Fairness Perceptions of Large Language Models

Benjamin Cookson , **Soroush Ebadian** , **Nisarg Shah**
University of Toronto
{bcookson, soroush, nisarg}@cs.toronto.edu,

Abstract

Large language models (LLMs) are increasingly used for decision-making tasks where fairness is an essential desideratum. But what does fairness even mean to an LLM? To investigate this, we conduct a comprehensive evaluation of how LLMs perceive fairness in the context of resource allocation, using both synthetic and real-world data.

We observe that various state-of-the-art LLMs, when asked to be fair, prioritize improving collective welfare over distributing benefits equally. Their perception of fairness is somewhat sensitive to how user preferences are provided, but less so to the real-world context of the decision-making task. Finally, we show that the best strategy for aligning an LLM's perception of fairness to a specific criterion is to provide it as a mathematical objective, without referencing "fairness", as this prevents the LLM from mixing the given criterion with its prior notions of fairness. Our results provide practical insights regarding when to use LLMs for fair decision-making and when using traditional algorithms may be more appropriate.

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 fairness criteria from the fair division literature to machine learning applications [Balcan *et al.*, 2019; Hossain *et al.*, 2020; Chen *et al.*, 2019; Micha and Shah, 2020; Kellerhals and Peters, 2024; Caragiannis *et al.*, 2024], 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 in recent years. These models are increasingly driving decision-making by sitting at the core of powerful AI agents that can autonomously act in the real world [News, 2025]. These models exhibit social understanding gleaned from their pretraining 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 enabled 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 investigate the perceptions of fairness exhibited by LLMs using fair division — specifically, fair allocation of indivisible goods to a set of agents — as our example domain. We choose fair division because there are several reasons that make LLMs aptly suited for adoption in real-world fair division applications. They are wildly popular, easy to use, and often freely available. Further, their unique ability to understand contextual nuance can give them an edge over traditional algorithms (see Section 7 for further discussion). Our objectives are threefold:

1. *What is fair in the eyes of LLMs?* When LLMs are asked to be "fair", what metrics do they prioritize?
2. *What influences fairness perception?* How does an LLM's understanding of fairness depend on factors such as the nature of agents and goods involved, and the framing of the agents' preferences?
3. *To what extent can we steer LLMs?* Do the LLMs have the reasoning abilities to optimize user-specified fairness criteria?

Under the first two objectives, our goal is to identify patterns that are common across different LLMs. These patterns may reflect perceptions of fairness encoded in the (largely common) pretraining datasets that the LLMs are trained with and, therefore, are likely to persist even as more capable LLMs are deployed in the future. Under the third objective, on the other hand, we seek to conduct an evaluation of the capabilities of the current state-of-the-art (SOTA) LLMs. While these models may soon be superseded, this portion of our work contributes a framework that can be used for continuous monitoring of the fairness capabilities of LLMs; thus, it contributes to the quickly-growing literature in AI on conducting LLM evaluations on various dimensions such as safety, trustworthiness, and inclination to hallucinate [Guo *et al.*, 2023; Chang *et al.*, 2024; Chu *et al.*, 2024].

Our results. We evaluate fairness perceptions of three state-of-the-art families of LLMs — `Claude` (by Anthropic) [Anthropic, 2024], `Gemini` (by Google) [DeepMind, 2023], and `GPT` (by OpenAI) [OpenAI, 2023] — using both synthetic data and real data from Spliddit.org. Using carefully designed prompts, 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 behavior to that of traditional algorithms based on (multiplicative) approximations to popular fairness and efficiency criteria, such as envy-freeness up to one good (EF1) and social welfare, with the goal of analyzing the fairness-efficiency tradeoff exhibited by LLM-generated allocations.

Our main takeaway is that the when asked for fairness, LLMs value high social welfare, seemingly at the expense of envy-based notions of fairness. This can be seen visually in Figure 1. Although the different models vary in the exact approximations they achieve of the criteria we examine, all three models largely follow the same trends. Namely, in instances where it is impossible to achieve high approximations of EF1 and social welfare simultaneously, the LLMs opt for high social welfare.

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, heirlooms to siblings after a parent's death, or machines to teams in a corporate setting, the context appears to make little difference in how LLMs perform the allocation, at least when given only a brief 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 prompted to explicitly seek EF1, as opposed to simply maximizing "fairness", their tradeoff between fairness and efficiency changes slightly. Specifically, they tend to achieve higher EF1 approximations on average, although the overall trend of EF1 approximation degrading as it becomes harder to achieve EF1 and social welfare simultaneously still remains. We also prompt the LLMs using a purely combinatorial definition of EF1, dropping the language of "fairness" and "allocations of goods" entirely. Here, `GPT` and `Gemini` both do not see the same drop off in EF1 approximation as previously, while `Claude` still appears to prioritize efficiency over fairness in this setting.

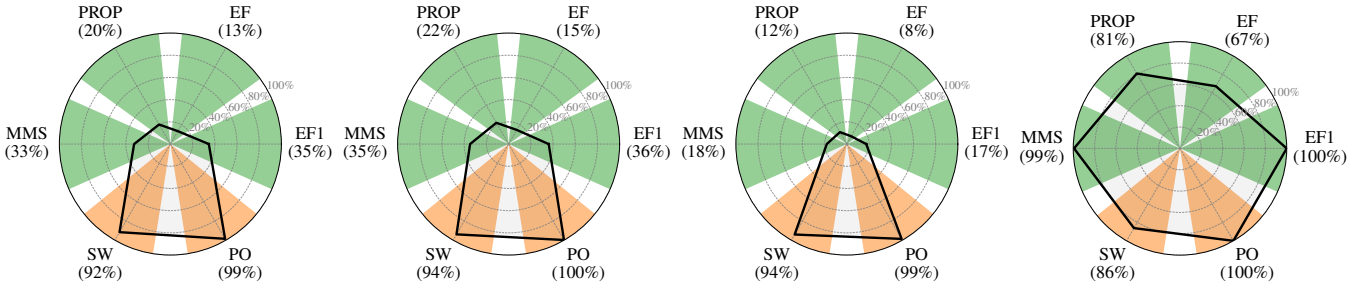
In our analysis of the real-world data from Spliddit.org, we find that under these instances, the LLMs do better at achieving good EF1 approximations. This is partially due to the fact that Spliddit.org forces agent valuations to be *normalized*, which generally means it is easier to find allocations that achieve good fairness and efficiency simultaneously. However, even when compared to the LLMs' results on the subset of our synthetic instances which are normalized, we find that LLMs perform better overall on the Spliddit.org instances. Due to the larger scale of the synthetic instances, and the control they allow in varying parameters, we focus on them rather than the Spliddit.org data for the majority of our analysis. However, we provide a detailed look at how all our tests performed on the Spliddit.org instances in Appendix E.

Although the main goal of our work is to dissect the interplay between EF1 and social welfare in the LLMs perception of fairness, we also include a detailed summary of the aggregate performance of LLMs under a variety of fairness and efficiency metrics. These summaries, shown in Figure 1, give a high-level overview of exactly what the LLMs are prioritizing in their allocations, with the key takeaway again being that they seem to value efficiency more than fairness.

1.1 Related Work

To the best of our knowledge, ours is the first work to explore the use of LLMs in fair division, with the exception of the simultaneous and independent recent work of Hosseini and Khanna [2025].

Hosseini and Khanna also investigate fairness perceptions of LLMs in the fair division context, but using a very different approach. They use 10 hand-crafted instances borrowed from the work of Herreiner and Puppe [2007], along with their slight variations. For each instance, they ask LLMs and humans to pick from a small menu of predetermined allocations. This menu is designed to include allocations that satisfy different subsets of four primary metrics they consider: envy-freeness, equitability, egalitarian welfare, and social welfare. In contrast, our study uses tens of thousands of instances generated in a randomized fashion and allows the LLMs to pick



(a) GPT (b) Claude (c) Gemini (d) MNW

Figure 1: Radar charts showing average approximation performance of LLMs and the MNW baseline across fairness (green) and efficiency (orange) metrics. Each axis corresponds to a criterion, with higher values (closer to the outer edge) indicating better approximation to that metric.

from the entire set of (exponentially many) feasible allocations. This allows us to evaluate the unhindered fairness perceptions of LLMs in a more robust manner.

More broadly, our work is tangentially related to three lines of work.

LLM → 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] discuss broader opportunities and risks of LLMs in deliberative platforms like 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 → LLM. In the opposite direction, researchers have recently explored applying social choice concepts to the design of LLMs. For example, Zhong *et al.*; Williams [2024; 2024] use the Nash social welfare in the 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.

LLM evaluations. A rapidly growing literature evaluates LLMs on safety, trustworthiness, hallucination, reasoning, etc.; see surveys by Guo *et al.*; Chang *et al.*; Chu *et al.* [2023; 2024; 2024]. Several studies focus on *fairness* of LLMs, either broadly [Li *et al.*, 2023] or in specific domains like recommendations [Zhang *et al.*, 2023] and ranking [Wang *et al.*, 2024]. To our knowledge, our work and the independent study by Hosseini and Khanna [2025] are the first to evaluate fairness of LLMs in resource allocation.

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.

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 $\cup_{i \in N} A_i = M$.

Synthetic data. For our synthetic data experiments, we build on the setup of Ebadian *et al.* [2024]. They draw agent utilities from the Dirichlet-multinomial distribution, defined as follows. First, a vector $\vec{p}$ is drawn uniformly from the $(m-1)$-simplex (i.e., from the Dirichlet distribution), where p_g represents the "market value" of good g. Then, for each agent i, a utility vector $(v_i(\{g\}) : g \in M)$ is independently drawn from the multinomial distribution with parameters T and $\vec{p}$, ensuring that $\mathbb{E}[v_i(\{g\})] = p_g$ for each $g \in M$ and $\sum_{g \in M} v_i(\{g\}) = T$. They choose this distribution to induce a sharper tradeoff between fairness and efficiency than simply drawing all utilities i.i.d. We sample a different total utility T_i for each agent i independently from the uniform distribution over the set of integers $\{(50 - \lambda) \cdot m, \ldots, (50 + \lambda) \cdot m\}$. When $\lambda = 0$, our sampling process coincides with theirs. As λ increases, the total utility varies more across agents, thereby intensifying the tension between fairness (equal distribution of goods) and efficiency (allocating more to higher-utility agents).

