# Prediction-Based Selective Negotiation for Refining Multi-Agent Resource Allocation

Madalina Croitoru[1], Cornelius Croitoru[2] and Gowrishankar Ganesh[3]
[1]*University of Montpellier, LIRMM, France*
[2]*Faculty of Computer Science, Al. I. Cuza Univerisity, Iasi, Romania*
[3]*LIRMM, CNRS (National Center of Scientific Research), France*

Keywords: Collective Decision Making, Multi Agent Systems, Negotiation.

Abstract: This paper proposes a 2-stage framework for multi-agent resource allocation. Following a Borda-based allocation, machine learning predictions about agent preferences are used to selectively choose agent pairs to perform negotiations to swap resources. We show that this selective negotiation improves overall satisfaction towards the resource redistribution.

## 1 INTRODUCTION

Resource allocation (Ibaraki and Katoh, 1988) is a core problem in a wide range of real-world multi-agent applications (Chevaleyre et al., 2005), from distributing medical supplies in emergency situations (Zhang et al., 2016) to assigning computational resources in cloud computing (Vinothina et al., 2012) or allocating advertising slots to bidders in digital marketplaces (Li et al., 2018). Such problems require dividing limited resources among competing agents, each with their own preferences. Finding optimal solutions to such problems is typically NP-hard, with linear programming methods used for approximate solutions (Katoh and Ibaraki, 1998; Croitoru and Croitoru, 2011).

Advancements in generative Artificial Intelligence (AI) and autonomous systems are shaping hybrid societies where human agents and artificial systems are deeply interconnected. In these environments, subjective, context-dependent human preferences must integrate with the pre-coded preferences of artificial agents. Resource allocation problems involving human agents should consider contextual factors such as socioeconomic status or geographic location, as these can influence priorities that may evolve during the allocation process (Dafoe et al., 2020). For instance, in digital marketplaces, preferences for goods or services can be inferred from browsing behavior or demographic profiles. Similarly, in education, predictive models can guide resource allocation to students based on their specific learning needs.

In hybrid societies, achieving the global good requires systems that facilitate fair trade-offs among diverse stakeholders, balancing individual preferences with collective outcomes. This paper examines a straightforward approach to negotiation: the swapping of goods. This method serves as a form of compromise, enabling mutually acceptable outcomes through direct exchanges while supporting the broader collective interest. We propose a 2-stage framework refining multi-agent resource allocation:

1. In the first step, individual preferences are aggregated using the Borda voting method to create a collective preference ranking. This aggregated ranking, combined with the individual preferences of each agent, is used to allocate goods in a greedy manner. Starting with the most preferred good in the Borda aggregated ranking, goods are allocated to the maximum number of agents, proceeding sequentially to less preferred goods.
2. In the second step, following a Condorcet-like approach, pairs of agents are identified based on machine learning predictions of their features, indicating potential for compromise. These agents are then considered for swapping their allocated goods to enhance overall societal satisfaction by aligning the allocations with predicted preferences.

In this work, our contributions are threefold:

- We formalize the 2-stage framework for prediction-based resource allocation.
- We design an algorithm that integrates preference prediction and preference aggregation for overall agent satisfaction.



Croitoru, M., Croitoru, C. and Ganesh, G.
Prediction-Based Selective Negotiation for Refining Multi-Agent Resource Allocation.
DOI: 10.5220/0013368000003890

In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 1*, pages 656-662
ISBN: 978-989-758-737-5; ISSN: 2184-433X

- Third, we analyse the satisfaction guarantees provided by the proposed framework.

