Don't Try This at Home: Examining How LLMs Perform Fair Division

Benjamin Cookson Soroush Ebadian Nisarg Shah

University of Toronto
{bcookson,soroush,nisarg}@cs.toronto.edu

Abstract

Large language models (LLMs) are increasingly being leveraged for decision-making tasks, yet their ability to make these decisions fairly remains underexplored. We conduct a fairness evaluation of LLM-based decision-making for allocating indivisible goods, using both synthetic data and real-world data from `Spliddit.org`.

Specifically, we examine what the state-of-the-art LLMs consider to be fair, depending on the real-world context and prompt framing, and how they compare to traditional algorithms on common fair division desiderata. We also conduct an evaluation of their reasoning abilities when tasked with meeting specific criteria. Our results indicate that LLMs are not yet ready to be used for performing fair division in the real world: while they strike a good trade-off between conflicting desiderata in small instances, they are still outperformed by state-of-the-art traditional algorithms, and their performance degrades quickly as the instances grow larger.

1 Introduction

The concept of *fairness* has captivated human thought for centuries, shaping the foundations of our core institutions, such as democracy, law, and healthcare. But what does fairness truly entail? While universally appealing, fairness is far from universally defined, and its interpretation often depends on the lens through which it is examined.

Fairness is a quintessential sociotechnical concept, explored extensively across disciplines. Philosophy deliberates the underlying principles of fairness, comparing Rawls' 1971 egalitarianism to Harsanyi's 1975 utilitarianism, and examining concepts such as desert, the right to a minimum, and fair equality of opportunity. Meanwhile, the machine learning literature takes a mathematical perspective on fairness, and often narrows its focus to deal with the most practically relevant issues such as mitigating race- or gender-based discrimination Mehrabi *et al.* [2021]. The fair division literature, at the intersection of economics and computer science, also takes a mathematical perspective, but formalizes individual and group fairness principles in an abstract resource allocation context devoid of specific attributes such as race or gender Amanatidis *et al.* [2022]; Shah [2023]. Finally, studies on human perceptions of fairness provide a descriptive counterpart to these normative approaches to fairness Grgic-Hlaca *et al.* [2018]; Srivastava *et al.* [2019]; Saxena *et al.* [2019].

Recently, researchers have begun bridging these disciplinary silos by, e.g., applying the criteria of envy-freeness and equitability from the fair division literature to gender or racial contexts Hossain *et al.* [2020], or connecting fairness definitions in machine learning to those from moral and political philosophy Binns

[2018]. However, a complete integration of these diverse perspectives has remained elusive, partly due to disciplinary boundaries and methodological divides.

Enter large language models (LLMs)! The advent of highly competent LLMs has been one of the most profound technological disruptions of recent years. These models exhibit social understanding gleaned from their pre-training on vast repositories of human-generated data, ethical considerations learned from academic research and post-training techniques such as reinforcement learning from human feedback (RLHF), and mathematical reasoning abilities. This unique blend of sociotechnical abilities has allowed them to yield breakthrough performance across domains such as healthcare, education, finance, engineering, and programming Hadi *et al.* [2023]. This makes LLMs particularly intriguing for exploring the multifaceted nature of fairness.

In this work, we evaluate fairness of LLM-based decision-making using fair division—specifically, the allocation of indivisible goods—as our example setting, thereby contributing to the quickly-growing literature in AI on conducting LLM evaluations on various dimensions such as safety, trustworthiness, or inclination to hallucinate Guo *et al.* [2023]; Chang *et al.* [2024]; Chu *et al.* [2024]. There are several reasons why LLMs are aptly suited to fair division in particular. They are wildly popular, freely available, and easy to use. They can also understand contextual nuance and adapt the allocations accordingly, in a manner that traditional algorithms may not be able to (see Section 7 for further discussion). So, *if* it turns out they can allocate resources well, they may be adopted quickly in real-world applications. This motivates our in-depth evaluation of how the state-of-the-art (SOTA) LLMs perform fair division. Our main research questions are:

1. *What is fair in the eyes of LLMs?* When LLMs are asked to optimize for "fairness", what do the resulting allocations look like? How do they perform on traditional metrics or in comparison to traditional algorithms?

2. *What influences fairness perception?* How does LLMs' understanding of fairness depend on the context, e.g., the nature of goods or the framing of the preferences?

3. *To what extent can we steer LLMs?* Do the LLMs have the reasoning abilities to be able to optimize user-specified fairness criteria?

1.1 Our Results

We evaluate fairness of three SOTA (state-of-the-art) families of LLMs — `Claude` (by Anthropic), `Gemini` (by Google), and `GPT` (by OpenAI) — using both synthetic data and real data from `Spliddit.org`. Through a carefully designed prompt, we ask the LLMs to allocate a set of goods fairly to a set of agents based on (additive) valuations provided as part of the prompt, and compare their performance to traditional algorithms based on (multiplicative) approximations to popular criteria such as envy-freeness up to one good (EF1), Nash welfare, and utilitarian welfare.

Our main takeaway is that even the SOTA LLMs are not yet ready to perform fair division in the real world autonomously; the SOTA algorithm, maximum Nash welfare Caragiannis *et al.* [2019], outperforms them on all the metrics we study. That said, `Claude` (specifically, `claude-3.5-sonnet-20241022`) shows evidence of allocating goods intelligently, at least providing a trade-off that is not entirely surpassed by a simpler Round Robin algorithm, which is not the case with `Gemini` and `GPT`.

To better understand what goes into the LLM's allocation process, we investigate three variations in prompt design:

- **Context variation.** Whether the task is to allocate objects to people, or heirlooms to siblings after recent death of their parent, or machines to teams in a corporate environment seems to make little difference to how LLMs perform the allocation, at least with a short description of the context.
- **Preference framing.** When agent preferences are provided grouped by goods (with each line specifying all agents' values for a given good), as opposed to grouped by agents (with each line specifying a given agent's values for all the goods), all models become a bit more efficient, with `Claude` and `Gemini` also becoming a bit fairer while `GPT` becoming a bit less fair. The effect size, however, is small.
- **Goal framing.** When LLMs are asked to explicitly seek EF1, maximum Nash welfare (MNW), or maximum utilitarian welfare (MUW), as opposed to simply maximizing "fairness", they slightly alter their behavior. Seeking EF1 generally hurts utilitarian welfare and only improves EF1 approximation for `GPT`, whereas seeking MUW slightly improves the utilitarian welfare for `GPT` and `Claude` while hurting `GPT`'s EF1 approxiation and `Claude`'s EF1 and Nash welfare approximations. Once again, however, the effect sizes are rather small, and letting LLMs optimize their own perception of fairness (as opposed to explicitly steering them to some criterion) comes out as a rather robust choice.

In the appendix, we also present additional experiments where we measure other desiderata (Appendices C and D) and dive deeper into the LLMs' "thought process" using a more controlled environment (Appendix F).

1.2 Related Work

To the best of our knowledge, ours is the first work to explore the use of LLMs in fair division. However, our work is tangentially related to three lines of work.

LLM $\rightarrow$ social choice. Use of LLMs in the adjacent world of voting has been explored recently. When the candidates to be voted on are (policy) statements, LLMs have the remarkable potential of finding consensus candidates that are widely agreeable out of the vast space of possible statements. Bakker *et al.* [2022] design a system in which a fine-tuned set of LLMs generate statements that would be agreeable to large groups of humans and a traditional voting rule picks a single winning statement ("winner selection"), showing that such a system can outperform humans. Fish *et al.* [2024] develop this into *generative social choice*, which can design a representative slate of statements ("committee selection"); they use *generative queries*, which ask LLMs to find statements that would be agreeable to a specified target group of users. Small *et al.* [2023] more broadly discuss the opportunities and risks of LLM usage in deliberative democracy platforms such as `Pol.is`. Our work suggests extending LLM use to social choice more broadly, possibly to other problems such as matching and coalition formation.

Social choice $\rightarrow$ LLM. In the opposite direction, researchers have recently explored applying social choice concepts to the design of LLMs. For example, Zhong *et al.* [2024]; Williams [2024] use the Nash social welfare in the reinforcement learning from human feedback (RLHF) stage of LLM training in order to get LLMs to proportionally represent the preferences of human annotators. This is related to (but a completely different approach to) our MNW prompt, which asks the LLM to maximize Nash welfare as part of the prompt rather than imbuing the principle in its design. Chakraborty *et al.* [2024] similarly use the egalitarian welfare to guide RLHF. It remains to be seen whether other social choice principles, such as envy-freeness or harm ratio Ebadian *et al.* [2024], can be applied to designing LLMs.

n	2	3	4	$\geqslant 5$
Instances	27.6%	67.9%	2.3%	2.2%

Table 1: Distribution of number of agents for `Spliddit.org` instances.

LLM evaluations. There is a quickly-growing literature on conducting evaluations of LLMs to assess their safety, trustworthiness, inclination to hallucinate, reasoning capabilities, etc.; see the comprehensive surveys by Guo *et al.* [2023]; Chang *et al.* [2024]; Chu *et al.* [2024]. A number of studies have specifically focused on evaluating *fairness* of LLMs overall Li *et al.* [2023], or in specific domains such as recommendations Zhang *et al.* [2023] or ranking Wang *et al.* [2024]. To the best of our knowledge, ours is the first work to evaluate fairness of LLMs in the resource allocation domain.

2 Experimental Setup

In this section, we describe the fair division model at the heart of our experiments, the data and LLMs we use, our experimental setup, and our evaluation criteria.

2.1 Fair Division Model

For any $t \in \mathbb{N}$, let $[t] = \{1, 2, \ldots, t\}$. A fair division instance consists of a set of n agents $N = [n]$ and a set of m indivisible goods $M = [m]$. Each agent $i \in N$ has a valuation function $v_i : 2^M \to \mathbb{R}_{\geqslant 0}$, which represents the utility of agent i for each subset of goods. We focus on *additive* valuation functions, meaning $v_i(S) = \sum_{g \in S} v_i(\{g\})$ for all $S \subseteq M$ and $v_i(\emptyset) = 0$. With slight abuse of notation, we write $v_i(g) \coloneqq v_i(\{g\})$ for a single good $g \in M$.

An allocation $A = (A_1, \ldots, A_n)$ is a partition of the set of goods M into n disjoint bundles, where $A_i \subseteq M$ is the bundle allocated to agent i, $A_i \cap A_j = \emptyset$ for all $i, j \in N$ with $i \neq j$, and $\cup_{i \in N} A_i = M$.

2.2 Data

Synthetic data. For our experiments with synthetic data, we sample utilities independently and identically distributed (i.i.d.) as $v_i(g) \sim \text{Uniform}(\{0, \ldots, 10\})$ for each agent $i \in N$ and good $g \in M$. We vary the number of agents $n \in \{2, \ldots, 8\}$ and the number of goods $m \in \{n, 2n, \ldots, 5n\}$,[1] and in each case, show results averaged over 200 random instances along with 95% confidence intervals.

Spliddit data. We utilize real-world goods division Spliddit instances. In these instances, the total utility of each agent for all goods is always 1000. Out of the 5295 instances, we focus on the 4835 instances in which a positive Nash welfare is attainable (see Footnote 2), and show results averaged over these instances. These instances involve between 2 to 15 agents and between 2 to 96 goods; some more statistics are provided in Tables 1 and 2.

2.3 Evaluation Criteria

We use three criteria in our evaluations.

[1]While this parameter range may seem small, so are most fair division instances in practice. For example, 98.2% of `Spliddit.org` instances have $n \leqslant 8$ and $m \leqslant 5n$.

m	2	3	4	5	6	$\geqslant 7$
Instances	3.1%	10.5%	12.9%	4.7%	62.9%	6.0%

Table 2: Distribution of number of goods for `Spliddit.org` instances.

EF1 approximation. The cornerstone notion of fairness in the fair division literature is *envy-freeness* Gamow and Stern [1958]; Foley [1967], which demands that no agent prefer the bundle allocated to another agent over their own bundle. For indivisible goods, this is not always attainable, so its relaxation known as envy-freeness up to one good (EF1) Budish [2011] is more widely used.

For our fairness evaluation, we measure the degree to which an allocation achieves EF1.

Definition 1 (EF1 Approximation)**.** The EF1 approximation of an allocation A is the largest value $\alpha \in [0, 1]$ such that, for all $i, j \in N$ with $A_j \neq \emptyset$, there exists a good $g \in A_j$ such that $v_i(A_i) \geqslant \alpha \cdot v_i(A_j \setminus \{g\})$. An allocation with an EF1 approximation of 1 is simply called an EF1 allocation.

EF1 allocations are guaranteed to exist and we use algorithms known to compute them as baselines (see Section 2.4). In Appendix C, we define and present results for two additional fairness criteria, but note that they do not offer any qualitatively different insights.

Welfare approximation. A desideratum sought commonly in conjunction with fairness is *efficiency*. This is often measured via a *social welfare function* $\mathrm{W} : \mathbb{R}^N_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$, which aggregates individual agent utilities $\vec{u} = (u_1, \ldots, u_n)$ into a single cardinal value. We focus on two widely studied welfare functions:

- The *utilitarian welfare* is defined as the arithmetic mean of agents' utilities: $\mathrm{UW}(\vec{u}) = \frac{1}{n} \sum_{i \in N} u_i$.
- The *Nash welfare* is defined as the geometric mean of agents' utilities: $\mathrm{NW}(\vec{u}) = (\prod_{i \in N} u_i)^{1/n}$.

With slight abuse of notation, we denote the welfare of allocation A under social welfare function W as $\mathrm{W}(A) \coloneqq \mathrm{W}((v_1(A_1), \ldots, v_n(A_n)))$.

For our efficiency evaluation, we measure the degree to which an allocation optimizes these welfare functions.

Definition 2 (Welfare Approximation)**.** Given a social welfare function W, the W approximation of an allocation A is the ratio between the highest welfare of any allocation to that of A, i.e., $\max_B \mathrm{W}(B)/\mathrm{W}(A)$.

2.4 Baseline Algorithms

We evaluate three popular fair division algorithms as baselines in our experiments:

- *Round Robin* (`RR`): An iterative algorithm that provably produces an EF1 allocation. Agents pick goods one by one in a cyclic fashion; specifically, in each round $k \in [m]$, agent $(k-1) \bmod n + 1$ is allocated her most preferred good among the ones remaining.
- *Maximum utilitarian welfare* (`MUW`): Returns an allocation with the highest utilitarian welfare. Under additive valuations, this simply allocates each good to an agent with the highest value for it. However, this does not guarantee any positive EF1 approximation.

- *Maximum Nash welfare* (`MNW`): Returns an allocation that maximizes the Nash welfare. This provably achieves EF1 Caragiannis *et al.* [2019], and is the state-of-the-art algorithm deployed to `Spliddit.org` due to its combination of fairness and efficiency guarantees.[2]

While `MNW` generally supersedes Round Robin, it requires a global optimization of a non-linear function over a combinatorial space, which LLMs cannot be expected to compete with. Hence, Round Robin serves as a more reasonable baseline that can still achieve EF1 with reasonable welfare approximations.

2.5 Large Language Models

We utilize three SOTA families of commercial LLMs: `GPT` from OpenAI, `Claude` from Anthropic, and `Gemini` from Google. To manage time and monetary costs, we first perform all our experiments (see Section 2.6) for the entire parameter range of n and m (see Section 2.2) using the faster and cheaper models: `gpt-4o-mini`, `claude-3-haiku-20240307` (henceforth, `claude-3-haiku`), and `gemini-1.5-flash`. Based on these experiments, we identify $m = 5n$ to be a representative regime,[3] and conduct experiments for $m = 5n$ again using the slower and more expensive, but more powerful models: `gpt-4o`,[4] `claude-3.5-sonnet-20241022` (henceforth, `claude-3.5-sonnet`), and `gemini-1.5-pro`. In the main body, we present the results of the more powerful models and their improvement over the weaker models for the $m = 5n$ case, relegating the rest to the appendix.