We vary the number of agents $n \in \{2, 3, \ldots, 10\}$ (de-

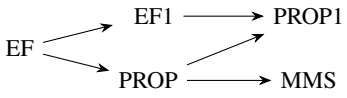


Figure 2: Relationships between fairness notions.

fault $n = 5$), the number of goods $m \in \{n, 2n, \ldots, 5n\}$ (default $m = 3n$), and the total utility variation parameter $\lambda \in \{0, 5, \ldots, 40\}$ (default $\lambda = 20$). When varying parameter, we fix the remaining two parameters to their default values, sample 200 instances, and plot the averages along with 95% confidence intervals.

Spliddit data. We utilize real-world goods division instances from Spliddit.org. In these instances, the total utility of each agent for all goods is always 1000. Out of the 5295, we focus on the 4835 instances in which a positive Nash welfare is attainable (see Footnote 1), and show results averaged over these instances. These instances involve between 2 to 15 agents and 2 to 96 goods, with more than 99% of the instances involving at most 5 agents and at most 15 goods.

Evaluation: fairness criteria. 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, i.e., $v_i(A_i) \geqslant v_i(A_j)$ for all $i, j \in N$. For indivisible goods, this is not always attainable. Hence, we measure its multiplicative approximation, and multiplicative approximations of its four widely studied relaxations: envy-freeness up to one good (EF1) [Budish, 2011],proportionality (PROP) [Steinhaus, 1948], proportionality up to one good (PROP1) [Conitzer *et al.*, 2017], and maximin share (MMS) [Budish, 2011].

Figure 2 depicts the logical relationships between these criteria. In the interest of space, we define and present results for only EF1 approximation in the main body, deferring the definitions of and results for the rest Appendix C.

- *EF1 approximation:* For an allocation A, this 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 $v_i(A_i) \geqslant \alpha \cdot v_i(A_j \setminus \{g\})$.

EF1 allocations are guaranteed to exist, and the maximum Nash welfare (MNW) algorithm [Caragiannis *et al.*, 2019], which provably satisfies EF1, serves as our primary baseline (see Section 2). MMS allocations need not exist [Kurokawa *et al.*, 2018], but a $\frac{3}{4} + \frac{3}{3836}$ approximation is achievable [Akrami and Garg, 2024].

While the MMS approximations are quantitatively similar to EF1, the PROP1 approximations are quite different. This is due to subtleties about how our synthetic instances were generated, which we also explain in Appendix C. We emphasize that our results are pessimistic for fairness of LLMs, and our use of the weaker EF1 criterion instead of the stronger EF criterion only makes them stronger.

Evaluation: efficiency criteria. We use two prominent efficiency criteria from the literature: (utilitarian) social welfare (SW) and Pareto optimality (PO). Since maximizing SW implies PO, and PO approximation is at least as high as SW, we focus on SW in the main text and defer the definition and similar results for PO to Appendix C.

- *SW approximation*: The (utilitarian) social welfare of an allocation A is the sum of agent utilities, i.e., $\mathrm{SW}(A) = \sum_{i \in N} u_i(A_i)$, and its SW approximation is its social welfare as a fraction of the highest possible social welfare, i.e., $\frac{\mathrm{SW}(A)}{\max_B \mathrm{SW}(B)}$.

Baseline algorithms. We compare the behavior of LLMs to that of three popular fair division algorithms:

- *Maximum Nash welfare* (`MNW`) [Caragiannis *et al.*, 2019] returns an allocation that maximizes the Nash welfare, i.e., $\prod_{i \in N} v_i(A_i)$.[1] This provably achieves EF1 and PO [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.
- *Round Robin* (`RR`) is an iterative algorithm that guarantees EF1 but not necessarily PO. 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 social welfare* (`MSW`) returns an allocation with the highest utilitarian social welfare. Under additive valuations, this simply allocates each good to an agent with the highest value for it. This is PO but does not guarantee any positive EF1 approximation.

Our primary focus is to investigate how LLMs behave when asked to be fair, and not to compare them with traditional algorithms. Hence, for clarity, we show only the MNW rule in the plots in the main body. In Appendix D, we compare LLMs to the other two baselines.

Large language models. We use three state-of-the-art commercial LLMs: `gpt-4o` (in short, `GPT`) from OpenAI, `claude-3.5-sonnet-20241022` (in short, `Claude`) from Anthropic, and `gemini-1.5-pro` (in short, `Gemini`) from Google.

In Appendix B, we report input/output token sizes, provide rough estimates of LLM costs for fair division, and show how costs scale with instance size.

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).

[1]The algorithm is more subtle in edge cases where all allocations yield zero Nash social welfare, but our experiments focus on instances that admit allocations with strictly positive utility for all agents (and thus positive Nash social welfare).

- *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.

- *"Fair"* (default): The model is asked to allocate goods "fairly," without an explicit definition of fairness.
- *EF1 fair*: The model is instructed to find an EF1 allocation, with EF1 introduced as a fairness criterion and defined mathematically.
- *EF1 combinatorial*: Same as the EF1 fair prompt, but framed as a purely combinatorial problem — without reference to "fairness" or the context of allocating goods.

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,[2] mapping each good to the index of its assigned agent. We explicitly instruct the model not to include any additional text or reasoning.

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

In Section 3, we compare all models and baselines using the default settings for the first three components. Then, in Sections 4 to 6, we vary each component individually while keeping the others at their default.

3 LLMs for Fair Division

The plots in Figure 3 highlight how the LLMs behave when prompted to simply find a "fair" allocation, with no further instruction on the problem context, or what "fairness" should entail. From these results, it is clear that all models generally prioritize efficiency (measured by approximation to SW) over fairness (measured by approximation to EF1). As a baseline, we first examine the performance of maximum Nash welfare (MNW), which is known to always return an EF1 allocation. This explains why, in figures (c) and (f), as λ, the utility variation parameter, increases the SW approximation of the MNW allocations decrease sharply. When one agent has a much higher utility for all goods compared to another agent, achieving high social welfare requires allocating all goods to that agent, which goes against fairness. In contrast to MNW, we observe that as λ increases, the EF1 approximation of all three LLM models declines rapidly, while their SW approximation remains high.

Takeaways. In plots (a) (d), and (b) (e), we can also see how the EF1 and SW approximation of the models change as we vary n and m respectively. These represent increasing the complexity of the instances. As n increases, we can again see that MNW becomes worse at approximating SW. Intuitively, this is because having more agents raises the probability that one agent a much lower utility sum than some other agent, making it so that some goods inefficiently allocate some goods in order to ensure fairness. Here we see that this worsening tradeoff causes the same behavior in the LLMs who get drastically worse at fairness in order to maintain high efficiency.

In contrast, when m increases, we can see that MNW's SW approximation does not see significant change. It can be seen that when $m = 5$, the models all perform much better at fairness than when m is higher. Between $m = 5$ and $m = 10$, we see a steep drop off in the level of fairness the models achieve, and an increase in efficiency. For all $m \geqslant 10$, the fairness and efficiency levels stay much more constant, with only small decreases. In all our experiments, it appeared that when LLMs are provided with the same number of goods as there are agents $n = m$, their behavior was much different than when $m > n$, with the models being more likely to provide a *balanced* allocation, where all agents received the same number of items, even if that led to inefficiencies. This behavior is what explains the steep drop off.

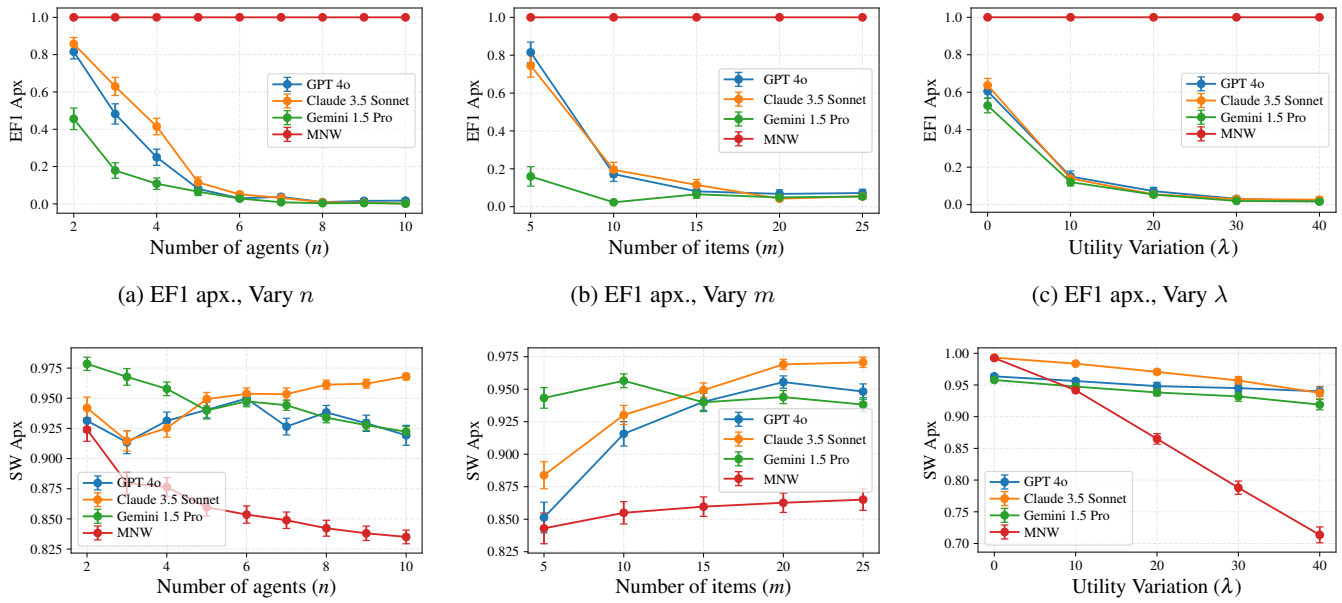
We also evaluate the LLMs performance against real world instances from Spliddit.org, with aggregate results shown in Figure 6. Interestingly, the LLMs perform significantly better on fairness for these real-world instances than for the synthetic ones. The most natural comparison is to the synthetic instances with $\lambda = 0$ in Figure 3, since Spliddit.org instances are normalized. On the Spliddit.org instances, the LLMs maintain relatively high efficiency, while achieving high EF1 approximations, around 0.8 to 0.9, compared to approximately 0.6 on comparable synthetic instances. This indicates that real-world instance are more likely to have a better fairness-efficiency tradeoff, allowing LLMs to find passable allocations despite their efficiency bias.