The paper is structured as follows. Section 2 provides a motivating example illustrating how the use of swaps can improve outcomes compared to a simple naive allocation. The naive allocation relies on a lexicographical ordering of agents, assigning each their most preferred goods in sequence until all goods are allocated. Section 3 formally defines the resource allocation problem and introduces the pseudocode for the naive algorithm used in the motivating example presented in Section 2. Section 4 details the Borda voting method, which serves as the basis for the refined allocation process described in the paper, and the Condorcet voting method, which relies on pairwise comparisons of preferences. The intuition behind Condorcet's pairwise comparisons is used to justify why agents might swap goods. Section 5 introduces our two-step framework and its theoretical satisfaction guarantees. The framework (i) uses Borda aggregation for an initial allocation, followed by (ii) optimization through swaps between agents, guided by feature-based preference predictions. Section 6 concludes the paper.

## 2 MOTIVATING EXAMPLE

This section shows an illustrative example involving 10 agents and 5 goods, each with a maximum availability of 4 units. The scenario explains how a simple greedy initial allocation, while respecting multiplicity constraints, may fail to achieve optimal satisfaction. Using predicted preferences provided by a machine learning model, we are then able to target the agents that might engage in negotiation to improve the allocation by augmenting overall satisfaction.

We consider 10 children $A = \{a_1, a_2, \ldots, a_{10}\}$ and a pool of 5 goods $G = \{g_1, g_2, g_3, g_4, g_5\}$, each with a multiplicity of 4. Each child has a valuation function $v_i : G \to \{15, 10, 5, 0\}$, representing their preferences for the goods. The top three preferences for each child are shown in Table 1.

An initial allocation respecting multiplicities (no good allocated to more than 4 children) is:

$$O_1 = \{g_1\}, O_2 = \{g_1\}, O_3 = \{g_1\}, O_4 = \{g_1\},$$
$$O_5 = \{g_2\}, O_6 = \{g_2\}, O_7 = \{g_3\}, O_8 = \{g_3\},$$
$$O_9 = \{g_4\}, O_{10} = \{g_5\}.$$

The overall initial allocation satisfaction is:

$$\begin{aligned} v_A(O) &= \sum_{i=1}^{10} v_i(O_i) \\ &= 15+15+15+15+10+15+15+10 \\ &\quad +15+0 = 135. \end{aligned}$$

Using negotiations children can adjust the allocation to improve overall satisfaction:

- $a_5$, receiving $g_2$ (value 10), negotiates with $a_{10}$ to swap $g_2$ for $g_1$, as $g_1$ is $a_5$'s top preference and $g_2$ is $a_{10}$'s second preference.
- $a_{10}$, currently receiving $g_5$ (value 0), negotiates with $a_7$ to swap $g_5$ for $g_3$, as $g_3$ is $a_{10}$'s top preference and $g_5$ is in $a_7$'s top three.

After negotiation, the final allocation is:

$$O_1^* = \{g_1\}, O_2^* = \{g_1\}, O_3^* = \{g_1\},$$
$$O_4^* = \{g_1\}, O_5^* = \{g_1\}, O_6^* = \{g_2\}, O_7^* = \{g_2\},$$
$$O_8^* = \{g_3\}, O_9^* = \{g_4\}, O_{10}^* = \{g_3\}.$$

The overall final allocation satisfaction is:

$$v_A(O^*) = \sum_{i=1}^{10} v_i(O_i^*) = 150.$$

## 3 RESOURCE ALLOCATION

A resource allocation problem is defined as a tuple $(A, G, M, V)$, where:

- $A = \{a_1, a_2, \ldots, a_n\}$ is the set of $n \geq 2$ agents.
- $G = \{g_1, g_2, \ldots, g_m\}$ is the set of $m \geq 1$ goods.
- $M = \{m_1, m_2, \ldots, m_m\}$ is the multiplicity $m_j \geq 1$ of each good $g_j \in G$, representing the maximum number of agents that can receive $g_j$. The total availability of goods is $\sum_{j=1}^{m} m_j$.
- $V = \{v_1, v_2, \ldots, v_n\}$ is the set of ordinal preference functions, one for each agent $a_i \in A$. Each $v_i$ defines a strict ranking over the goods $G$, such that for any two goods $g_j, g_k \in G$, $v_i(g_j) < v_i(g_k)$ implies that $g_j$ is strictly preferred to $g_k$ by agent $a_i$.