In Appendix B, we provide details on the sizes of these experiments in terms of the numbers of input and output tokens, giving a rough estimate on the cost of using LLMs for fair division problems, and showing how these costs scale as the instance size increases.

2.6 Experiments and Prompts

Each datum in our experiments is generated by sending a prompt to an LLM, which fully described the fair division problem at hand, and asking the model to return an allocation. At a high level, all prompts have the same structure involving four components, whose designs we experiment with. We provide a summary below; full details are available in Appendix A.

1) Context. First, the prompt describes the contextual scenario including the nature of agents and goods, which may affect LLMs' perceptions of fairness. We test three contexts:

- *Person/Object* (default): An abstract scenario with "objects" (goods) to be allocated to "people" (agents).
- *Sibling/Heirloom*: A "subjective" inheritance division scenario with "heirlooms" (goods) to be allocated to "siblings" (agents) following the passing of their parent.
- *Team/Machine*: An "objective" corporate scenario with "machines" (goods) to be allocated to "teams" (agents).

2) Goal. Next, the prompt describes the goal we want the LLM to achieve in the allocation it returns.

[2]The rule, as defined by Caragiannis *et al.* [2019], is a bit more subtle for edge-case instances in which all allocations have zero Nash welfare, but we restrict our experiments to instances in which allocations yielding a positive utility to all agents (thus having a positive Nash welfare) exist.

[3]The case of $m = n$ involved several oddities, as described in Appendix E.

[4]We did not include the `o1` and `o1-mini` models due to the excessive cost of reasoning tokens.

- *"Fairness"* (default): Here, we ask the model to allocate the goods "fairly" without providing an explicit definition of fairness.
- *EF1*, *MNW*, and *MUW*: Here, we explicitly instruct the model to find an allocation that satisfies envy-freeness up to one item (EF1), or one that maximizes either the Nash welfare (MNW) or the utilitarian welfare (MUW).

3) Preference framing. Next, we provide agents' valuations in one of two formats:

- *Person/Object* (default): For each agent, we provide a separate line listing their values for the m goods as integers, where the k-th value corresponds to good k:

```
Person 1: [1, 0, ...]    // m values
Person 2: [5, 8, ...]    // m values
```

- *Object/Person*: For each good, we provide a separate line listing the values of all n agents for that good as integers, where the i-th value corresponds to agent i:

```
Object 1: [1, 5, ...]    // n values
Object 2: [0, 8, ...]    // n values
```

4) Output format. We instruct the model to return a JSON object,[5] in which each good is mapped to the index of the agent it is allocated to. We explicitly instruct the model not to include any additional text or reasoning about its allocation decisions.

```
{ Object 1:  3,   // index (from 1 to n)
  Object 2:  2, ... }.
```

In Section 3, we show compare all the models and baselines under the default choices of the first three components. Then, in Sections 4 to 6, we compare the different designs of each of those three components individually, while using the default choice in the other components.

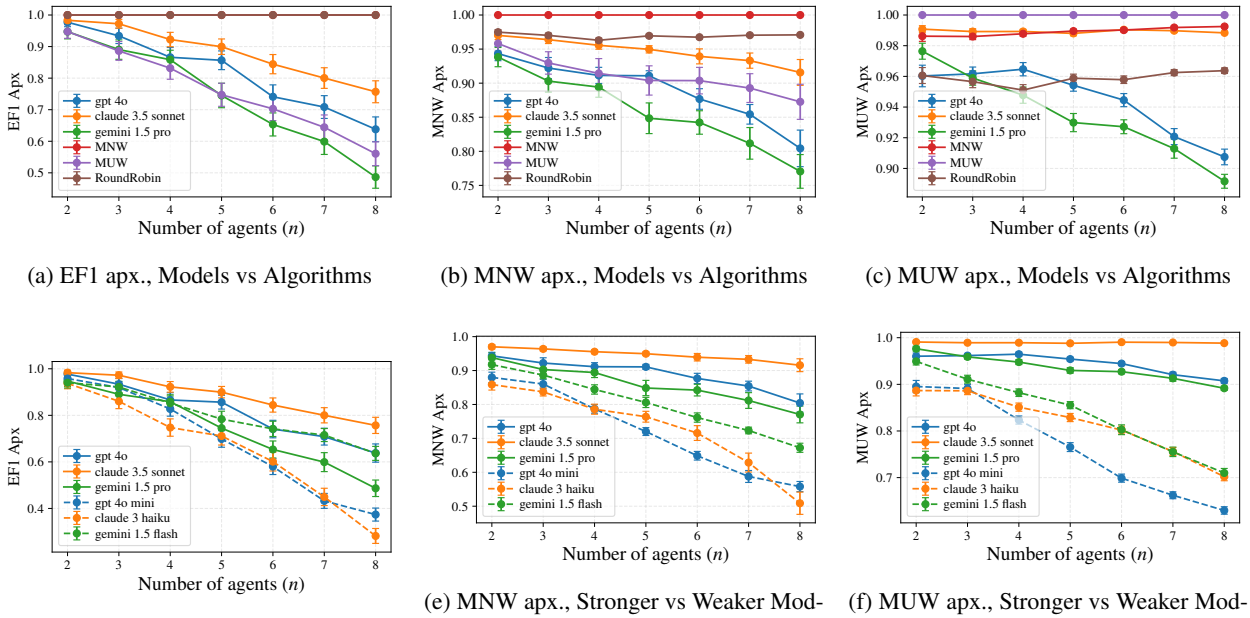
3 LLMs for Fair Division

In this section, we compare all our models and baselines under the default choice of each prompt component.

Results. Figure 1 shows how the different LLMs perform with regard to EF1, MNW, and MUW approximations. In plots (a), (b), and (c), the powerful models are compared to the traditional fair division algorithms (baselines). Out of the three LLMs, `Claude` is clearly the best performer in all three fairness metrics. Additionally, `Claude` is even competitive with the traditional algorithms: in EF1 approximation, it is better than MUW, and in MUW approximation, it is better than Round Robin and comparable to MNW.

The other models, `GPT` and `Gemini`, are less impressive. While `GPT` is competitive with MUW in EF1 approximation, it achieves a much worse Nash and utilitarian welfare when looking at large instances. `Gemini` achieves a similarly poor welfare approximation while performing even worse than MUW in EF1 approximation.

[5]For `GPT` and `Gemini`, we use an in-built feature to restrict their output to the JSON schema. For `Claude` (and one Spliddit instance with 5 agents and 96 goods for which `Gemini` rejected the schema for being too long), we simply requested the models to follow the schema as part of the prompt, which they do very well.



(a) EF1 apx., Models vs Algorithms

(b) MNW apx., Models vs Algorithms

(c) MUW apx., Models vs Algorithms

(d) EF1 apx., Stronger vs Weaker Models

(e) MNW apx., Stronger vs Weaker Models

(f) MUW apx., Stronger vs Weaker Models

Figure 1: Comparison of models using the default prompt with $m = 5n$.

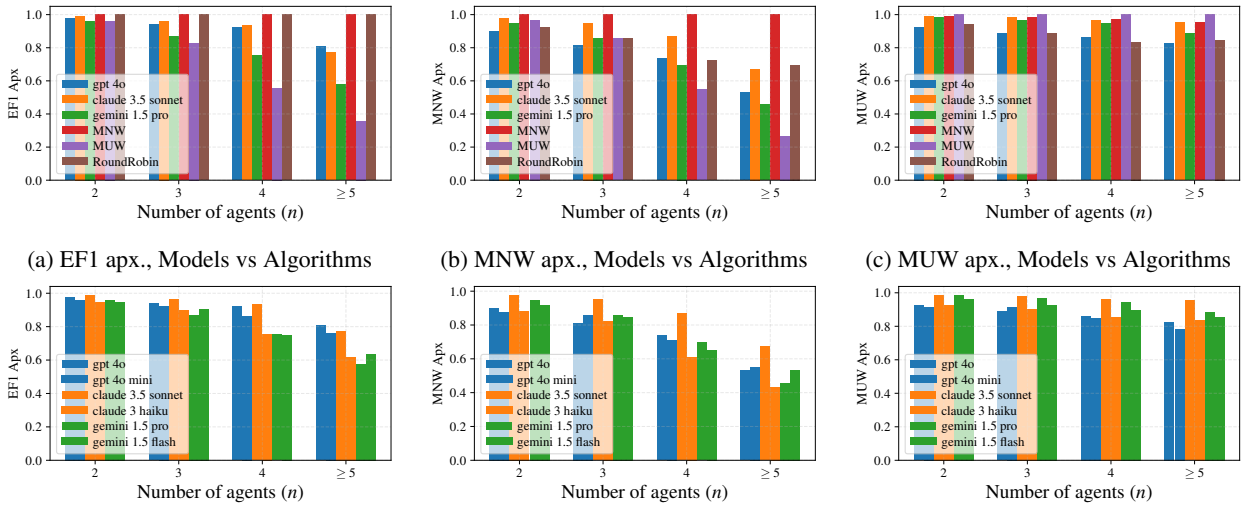
Figure 1 (d)-(f) compare the three stronger models to their weaker counterparts. For the large instances, the stronger models unsurprisingly performed better in all metrics for `Claude` and `GPT`, but for `Gemini`, the stronger model achieves higher welfare at the cost of a lower EF1 approximation than the weaker model.

Figure 2 shows the results for Spliddit instances, where we largely see the same patterns emerging. `Claude` is the dominant model outperforming `GPT` and `Gemini` in all metrics, except for `GPT` being better in EF1 approximation when $n \geqslant 5$, but $n \geqslant 5$ captures only 2.2% of all Spliddit instances.

A stark contrast between synthetic data and real Spliddit data is observed in the performance of the MUW algorithm: it achieves significantly worse approximations of EF1 and MNW on Spliddit data compared to similarly-sized synthetic data. This tension between fairness and efficiency in Spliddit data may explain `Claude`'s performance: it still outperforms MUW on EF1 and MNW approximations, but now this comes at a somewhat more noticeable cost in MUW approximation ($\approx 95\%$ on Spliddit data as opposed to $\approx 99\%$ on synthetic data).

Takeaway. The main takeaway is that LLMs are not yet ready to replace the state-of-the-art Maximum Nash Welfare algorithm; maximizing the Nash welfare still yields the best trade-off between fairness (exact EF1) and efficiency (exact MNW and near-perfect MUW). But the best of the three models, namely `Claude`, is clearly allocating goods intelligently, offering a better fairness-efficiency trade-off than MUW and achieving higher utilitarian welfare than Round Robin at the cost of EF1 and Nash approximations. `GPT` and `Gemini`, on the other hand, perform worse than Round Robin, and similar to or worse than MUW, across all three metrics.

In subsequent experiments, we use these results as a baseline to examine how the three models react to variations in prompt design.



(a) EF1 apx., Models vs Algorithms (b) MNW apx., Models vs Algorithms (c) MUW apx., Models vs Algorithms

(d) EF1, Stronger vs Weaker Models (e) MNW, Stronger vs. Weaker Models (f) MUW, Stronger vs Weaker Models

Figure 2: Comparison of models using the default prompt with the Spliddit instances.

4 Does the Allocation Context Matter?

In this section, we examine whether the context of the allocation—be it abstract objects allocated to people, heirlooms divided among siblings following a parent's death, or machines distributed among corporate teams—affects how LLMs chart the fairness-efficiency tradeoff.

Results. As shown in Figure 3 with EF1, MNW, and MUW approximations for synthetic data, and Figure 4 for Spliddit data, the context makes no noticeable impact on the performance of any of the three models under any of the three metrics. The only exception is that for synthetic data, the Sibling/Heirloom context seems to be slightly improving the EF1 and MNW approximations of `Gemini` and `Claude`.

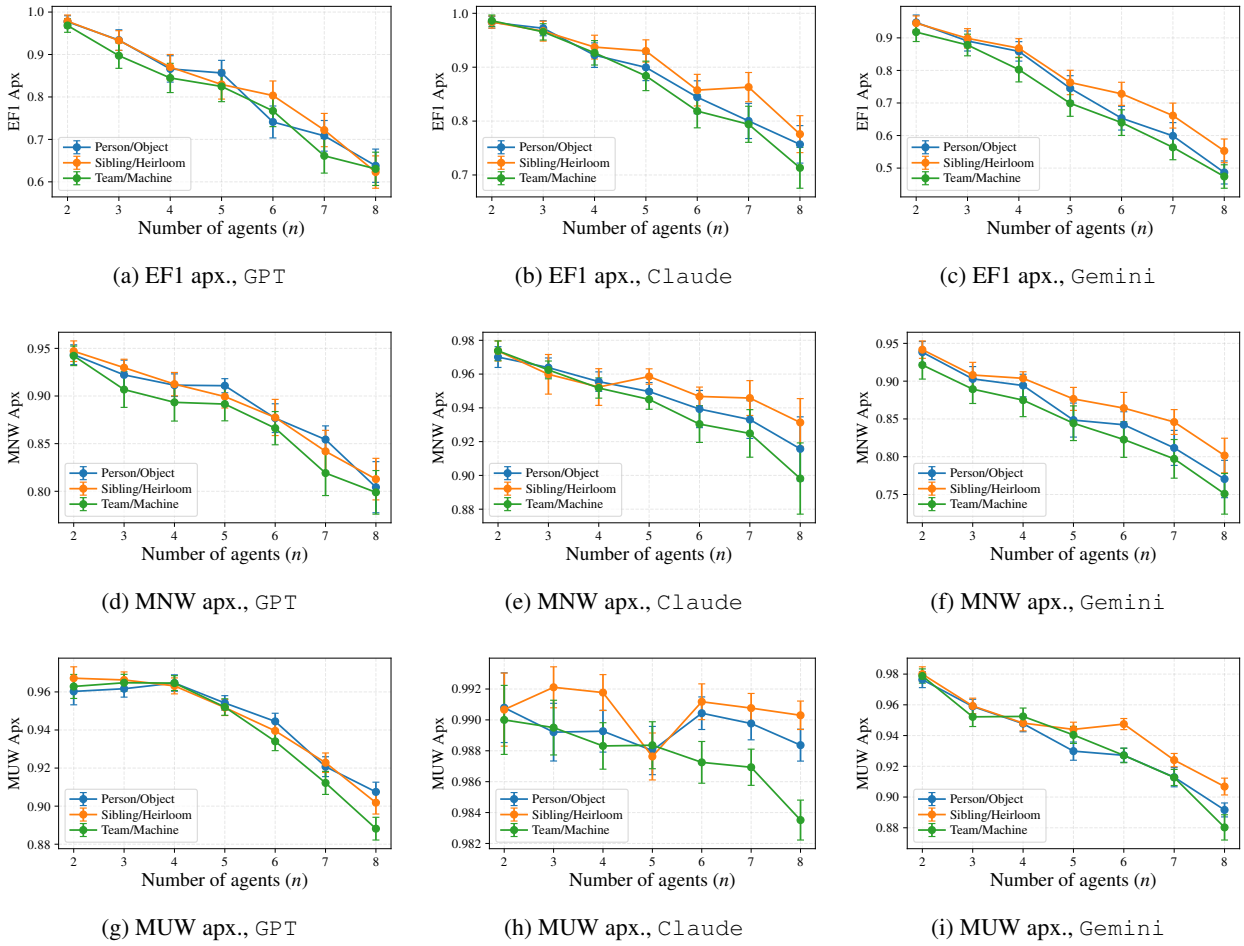
Takeaway. As mentioned in Section 1, LLMs have a unique ability to understand the context of a fair division problem in ways that traditional algorithms cannot. However, our results show that at least a one-line context description along with a relabeling of agents and goods does not sufficiently impact how (the most powerful) LLMs allocate goods or perceive fairness. Whether an in-depth context description influences the models remains to be seen; see Section 7 for a discussion.