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.

Takeaways. The results in Figure 4 show that contextual changes have little effect on the LLMs' fairness

[2] 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., Vary n (b) EF1 apx., Vary m (c) EF1 apx., Vary λ

(d) SW apx., Vary n (e) SW apx., Vary m (f) SW apx., Vary λ

Figure 3: Comparison of models for the default prompt by varying n, m, or λ.

and efficiency behavior. In both the Siblings/Heirlooms and Teams/Machines scenarios, the models' approximations closely mirror those of the default setting, suggesting that small contextual shifts do not alter the tradeoffs these models make.

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).

Takeaways. Figure 5 shows that how preferences are framed does affect 2 out of 3 models. For `Claude` and `Gemini`, the Object/Person framing leads to lower EF1 approximations but higher social welfare, suggesting a shift toward efficiency at the expense of fairness. One possible explanation is that presenting all agents' valuations for each object in a single list makes it easier for the LLM to compare utilities across agents and assign each object to the agent who values it most. This raises an important question: when LLMs fail to find a maximum social welfare allocation, is it due to a preference for fairness, or simply an inability to identify the optimal outcome? Interestingly, `GPT` appears largely unaffected by preference framing, with near-identical scores across both settings.

6 Steer LLMs or Let Them Be Free?

In this section, we evaluate how LLMs perform when specifically asked to aim for fairness, both by asking them directly to find an allocation that is EF1, and by providing them the instance as a purely combinatorial problem, and asking them to find an allocation with a property equivalent to EF1.

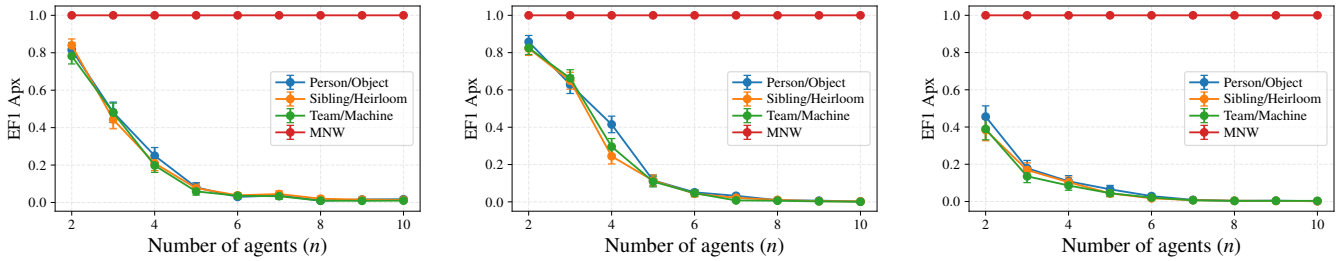
Takeaways. Figure 7 varies λ to control how difficult it is to satisfy fairness and efficiency simultaneously. For two of the three models (`GPT` and `Gemini`), we observe a very interesting difference between the EF1 and Combinatorial prompts. Across all models, allocations from the EF1 prompt are consistently fairer than those from the default prompt. However, EF1 approximations still decline as λ increases, reflecting the growing difficulty of the task.

In contrast, for `GPT` and `Gemini`, the Combinatorial prompt produces allocations whose fairness remains stable as λ increases. This suggests that when the task is framed as explicitly satisfying EF1 in a combinatorial setting, without the usual allocation context, LLMs deprioritize efficiency and focus more narrowly on the specified goal. When the allocation context is present, however, even explicit instructions to satisfy EF1 may be overridden by implicit reasoning about tradeoffs. Interestingly, `Claude` does not follow this pattern — it appears to favor efficiency over fairness even when the prompt strips away allocation context.

In Figure 9, we again observe that all prompt types degrade similarly as n increases, likely due to the increasing complexity of achieving fair and efficient allocations.

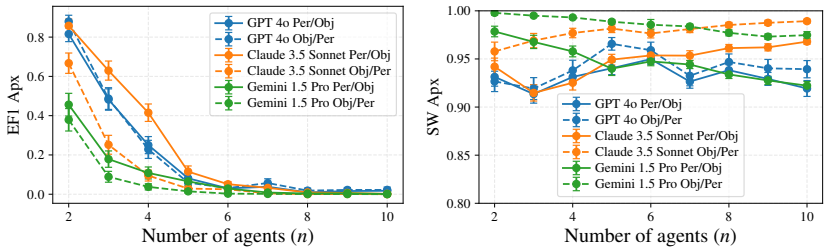
7 Discussion

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.



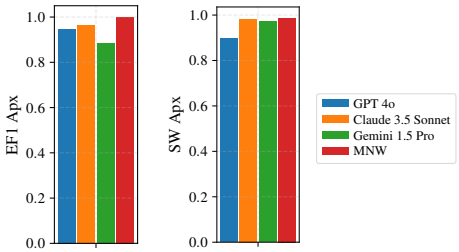
(a) EF1 apx., `GPT` (b) EF1 apx., `Claude` (c) EF1 apx., `Gemini`

Figure 4: Comparison of models based on varying context with $m = 3n$ and $\lambda = 20$.



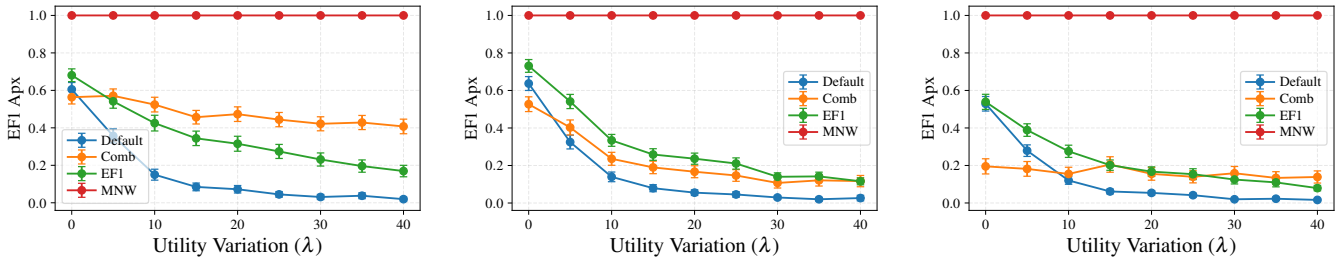
(a) EF1 apx. (b) SW apx.

Figure 5: Comparison of models under different input valuation framings with $m = 3n$ and $\lambda = 20$.



(a) EF1 apx. (b) SW apx.

Figure 6: Comparison of models on Spliddit.org.



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

Figure 7: Comparison of models based on varying goals with $n = 5$ and $m = 15$.

Prompt engineering. While we experimented with variations 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.

Task generalization. We focused on 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 approximations to EF1, SW, and other fairness and efficiency notions 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 the current capabilities of existing models rather than introducing new ones, which somewhat limits the ethical risks involved. Nevertheless, there remains a potential risk that our methodology may be used to "validate" a model in terms of fairness, even when the model exhibits significant unfairness along dimensions not captured in our analysis. We stress that our evaluation focuses on *specific* fairness aspects in how LLMs allocate indivisible goods, and should not be interpreted as a comprehensive audit of fairness.

References

[Akrami and Garg, 2024] Hannaneh Akrami and Jugal Garg. Breaking the 3/4 barrier for approximate maximin share. In *Proceedings of the 35th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 74–91, 2024.

[Amanatidis *et al.*, 2022] 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.

[Anthropic, 2024] Anthropic. Introducing claude 3, 2024.

[Bakker *et al.*, 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.

[Balcan *et al.*, 2019] Maria-Florina Balcan, Travis Dick, Ritesh Noothigattu, and Ariel D Procaccia. Envy-free classification. In *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 1238–1248, 2019.

[Binns, 2018] 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.

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

[Caragiannis *et al.*, 2019] 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.

[Caragiannis *et al.*, 2024] Ioannis Caragiannis, Evi Micha, and Nisarg Shah. Proportional fairness in non-centroid clustering. In *Proceedings of the 37th Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 19139–19166, 2024.

[Chakraborty *et al.*, 2024] 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.

[Chang *et al.*, 2024] 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.

[Chen *et al.*, 2019] Xingyu Chen, Brandon Fain, Liang Lyu, and Kamesh Munagala. Proportionally fair clustering. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 1032–1041, 2019.

[Chu *et al.*, 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.

[Conitzer *et al.*, 2017] 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.

[DeepMind, 2023] Google DeepMind. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[Ebadian *et al.*, 2024] 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.

[Fish *et al.*, 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.

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

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

[Grgic-Hlaca *et al.*, 2018] 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.

[Guo *et al.*, 2023] 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.

[Hadi *et al.*, 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.

[Harsanyi, 1975] 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.

[Herreiner and Puppe, 2007] Dorothea K Herreiner and Clemens Puppe. Distributing indivisible goods fairly: Evidence from a questionnaire study. *Analyse & Kritik*, 29(2):235–258, 2007.

[Hossain *et al.*, 2020] 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.

[Hosseini and Khanna, 2025] Hadi Hosseini and Samarth Khanna. Distributive fairness in large language models: Evaluating alignment with human values. arXiv:2502.00313, 2025.

[Kellerhals and Peters, 2024] Leon Kellerhals and Jannik Peters. Proportional fairness in clustering: A social choice perspective. In *Proceedings of the 37th Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 111299–111317, 2024.

[Kurokawa *et al.*, 2018] David Kurokawa, Ariel D. Procaccia, and Junxing Wang. Fair enough: Guaranteeing approximate maximin shares. *Journal of the ACM*, 64(2): article 8, 2018.

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

[Mehrabi *et al.*, 2021] 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.

[Micha and Shah, 2020] Evi Micha and Nisarg Shah. Proportionally fair clustering revisited. In *Proceedings of the 47th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 85:1–85:16, 2020.

[News, 2025] OpenAI News. Introducing Operator. https://openai.com/index/introducing-operator, 2025.

[OpenAI, 2023] OpenAI. Gpt-4 technical report, 2023.