An allocation $O = \{O_1, O_2, \ldots, O_n\}$ is a mapping of goods to agents, where $O_i \subseteq G$ denotes the subset of goods allocated to agent $a_i$. The allocation must satisfy the multiplicity constraints:

$$\sum_{a_i \in A} \mathbb{1}_{g_j \in O_i} \leq m_j, \quad \forall g_j \in G,$$

where $\mathbb{1}_{g_j \in O_i}$ is an indicator function that equals 1 if $g_j \in O_i$ and 0 otherwise.

The (individual) satisfaction of agent $a_i$ is a function $S_i$ that associates to each subset $G' \subseteq G$ of goods

Table 1: Top 3 goods and their valuations for each child.

| Child | Top 3 Goods (Values $v_i(g)$) |
|---|---|
| $a_1$ | $v_1(g_1) = 15, v_1(g_2) = 10, v_1(g_3) = 5$ |
| $a_2$ | $v_2(g_1) = 15, v_2(g_3) = 10, v_2(g_4) = 5$ |
| $a_3$ | $v_3(g_1) = 15, v_3(g_4) = 10, v_3(g_5) = 5$ |
| $a_4$ | $v_4(g_1) = 15, v_4(g_5) = 10, v_4(g_2) = 5$ |
| $a_5$ | $v_5(g_1) = 15, v_5(g_2) = 10, v_5(g_3) = 5$ |
| $a_6$ | $v_6(g_1) = 15, v_6(g_3) = 10, v_6(g_4) = 5$ |
| $a_7$ | $v_7(g_1) = 15, v_7(g_4) = 10, v_7(g_5) = 5$ |
| $a_8$ | $v_8(g_2) = 15, v_8(g_3) = 10, v_8(g_4) = 5$ |
| $a_9$ | $v_9(g_2) = 15, v_9(g_3) = 10, v_9(g_5) = 5$ |
| $a_{10}$ | $v_{10}(g_3) = 15, v_{10}(g_4) = 10, v_{10}(g_1) = 5$ |

a positive real value $S_i(G')$. This value is strongly dependent of the preferences values $v_i(g)$, for $g \in G'$. The satisfaction of an allocation $O$ is the sum of the satisfactions of the agents for their respective allocations: $S(O) = \sum_{i=1}^{n} S_i(O_i)$. The objective is to find an allocation $O^*$ that maximizes satisfaction over all possible allocations.

A straightforward allocation algorithm can allocate goods to agents based on their ordinal preferences in a sequential manner. For each agent $a_i \in A$ (in a pre-defined order) the algorithm will allocate the most preferred good $g_{(1)}$ of $a_i$ that is still available, if $g_{(1)}$ is no longer available, allocate the next most preferred good $g_{(2)}$, and so on, until either a good is allocated or all preferred goods are unavailable.

This greedy approach ensures that each agent receives the best possible good based on availability, prioritizing agents in the order they are processed. In order to keep the code simple, we used the command *Proceed to the next agent* which tries to be equitable enough and which can restart from the first agent if there are unallocated goods and also agents $a_i$ with $O_i \neq G$. Obviously, this algorithm may not result in optimal overall satisfaction.

**Input:** $A = \{a_1, a_2, \ldots, a_n\}$ (agents in a predefined order),
$G = \{g_1, g_2, \ldots, g_m\}$ (set of goods),
$M = \{m_1, m_2, \ldots, m_m\}$ (multiplicities of goods),
$V = \{v_1, v_2, \ldots, v_n\}$ (ordinal preference functions for each agent).
**Output:** An allocation $O = \{O_1, O_2, \ldots, O_n\}$ where $O_i$ is the set of goods allocated to agent $a_i$.

```
Initialize O_i ← ∅ for all a_i ∈ A ;
No_goods ← Σ_{j=1}^{m} m_j;
while No_goods > 0 do
    foreach a_i ∈ A do
        foreach g_j ∈ G in order given by v_i do
            if m_j > 0 and g_j ∉ O_i then
                O_i ← O_i ∪ {g_j} ;
                m_j ← m_j − 1 ;
                No_goods ← No_goods − 1 ;
                Proceed to the next agent;
            end
        end
    end
end
return O
```

Algorithm 1: Simple Allocation Algorithm.