5 Does the Preference Framing Matter?

In this section, we test providing the preferences one agent at a time (Person/Object) versus one good at a time (Object/Person). This simply transposes the valuation matrix, which does not affect traditional algorithms' ability to access the values, but it may affect how an LLM interprets the preference data (just as it might affect a human too, at least in larger instances).

Results. For synthetic data (Figure 5), `Claude` and `Gemini` both appear to weakly improve in all three metrics—both slightly in EF1, both more noticeably in MNW, and `Gemini` significantly in MUW (`Claude` is near-perfect in MUW in either case)—when utilities are grouped by object (Object/Person).

`GPT` tells a different story: the Object/Person framing slightly increases MUW approximation, and



(a) EF1 apx., `GPT` (b) EF1 apx., `Claude` (c) EF1 apx., `Gemini`

(d) MNW apx., `GPT` (e) MNW apx., `Claude` (f) MNW apx., `Gemini`

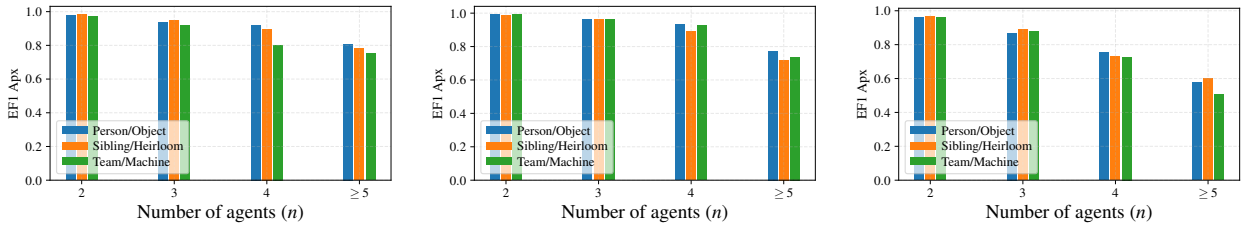
(g) MUW apx., `GPT` (h) MUW apx., `Claude` (i) MUW apx., `Gemini`

Figure 3: Comparison of models based on varying context with $m = 5n$.

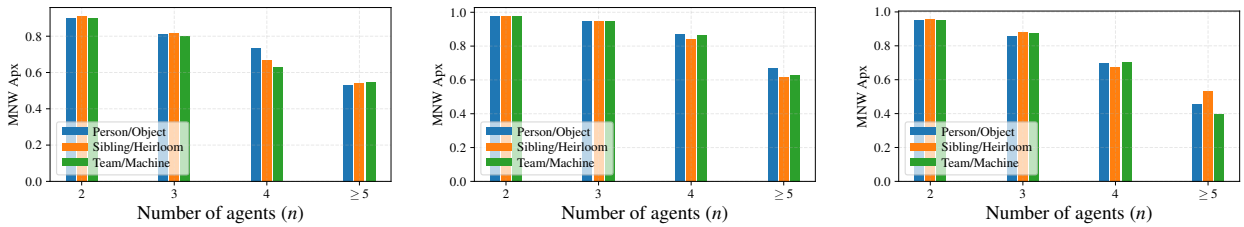
achieves roughly the same MNW approximation, but causes a notable decrease in the EF1 approximation (fairness).

For Spliddit data (Figure 6), the differences are much less pronounced, with the exception that the Person/Object framing slightly improves the EF1 and MNW approximations of `Claude` and `Gemini` on larger instances.

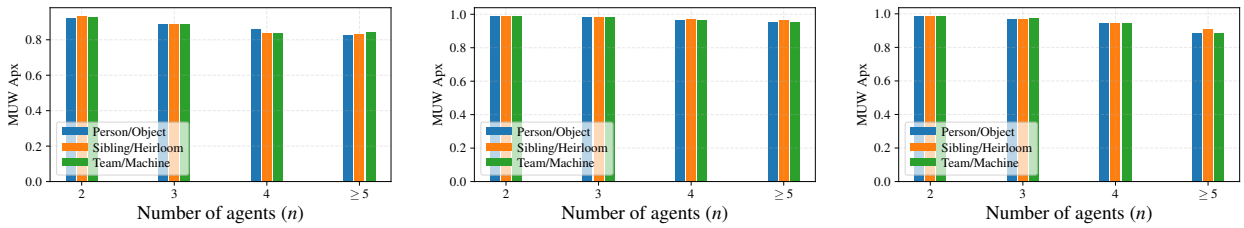
Takeaway. The utilitarian welfare of all models improves when preferences are grouped by object; this makes sense because utilitarian welfare is maximized by allocating each object to the person who values it the most, which is easier to look up under the Object/Person framing. It is somewhat surprising that this framing also improves fairness of `Claude` and `Gemini` in synthetic data, even though it may make the agent-wise consideration required for fairness more difficult, but the effect sizes are very small and not observed on Spliddit data.



(a) EF1 apx., `GPT` (b) EF1 apx., `Claude` (c) EF1 apx., `Gemini`

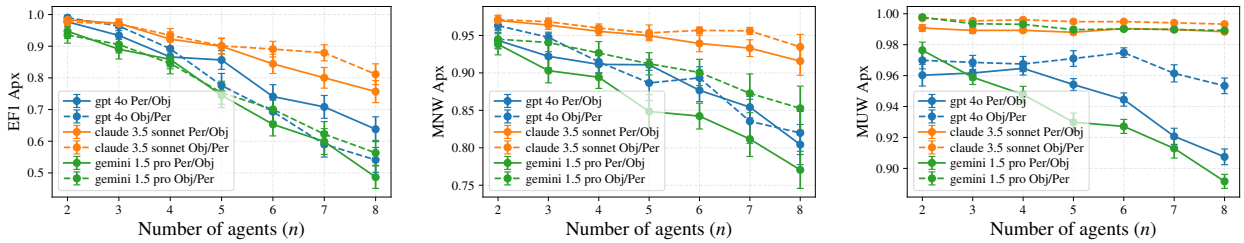


(d) MNW apx., `GPT` (e) MNW apx., `Claude` (f) MNW apx., `Gemini`



(g) MUW apx., `GPT` (h) MUW apx., `Claude` (i) MUW apx., `Gemini`

Figure 4: Comparison of models based on varying context with the Spliddit instances.



(a) EF1 apx. (b) MNW apx. (c) MUW apx.

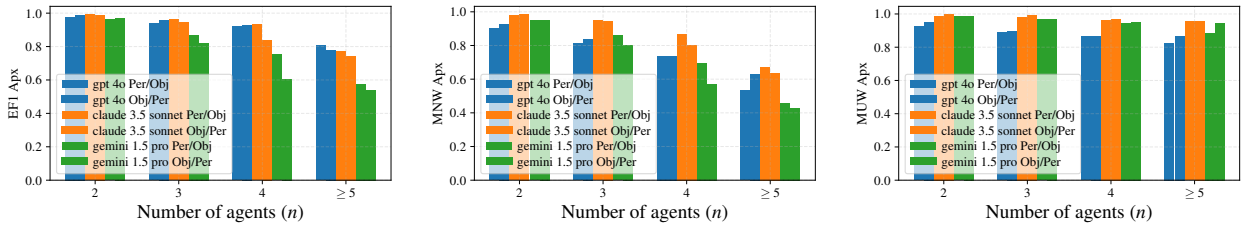
Figure 5: Comparison of models under different input valuation framings with $m = 5n$.

6 Steer LLMs or Let Them Be Free?

In this section, we compare asking the LLMs to achieve specific desiderata (EF1/MNW/MUW) as opposed to simply asking them to allocate "fairly".

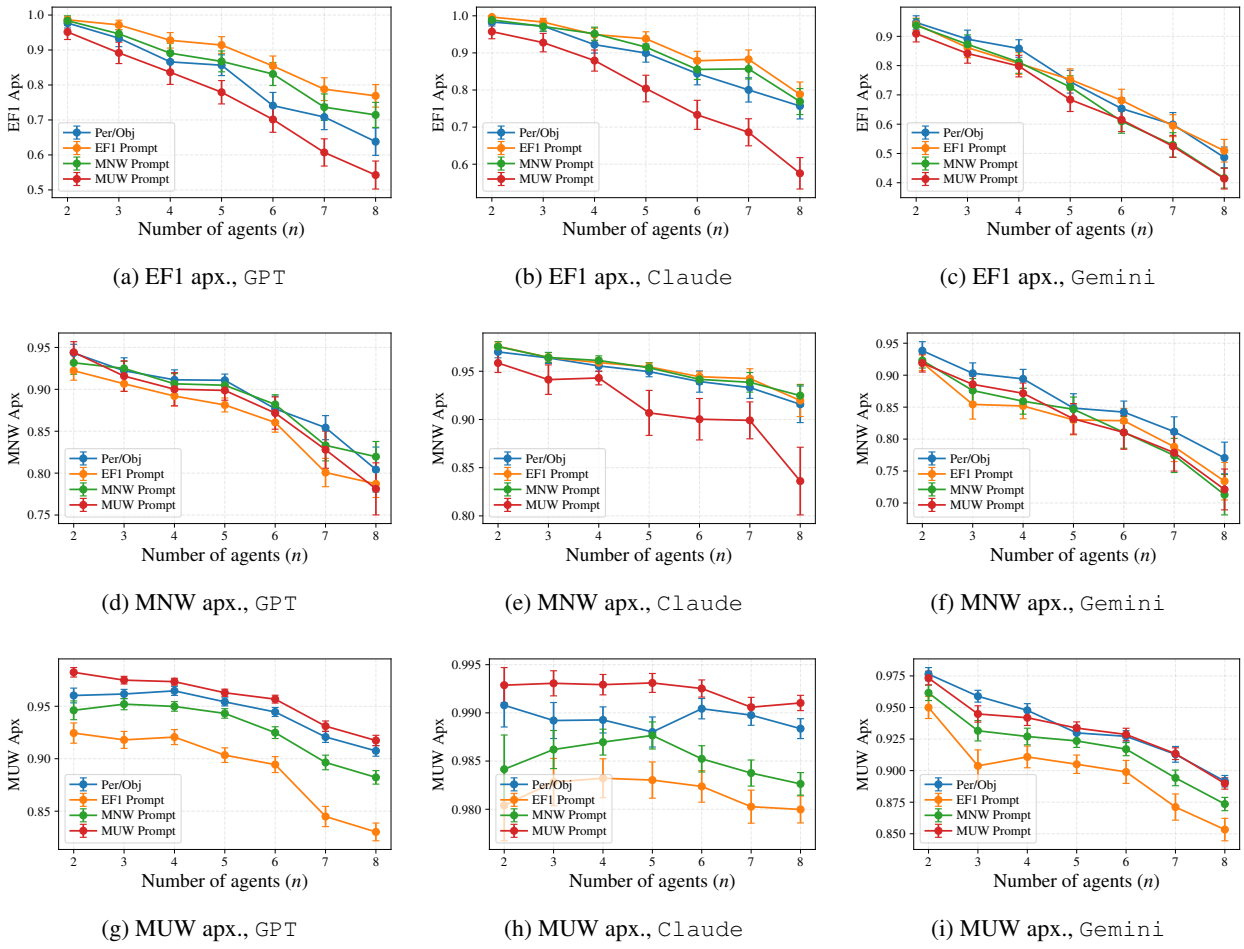
Results. Figure 7 compares the default prompt to the EF1/MNW/MUW-guided prompts separately for each model based on for synthetic data. Figure 8 does the same for Spliddit data.

For `Claude`, the MUW prompt achieves a slightly higher MUW approximation than the default prompt,



(a) EF1 apx. (b) MNW apx. (c) MUW apx.

Figure 6: Comparison of models under different input valuation framings with Spliddit instances.



(a) EF1 apx., `GPT` (b) EF1 apx., `Claude` (c) EF1 apx., `Gemini`

(d) MNW apx., `GPT` (e) MNW apx., `Claude` (f) MNW apx., `Gemini`

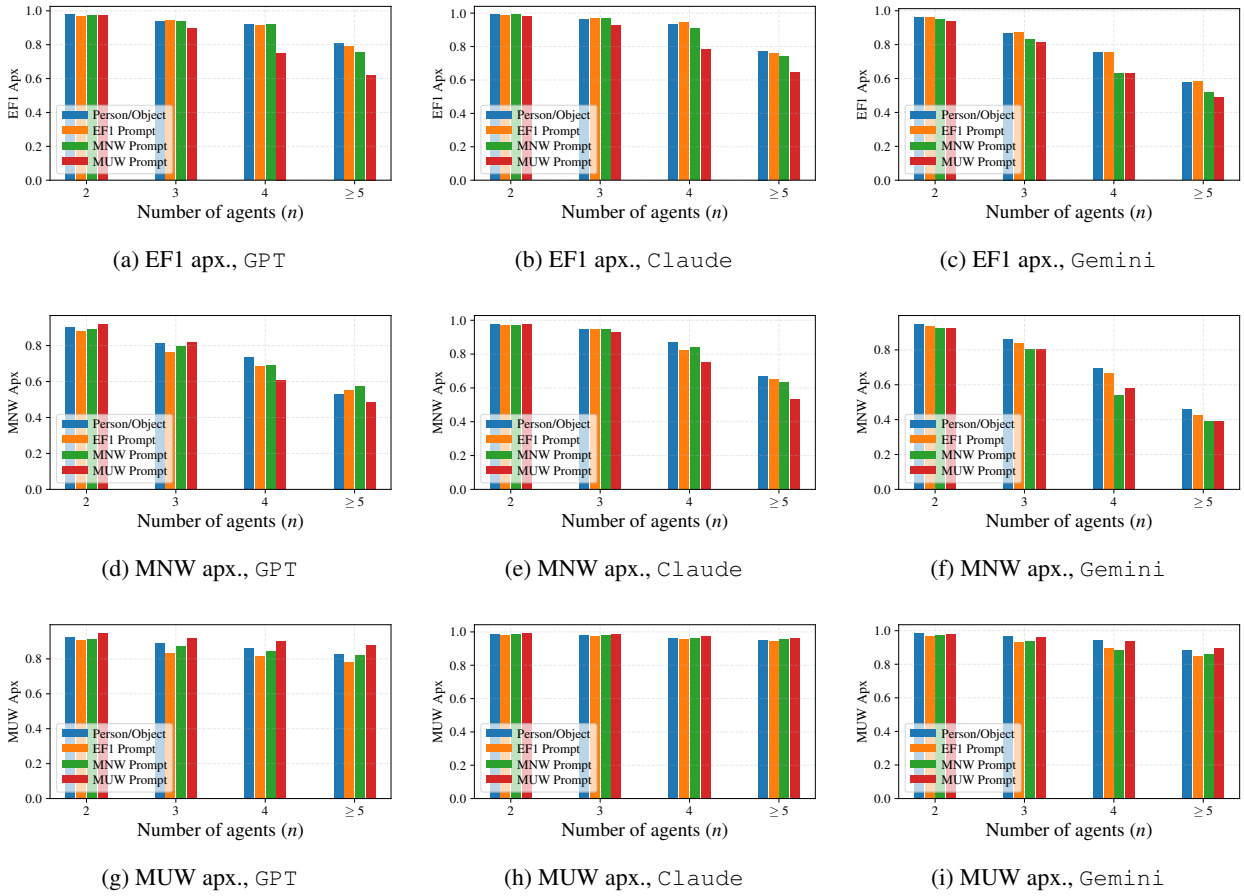
(g) MUW apx., `GPT` (h) MUW apx., `Claude` (i) MUW apx., `Gemini`

Figure 7: Comparison of models based on varying goals with $m = 5n$.

but at a noticeable loss in fairness. Given that the MUW approximation was 99% to begin with, the trade-off offered is not worth it. The EF1 and MNW prompts only degrade the MUW approximation without offering significant (or any) improvement in EF1 or MNW approximations.