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

[Saxena *et al.*, 2019] 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.

[Shah, 2023] 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.

[Small *et al.*, 2023] 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.

[Srivastava *et al.*, 2019] 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.

[Steinhaus, 1948] Hugo Steinhaus. The problem of fair division. *Econometrica*, 16:101–104, 1948.

[Wang *et al.*, 2024] 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.

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

[Zhang *et al.*, 2023] 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.

[Zhong *et al.*, 2024] 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 of Submission #7010

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

A Prompts

To reiterate on the discussion of our experiments in Section 2, in total, our experiments involved 12 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.
- 2 *Reasoning* prompt: 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, both in a straightforward way, and in a purely combinatorial way to mask the fact that it was a fair allocation problem.
- For each of the above 6 prompts, 2 copies of that prompt were needed for both the synthetic and the Spliddit.org data. For the Spliddit.org data, we informed the LLM that each agents' utility had a normalized sum of 1000, while for the synthetic instances, we did not provide any bound for the utility sum. 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.

Below, we break down the exact contents of the prompts section-by-section. Most of the prompts are very similar. The only exception is the combinatorial EF1 prompt, which takes a very different form due to it avoiding using terms from fair division. For simplicity, we show that prompt seperately at the end of this section.

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 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 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 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 EF1 prompt, which specifically instructs the LLM to find an allocation meeting the EF1 fairness criterion, the *Fairness Explanation* 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 $v_i(A_i) \geq
↪  v_i(A_j \setminus \{o\})$.
```

The Combinatorial EF1 prompt is quite different, and does not involve changing only the *Fairness Explanation* paragraph. We found it simplest to explain it by putting it in its entirety below:

```
Your task is to find a solution to the following combinatorics problem.

Given 3 functions $v_1,\dots,v_3: \{1,\dots,6\} \to \mathbb{N} \cup \{0\}$,
↪  partition $\{1,\dots,6\}$ into 3 sets $A_1,\dots,A_3$ such that $\sum_{t \in
↪  A_i}v_i(t) \ge \sum_{t \in A_j}v_i(t) - \max_{t \in A_j}v_i(t)$ for all distinct
↪  $i,j \in \{1,\dots,3\}$, where the right hand side of the inequality is treated
↪  as 0 when $A_j = \emptyset$.

$v_1,\dots,v_3$ are provided below in the following format: for each function $v_i$,
↪  there is an associated list of length {m}. For each $t \in \{1,\dots,6\}$, the
↪  $t$th entry in this list corresponds to $v_i(t)$.

-----FUNCTIONS-----
$v_1$: [1, 5, 7, 3, 4, 0]
$v_2$: [5, 9, 1, 6, 3, 3]
$v_3$: [8, 0, 2, 1, 5, 4]
-----END OF FUNCTIONS-----

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 specifying, for each $t \in \{1,\dots,{m}\}$, the unique index
↪  $i \in \{1,\dots,3\}$ for which $t \in A_i$.

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

Remember, your goal is to find a partitioning that meets the property described
↪  above.
```

A.4 JSON Formatting Instructions

When we prompt the LLMs, we use the "JSON Schema"

Due to the large amount of prompts required to complete these experiments, and output tokens being the main contributing factor to costs in each prompt, we chose to make heavy use of the "JSON Schema" functionality included in the API for `GPT` and `Claude`. Specifically, this feature allowed us to restrict the output of the LLM to return only the JSON template

containing the final allocation. Since `Gemini` did not include the same functionality in its API, the part of our prompt that talks about JSON formatting explicitly instructs the LLM to to only include the LLM to only include the final JSON template in its response. For consistency, we include this in our prompt for all models, even the ones using the JSON schema.

The paragraph that instructs the LLMs on 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.
```

A complete example of the default 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 1 to 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 number of tokens.

n	2	3	4	5	6	7	8	9	10
GPT	96400	121200	149600	181600	217200	256400	299200	345600	395600
Gemini	92961	119610	153264	191249	233997	281337	333447	390507	458992
Claude	88600	109400	133800	161800	193400	228600	267400	309800	355800

Table 1: Number of input tokens required to run 200 tests for n agent, $3n$ goods synthetic instances

n	2	3	4	5	6	7	8	9	10
GPT	7628	11268	15020	18575	22824	26135	29795	34023	37052
Gemini	9600	14400	19800	25210	30592	36000	41400	46864	52695
Claude	10200	15000	19800	24600	29400	34200	39000	43800	48600

Table 2: Number of output tokens required to run 200 tests for n agent, m goods synthetic instances

m	5	10	15	20	25
GPT	101600	141600	181600	221600	261600
Gemini	101699	145439	191249	236857	282718
Claude	197400	270400	343400	416400	489400

Table 3: Number of input tokens required to run 200 tests for 5 agents, m goods synthetic instances

m	5	10	15	20	25
GPT	6456	12525	18575	24498	30795
Gemini	8000	16205	25210	34238	43224
Claude	15056	29125	43175	57098	71395

Table 4: Number of output tokens required to run 200 tests for 5 agents, m 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

Next, in Table 7, we also record the average time it took for each of the LLMs to return a query. Note that the time an LLM takes to return a query through an API call is dependent on the traffic to the API when making the call. This can be seen clearly

in the $n = 5$ column of Table 7. These tests were run at a different time then the others, causing them to be a notable outlier to the general increasing trend of other values of n in `GPT` and `Claude`. Also in Table 7, we record the amount of time it took for each of our baseline algorithms to compute.

n	2	3	4	5	6	7	8	9	10
`GPT`	1.03	1.32	1.57	2.08	1.99	2.29	2.72	2.84	2.92
`Gemini`	1.12	1.37	1.68	2.00	2.29	2.61	2.94	3.33	3.69
`Claude`	1.62	3.49	4.84	2.69	3.73	3.76	5.39	5.73	6.55
`MNW`	0.01	0.04	0.11	0.18	0.35	0.60	0.97	1.82	2.53
`MSW`	$1.4e{-5}$	$2.4e{-5}$	$3.3e{-5}$	$4.8e{-5}$	$6.3e{-5}$	$7.9e{-5}$	$1.0e{-4}$	$1.3e{-4}$	$1.5e{-4}$
`RR`	$8.0e{-6}$	$1.3e{-5}$	$1.9e{-5}$	$2.7e{-5}$	$3.7e{-5}$	$4.4e{-5}$	$5.4e{-5}$	$6.4e{-5}$	$7.6e{-5}$

Table 7: Average runtime (in seconds) to solve a default query instance with n agents and $3n$ goods, reported for each model and algorithm.

C Additional Criteria

In this section, we provide the formal definitions of all fairness and efficiency criteria we use. Then, in the subsequent subsections, we provide all the plots corresponding to various criteria that were omitted from the main body due to the lack of space.

Fairness criteria. For an allocation A, we measure the following five fairness criteria.

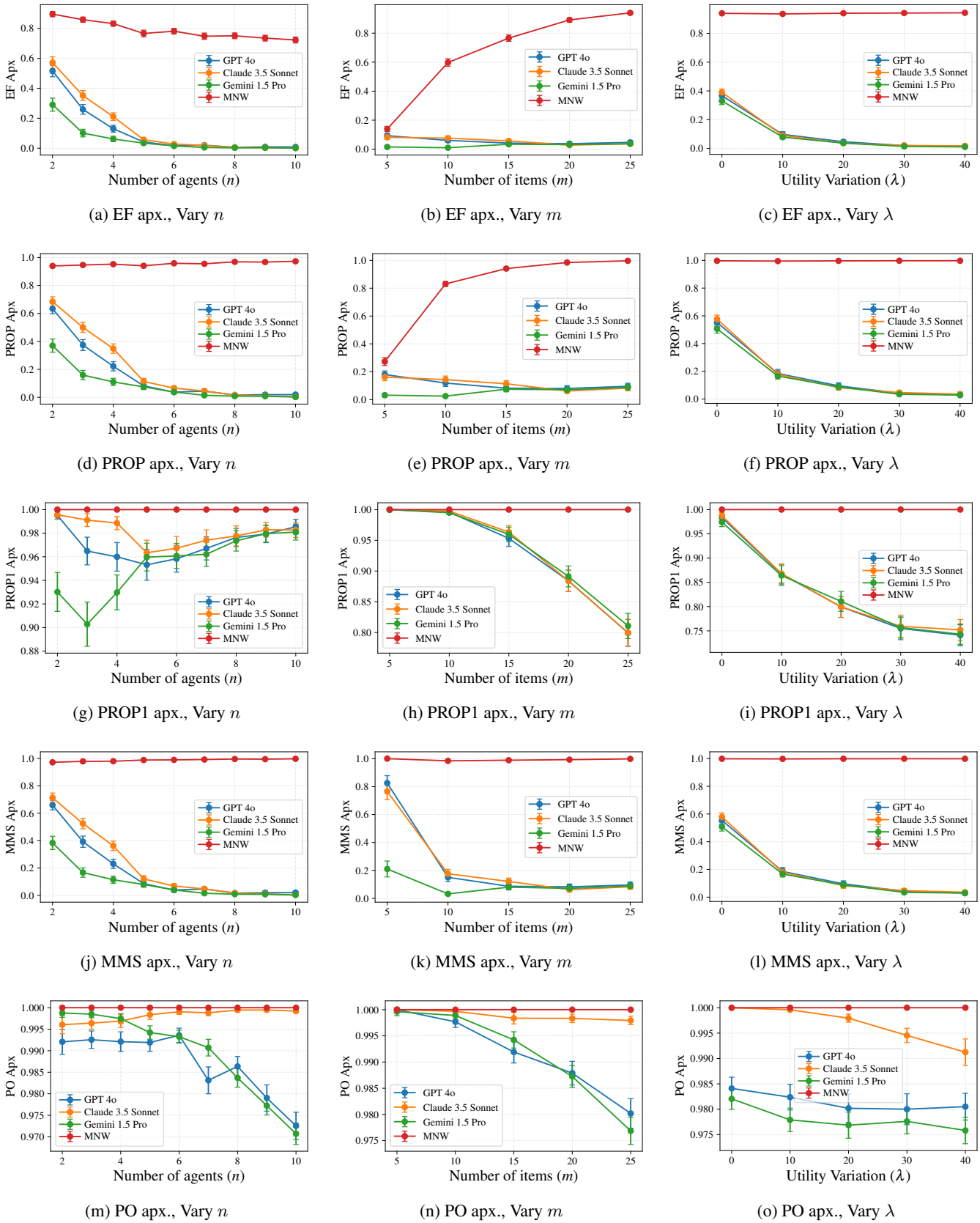
- *EF approximation:* The largest value $\alpha \in [0, 1]$ such that, for all $i, j \in N$ $v_i(A_i) \geqslant \alpha \cdot v_i(A_j)$.
- *EF1 approximation:* The largest value $\alpha \in [0, 1]$ such that, for all $i, j \in N$ with $A_j \neq \emptyset$, there exists a good $g \in A_j$ such that $v_i(A_i) \geqslant \alpha \cdot v_i(A_j \setminus \{g\})$.
- *PROP approximation:* The largest value $\alpha \in [0, 1]$ such that, for all $i \in N$ $v_i(A_i) \geqslant \alpha \cdot \frac{v_i(M)}{n}$.
- *PROP1 approximation:* 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 $v_i(A_i \cup \{g\}) \geqslant \alpha \cdot \frac{v_i(M)}{n}$.
- *MMS approximation:* The largest value $\alpha \in [0, 1]$ such that, for all $i \in N$ and partition $B = (B_1, \ldots, B_n)$ of M into n bundles, $v_i(A_i) \geqslant \alpha \cdot \min_{k \in [n]} v_i(B_k)$.