# 4 PREDICTIVE-BASED RESOURCE ALLOCATION

In this section, we introduce the concepts that underpin the two-step framework for refining multi-agent resource allocation. The first step relies on the aggregated preferences of agents, calculated using social choice methods outlined in Section 4.1 and detailed in (Brandt et al., 2016). The second step involves the use of classifiers (Jordan and Mitchell, 2015), discussed in Section 4.2, to identify pairs of agents for good-swapping, inspired by Condorcet principles.

## 4.1 Social Choice

Let us now define an aggregated preference $v_A$ that combines the individual preferences $v_1, v_2, \ldots, v_n$ of all agents $A = \{a_1, a_2, \ldots, a_n\}$ into a single collective preference over the goods $G = \{g_1, g_2, \ldots, g_m\}$. The aggregated preference $v_A$ can be formally expressed as:

$$v_A = \text{Aggregation}(v_1, v_2, \ldots, v_n),$$

where, Aggregation is the function (e.g., Borda or Condorcet) used to combine the individual preferences into a collective ranking.

In the Borda method, a score is assigned to goods based on their ordinal rankings in each agent's prefer-

ence. The aggregated preference $v_A$ is defined as:

$$v_A(g_j) = \sum_{i=1}^{n} (n - v_i(g_j) + 1),$$

- $v_i(g_j)$ is the rank of $g_j$ in agent $a_i$'s preference list (lower ranks indicate higher preference),
- $(n - v_i(g_j) + 1)$ converts the rank into a score (higher scores indicating higher preference).

The goods are then ordered in descending order of $v_A(g_j)$ to form the aggregated preference.

In the Condorcet method, the aggregated preference is determined through pairwise comparisons. For each pair of goods $(g_j, g_k$, a preference matrix $M$ is constructed:

$$M(g_j, g_k) = \sum_{i=1}^{n} \mathbb{1}_{v_i(g_j) < v_i(g_k)},$$

- $M(g_j, g_k)$ is the number of agents preferring $g_j$ to $g_k$,
- $\mathbb{1}_{v_i(g_j) < v_i(g_k)} = 1$ if $g_j$ is ranked higher than $g_k$ by agent $a_i$, and 0 otherwise.

A good $g_j$ is a Condorcet winner if it is preferred to every other good in pairwise comparisons:

$$M(g_j, g_k) > M(g_k, g_j) \quad \forall g_k \neq g_j.$$

An aggregation function for collective preferences should adhere to several key principles to ensure fairness and rationality: Pareto efficiency, Non-Dictatorship, Independence of Irrelevant Alternatives (IIA), and Anonymity.

Pareto Efficiency requires that if all agents prefer a good $g_j$ over another good $g_k$, the collective preference $v_A$ ranks $g_j$ higher than $g_k$:

$$v_i(g_j) < v_i(g_k) \quad \forall i \in A,$$

then the aggregated preference must satisfy:

$$v_A(g_j) < v_A(g_k).$$

Non-Dictatorship ensures that no single agent $a_i \in A$ can unilaterally determine the collective preference $v_A$, unless their preferences align with all agents unanimous preference:

$$\exists g_j, g_k \in G \text{ such that } v_i(g_j) < v_i(g_k) \text{ and } v_A(g_k) < v_A(g_j),$$

for at least one pair of goods $g_j, g_k$ and $a_i$.