`Gemini` concurs: the default prompt is undefeated in each of the three metrics, with the other prompts having either no difference or negative impact on performance up to statistical significance.

`GPT` seems a bit better at following the specified goal. The EF1 prompt leads to a significant improve-



(a) EF1 apx., `GPT` (b) EF1 apx., `Claude` (c) EF1 apx., `Gemini`

(d) MNW apx., `GPT` (e) MNW apx., `Claude` (f) MNW apx., `Gemini`

(g) MUW apx., `GPT` (h) MUW apx., `Claude` (i) MUW apx., `Gemini`

Figure 8: Comparison of models based on varying goals with the Spliddit instances.

ment in EF1 approximation but at a significant cost to welfare approximations. Similarly, the MUW prompt leads to a better MUW approximation (although it was high to begin with) but at a significant cost to the EF1 approximation.

Takeaway. For `Claude` and `Gemini`, the takeaway is simple: the default prompt, which lets the models freely interpret how they wish to "optimize for fairness", is the best. When using `GPT`, one may wish to guide it to explicitly seek EF1, if improved fairness is worth the cost of reduced efficiency, but letting the model freely interpret fairness also provides a compelling trade-off.

7 Discussion

While our work charts a rather large experimental landscape, it represents merely the tip of the iceberg in the exploration of LLM applications in fair division, let alone in the comprehensive evaluation of their fairness. There are many directions in which one can deepen our investigation.

Prompt engineering. While we experimented with different designs of a few components of our base prompt, the possibilities of prompt engineering are vast, ranging from a mere reordering of the components to testing entirely novel task and goal descriptions. This requires an in-depth systematic study.

Task generalization. We limited our attention to the allocation of indivisible goods under additive valuations. Do our observations generalize to other fair division tasks, such as allocation of divisible goods, chore division, allocation under feasibility constraints, or allocating to agents with non-additive valuations? These tasks are notably more difficult, even for traditional algorithms, but that is precisely what may allow LLMs to be more competitive with traditional algorithms.

Better fairness evaluation. Our use of EF1, MNW, and MUW approximations are only proxy criteria; after all, if that is all that we care about, traditional algorithms already offer appealing trade-offs. The true power of LLMs lie in their unique sociotechnical understanding of fairness, so their efficacy must also be evaluated by human subjects (or, perhaps, other LLMs).

Leveraging contextual understanding. In Section 4, we found that a mere one-line description of the context does not significantly alter LLMs' behavior, but this may change if more context is provided. For example, an LLM performing inheritance division may lean towards optimizing fairness if there is a history of rivalry between the siblings, but optimizing efficiency if their relationships are largely harmonious. One can also follow the "generative social choice" style approach Fish *et al.* [2024]; Bakker *et al.* [2022], whereby LLM's contextual understanding is used to shape the problem instance (e.g., by detecting likely substitutes and complements among the goods based on their descriptions or likely cases of human error in providing valuations), but a traditional algorithm is used thereafter to hammer out the allocation, thereby achieving the best of both worlds.

Ethics Statement

Our work investigates existing capabilities of existing models rather than adding capabilities to existing models or designing new models, which somewhat limits the ethical risks involved. That said, we have acknowledged a number of limitations of our fairness evaluation methodology in Section 7, which raises the potential risk that our methodology may be used to "validate" a model in terms of fairness when the model still exhibits significant unfairness in aspects not captured in our work. To that end, we emphasize that our work examines *specific* fairness aspects of how LLMs allocate indivisible goods, and should not be assumed to be a comprehensive evaluation of all possible fairness aspects.

Acknowledgments

This research was partially supported by an NSERC Discovery grant and an NSERC-CSE Research Communities Grant. Researchers funded through the NSERC-CSE Research Communities Grants do not represent the Communications Security Establishment Canada or the Government of Canada. Any research, opinions or positions they produce as part of this initiative do not represent the official views of the Government of Canada.

References

Georgios Amanatidis, Georgios Birmpas, Aris Filos-Ratsikas, and Alexandros A Voudouris. Fair division of indivisible goods: A survey. In *Proceedings of the 31st European Conference on Artificial Intelligence (ECAI)*, pages 5385–5393, 2022.

Michiel A. Bakker, Martin J. Chadwick, Hannah R. Sheahan, Michael Henry Tessler, Lucy Campbell-Gillingham, Jan Balaguer, Nat McAleese, Amelia Glaese, John Aslanides, Matthew M. Botvinick, and Christopher Summerfield. Fine-tuning language models to find agreement among humans with diverse preferences. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 38176–38189, 2022.

Reuben Binns. Fairness in machine learning: Lessons from political philosophy. In *Proceedings of the 2018 Conference on Fairness, Accountability and Transparency*, pages 149–159, 2018.

Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.

Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum Nash welfare. *ACM Transactions on Economics and Computation*, 7(3): Article 12, 2019.

Souradip Chakraborty, Jiahao Qiu, Hui Yuan, Alec Koppel, Dinesh Manocha, Furong Huang, Amrit S. Bedi, and Mengdi Wang. Maxmin-rlhf: Alignment with diverse human preferences. In *Proceedings of the 41sticml*, 2024. Forthcoming.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

Zhibo Chu, Zichong Wang, and Wenbin Zhang. Fairness in large language models: A taxonomic survey. *ACM SIGKDD explorations newsletter*, 26(1):34–48, 2024.

Vincent Conitzer, Rupert Freeman, and Nisarg Shah. Fair public decision making. In *Proceedings of the 18th ACM Conference on Economics and Computation (EC)*, pages 629–646, 2017.

Soroush Ebadian, Rupert Freeman, and Nisarg Shah. Harm ratio: A novel and versatile fairness criterion. In *Proceedings of the 4th ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization*, pages 1–14, 2024.

Sara Fish, Paul Gölz, David C. Parkes, Ariel D. Procaccia, Gili Rusak, Itai Shapira, and Manuel Wüthrich. Generative social choice. In *Proceedings of the 25th ACM Conference on Economics and Computation (EC)*, page 985, 2024.

Duncan Karl Foley. Resource allocation and the public sector. *Yale Economics Essays*, 7:45–98, 1967.

George Gamow and Marvin Stern. *Puzzle-Math*. Viking, 1958.

Nina Grgic-Hlaca, Elissa M Redmiles, Krishna P Gummadi, and Adrian Weller. Human perceptions of fairness in algorithmic decision making: A case study of criminal risk prediction. In *Proceedings of the 2018 International World Wide Web Conference (TheWebConf)*, pages 903–912, 2018.

Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, Deyi Xiong, et al. Evaluating large language models: A comprehensive survey. arXiv:2310.19736, 2023.

Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. Large language models: a comprehensive survey of its applications, challenges, limitations, and future prospects. Authorea Preprints, 2023.

John C Harsanyi. Can the maximin principle serve as a basis for morality? a critique of john rawls's theory. *American political science review*, 69(2):594–606, 1975.

Safwan Hossain, Andjela Mladenovic, and Nisarg Shah. Designing fairly fair classifiers via economic fairness notions. In *Proceedings of the International World Wide Web Conference (TheWebConf)*, pages 1559–1569, 2020.

Yingji Li, Mengnan Du, Rui Song, Xin Wang, and Ying Wang. A survey on fairness in large language models. arXiv:2308.10149, 2023.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021.

John Rawls. *A Theory of Justice*. Harvard University Press, 1971.

Nripsuta Ani Saxena, Karen Huang, Evan DeFilippis, Goran Radanovic, David C Parkes, and Yang Liu. How do fairness definitions fare? Examining public attitudes towards algorithmic definitions of fairness. In *Proceedings of the 2nd AAAI/ACM Conference on Artificial Intelligence, Ethics, and Society (AIES)*, pages 99–106, 2019.

Nisarg Shah. Pushing the limits of fairness in algorithmic decision-making. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 7051–7056, 2023. Early Career Spotlight.

Christopher T. Small, Ivan Vendrov, Esin Durmus, Hadjar Homaei, Elizabeth Barry, Julien Cornebise, Ted Suzman, Deep Ganguli, and Colin Megill. Opportunities and risks of llms for scalable deliberation with polis. arXiv:2306.11932, 2023.

Megha Srivastava, Hoda Heidari, and Andreas Krause. Mathematical notions vs. human perception of fairness: A descriptive approach to fairness for machine learning. In *Proceedings of the 25th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 2459–2468, 2019.

Yuan Wang, Xuyang Wu, Hsin-Tai Wu, Zhiqiang Tao, and Yi Fang. Do large language models rank fairly? an empirical study on the fairness of llms as rankers. arXiv:2404.03192, 2024.

Marcus Williams. Multi-objective reinforcement learning from ai feedback. arXiv:2406.07295, 2024.

Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 993–999, 2023.

Huiying Zhong, Zhun Deng, Weijie J Su, Zhiwei Steven Wu, and Linjun Zhang. Provable multi-party reinforcement learning with diverse human feedback. arXiv:2403.05006, 2024.

Appendix

A Prompts

To reiterate on the discussion of our experiments in Section 2, in total, our experiments involved 14 unique prompts, broken down as follows:

- 1 *Default* prompt: These formed the skeleton of all subsequent prompts, in this prompt, we referred to the agents and goods as "People" and "Objects" respectively. We presented agents' utilities to the LLM grouped by person, and we simply instructed the LLM to find the fairest allocation possible, leaving it up to each model to decide what "fairness" entailed.
- 2 *Context* prompts: In these prompts, we changed the context of the fair division scenario. We changed the names of the agents and goods to "Siblings" and "Heirlooms", and to "Teams" and "Machines" respectively. The preference framing, and fairness instructions remained the same as the default prompt.
- 1 *Framing* prompt: This prompt presented the agents' preferences grouped by object instead of by person. The prompts used the default context of decribing the agents and goods as "People" and "Objects", and simply instructed the LLM to find the fairest allocation possible.
- 3 *Reasoning* prompts: These prompts specifically asked the LLMs to find allocations that satisfied certain fairness criteria. Instead of simply asking the LLM to find the fairest allocation possible, these prompts receptively described EF1, MNW, and MUW, and asked the LLM to ensure fairness by satisfying that criteria. These prompts used the default context of "People" and "Objects", and used the default preference framing where agents' preferences were presented to the LLM grouped by person.
- For each of the above 7 prompts, 2 copies of that prompt were needed for both the synthetic and the `Spliddit.org` data. This is due to the fact that the synthetic data was drawn from an i.i.d. distribution with each utility value ranging from 0 to 10. While the `Spliddit.org` data did not have a strict upper-bound on each individual utility value, but rather had an upper-bound of 1000 for the sum of a single agent's utilities. For each synthetic data prompt, the corresponding `Spliddit.org` prompt was identical except for the fact that the preference description was described to the LLM to match this change.

All the prompts that were used can be broken down into the following sections:

```
<Opening Paragraph>: Explains the context of the problem, lists the
 ↪  number of agents and goods involved, and the structure in which
 ↪  utilities are assigned to the goods.

<Introducing Utilities>: Explains the framing of the utilities, then
 ↪  lists the utilities of each agent according to the framing
 ↪  technique being used.
```

```
<Fairness Explanation>: If the LLM is being instructed to follow a
 ↪  specific fairness definition, it will be explained here. Otherwise,
 ↪  the LLM will simply be instructed to find the fairest allocation
 ↪  possible.

<JSON Formatting Instructions>: Instructs the LLM how to format their
 ↪  response, and provides a JSON template to follow.

<Closing Statement>: Reiterates the goals of the prompt (either finding
 ↪  a specific fairness criteria, or finding the fairest allocation
 ↪  possible).
```

A.1 Opening Paragraph

For all synthetic experiments that use the default context, describing the agents as "People" and the goods as "Objects", the opening paragraph is as follows:

```
Your task is to fairly allocate {m} objects between {n} people.
 ↪  Each person was asked to assign each object a score between 0
 ↪  and 10 that represents their subjective value for that object,
 ↪  with a higher score representing a greater desire to receive
 ↪  that object.
```

For prompts with different context, the opening paragraph is changed to reflect the different storyline that the additional context is portraying

For the Sibling/Heirloom context, the opening paragraph is as follows:

```
Your task is to fairly allocate {m} family heirlooms between {n}
 ↪  siblings after the recent death of their father. Each sibling
 ↪  was asked to assign each heirloom a score between 0 and 10 that
 ↪  represents their subjective value for that heirloom, with a
 ↪  higher score representing a greater desire to receive that
 ↪  heirloom.
```

For the Team/Machine context, the opening paragraph is as follows:

```
Your task is to fairly allocate {m} machines between {n} teams in
 ↪  an engineering firm. Each team was asked to assign each machine
 ↪  a score between 0 and 10 that represents how helpful that
 ↪  machine would be to them in their day-to-day operations, with a
 ↪  higher score representing a greater value for that machine.
```

For all prompts run against `Spliddit.org` data, the opening paragraph was changed slightly to reflect the difference in utilities for that dataset.

```
Your task is to fairly allocate {m} objects between {n} people.
 ↪  Each person was asked to assign each object a score that
 ↪  represents their subjective value for that object, with a
 ↪  higher score representing a greater desire to receive that
 ↪  object. For each person, the sum of all the scores they
 ↪  assigned will equal 1000.
```

A.2 Introducing Utilities

For all prompts that used the default style of preferences framing, where a list of utilities is provided for each person, the *Introducing Utilities* paragraph is described as follows:

```
The scores that each person assigned to the objects are provided
 ↪ below in the following format: Each person is labeled using
 ↪ indices from 1 to {n} ("Person 1", "Person 2", etc.). For each
 ↪ person, there is an associated list of length {m}. The nth
 ↪ entry in this list will correspond to the score that person
 ↪ assigned to the nth object.

-----SCORES-----
Person 1: [1, 0, ...] // m values
Person 2: [2, 5, ...] // m values
...
Person {n}: [4, 9, ...] // m values
-----END OF SCORES-----
```

For each prompt that uses different context, the names of "person" and "object" were changed to reflect this context (to either "sibling" and "heirloom", or to "team", and "machine").

For the prompts that use the alternate style of preferences framing, where a list of utilities is provided for each object, the *Introducing Utilities* paragraph is described as follows:

```
The scores that each person assigned to the objects are provided
 ↪ below in the following format: Each person is labeled using
 ↪ indices from 1 to {n} ("Person 1", "Person 2", etc.). For each
 ↪ person, there is an associated list of length {m}. The nth
 ↪ entry in this list will correspond to the score that person
 ↪ assigned to the nth object.

-----SCORES-----
Object 1: [1, 0, ...] // n values
Object 2: [2, 5, ...] // n values
...
Object {m}: [4, 9, ...] // n values
-----END OF SCORES-----
```

A.3 Fairness Explanation

For all prompts that do not ask for a specific definition of fairness, the *Fairness Explanation* paragraph simply tells the LLM to find the fairest allocation possible:

```
Using the people's scores, you should allocate the objects to the
 ↪ people in the fairest way possible.
```

Again, in the prompts with different contexts, "person/people" and "object" were changed to reflect this context.

For the prompts that specifically instruct the LLM to find an allocation that is EF1/MNW/MUW, the *Fairness Explanation* paragraph provides a formal description of that fairness property.