Efficiency criteria. For an allocation A, we measure the following two efficiency criteria.

- *SW approximation:* The largest value $\alpha \in [0, 1]$ such that for all allocations B, $\sum_{i \in N} v_i(A_i) \geqslant \alpha \cdot \sum_{i \in N} v_i(B_i)$.
- *PO approximation:* The largest value $\alpha \in [0, 1]$ such that for all allocations B, there exists an agent $i \in N$ with $v_i(A_i) \geqslant \alpha \cdot v_i(B_i)$.

Above, we highlighted that EF1 is a stronger criterion than PROP1. In the plots in this section, one may notice the PROP1 approximations are especially high for all models compared to any other fairness criterion. This is because when an agent's maximum value for any object is at least an α-fraction of its proportionality share, α-PROP1 is "free" in that every feasible allocation is α-PROP1. In the synthetic instances sampled from our distribution, this occurs frequently with a very high value of α (e.g., more than 90% of the instances we generated satisfied this condition with $\alpha \geqslant 0.7$). For this reason, we believe PROP1 is a less interesting fairness metric for evaluating allocations returned by LLMs, at least for our class of synthetic valuations. However, we still include the PROP1 approximation plots for completeness.

C.1 Default Prompt Plots, Synthetic Data



(a) EF apx., Vary n (b) EF apx., Vary m (c) EF apx., Vary λ

(d) PROP apx., Vary n (e) PROP apx., Vary m (f) PROP apx., Vary λ

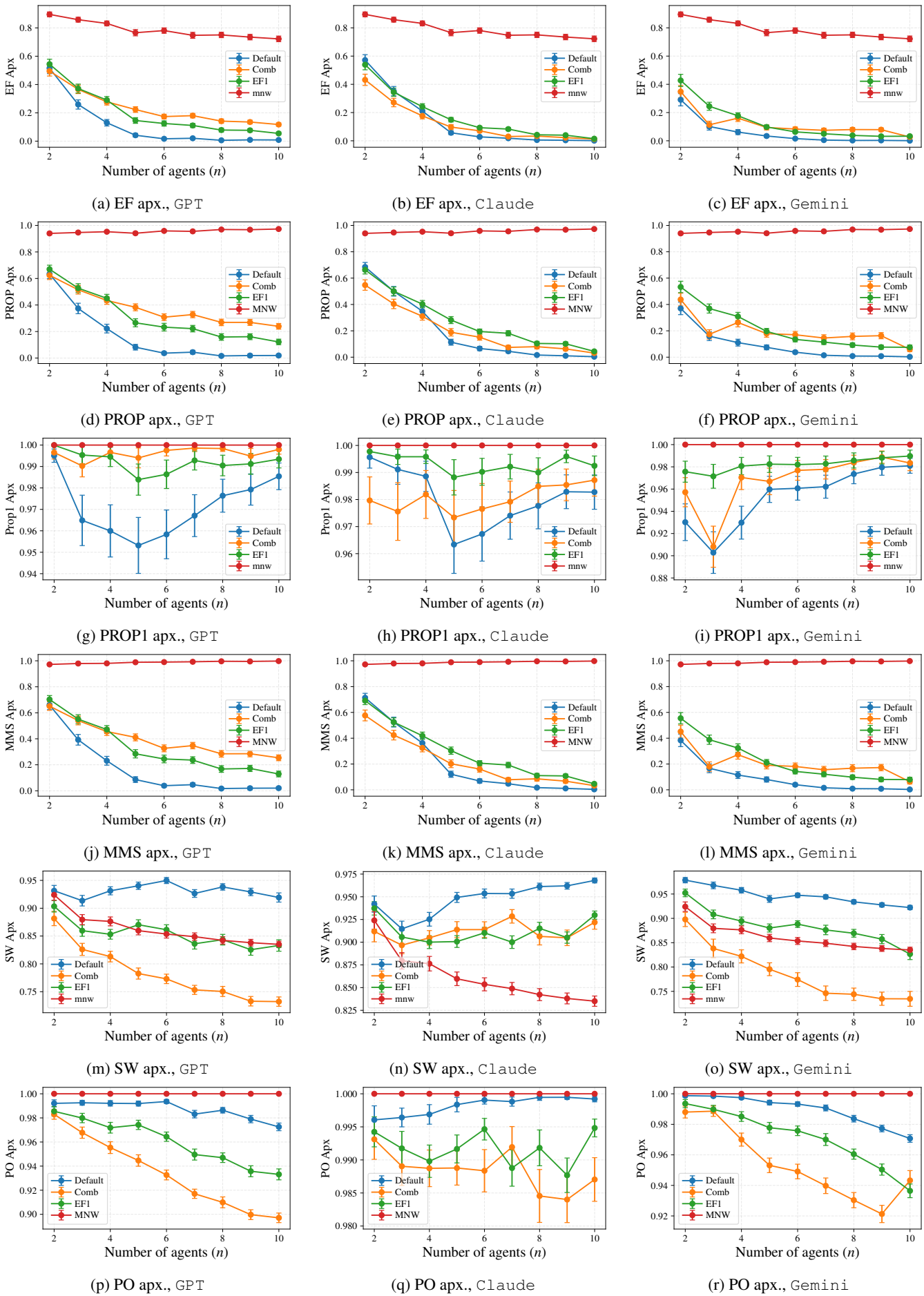
(g) PROP1 apx., Vary n (h) PROP1 apx., Vary m (i) PROP1 apx., Vary λ

(j) MMS apx., Vary n (k) MMS apx., Vary m (l) MMS apx., Vary λ

(m) PO apx., Vary n (n) PO apx., Vary m (o) PO apx., Vary λ

Figure 8: Comparison of models for the default prompt by varying n, m, or λ.

C.2 Varying Goals Plots, Synthetic Data



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

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

(g) PROP1 apx., `GPT` (h) PROP1 apx., `Claude` (i) PROP1 apx., `Gemini`

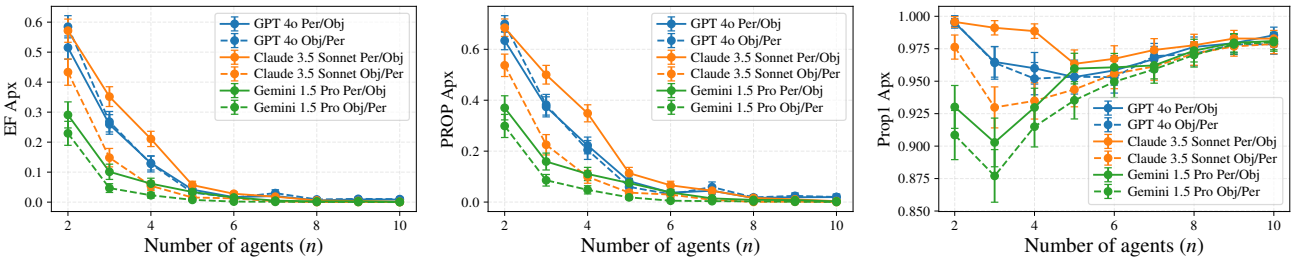
(j) MMS apx., `GPT` (k) MMS apx., `Claude` (l) MMS apx., `Gemini`

(m) SW apx., `GPT` (n) SW apx., `Claude` (o) SW apx., `Gemini`

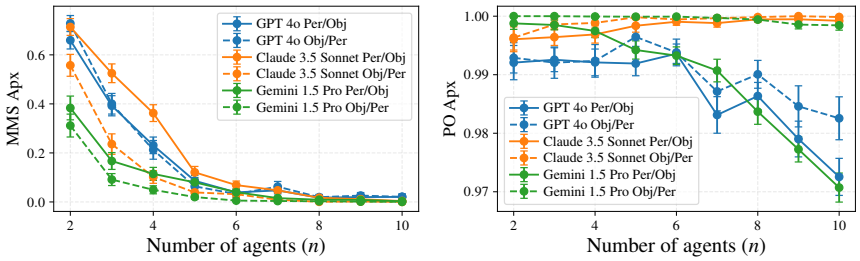
(p) PO apx., `GPT` (q) PO apx., `Claude` (r) PO apx., `Gemini`

Figure 9: Comparison of models based on varying goals with $m = 3n$ and $\lambda = 20$.