Independence of Irrelevant Alternatives (IIA) ensures that the collective ranking of two goods $g_j$ and $g_k$ depends on their relative rankings in individual preferences, unaffected by the presence or absence of other goods. Formally, if:

$$v_i(g_j) < v_i(g_k) \quad \forall i \in A,$$

then the aggregated preference must also preserve this ordering:

$$v_A(g_j) < v_A(g_k).$$

Anonymity requires that the aggregation mechanism treats all agents equally, i.e. the outcome is invariant to the permutation of agent indices.

The Condorcet method satisfies Pareto efficiency, as any unanimously preferred good will dominate others in pairwise comparisons. It also respects Non-Dictatorship, as no single agent can dictate outcomes unless their preferences align with the unanimous preference of all agents. Condorcet also satisfies Anonymity, as all agents are treated symmetrically in the aggregation process. However, Condorcet violates IIA because the addition or removal of other goods can alter pairwise comparison results. A Condorcet winner might not always exist (Arrow, 2012). The Borda method equally violates IIA and respects Anonymity, as the scoring mechanism treats all agents equally. However, Borda fails to satisfy Pareto efficiency because the collective ranking can assign higher aggregate scores to a good $g_k$ that is unanimously less preferred than $g_j$. Furthermore, Borda does not adhere to the Condorcet criterion, as it can select a good that loses in pairwise comparisons to another (Arrow, 2012).

As an example consider four agents $A = \{a_1, a_2, a_3, a_4\}$ and three goods $G = \{g_1, g_2, g_3\}$ with the following preferences:

$$a_1 : g_1 \succ g_2 \succ g_3, \quad a_2 : g_2 \succ g_3 \succ g_1,$$
$$a_3 : g_3 \succ g_1 \succ g_2, \quad a_4 : g_3 \succ g_2 \succ g_1.$$

Using Borda points are assigned as 2 for first place, 1 for second, and 0 for third:

$$g_1 : 3, \quad g_2 : 4, \quad g_3 : 5.$$

The Borda ranking is:

$$g_3 \succ g_2 \succ g_1.$$

For Condorcet each pair of goods is compared across agents:

$$g_1 \text{ vs. } g_2 : 2 \text{ votes for each} \quad \text{(Tie)},$$
$$g_1 \text{ vs. } g_3 : 2 \text{ votes for each} \quad \text{(Tie)},$$
$$g_2 \text{ vs. } g_3 : 2 \text{ votes for each} \quad \text{(Tie)}.$$

Since no good wins all pairwise comparisons, no Condorcet winner exists.

## 4.2 Preference Classifiers

Classifiers in Machine Learning analyse data features to learn patterns that distinguish between different categories. Features are measurable attributes or

properties of the data that are considered relevant for the classification task. The classifier evaluates these features to identify patterns or decision boundaries that separate classes.

A classifier can be formalized as a function $f : \mathcal{X} \to \mathcal{Y}$, where:

- $\mathcal{X}$ is the input space, representing the set of possible feature vectors, $x = (x_1, x_2, \ldots, x_d) \in \mathcal{X}$, where $d$ is the number of features.
- $\mathcal{Y}$ is the output space, which contains the set of possible labels or classes, typically $\mathcal{Y} = \{1, 2, \ldots, C\}$, where $C$ is the number of classes.
- $f(x;\theta)$ is the decision function, parameterized by $\theta$, which maps input features $x$ to a predicted class $\hat{y} \in \mathcal{Y}$.

The classifier is trained on a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, where $x_i \in \mathcal{X}$ are feature vectors and $y_i \in \mathcal{Y}$ are the corresponding ground truth labels. The objective during training is to optimize the parameters $\theta$ by minimizing a loss function $L$, i.e. the discrepancy between the predicted labels $\hat{y}_i = f(x_i;\theta)$ and the true labels $y_i$:

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(f(x_i;\theta), y_i).$$

The classifier can also incorporate a hypothesis space $\mathcal{H}$, representing the set of all possible decision functions $f$ that can be chosen given the parameterization:

$$f \in \mathcal{H}, \quad \mathcal{H} = \{f(x;\theta) \mid \theta \in \Theta\},$$

where, $\Theta$ is the parameter space.