For EF1, the paragraph is as follows:

```
You should make the allocation fair by ensuring that it meets the
↪ fairness criteria of "Envy-Freeness Up to 1 Good (EF1)". An
↪ allocation is EF1 if no person would rather have another
↪ person's bundle of objects over their own bundle after removing
↪ some object from that other person's bundle.

Formally, for any set $S$ of the objects, and any $i \in
↪ \{1,\dots,{n}\}$, we say that $v_i(S)$ is person $i$'s score
↪ for that set, derived by summing person $i$'s score for each
↪ object in $S$. For each person $i$, let $A_i$ be the set of
↪ objects assigned to person $i$ in an allocation $A$. An
↪ allocation $A$ is EF1 if for every person $i$ and person $j$
↪ with $A_j \neq \emptyset$, there exists an object $o \in A_j$
↪ such that $v_i(A_i) \geq v_i(A_j \setminus \{o\})$.
```

For MNW, the paragraph is as follows:

```
You should make the allocation fair by ensuring that it is an
↪ allocation with Maximum Nash Welfare. An allocation has Maximum
↪ Nash Welfare if it maximizes the product of all the people's
↪ scores for their bundles of objects.

Formally, for any set $S$ of the objects, and any $i \in
↪ \{1,\dots,{n}\}$, we say that $v_i(S)$ is person $i$'s score
↪ for that set, derived by summing person $i$'s score for each
↪ object in $S$. For each person $i$, let $A_i$ be the set of
↪ objects assigned to person $i$ in an allocation $A$. The Nash
↪ Welfare of an allocation $A$ is the value $\prod_{i \in
↪ \{1,\dots,{n}\}}{v_i(A_i)}$, i.e., it is derived by multiplying
↪ together $v_i(A_i)$ for all people $i$. An allocation $A$
↪ maximizes Nash Welfare if its Nash Welfare is the largest
↪ possible among all ways to allocate the objects.
```

For MUW, the paragraph is as follows:

```
You should make the allocation fair by ensuring that it is an
↪ allocation with Maximum Utilitarian Welfare. An allocation has
↪ Maximum Utilitarian Welfare if it maximizes the sum of all the
↪ people's scores for their assigned sets of objects.
```

```
Formally, for any set $S$ of the objects, and any $i \in
 ↪  \{1,\dots,{n}\}$, we say that $v_i(S)$ is person $i$'s score
 ↪  for that set, derived by summing person $i$'s score for each
 ↪  object in $S$. For each person $i$, let $A_i$ be the set of
 ↪  objects assigned to person $i$ in an allocation $A$. The
 ↪  Utilitarian Welfare of an allocation $A$ is the value $\sum_{i
 ↪  \in \{1,\dots,{n}\}}{v_i(A_i)}$, i.e., it is derived by summing
 ↪  $v_i(A_i)$ for all people $i$. An allocation $A$ maximizes
 ↪  Utilitarian Welfare if its Utilitarian Welfare is the largest
 ↪  possible among all ways to allocate the objects.
```

A.4 JSON Formatting Instructions

The paragraph that instructs the LLMs how to format their responses varies slightly based on which preference framing was used in the prompt. For prompts that provide preferences to the LLM grouped by people, the *JSON Formatting Instructions* paragraph is as follows:

```
Included below is a json template indicating how your response
 ↪  should be formatted. Please format your response EXACTLY
 ↪  according to the following json template. DO NOT respond with
 ↪  any additional text or reasoning about your decision. The json
 ↪  template requires that for each object, a single person be
 ↪  specified to receive that object. The person should be
 ↪  specified using their index ranging from 1 to {n}.

-----JSON TEMPLATE-----
{"Object 1": "index (from 1 to {n})", "Object 2": "index (from 1 to
 ↪  {n})", ..., "Object {m}":"index (from 1 to {n})"}
-----END OF JSON TEMPLATE-----
```

Again, in the prompts with different contexts, the words "person" and "object" were changed to reflect this context.

For the prompts that provide preferences grouped by objects, the paragraph is slightly changed to better explain to the LLM which index corresponds to which person:

```
Using the people's scores, you should allocate the objects to the
 ↪  people in the fairest way possible. Included below is a json
 ↪  template indicating how your response should be formatted.
 ↪  Please format your response EXACTLY according to the following
 ↪  json template. DO NOT respond with any additional text or
 ↪  reasoning about your decision. The json template requires that
 ↪  for each object, a single person be specified to receive that
 ↪  object. The person should be specified using their index
 ↪  ranging from 1 to {n}, corresponding to their position in the
 ↪  above scores lists.

-----JSON TEMPLATE-----
```

```
{"Object 1": "index (from 1 to {n})", "Object 2": "index (from 1 to
↪  {n})", ..., "Object {m}":"index (from 1 to {n})"}
-----END OF JSON TEMPLATE-----
```

A.5 Closing Statement

For the closing statement, all prompts that do not ask the LLM to find a specific fairness criteria simply state the following:

```
Remember, your goal is to allocate these objects in the fairest way
↪  possible.
```

Again, in the prompts with different contexts, the words "person" and "object" were changed to reflect this context.

For the prompts that specify certain fairness criteria, the prompt reminds the LLM that fairness means finding that criteria.

For the EF1 prompt:

```
Remember, your goal is to make the allocation that you respond with
↪  fair by ensuring that it is EF1.
```

For the MNW prompt:

```
Remember, your goal is to make the allocation that you respond with
↪  fair by ensuring that it maximizes Nash Welfare.
```

For the MUW prompt:

```
Remember, your goal is to make the allocation that you respond with
↪  fair by ensuring that it maximizes Utilitarian Welfare.
```

A.6 A Complete Example of the Base Prompt

The following is the complete base prompt (using the default choice for each component) for the synthetic experiments, formatted to run on an example instance with 3 agents and 6 goods:

```
Your task is to fairly allocate 6 objects between 3 people. Each
 ↪  person was asked to assign each object a score between 0 and 10
 ↪  that represents their subjective value for that object, with a
 ↪  higher score representing a greater desire to receive that
 ↪  object.

The scores that each person assigned to the objects are provided
 ↪  below in the following format: Each person is labeled using
 ↪  indices from 1 to 3 ("Person 1", "Person 2", etc.). For each
 ↪  person, there is an associated list of length 6. The nth entry
 ↪  in this list will correspond to the score that person assigned
 ↪  to the nth object.

-----SCORES-----
Person 1: [1, 5, 7, 3, 4, 0]
Person 2: [5, 9, 1, 6, 3, 3]
Person 3: [8, 0, 2, 1, 5, 4]
-----END OF SCORES-----

Using the people's scores, you should allocate the objects to the
 ↪  people in the fairest way possible. Included below is a json
 ↪  template indicating how your response should be formatted.
 ↪  Please format your response EXACTLY according to the following
 ↪  json template. DO NOT respond with any additional text or
 ↪  reasoning about your decision. The json template requires that
 ↪  for each object, a single person be specified to receive that
 ↪  object. The person should be specified using their index
 ↪  ranging from 1 to 3.

-----JSON TEMPLATE-----
{"Object 1": "index (from 1 to 3)", "Object 2": "index (from 1 to
 ↪  3)", "Object 3": "index (from 1 to 3)", "Object 4": "index
 ↪  (from 1 to 3)", "Object 5": "index (from 1 to 3)", "Object 6":
 ↪  "index (from 1 to 3)"}
-----END OF JSON TEMPLATE-----

Remember, your goal is to allocate these objects in the fairest way
 ↪  possible.
```

B Technical Experiment Details

In Tables 3 and 4, we highlight the number of input and output tokens required for each model to run the the default prompt experiments against the synthetic data. In Tables 5 and 6, we show the tokens required for running the default prompt experiments against the `Spliddit.org` data. The other experiments (Context, Framing, and Reasoning prompts) took roughly the same order of tokens.

n	2	3	4	5	6	7	8
`GPT`	120400	160200	206000	257800	315600	379400	449200
`Gemini`	100375	132821	171450	216271	267261	324443	387870
`Claude`	105600	137600	175600	219600	269600	325600	387600

Table 3: Number of input tokens required to run 200 tests for n agent, $5n$ goods synthetic instances

n	2	3	4	5	6	7	8
`GPT`	12414	18510	24416	30418	36422	42498	48467
`Gemini`	16195	25186	34203	43328	52151	61592	70672
`Claude`	16718	24618	32686	40812	48726	56819	66841

Table 4: Number of output tokens required to run 200 tests for n agent, $5n$ goods synthetic instances

n	2	3	4	$\geqslant 5$
`GPT`	601695	1676071	71220	83306
`Gemini`	547436	1541500	64450	78832
`Claude`	558487	1539268	64084	74976

Table 5: Number of input tokens required to run tests for one round of tests on the `Spliddit.org` instances

n	2	3	4	$\geqslant 5$
`GPT`	39079	121541	6073	7106
`Gemini`	48828	153333	8004	9558
`Claude`	52583	163051	8144	9486

Table 6: Number of output tokens required to run tests for one round of tests on the `Spliddit.org` instances

C Additional Criteria

In this section, we discuss two fairness criteria alternative to the EF1 approximation we used throughout the paper; in Appendix D.1, we show that they produce qualitatively the same results as EF1 approximation.

Definition 3 (EF Approximation)**.** The envy-freeness (EF) approximation of an allocation A is the largest value $\alpha \in [0, 1]$ for which $v_i(A_i) \geqslant \alpha \cdot v_i(A_j)$ for all $i, j \in N$. An allocation with an EF approximation of 1 is simply called EF.

The indivisibility of goods often makes envy-free allocations impossible; for example, when dividing a single good between two agents, only 0-EF allocations exist. This is why EF1 and its approximations are more commonly used with indivisible goods.

A weaker fairness notion than EF is *proportionality* (Prop), which demands that each agent receive utility that is at least $1/n$ of their total value for all the goods combined. We also use the approximation of its attainable relaxation, Prop1 Conitzer *et al.* [2017], as a fairness metric in our evaluation.

Definition 4 (Prop1 Approximation)**.** The proportionality up to one good (Prop1) approximation of an allocation A is the largest value $\alpha \in [0, 1]$ such that, for all $i \in N$ with $A_i \neq M$, there exists a good $g \in M \setminus A_i$ such that

$$v_i(A_i \cup \{g\}) \geqslant \alpha \cdot \frac{1}{n} v_i(M).$$

An allocation with a Prop1 approximation of 1 is simply called Prop1.

For any allocation, it is easy to see that its EF approximation is at most as large as its EF1 approximation, which is at most as large as its Prop1 approximation. The former relation is trivial, and a short proof of the latter is provided below.

Proposition 1. *The EF1 approximation of any allocation is at most as large as its Prop1 approximation.*

Proof. Consider any allocation A and let α be its EF1 approximation. Then, by the definition of EF1 approximation, we have that for all $i, j \in N$ with $A_j \neq \emptyset$, there exists a good $g_j \in A_j$ such that

$$v_i(A_i) \geqslant \alpha \cdot v_i(A_j \setminus \{g_j\}). \tag{1}$$

Let $g^* \in \arg\max_{j \in N : A_j \neq \emptyset} v_i(g_j)$ be agent i's most valuable good among all such g_j's. Then, for all $j \in N$,

$$v_i(A_i \cup \{g^*\}) \geqslant v_i(A_i) + \alpha \cdot v_i(g^*) \geqslant \alpha \cdot v_i(A_j),$$

where the last inequality holds trivially if $A_j = \emptyset$ and from Equation (1) otherwise.

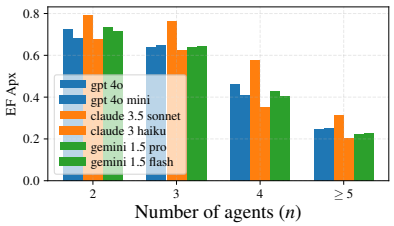
Averaging over all $j \in N$, we get

$$v_i(A_i \cup \{g^*\}) \geqslant \frac{\alpha}{n} \cdot v_i(M),$$

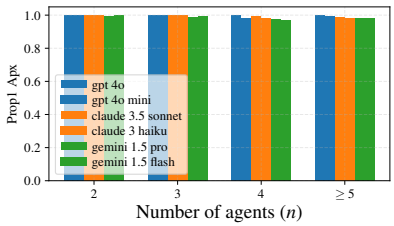
which means the Prop1 approximation of A is at least as large as α, as desired. □

D Additional Plots for Stronger Models

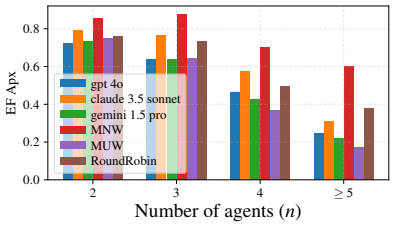
D.1 EF and Prop1 Approximations



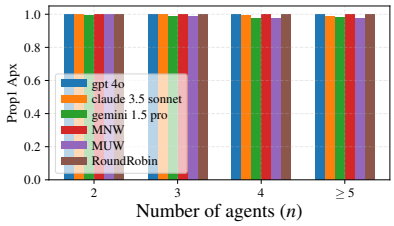
(a) EF, Stronger vs Weaker Models



(b) Prop1, Stronger vs Weaker Models

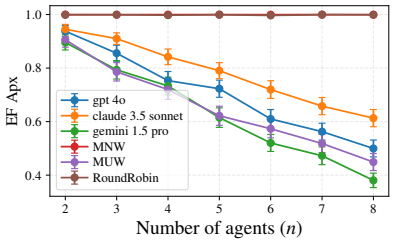


(c) EF, Models vs Algorithms

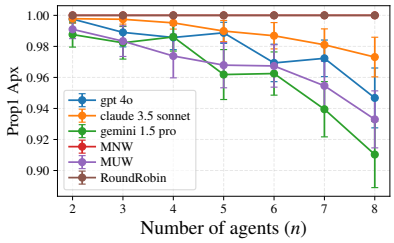


(d) Prop1, Models vs Algorithms

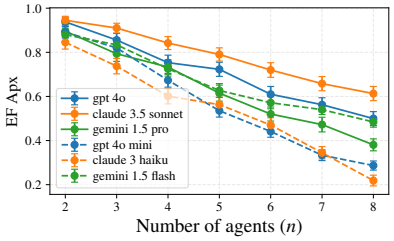
Figure 9: Comparison of stronger and weaker models using the default prompt with the Spliddit instances.



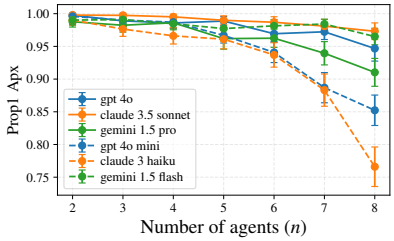
(a) EF apx., Models vs Algorithms



(b) Prop1 apx., Models vs Algorithms



(c) EF apx., Stronger vs Weaker Models



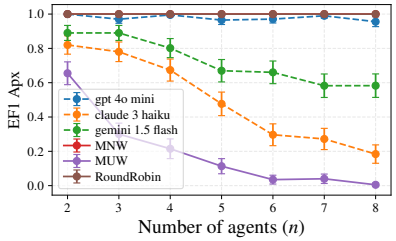
(d) Prop1 apx., Stronger vs Weaker Models

Figure 10: Comparison of models using the default prompt with $m = 5n$.

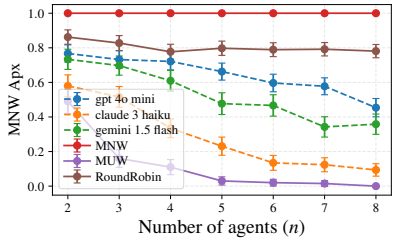
E Additional Plots for Smaller Models

In this section, we present a series of results for the smaller versions of each LLM model that our main experiments were conducted on. Namely, `gpt-4o-mini`, `claude-3-haiku-20240307`, and `gemini-1.5-flash`. We ran the smaller model tests for instances of size $n \in \{2, \ldots, 8\}$ and $m \in \{n, 2n, 3n, 4n, 5n\}$. We present the most interesting subset of those experiments here.