C.3 Varying Input Valuation Framing Results, Synthetic Data



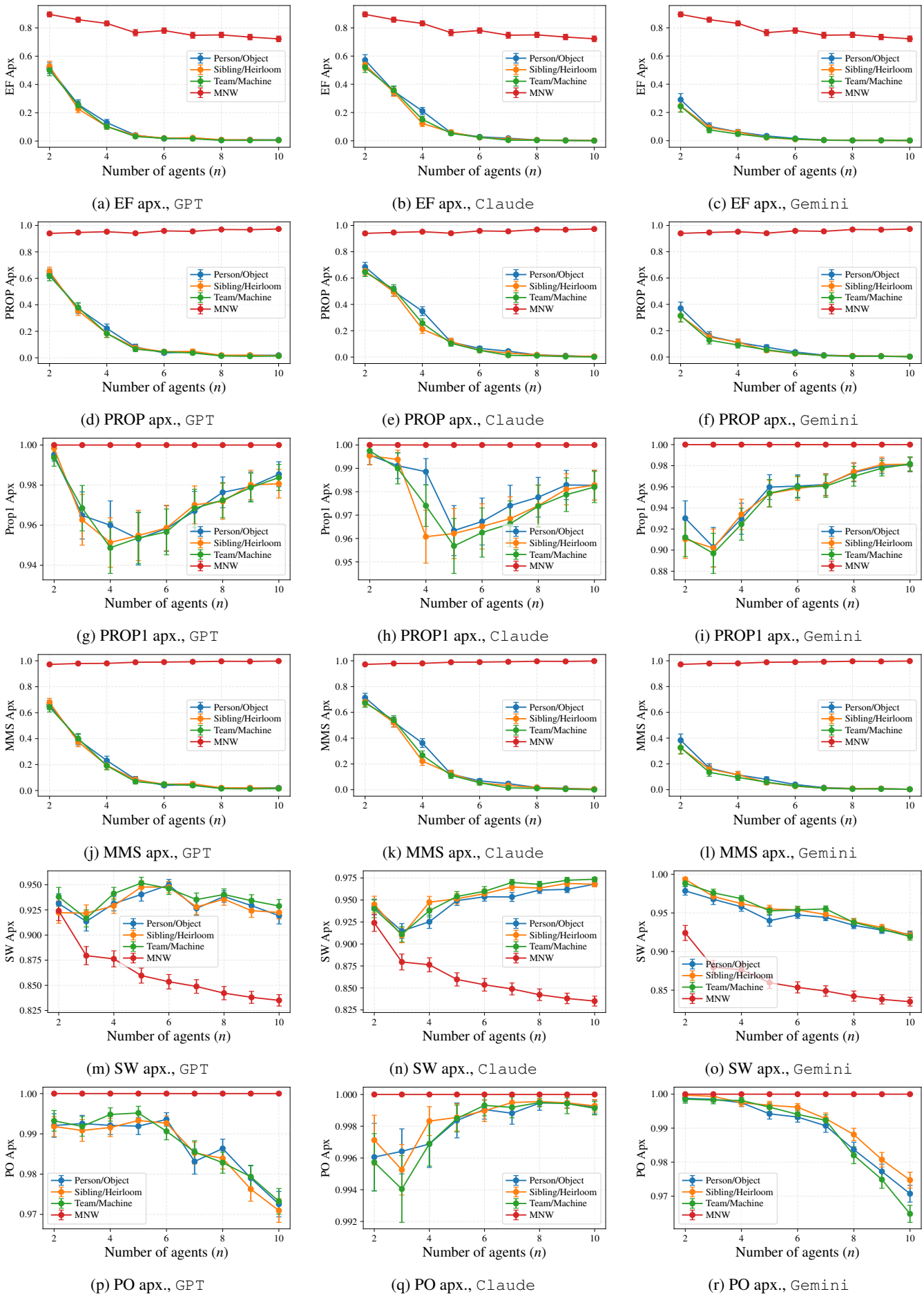
(a) EF apx. (b) PROP apx. (c) PROP1 apx.



(d) MMS apx. (e) PO apx.

Figure 10: Comparison of models under different input valuation framings with $m = 3n$ and $\lambda = 20$.

C.4 Varying Context Plots, Synthetic Data



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

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

(g) PROP1 apx., `GPT` (h) PROP1 apx., `Claude` (i) PROP1 apx., `Gemini`

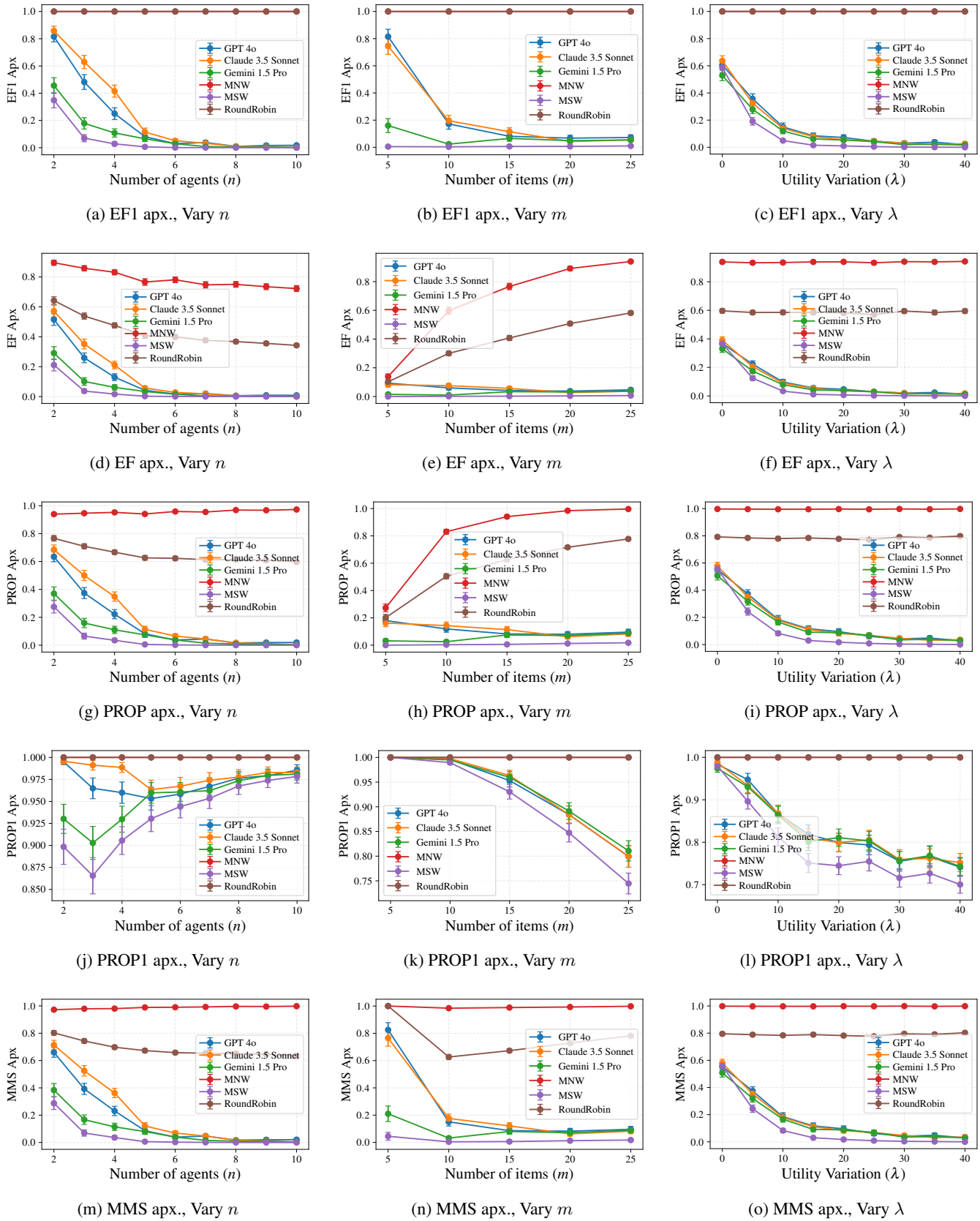
(j) MMS apx., `GPT` (k) MMS apx., `Claude` (l) MMS apx., `Gemini`

(m) SW apx., `GPT` (n) SW apx., `Claude` (o) SW apx., `Gemini`

(p) PO apx., `GPT` (q) PO apx., `Claude` (r) PO apx., `Gemini`

Figure 11: Comparison of models based on varying context with $m = 3n$ and $\lambda = 20$.

D Comparison of Models with Additional Baselines



(a) EF1 apx., Vary n (b) EF1 apx., Vary m (c) EF1 apx., Vary λ

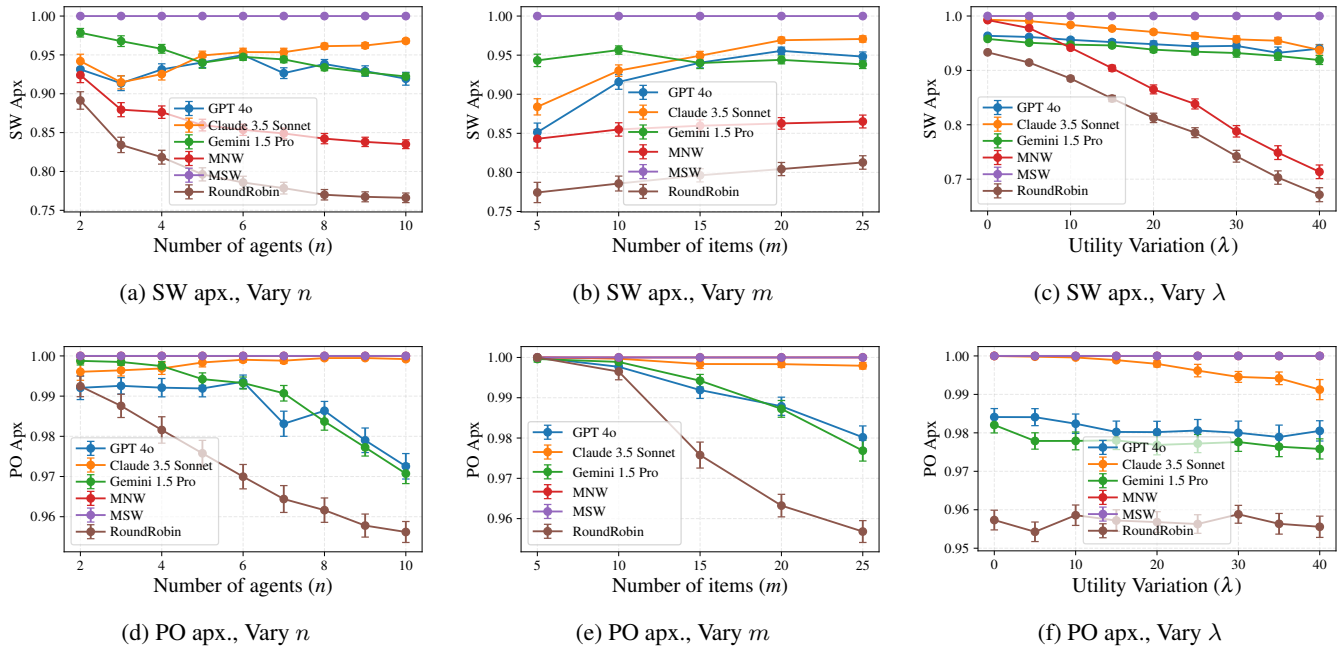
(d) EF apx., Vary n (e) EF apx., Vary m (f) EF apx., Vary λ