Once trained, the classifier predicts the label for a new input $x^*$ by computing:

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} P(y \mid x^*; \hat{\theta}),$$

where, $P(y \mid x^*; \hat{\theta})$ represents the model's estimated probability of class $y$ for the input $x^*$, given the optimized parameters $\hat{\theta}$.

The Predictive-Based Resource Allocation Problem extends the traditional resource allocation problem by incorporating classifiers to predict preferences based on agent feature as follows:

$$(A, G, M, \mathcal{F}, \mathcal{C}),$$

where:

- $A = \{a_1, a_2, \ldots, a_n\}$ is the set of $n$ agents.
- $G = \{g_1, g_2, \ldots, g_m\}$ is the set of $m$ goods.
- $M = \{m_1, m_2, \ldots, m_m\}$ specifies the maximum number of agents $m_j$ that can receive each good $g_j$, satisfying $\sum_{j=1}^{m} m_j \geq n$.
- $\mathcal{F}$ is the feature space, where each agent $a_i$ is associated with a feature vector $x_i = (f_{i1}, \ldots, f_{ip})$.
- $\mathcal{C}$ is the set of classifiers, where each classifier $f_{c_k} : \mathcal{X} \to \mathcal{Y}$ predicts the preference rankings of agents based on their features.

For each agent $a_i$, the predicted preference ranking over goods is given by:

$$\hat{v}_i = f_{c_k}(x_i),$$

where $f_{c_k}$ is a classifier applied to $x_i$, and $\hat{v}_i$ defines the predicted strict ranking of goods $G$.

An *allocation* $O = \{O_1, \ldots, O_n\}$ maps subsets of goods to agents, satisfying the multiplicity:

$$\sum_{a_i \in A} \mathbb{1}_{g_j \in O_i} \leq m_j, \quad \forall g_j \in G,$$

where $\mathbb{1}_{g_j \in O_i} = 1$ if $g_j \in O_i$, and 0 otherwise.

The satisfaction of agent $a_i$ is a function $S_i$ that associates to each subset $G' \subseteq G$ of goods a positive real value $S_i(G')$. Note that here, the individual satisfaction $S_i$ of agent $a_i$ depends on the predicted preference ranking $\hat{v}_i$. The satisfaction of an allocation $O$ is the sum of the satisfaction of the agents for their respective allocations: $S(O) = \sum_{i=1}^{n} S_i(O_i)$. The objective is to find an allocation $O^*$ that maximizes satisfaction.

# 5 OPTIMISING THE ALLOCATION

The algorithm introduced in this section operates in four phases, each presented as a distinct algorithm for lisibility. The first phase, which corresponds to the naive allocation based on the lexicographical order of agents, is identical to the algorithm described in Section 3 and is repeated here for readability. In the second phase, the Borda aggregation is computed, which is then used in the third phase to perform a Borda-based allocation of goods. Finally, in the fourth phase, the allocation is refined through classifier-driven swaps to further enhance satisfaction. The last two phases form the two-step framework for prediction-based refinement of the allocation.

In the second phase, the Borda method is applied to aggregate individual preferences into a collective ranking of goods. Each good receives a score based on its rank in the agents' preference lists. The goods are then ordered by their scores to produce a collective preference ranking that reflects the priorities of the group.

**Input:** $A = \{a_1, a_2, \ldots, a_n\}$: Set of agents;
$G = \{g_1, g_2, \ldots, g_m\}$: Set of goods;
$M = \{m_1, m_2, \ldots, m_m\}$: Multiplicities of goods;
$\{v_1, v_2, \ldots, v_n\}$: Agents' preferences over goods;
A lexicographical order on $A$;
**Output:** Initial allocation $O = \{O_1, O_2, \ldots, O_n\}$
Construct allocation $O$ using Algorithm in Section 3 or using any else Heuristic;
Compute overall satisfaction $S(O)$ of the initial allocation;
**return** $O$;

Algorithm 2: Phase 1: Initial Allocation.