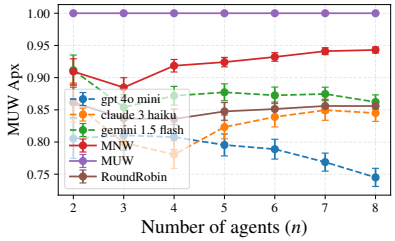
Weaker Models vs Algorithms, Default Prompts



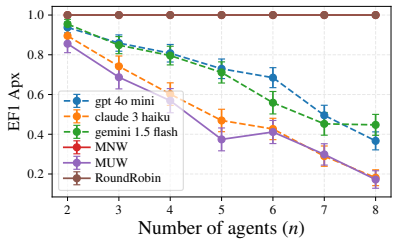
(a) EF1, Models vs Algs., $m = n$



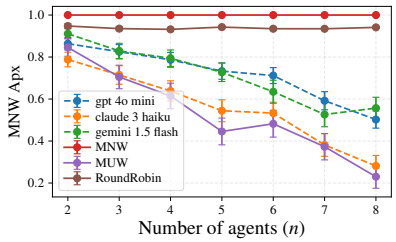
(b) MNW, Models vs Algs., $m = n$



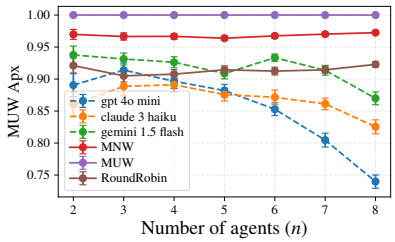
(c) MUW, Models vs Algs., $m = n$



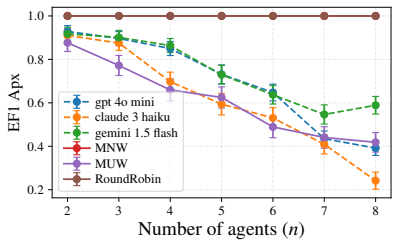
(d) EF1, Models vs Algs., $m = 2n$



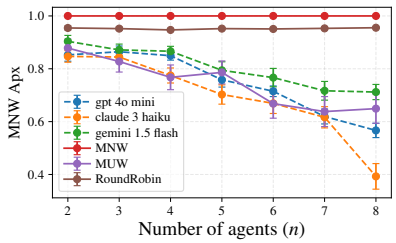
(e) MNW, Models vs Algs., $m = 2n$



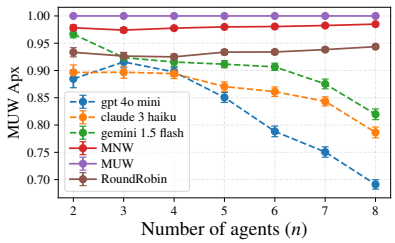
(f) MUW, Models vs Algs., $m = 2n$



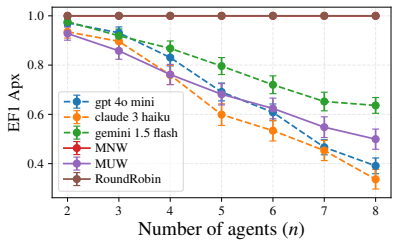
(g) EF1, Models vs Algs., $m = 3n$



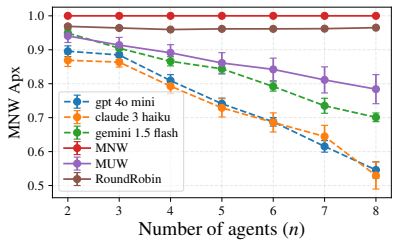
(h) MNW, Models vs Algs., $m = 3n$



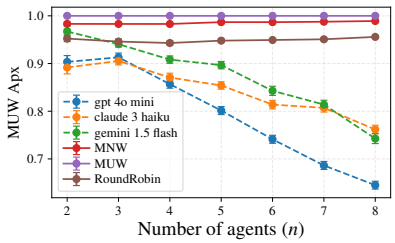
(i) MUW, Models vs Algs., $m = 3n$



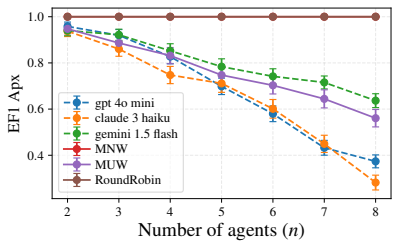
(j) EF1, Models vs Algs., $m = 4n$



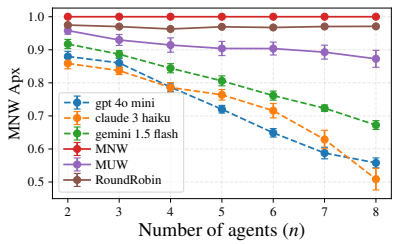
(k) MNW, Models vs Algs., $m = 4n$



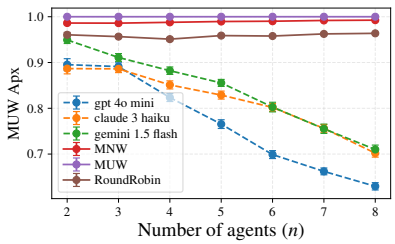
(l) MUW, Models vs Algs., $m = 4n$



(m) EF1, Models vs Algs., $m = 5n$



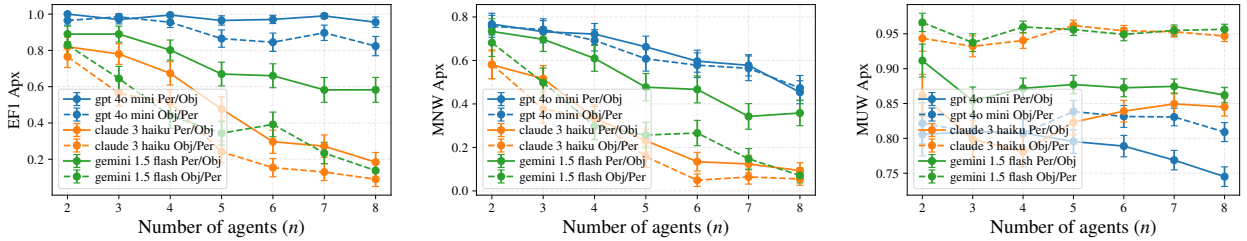
(n) MNW, Models vs Algs., $m = 5n$



(o) MUW, Models vs Algs., $m = 5n$

Figure 11: Comparison of weaker models with algorithms using the default prompt with $m \in \{n, 2n, \ldots, 5n\}$.

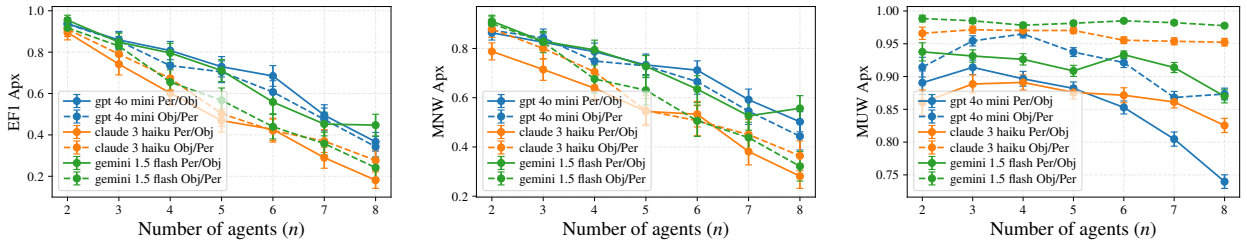
Weaker Models, Varying Input Valuation Framing



(a) EF1, $m = n$

(b) MNW, $m = n$

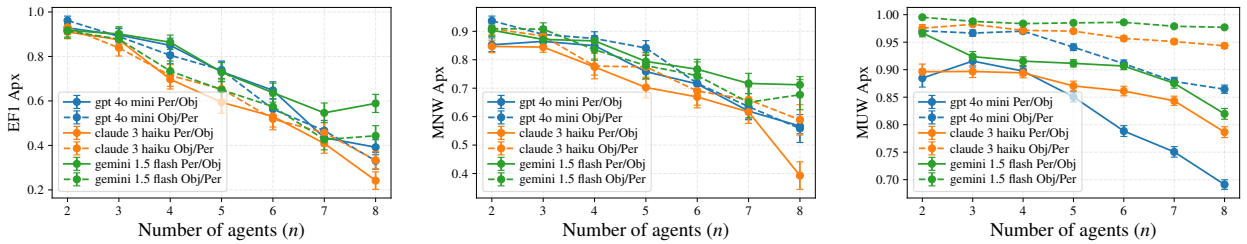
(c) MUW, $m = n$



(d) EF1, $m = 2n$

(e) MNW, $m = 2n$

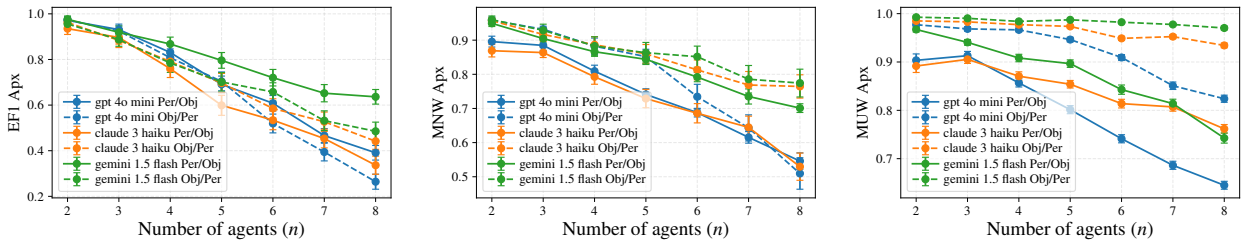
(f) MUW, $m = 2n$



(g) EF1, $m = 3n$

(h) MNW, $m = 3n$

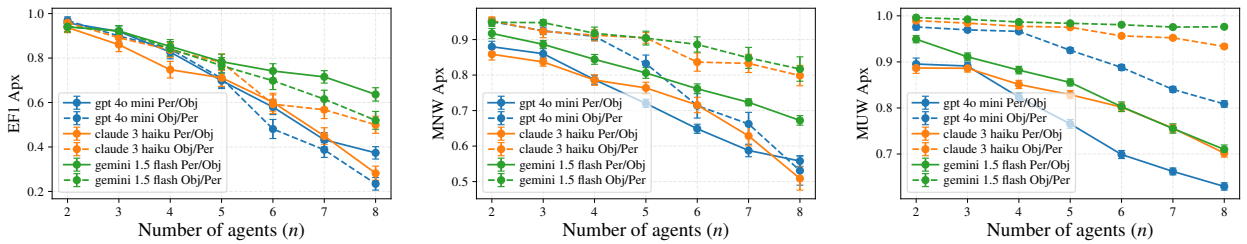
(i) MUW, $m = 3n$



(j) EF1, $m = 4n$

(k) MNW, $m = 4n$

(l) MUW, $m = 4n$



(m) EF1, $m = 5n$

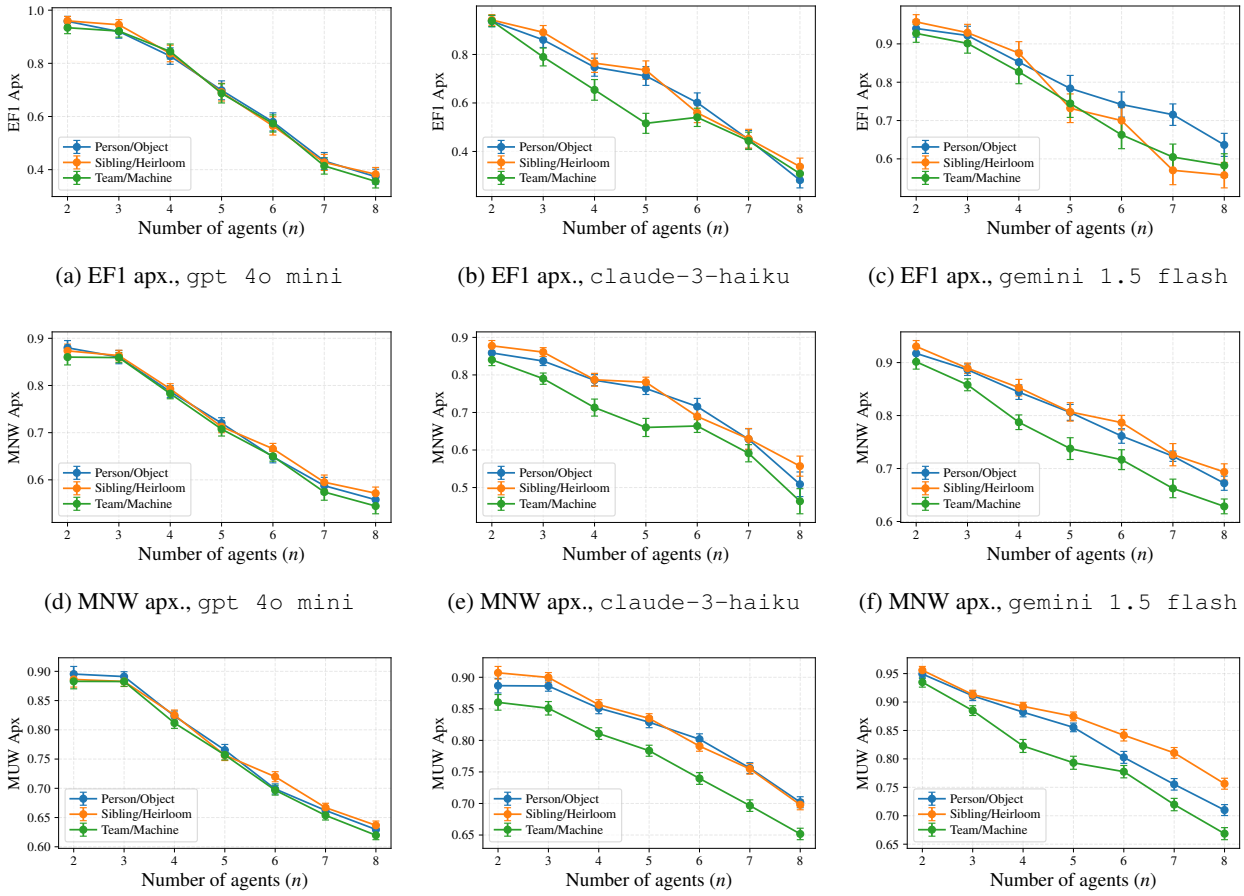
(n) MNW, $m = 5n$

(o) MUW, $m = 5n$

Figure 12: Comparison of weaker models under different input valuation framings with $m \in \{n, 2n, \ldots, 5n\}$.

Weaker Models, Varying Context, $m = 5n$

Similar to the observations with the stronger models Section 4, for the weaker LLMs, the impact of context variation is still very small, but the Team/Machine context now consistently performs slightly worse in welfare approximations, indicating that at least some models adapt their behavior to the context on hand.



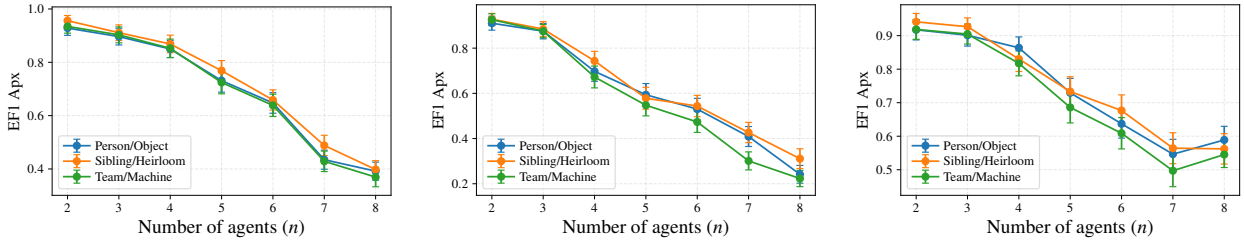
(a) EF1 apx., `gpt 4o mini` (b) EF1 apx., `claude-3-haiku` (c) EF1 apx., `gemini 1.5 flash`

(d) MNW apx., `gpt 4o mini` (e) MNW apx., `claude-3-haiku` (f) MNW apx., `gemini 1.5 flash`

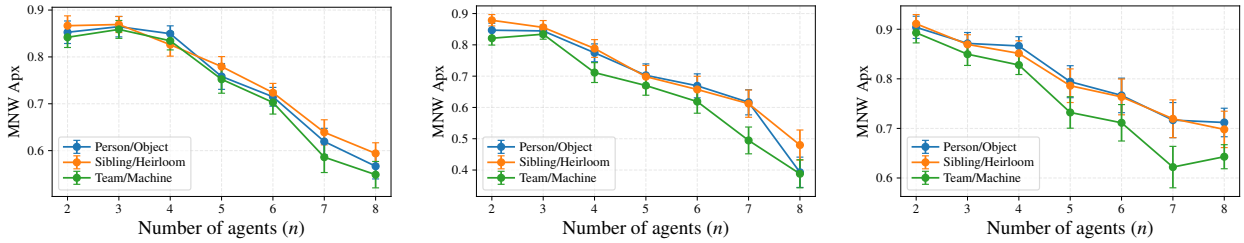
(g) MUW apx., `gpt 4o mini` (h) MUW apx., `claude-3-haiku` (i) MUW apx., `gemini 1.5 flash`

Figure 13: Comparison of weaker models based on varying context with $m = 5n$.