(g) PROP apx., Vary n (h) PROP apx., Vary m (i) PROP apx., Vary λ

(j) PROP1 apx., Vary n (k) PROP1 apx., Vary m (l) PROP1 apx., Vary λ

(m) MMS apx., Vary n (n) MMS apx., Vary m (o) MMS apx., Vary λ

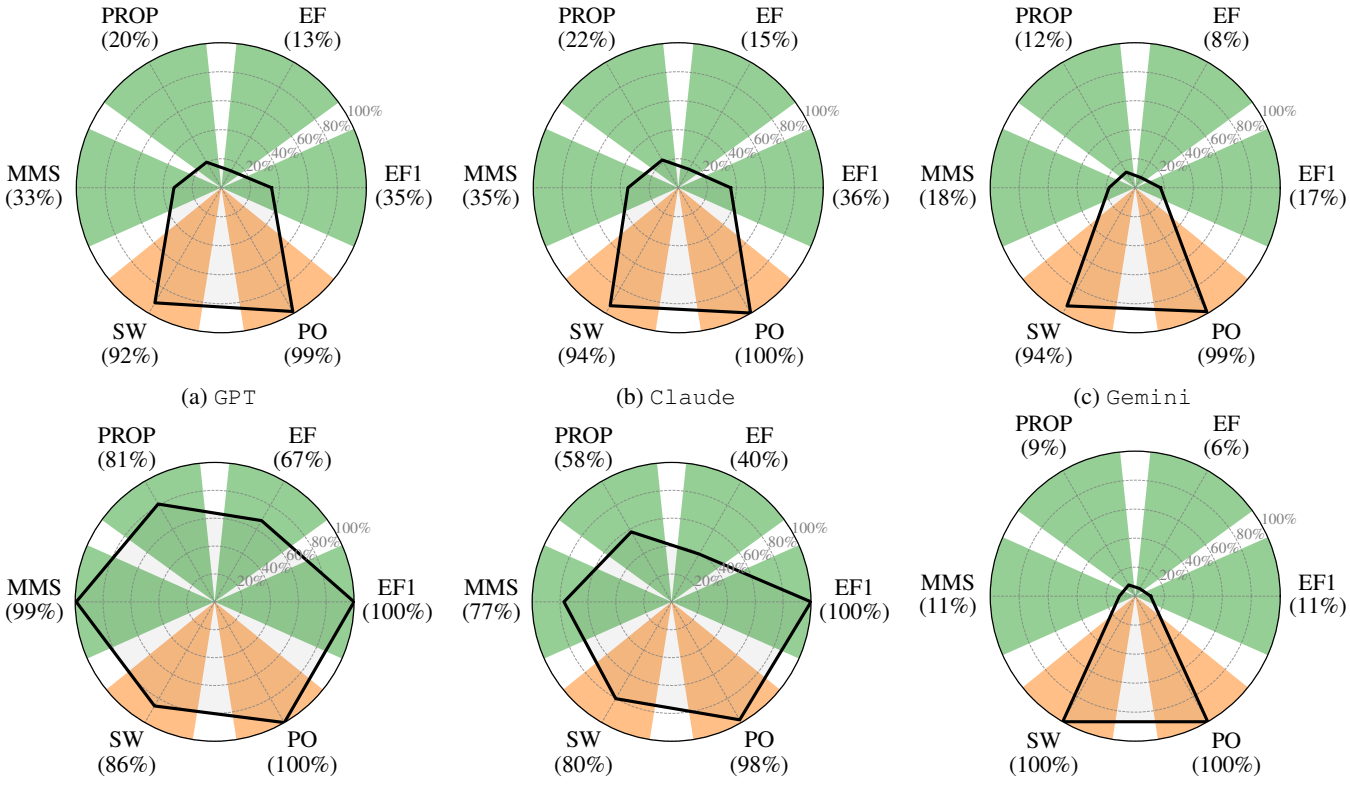
Figure 12: Comparison of models and algorithms based on fairness criteria for the default prompt by varying n, m, or λ.



(a) SW apx., Vary n (b) SW apx., Vary m (c) SW apx., Vary λ

(d) PO apx., Vary n (e) PO apx., Vary m (f) PO apx., Vary λ

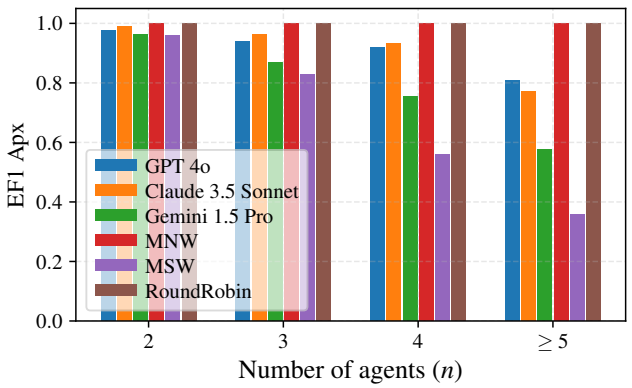
Figure 13: Comparison of models and algorithms based on efficiency criteria for the default prompt by varying n, m, or λ.



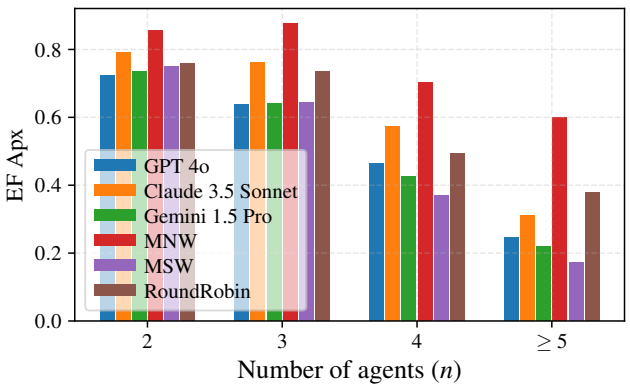
(a) GPT (b) Claude (c) Gemini

(d) MNW (e) RR (f) MSW

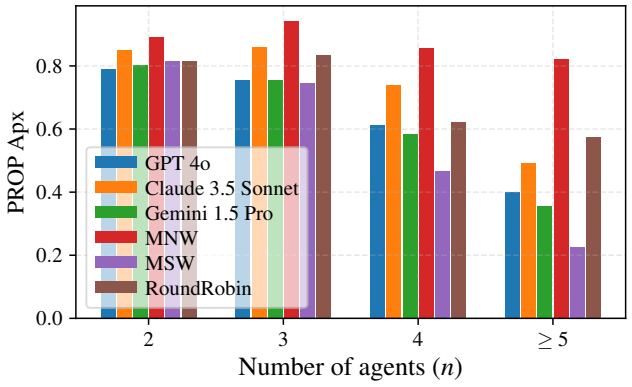
Figure 14: Radar charts showing average approximation performance of LLMs and the baselines across fairness (green) and efficiency (orange) metrics. Each axis corresponds to a criterion, with higher values (closer to the outer edge) indicating better approximation to that metric.



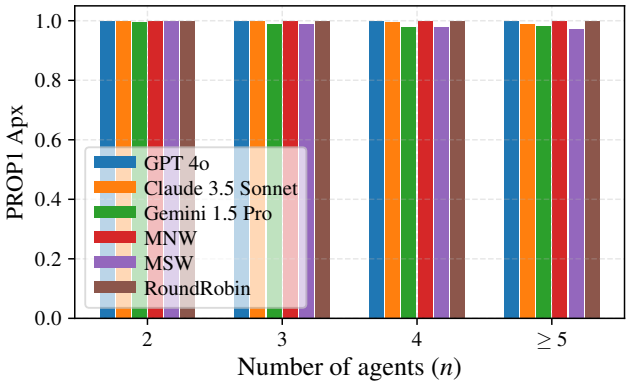
(a) EF apx.



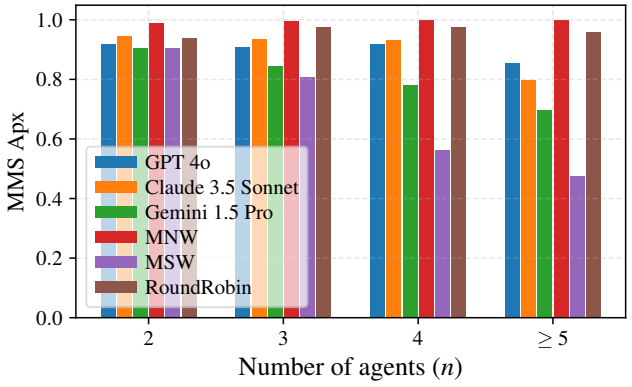
(b) EF apx.



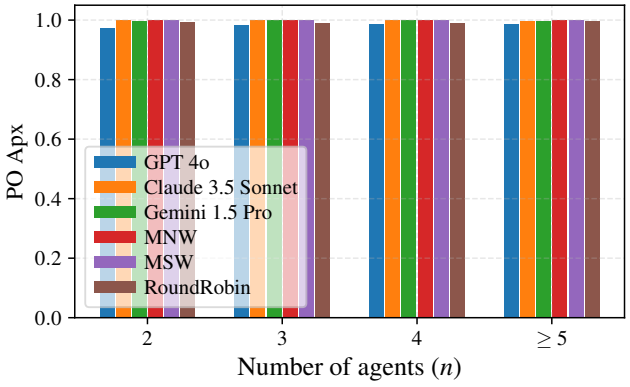
(c) PROP apx.