**Input:** $A = \{a_1, a_2, \ldots, a_n\}$: Set of agents;
$G = \{g_1, g_2, \ldots, g_m\}$: Set of goods;
$\{v_1, v_2, \ldots, v_n\}$: Agents' preferences over goods;
**Output:** Aggregated preference $v_A$
**foreach** $g_j \in G$ **do**
    Compute Borda score:

$$B(g_j) = \sum_{i=1}^{n} \text{score}(g_j \text{ in } v_i)$$

**end**
Sort $G$ in descending order of Borda scores to form aggregated preference $v_A$;
**return** $v_A$;

Algorithm 3: Phase 2: Aggregated Preferences (Borda Method).

In the third phase, the algorithm refines the Borda allocation to improve overall satisfaction according to the collective preferences. Starting with the most preferred good in the collective ranking, the algorithm allocates goods to agents in a way that maximizes the number of agents receiving goods they rank highly. This process continues for successive goods until either the goods are exhausted or further allocations no longer align with the collective ranking.

In the final phase, predictive models are used to adjust the allocation further. For each agent, a classifier predicts preferences based on their feature profile. Agents whose predicted preferences differ from their true preferences are identified, and the algorithm proposes swaps with other agents in order to increase the overall satisfaction of the allocation. The swaps are done iteratively to improve the alignment of the allocation. More precisely, a swap between agents $a_i$ and $a_k$ means that goods $ginO_i$ and $g' \in O_k$ are identified such that

$$S_i((O_i - \{g\}) \cup \{g'\}) + S_k((O_k - \{g'\}) \cup \{g\}) > S_i(O_i) + S_k(O_k).$$

**Input:** $A = \{a_1, a_2, \ldots, a_n\}$: Set of agents;
$G = \{g_1, g_2, \ldots, g_m\}$: Set of goods;
Aggregated preference $v_A$;
Current allocation $O$;
$\{m'_1, m'_2, \ldots, m'_m\}$: Current multiplicity of available ($m'_j > 0$) goods;
**Output:** Updated allocation $O$
**foreach** $g_j \in G$ *(in $v_A$-descending order)* **do**
    **foreach** $a_i \in A$ **do**
        **if** $m'_j > 0$ *and* $g_j \notin O_i$ **then**
            Allocate $g_j$ to $a_i$: $O_i \leftarrow O_i \cup \{g_j\}$;
            Update $m'_j \leftarrow m'_j - 1$;
        **end**
    **end**
**end**
**return** $O$;

Algorithm 4: Phase 3: Satisfaction Aggregated Preference Boosting.

**Input:** $A = \{a_1, a_2, \ldots, a_n\}$: Set of agents;
$G = \{g_1, g_2, \ldots, g_m\}$: Set of goods;
Feature space $\mathcal{F}$;
Classifiers $\mathcal{C}$; Current allocation $O$;
**Output:** Optimized allocation $O^*$
**foreach** $a_i \in A$ **do**
    Predict preferences using classifier:

$$\hat{v}_i = f_{c_k}(x_i)$$

    **if** $\hat{v}_i \neq v_i$ **then**
        Identify agents $a_k$ such that a swap with $a_i$ is possible;
        Propose swaps between $a_i$ and $a_k$ to improve satisfaction;
        Update $O_i$ and $O_k$ if swaps are accepted;
    **end**
**end**
**return** $O^* = O$;

Algorithm 5: Phase 4: Optimization.

The algorithm outputs an optimized allocation that improves on the naive approach by incorporating both collective preferences and predictions. In each of the Phases 1, 3 and 4, an implicit assumption is made: each (individual) satisfaction function $S_i$ of the agent $a_i$ is monotone:

$$\text{if } G_1 \subseteq G_2 \subseteq G \text{ then } S_i(G_1) \leq S_i(G_2).$$

Then, it is not difficult to see that the following theorem holds.