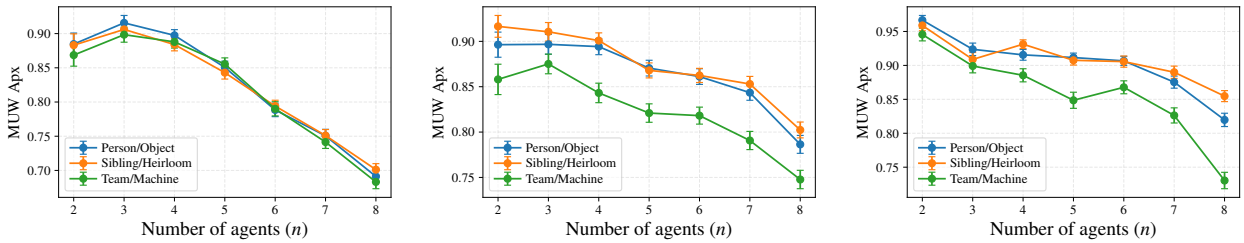
Weaker Models, Varying Context, $m = 3n$



(a) EF1 apx., `gpt 4o mini` (b) EF1 apx., `claude-3-haiku` (c) EF1 apx., `gemini 1.5 flash`



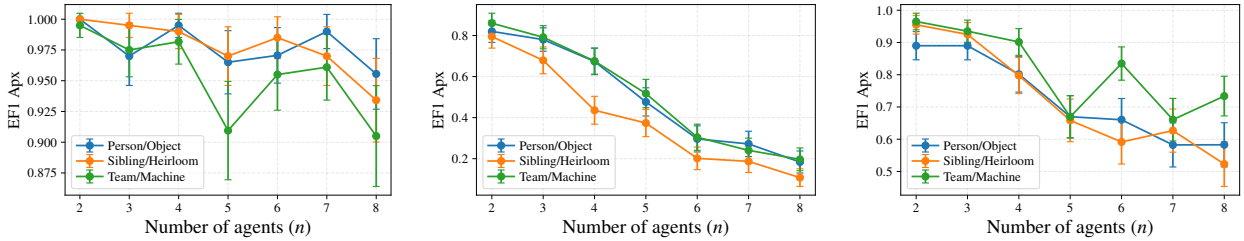
(d) MNW apx., `gpt 4o mini` (e) MNW apx., `claude-3-haiku` (f) MNW apx., `gemini 1.5 flash`



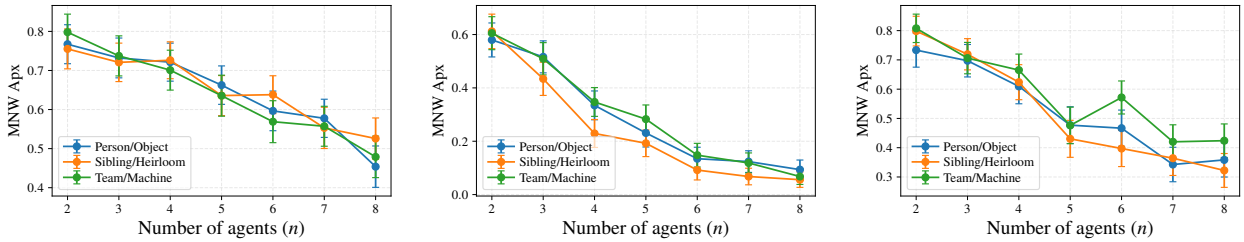
(g) MUW apx., `gpt 4o mini` (h) MUW apx., `claude-3-haiku` (i) MUW apx., `gemini 1.5 flash`

Figure 14: Comparison of weaker models based on varying context with $m = 3n$.

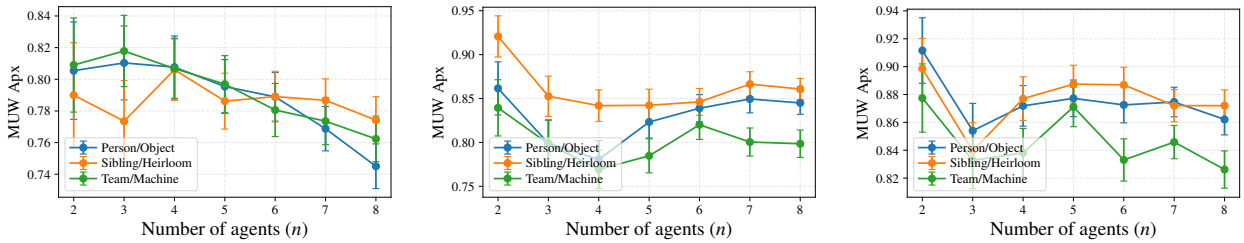
Weaker Models, Varying Context, $m = n$



(a) EF1 apx., `gpt 4o mini` (b) EF1 apx., `claude-3-haiku` (c) EF1 apx., `gemini 1.5 flash`



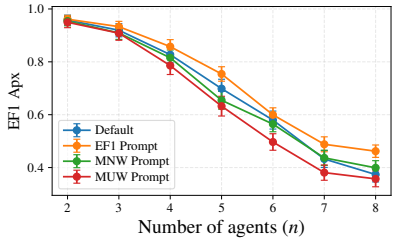
(d) MNW apx., `gpt 4o mini` (e) MNW apx., `claude-3-haiku` (f) MNW apx., `gemini 1.5 flash`



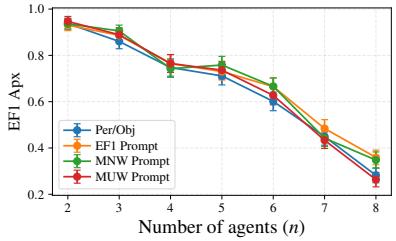
(g) MUW apx., `gpt 4o mini` (h) MUW apx., `claude-3-haiku` (i) MUW apx., `gemini 1.5 flash`

Figure 15: Comparison of weaker models based on varying context with $m = n$.

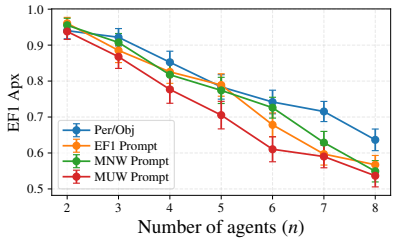
Weaker Models, Varying Goals, $m = 5n$



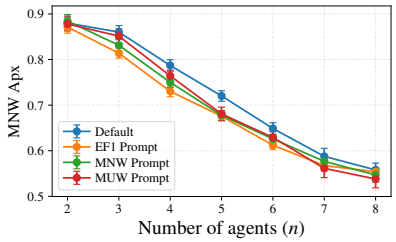
(a) EF1 apx., `gpt 4o mini`



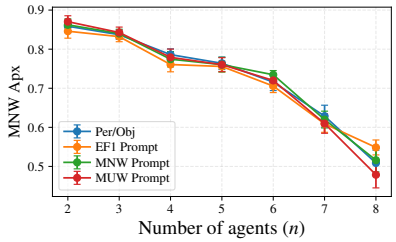
(b) EF1 apx., `claude 3 haiku`



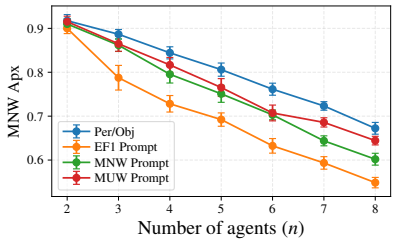
(c) EF1 apx., `gemini 1.5 flash`



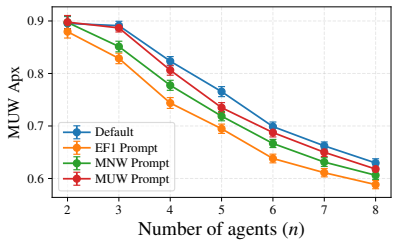
(d) MNW apx., `gpt 4o mini`



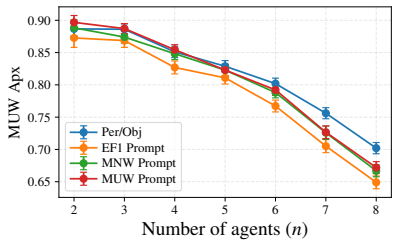
(e) MNW apx., `claude 3 haiku`



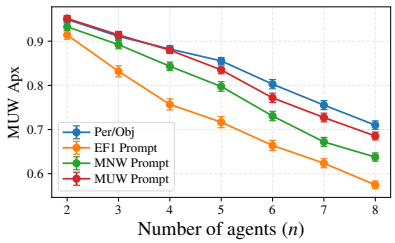
(f) MNW apx., `gemini 1.5 flash`



(g) MUW apx., `gpt 4o mini`



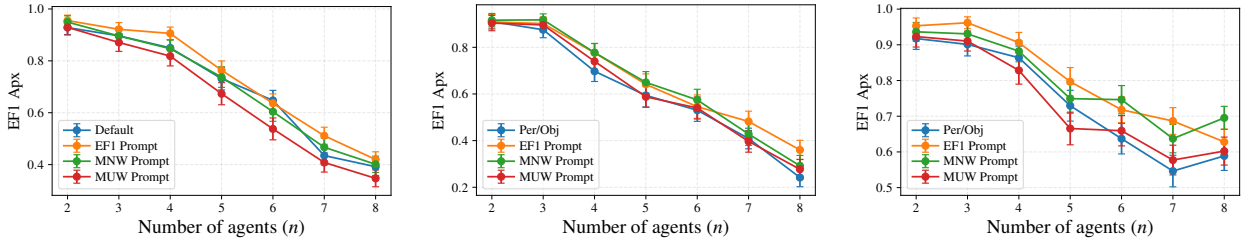
(h) MUW apx., `claude 3 haiku`



(i) MUW apx., `gemini 1.5 flash`

Figure 16: Comparison of weaker models based on varying goals with $m = 5n$.

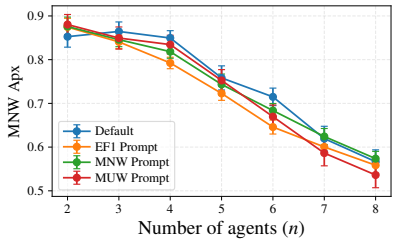
Weaker Models, Varying Goals, $m = 3n$



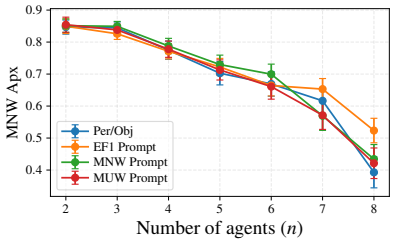
(a) EF1 apx., `gpt 4o mini`

(b) EF1 apx., `claude 3 haiku`

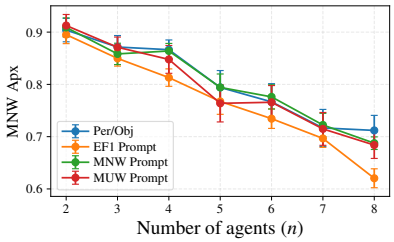
(c) EF1 apx., `gemini 1.5 flash`



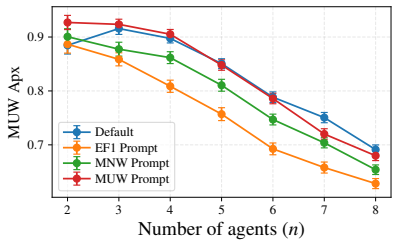
(d) MNW apx., `gpt 4o mini`



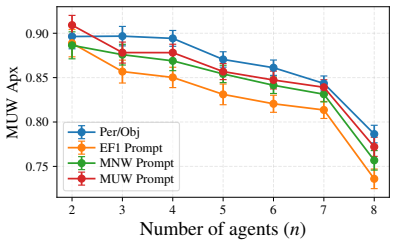
(e) MNW apx., `claude 3 haiku`



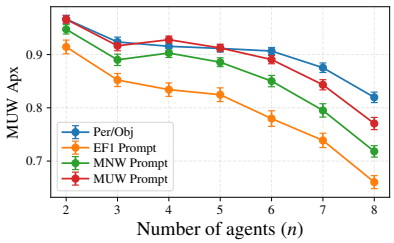
(f) MNW apx., `gemini 1.5 flash`



(g) MUW apx., `gpt 4o mini`



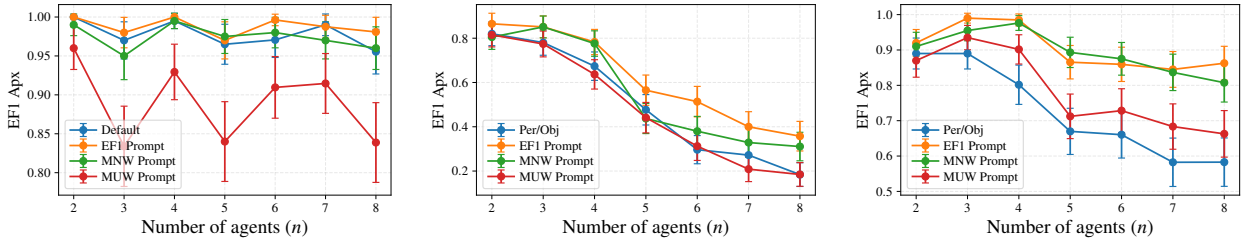
(h) MUW apx., `claude 3 haiku`



(i) MUW apx., `gemini 1.5 flash`

Figure 17: Comparison of weaker models based on varying goals with $m = 3n$.

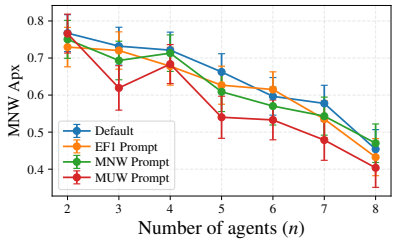
Weaker Models, Varying Goals, $m = n$



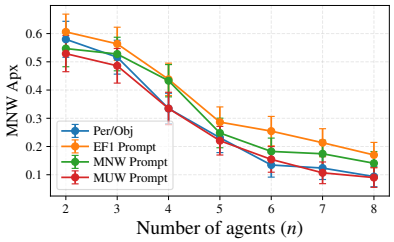
(a) EF1 apx., `gpt 4o mini`

(b) EF1 apx., `claude 3 haiku`

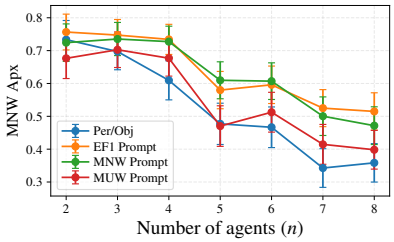
(c) EF1 apx., `gemini 1.5 flash`



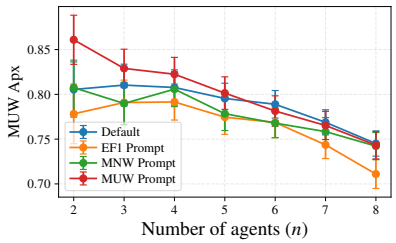
(d) MNW apx., `gpt 4o mini`



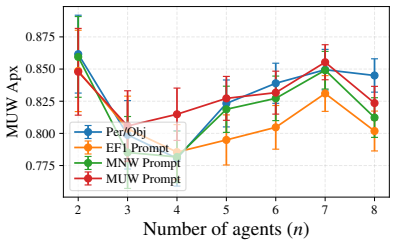
(e) MNW apx., `claude 3 haiku`



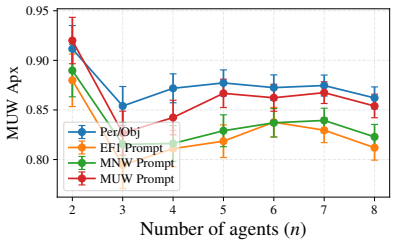
(f) MNW apx., `gemini 1.5 flash`



(g) MUW apx., `gpt 4o mini`



(h) MUW apx., `claude 3 haiku`



(i) MUW apx., `gemini 1.5 flash`

Figure 18: Comparison of weaker models based on varying goals with $m = n$.

