each

Table 3. Simulation results for AWNT and AWNS for the Spliddit. In each of the plots: the x-axis is the total number m of goods; the y-axis is the number k of goods that are allowed to be removed; and the value in each cell is the average (over the simulation repetitions) of the difference in utility of the two agents.

AWNT



AWNS



Spliddit

agent's utility for the goods in V from a symmetric Dirichlet distribution with parameter α, scaling it by 100, and rounding it up. For the Attributes model, we sample vectors representing goods and agents' utility over attributes. The parameter $d \in \{2, 5\}$ indicates the number of attributes. The procedure involves sampling vectors for goods and agents uniformly at random from $[0, 1]^d$, where the agent's utility for a good is proportional to the dot product of the agent's attribute weights vector and the good's desirability vector. Lastly, for the Euclidean model, we sample vectors for goods and agents' utility over attributes and calculate utility based on Euclidean distance. The parameter $d \in \{2, 5\}$ denotes the dimensionality for sampling vectors. The procedure involves sampling vectors for agents and goods uniformly from $[0, 1]^d$ and calculating the agent's utility for a good based on the Euclidean distance between their attribute vectors.

Results. We have performed two experiments – one in which we wanted to see the average effectiveness of removing goods that will be shown in the full version, and one in which we wanted to see the structural properties that affect the effectiveness of removing goods as that will also be shown in the full version of this paper.

Experiment 1. In the first experiment, whose full results are in the full version, we tested how effective deleting goods is in the sense of reducing the utility difference between the two agents. To this end, we have considered all instances; for each, we considered k spanning from 0 to $m - 2$ (recall that m is the total number of goods and k is the number of goods to delete), and calculated the optimal difference d between the utility of the agents when up to k goods are deleted; both for AWNT and for AWNS. In Table 3 we present the results only for the Spliddit data-set for AWNT and AWNS, in which k is the x-axis, m is the y-axis, and the value in each cell is the optimal utility difference d for a combination of m total goods and up to k goods to delete.

Experiment 2. In the second experiment, our objective was to elucidate the correlation between the structure of an instance and its response to good deletion, specifically examining its receptiveness to removing goods. To accomplish this, building upon the findings of the previous experiment which indicated that the maximum gain from deleting goods, in terms of the change in d, was achieved when transitioning from no goods to delete to one good to delete, we conducted a comprehensive analysis across all datasets. We aimed to model the ℓ_1 distance between the utilities of the two agents as a function of the gain, denoted as $d_0 - d_1$, where d_0 represents the difference when no goods are removed, and d_1 is the minimal difference observed when one good is removed. While the artificial data showed a very low correlation, the Spliddit data, resulted in a high correlation of 0.677 for AWNS and 0.569 for AWNT. It will also be illustrated in the the full version of this paper in the future.

Discussion. We discuss our conclusions from the simulations: (1) As expected, for all artificial and real instances, deleting the first column had the most significant impact. Interestingly, after deleting one or two more columns, this impact decreases drastically; (2) The maximum gain of an instance from deleting columns can be expressed as a function of the distribution difference between the rows. This implies that, the more different the instances are, the more gain an instance will receive from deleting a column; (3) It seems that the more goods needed to be deleted in order to reach half of the optimal d, the more likely the difference between the two rows and the distribution, will be low.

6 Outlook

We have investigated an approach to avoiding conflicts in AW – in the sense of avoiding transferring or splitting goods within the process of Adjusted Winner – by removing a few goods. Our theoretical results suggest that, while the corresponding combinatorial problems are generally intractable, there are efficient parameterized algorithms, pseudopolynomial algorithms, and approximation algorithms for the problems. Our computer-based simulations show the effectiveness of the approach in reducing the utility difference between the agents and show how the more misaligned the agents are the more effective is the approach. Some avenues for future research include: (1) pinpointing whether an FPTAS for AWNS-r exists; (2) identifying structural properties of instances that affect their behavior w.r.t. removing goods (theoretically, and via simulations that follow the map of elections framework for fair division [5]); (3) studying other actions besides removing goods (e.g., modifying goods, removing utility tokens, or adding goods); and (4) considering other optimization goals (e.g., minimizing the number of discarded tokens denoted by z).

References

[1] A. Abboud, K. Lewi, and R. Williams. On the parameterized complexity of k-sum. *CoRR, abs/1311.3054*, 2013.

[2] G. Amanatidis, G. Birmpas, A. Filos-Ratsikas, and A. A. Voudouris. Fair division of indivisible goods: A survey. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI '22)*, pages 5385–5393. ijcai.org, 2022.

[3] H. Aziz, S. Brânzei, A. Filos-Ratsikas, and S. K. S. Frederiksen. The adjusted winner procedure: Characterizations and equilibria. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI '15)*, pages 454–460. AAAI Press, 2015.

[4] J. J. Bartholdi III, C. A. Tovey, and M. A. Trick. How hard is it to control an election? *Mathematical and Computer Modelling*, 16(8-9): 27–40, 1992.

[5] P. Böhm, R. Bredereck, P. Gölz, A. Kaczmarczyk, and S. Szufa. Putting fair division on the map, 2023.

[6] S. Bouveret and M. Lemaître. Characterizing conflicts in fair division of indivisible goods using a scale of criteria. *Autonomous Agents and Multi-Agent Systems*, 30(2):259–290, 2016.

[7] Brams. Fair division: A new approach to the spratly islands controversy. *International Negotiation*, 2(2):303–329, 1997.

[8] S. J. Brams and A. D. Taylor. *Fair Division: From cake-cutting to dispute resolution*. Cambridge University Press, 1996.

[9] N. Dupuis-Roy and F. Gosselin. An empirical evaluation of fair-division algorithms. In *Proceedings of the annual meeting of the cognitive science society*, volume 31, 2009.

[10] P. Faliszewski and J. Rothe. Control and bribery in voting., 2016.

[11] M. R. Garey and D. S. Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979.

[12] J. Goldman and A. D. Procaccia. Spliddit: Unleashing fair division algorithms. *ACM SIGecom Exchanges*, 13(2):41–46, 2015.

[13] D. M. Kilgour and R. Vetschera. Two-person fair division with additive valuations. *Group Decision and Negotiation*, pages 1–30, 2024.

[14] K. Koiliaris and C. Xu. Subset sum made simple. *arXiv preprint arXiv:1807.08248*, 2018.

[15] T. G. Massoud. Fair division, adjusted winner procedure (aw), and the Israeli-Palestinian conflict. *Journal of Conflict Resolution*, 44(3):333–358, 2000.

[16] E. Pacuit. Towards a logical analysis of adjusted winner. *Proof, Computation and Agency: Logic at the Crossroads*, pages 229–239, 2011.

[17] S. Salame, E. Pacuit, and R. Parikh. Some results on adjusted winner. *Synthese*, 2005.