**Theorem 1.** *Let $O$ be the naive initial allocation and $O^*$ the final allocation produced by the Predictive-Based Resource Allocation Algorithm. If all satisfaction function $S_i$ of the agents are monotone, then*

$$S(O^*) \geq S(O),$$

*where $S(\cdot)$ is the overall satisfaction of an allocation.*

## 6 FUTURE WORK

In this paper, we proposed a predictive-based resource allocation algorithm that combines machine learning predictions with preference aggregation techniques to optimize resource allocation in multi-agent systems.

- Experimental validation is necessary to assess the framework performance in diverse practical settings (cloud resource allocation, logistics, public goods distribution etc.). This includes a formal computational complexity analysis evaluating scalability of the proposed algorithm as well as execution time, memory usage, and performance to assess the scalability of your approach while varying distributions of preferences and goods.
- Apart from the theoretical study of the properties such as envy-freeness or equitability another notion that should be investigated is that of "stable" allocation (the agents do not have any incentive to further swap).
- When several agents are eligible for a swap based on their profiles and predicted preferences, criteria for choosing the most appropriate participants need to be defined. This decision introduces potential concerns regarding ethics and bias, particularly when prioritizing agents could inadvertently favor certain groups over others (Hurwicz, 1973).
- Furthermore, the notion of overall satisfaction could be refined to incorporate subpopulation-specific goals. For instance, rather than optimizing global satisfaction, the algorithm could prioritize improving the satisfaction of specific subpopulations (e.g., AI-agents vs humans in hybrid societies) based on implicit or explicit norms (Aldewereld et al., 2016). Similarly to as above, this issue is directly related to the fairness and ethical concerns of our approach.

## REFERENCES


Aldewereld, H., Dignum, V., and Vasconcelos, W. W. (2016). Group norms for multi-agent organisations. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 11(2):1–31.

Arrow, K. J. (2012). *Social choice and individual values*, volume 12. Yale university press.

Brandt, F., Conitzer, V., Endriss, U., Lang, J., and Procaccia, A. D. (2016). *Handbook of computational social choice*. Cambridge University Press.

Chevaleyre, Y., Dunne, P. E., Endriss, U., Lang, J., Lemaitre, M., Maudet, N., Padget, J., Phelps, S., Rodrígues-Aguilar, J. A., and Sousa, P. (2005). Issues in multiagent resource allocation.

Croitoru, M. and Croitoru, C. (2011). Generalised network flows for combinatorial auctions. In *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 313–316. IEEE.

Dafoe, A., Hughes, E., Bachrach, Y., Collins, T., McKee, K. R., Leibo, J. Z., Larson, K., and Graepel, T. (2020). Open problems in cooperative ai. *arXiv preprint arXiv:2012.08630*.

Hurwicz, L. (1973). The design of mechanisms for resource allocation. *The American Economic Review*, 63(2):1–30.

Ibaraki, T. and Katoh, N. (1988). *Resource allocation problems: algorithmic approaches*. MIT press.

Jordan, M. I. and Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260.

Katoh, N. and Ibaraki, T. (1998). Resource allocation problems. *Handbook of Combinatorial Optimization: Volume1–3*, pages 905–1006.

Li, J., Ni, X., Yuan, Y., and Wang, F.-Y. (2018). A hierarchical framework for ad inventory allocation in programmatic advertising markets. *Electronic Commerce Research and Applications*, 31:40–51.

Vinothina, V. V., Sridaran, R., and Ganapathi, P. (2012). A survey on resource allocation strategies in cloud computing. *International Journal of Advanced Computer Science and Applications*, 3(6).

Zhang, J., Zhang, M., Ren, F., and Liu, J. (2016). An innovation approach for optimal resource allocation in emergency management. *IEEE Transactions on Computers*.