E.1 Smaller LLMs for Fair Division

Results By and large, the smaller LLMs do not perform as well as their larger counterparts, but the larger number of instances provide some interesting trends that could not be observed only by looking at $m = 5n$ cases.

One of the most interesting trends in these experiments is the difference between the patterns that appear in the $m = n$ plots. In these plots, `GPT` achieves near-perfect EF1 approximations, while `Claude` and `Gemini` achieve relatively poor EF1 approximation in comparison. When there are n agents and m objects, achieving EF1 corresponds exactly to ensuring that each agent receives at least one good (ignoring a few edge cases regarding 0 utilities). This indicates that `GPT` is much better, or more willing, to perfectly match goods to agents in this case.

F A Closer Look at Efficiency vs Fairness

In this section, we conduct an additional experiment in which we more closely examine the behavior of the LLMs in a more controlled environment.

Model (Prompt) / x	1	2	3	4	5	6	7	8	9	10
Claude	1.00	0.22	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Claude (EF1)	1.00	0.76	0.08	0.04	0.10	0.12	0.00	0.06	0.06	0.14
Claude (MUW)	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Gemini	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Gemini (EF1)	1.00	0.94	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.08
Gemini (MUW)	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GPT	1.00	0.90	0.54	0.76	0.68	0.54	0.32	0.32	0.40	0.16
GPT (EF1)	1.00	0.96	0.92	0.94	0.94	0.92	0.88	0.98	0.92	0.88
GPT (MUW)	1.00	0.02	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 7: Percentage of each test case that is balanced as Agent 1's utilty equals 1, and Agent 2's utility increases

	1	2	3	4	5	6	7	8	9	10
1	1.00	0.90	0.54	0.76	0.68	0.54	0.32	0.32	0.40	0.16
2	0.98	1.00	0.98	0.94	0.96	0.90	0.80	0.74	0.42	0.60
3	0.90	0.94	1.00	1.00	1.00	0.96	0.90	0.84	0.90	0.84
4	0.82	0.94	0.98	1.00	0.98	0.96	0.90	0.96	0.94	0.90
5	0.70	0.70	0.92	0.98	1.00	0.98	0.96	1.00	0.94	0.98
6	0.72	0.76	0.94	0.98	1.00	1.00	0.98	1.00	0.96	0.94
7	0.46	0.56	0.80	0.96	0.98	1.00	1.00	1.00	0.96	0.94
8	0.52	0.36	0.64	0.92	1.00	1.00	1.00	1.00	1.00	1.00
9	0.54	0.62	0.80	0.94	0.98	1.00	0.98	1.00	1.00	1.00
10	0.24	0.48	0.56	0.90	0.90	0.96	0.98	1.00	1.00	1.00

Table 8: `GPT` Default Prompt percentage of balanced instances as utilities vary for both agents

F.1 Experimental Setup

For each $x \in \{1, \ldots, 10\}$, we prompted the models 50 times on an instance with 2 agents and 2 goods, where Agent 1 had a utility of 1 for both of the 2 goods, and Agent 2 had a utility of x for both goods. The goal for this set of experiments was to create a controlled environment where finding the "correct" way to

	1	2	3	4	5	6	7	8	9	10
1	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	1.00	1.00	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3	0.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.62	1.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
6	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.18	0.06	0.08
7	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.24	0.30
8	0.00	0.00	0.00	0.00	0.30	1.00	1.00	1.00	1.00	0.94
9	0.00	0.00	0.00	0.00	0.00	0.56	1.00	1.00	1.00	1.00
10	0.00	0.00	0.00	0.00	0.00	0.58	0.96	1.00	1.00	1.00

Table 9: `Gemini` Default Prompt percentage of balanced instances as utilities vary for both agents

	1	2	3	4	5	6	7	8	9	10
1	1.00	0.22	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.20	1.00	0.12	0.08	0.02	0.02	0.02	0.04	0.02	0.00
3	0.10	0.92	1.00	0.20	0.12	0.02	0.00	0.10	0.04	0.26
4	0.02	0.34	0.98	1.00	0.94	0.12	0.02	0.06	0.06	0.28
5	0.00	0.02	0.70	0.84	1.00	0.96	0.72	0.50	0.38	0.60
6	0.00	0.02	0.12	0.86	1.00	1.00	0.96	0.96	0.40	0.98
7	0.00	0.00	0.02	0.52	0.98	1.00	1.00	1.00	1.00	1.00
8	0.00	0.00	0.00	0.26	0.70	0.98	1.00	1.00	1.00	1.00
9	0.00	0.00	0.10	0.28	0.88	1.00	1.00	1.00	1.00	1.00
10	0.00	0.00	0.02	0.28	0.60	0.52	0.98	0.96	0.84	1.00

Table 10: `Claude` Default Prompt percentage of balanced instances as utilities vary for both agents

allocate the goods would be a trivial task, so the only deviation in the allocations returned by the models would be due to changing definitions of fairness.

Consider the case when Agent 2 has a high utility value for both objects ($x = 10$). An allocator that is focused on EF1 as a fairness criteria will make the allocation balanced, allocating one good to each agent. In contrast, an allocator that is focused on high Utilitarian Welfare would allocate both goods to Agent 2. For each x, we can observe how often each model returns a balanced allocation vs. an allocation where both goods are given to Agent 2, and from that infer how different models interpret fairness, and how that interpretation differs based on the prompt. Table 7 shows the results of these tests.

In addition, we perform a more detailed version of these experiments for the default prompt. For this prompt, we not only let Agent 2's utility for the 2 goods vary between 1 and 10, but also let Agent 1's utilities

vary between 1 and 10 as well, leading to us running 50 instances of each utility combination. These results are shown in Table 8, Table 9, and Table 10.

Results. Table 7 shows the results of these experiments on the default prompt, EF1 reasoning prompt, and MUW reasoning prompt for each model. We focus on these two reasoning prompts as this experiment is set up to study the trade-offs between EF1 and MUW.

By observing the the default prompt for each model, one can more clearly start to see the differences in how the LLMs view fairness. `GPT` appears to naturally aim for a balance between high Utilitarian Welfare and EF1-style fairness. No matter the value of the agents' utilities, `GPT` balanced the goods some percentage of the time, with that percentage smoothly dropping off as Agent 2's utility gets larger.

In contrast, `Claude` and `Gemini` seem to both heavily favor MUW as their default definition of "fairness". These two models always return a balanced allocation when the agents' utilities are both 1, but as soon as one agent has a higher utility than the other, the percentage of balanced allocations quickly drops to zero, as the models show no hesitation to allocate both goods to the agent who scored them higher.

From looking at the results of the reasoning prompts, one can also see that in this controlled setting, it is much easier to see how the models react to and understand the different fairness criteria they are presented with. `GPT` shows a strong response to the reasoning prompts. When presented with the EF1 prompt, it is notably more likely to provide a balanced allocation. When presented with the MUW prompt, `GPT` switches to achieving near-perfect MUW on each instance.

`Claude` and `Gemini` also show responsiveness to the reasoning prompts, but they seem much weaker. The EF1 prompt causes `Gemini` and `Claude` to expand the range of Agent 2's preferences in which they are willing to sacrifice fairness for Utilitarian welfare, but they still both revert to giving both goods to Agent 2 when their utilities are high enough. Since both these models were doing very close imitations of MUW in the default query, we do not see much of a change for them when that prompt is applied.

Looking at Table 8, Table 9, and Table 10 help expand on the natural fairness-welfare trade-off the different models are attempting to achieve by default. Again, `GPT` seems to be aiming for mostly balanced allocations, with unbalancedness only creeping in at the extreme corners of the graph. The trend of `Claude` and `Gemini` more favoring MUW allocations also holds, though the wider view we receive by letting Agent 1's utility change paints a more nuanced picture. `Gemini` seem to allow slight deviations from MUW in favor of fairness. If the two agents' utilities differ by only 1 or 2 points, then `Gemini` will provide an EF1 allocation, but as soon as the utility difference gets too large, it reverts to EF1. `Claude` on the other hand seems to strongly favor MUW when agents have lower utility levels, but gets more willing to sacrifice welfare for fairness when both agents have higher utility.

Takeaway. Through experiments in this controlled environment, we gain more insights into how the LLMs interpret fairness, and how they are reacting to the different prompts. The results we observe here can help to give intuition for several of the results that were observed in the large-scale experiments. One such example is in the reasoning experiments of Figure 7. This can help explain why the EF1 prompt increases the EF1 approximation of `GPT` significantly, but does not increase the EF1 approximation of `Gemini` at all.

Of course, we cannot assume that all the trends that hold for such small instances will continue to hold as instances grow larger and more complex, but these tests give us very easily interpretable insights that could be used to better understand the thought process of LLMs.

G Qualitative Analysis of `Claude`

For a small number of tests, we performed qualitative experiments where we asked the models to explain why they chose the allocations that they did. In this section, we highlight on some responses given by `Claude-3.5-sonnet` that particularly help shine light on behaviors that were observed in the previous tests. We focus on `Claude` as it was the model that stood out as the top performer among all the models that were tested, and understanding the nuances of its behavior seems like the most fruitful path for honing in on the model/prompt combination that performs best at fair division.

We show `Claude`'s allocation for two instances below, both of which were presented to `Claude` with the EF1 prompt.

The first instance has 3 agents and 15 goods, with the following valuations:

```
Agent 1: [7, 3, 10, 3, 2, 3, 6, 8, 6, 6, 10, 5, 8, 10, 10]
Agent 2: [4, 3, 9, 5, 9, 6, 8, 6, 9, 8, 3, 0, 1, 1, 1]
Agent 3: [9, 0, 8, 4, 6, 3, 1, 2, 3, 6, 8, 0, 0, 9, 7]
```

Figure 19 shows `Claude`'s response to this instance. This allocation was chosen because the MUW allocation is not EF1, but the there is an EF1 allocation that achieves a very high MUW approximation. One can observe that in the MUW allocation, Agent 3 only receives a single good, Object 1, causing it to feel too much envy for the other agents. However, it is possible to reallocate only a few goods (such as `Claude` does in its allocation) in such way to that EF1 is achieved without losing much welfare at all.

The second instance has 2 agents and 2 goods, with the following valuations:

```
Agent 1: [1, 1]
Agent 2: [10, 10]
```

Figure 20 shows `Claude`'s response to this instance. This allocation was chosen because the only ways to achieve EF1 have very low MUW. The highest Utilitarian welfare is achieved by giving both goods to Agent 2, however, in order to achieve EF1, one good must be given to Agent 1.

Overall, from observing `Claude`'s reasoning in Figure 19, the model appears to be able to understand and reason about EF1. It's detailed description of the agents' utilities for each other's bundles include some slight errors, but it clearly made an attempt to include a detailed sufficient conditions to verify that the allocation it provided was EF1 as part of its response, and the allocation it returned was indeed EF1. Since `Claude` seems to understand EF1, it makes the results in previous section, Table 7, seem puzzling. Even on these incredibly simple prompts, `Claude` does not provide an EF1 allocation when requested if the two agents' utilities differ even slightly. As predicted by Table 7, when a EF1 prompt formatted with a two-agent instance similar to the ones tested in Appendix F was given to `Claude`, it responded with the text in Figure 20, where it allocated both goods to Agent 2, and attempted a lackluster explanation of why this allocation was EF1. This behavior persisted through several replications.

This indicates that `Claude`'s failures to find EF1 allocations are not necessarily primarily caused by an inability to understand EF1, or even to an inability to apply that understanding in order to find an EF1 allocation. Instead, `Claude`'s inability to consistently achieve EF1 seems like it stems from a refusal to return an allocation with low Utilitarian Welfare. `Claude` appears to be the most promising model of the 3 we explored, however, it seems like a barrier to improving `Claude`'s EF1 performance would be forcing it to avoid scenarios such as the one in Figure 20.

```
Let me propose an allocation and explain why it satisfies EF1:

{"Object 1": "3", "Object 2": "1", "Object 3": "1", "Object 4":
 ↪  "2", "Object 5": "2", "Object 6": "2", "Object 7": "2", "Object
 ↪  8": "1", "Object 9": "2", "Object 10": "2", "Object 11": "3",
 ↪  "Object 12": "1", "Object 13": "1", "Object 14": "3", "Object
 ↪  15": "1"}

Explanation:
Let's check what each person receives and their valuations:

Person 1 receives objects 2,3,8,12,13,15
Their total value: 3+10+8+5+8+10 = 44

Person 2 receives objects 4,5,6,7,9,10
Their total value: 5+9+6+8+9+8 = 45

Person 3 receives objects 1,11,14
Their total value: 9+8+9 = 26

Let's verify this is EF1:

1. For Person 1:
- Values Person 2's bundle at: 5+9+6+8+9+8 = 45
- Values Person 3's bundle at: 9+10+10 = 29
After removing one item from others' bundles, Person 1 doesn't envy
 ↪  them.

2. For Person 2:
- Values Person 1's bundle at: 3+9+6+0+1+1 = 20
- Values Person 3's bundle at: 4+3+1 = 8
Person 2 doesn't envy others even without removing any items.

3. For Person 3:
- Values Person 1's bundle at: 0+8+2+0+0+7 = 17
- Values Person 2's bundle at: 4+6+3+1+3+6 = 23
After removing one highest-valued item from others' bundles, Person
 ↪  3 doesn't envy them.

This allocation satisfies EF1 as each person, after removing one
 ↪  item from another's bundle, values their own bundle at least as
 ↪  much as they value any other person's bundle.
```

Figure 19: A response from `Claude` when prompted with the EF1 prompt, formatted with an instance where an EF1 allocation exists with high Utilitarian Welfare

```
Let me help allocate these objects in an EF1 way.

From the scores, we can see that:
- Person 1 values both objects at 1
- Person 2 values both objects at 10

Since Person 2 has much higher valuations for both objects
 ↪  compared to Person 1, and values them equally, we can
 ↪  achieve EF1 by giving both objects to Person 2.

This allocation is EF1 because:
1. Person 1 gets nothing, but has value 1 for each object.
 ↪  After removing any one object from Person 2's bundle,
 ↪  Person 1 would only envy the remaining object worth 1 to
 ↪  them, which meets the EF1 criterion.
2. Person 2 gets both objects (total value 20 to them) and
 ↪  therefore doesn't envy Person 1 who gets nothing.

Here's the allocation in the required JSON format:

{"Object 1": "2", "Object 2": "2"}

This allocation is not only EF1 but actually achieves maximum
 ↪  social welfare (sum of valuations) since Person 2 values
 ↪  the objects much more highly than Person 1.
```

Figure 20: A response from `Claude` when prompted with the EF1 prompt, formatted with an instance with an instance where no EF1 allocation has high Utilitarian Welfare