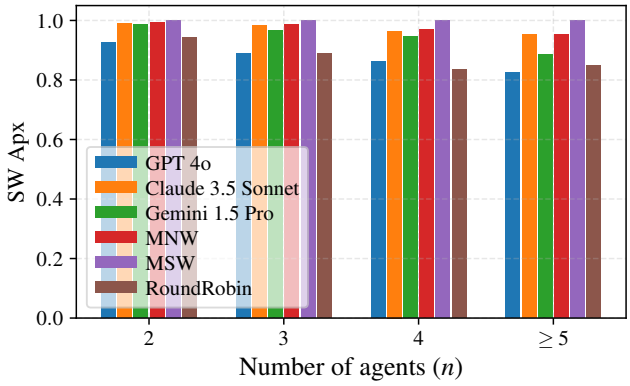
(d) PROP1 apx.



(e) MMS apx.



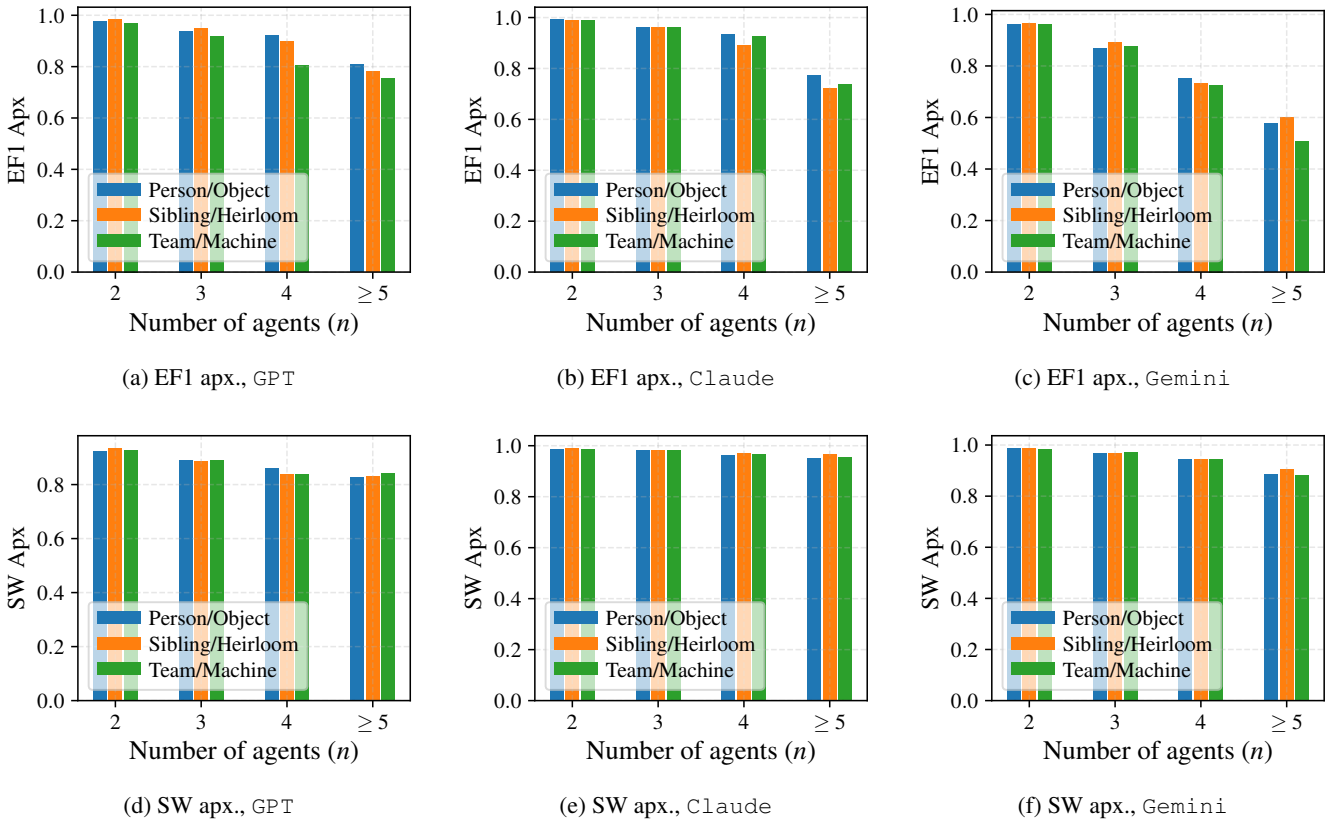
(f) PO apx.



(g) SW apx.

Figure 15: Comparison of models and algorithms with the default problem on Spliddit instances.

E.2 Varying Context

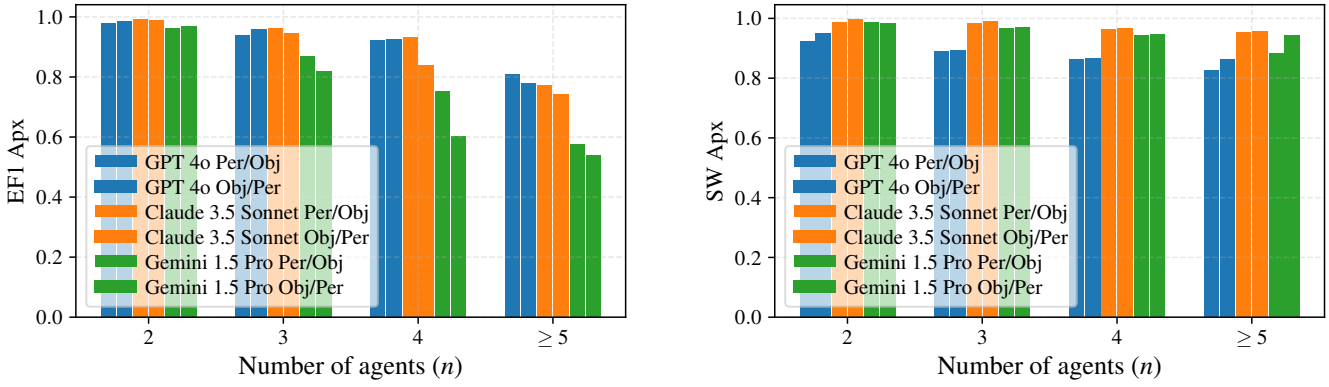


(a) EF1 apx., `GPT` (b) EF1 apx., `Claude` (c) EF1 apx., `Gemini`

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

Figure 16: Comparison of models based on varying context with the Spliddit instances.

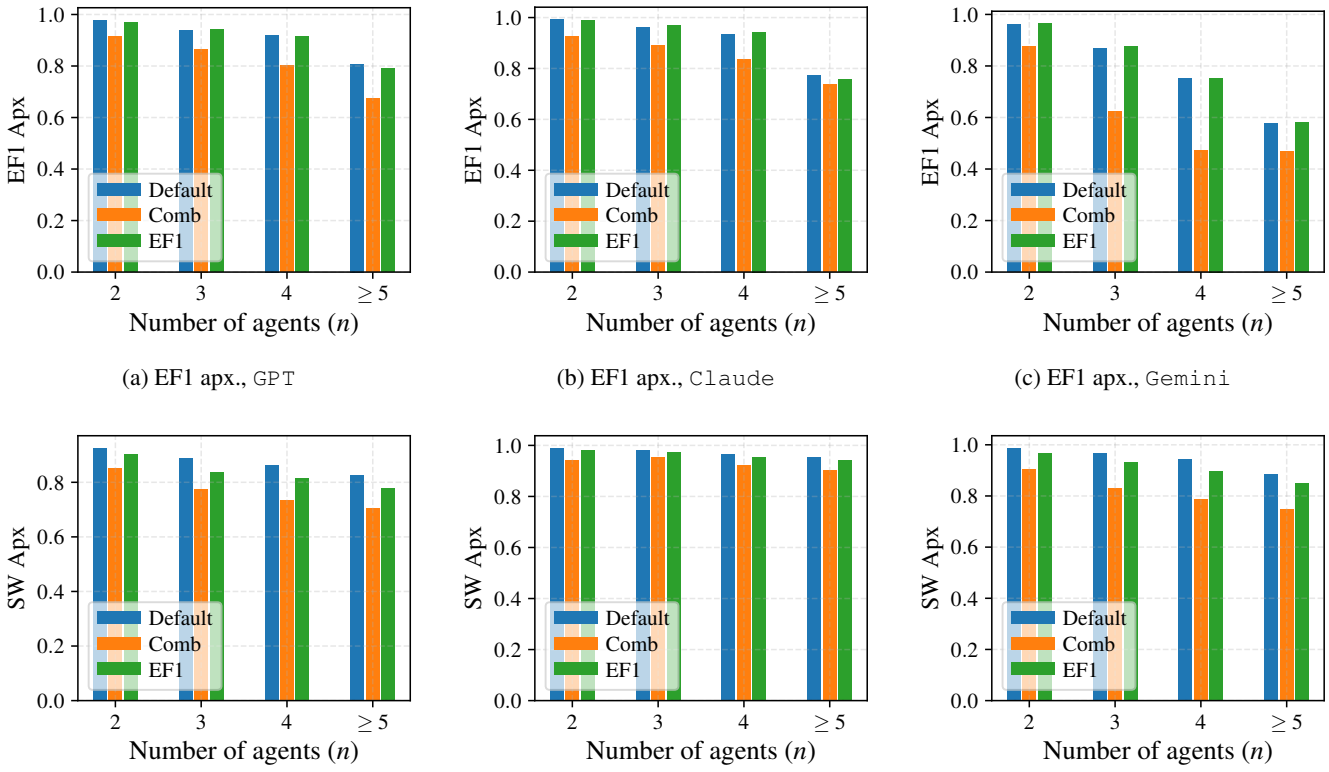
E.3 Varying Input Valuation Framing



(a) EF1 apx. (b) SW apx.

Figure 17: Comparison of models under different input valuation framings with Spliddit instances.

E.4 Varying Goals



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

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

Figure 18: Comparison of models based on varying goals with the Spliddit instances.

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.

	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

	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

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 x for both of the 2 goods, and Agent 2 had a utility of y for both goods. The goal for this set of experiments was to create a controlled environment where finding the "correct" way to 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 social 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 the agent with the highest utility, and from that infer how different models interpret fairness. These results are shown in Table 8, Table 9, and Table 10.

Results. Looking at Table 8, Table 9, and Table 10 help expand on the natural fairness-efficienct 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 in term of the fairness-efficiency trade-offs we are viewing.

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.