Balancing Act: Prioritization Strategies for LLM-Designed Restless Bandit Rewards

Shresth Verma*[1], Niclas Boehmer*[1,2], Lingkai Kong[1], and Milind Tambe[1]

[1] Harvard University, USA
[2] Hasso Plattner Institute, Germany

Abstract. LLMs are increasingly used to design reward functions based on human preferences in multiagent Reinforcement Learning (RL). We focus on LLM-designed rewards for Restless Multi-Armed Bandits, a framework for allocating limited resources among agents. In applications such as public health, this approach empowers grassroots health workers to tailor automated allocation decisions to community needs. In the presence of multiple agents, altering the reward function based on human preferences can impact subpopulations very differently, leading to complex tradeoffs and a multi-objective resource allocation problem. We are the first to present a principled method termed *Social Choice Language Model* for dealing with these tradeoffs for LLM-designed rewards for multiagent planners in general and restless bandits in particular. The novel part of our model is a transparent and configurable selection component, called an *adjudicator*, external to the LLM that controls complex tradeoffs via a user-selected social welfare function. Our experiments demonstrate that our model reliably selects more effective, aligned, and balanced reward functions compared to purely LLM-based approaches.

Keywords: Restless Bandits, Multi-Objective RL, Mobile Health

1 Introduction

Reward functions play a fundamental role in the generation of optimal policies for sequential decision-making via reinforcement learning. Previous work has shown that LLMs are an effective tool for designing reward functions that can be guided and customized via human language prompts (15; 7; 14; 37; 16; 39; 11). Focusing on optimization and planning scenarios, we study the problem of designing high-quality reward functions aligned with human preference prompts in a *multiagent* context, rendering the underlying problem inherently multi-objective. We present a transparent framework around LLMs that constructs effective, aligned, and balanced reward functions for complex human prompts.

We study the reward design problem for restless multi-armed bandits (RMABs), a popular model in multiagent systems for sequentially allocating a limited number of resources to a set of agents (36; 22). In RMABs, there are

*These authors contributed equally

multiple, independently evolving agents, with each agent being represented by an individual Markov Decision Process including a reward function. By choosing these reward functions, one can control which agents are more or less likely to receive a resource. RMABs have been applied to multiagent problems in various domains such as machine maintenance (1), anti-poaching (25), and healthcare (5; 34). In many of them, system organizers have evolving allocation priorities based on agents' features that need to be incorporated into the resource allocation process (9; 35). For instance, in a healthcare program, a healthcare worker might want to change the allocation policy to prioritize low-income beneficiaries who are at higher risk or older beneficiaries who have transportation barriers for healthcare access (21; 29) via the following preference prompt: *Prioritize low-income beneficiaries and older beneficiaries.*

Unfortunately, handcrafting reward functions is often a challenging and time-consuming task for humans because of the complex relationship between reward functions and policy outcomes (11; 7; 15). Further, the multiagent nature of the RMAB problem adds a new twist to the problem of reward design in RL: It becomes fundamentally *multi-objective*. Consider the above example prompt asking for the prioritization of two subpopulations. As these subpopulations may contain different agents, selecting a reward function will most likely involve trading off the interests of the low-income vs. older beneficiaries, making this a multi-objective problem. If this multi-objective nature is ignored, a selected reward function might heavily favor one of the two groups (e.g., leading to the allocation of many resources to low-income beneficiaries, and no resources to older ones).

This problem of multi-objective reward function modification even extends beyond Restless Multi-Armed Bandits, to challenges relevant in using Game Theory and AI for security, for instance, in Stackelberg Security Games (SSGs) applications (30). In SSGs, optimal security strategies are highly sensitive to the defined payoff functions, which typically reflect the defender's and attacker's objectives. Using a natural language interface offers defenders to refine and balance these payoff functions when faced with conflicting objectives can act as a powerful tool. This is in contrast with traditional iterative and manual adjustments presented in prior SSG applications. For instance, the ARMOR system deployed at LAX airport for security resource allocation (24) originally included a graphical user interface to enable payoff adjustments, but this interface was rudimentary. A natural language interface, as proposed in our work, could significantly enhance the practical utility of SSGs by allowing security personnel to express nuanced objectives (e.g., "prioritize slightly more air marshals on routes to Tokyo and Paris" using a system like IRIS (31)) directly, and have these preferences be translated into balanced and effective security strategies. This capability addresses a critical need for more flexible and human-centric control over complex security resource allocation.

To our knowledge, we are the first to address the multi-objective nature of LLM-powered reward design with application in RMABs in particular and multiagent planners in general. Closest to our paper is the work by (6) who

proposed a fully LLM-based Decision-Language Model for RMABs to generate and select reward functions (as code) from human language prompts. However, as argued in Sections 2 and 4, the DLM model is not properly equipped to handle the multi-objective nature of the problem, as the LLM selects functions in an unpredictable, hard-to-control and sometimes (clearly) suboptimal way that does not adequately take into account and balance the different objectives.

We present a Social Choice Language Model (SCLM) that designs reward functions (as Python code) aligned with complex, multi-objective human language preferences; see Figure 3 for an overview of SCLM. Our pipeline separates the generation of candidate reward functions in the *generator* from the selection of one function in the *adjudicator*. For the generator, we use LLM-powered evolutionary search to generate a pool of reward functions (6). In the transparent and customizable adjudicator, we take a new social choice perspective to address the multi-objective nature of our problem: We create a scorer component that evaluates the quality of generated reward functions according to the different objectives (e.g., different prioritization requests). Subsequently, a social welfare function aggregates these "alignment scores" to select the best reward function. The user can select the social welfare function and thereby has additional control over the preferred trade-off between objectives, e.g., maximizing the summed vs. minimum alignment of all objectives. We show that SCLM returns high-quality reward functions even if the computed alignment scores are noisy. In our experiments, we demonstrate that SCLM leads to the selection of reward functions significantly better aligned with complex, multi-objective prompts.

Moreover, we also show how it can be used to effectively mitigate the risks of using rewards designed from human prompts: unintended effects for other agents and the ineffective allocation of resources. *Overall, SCLM combines the generative power of LLMs to design reward functions with the capabilities of social choice to handle multi-objective decision-making scenarios.*

2 Related Works

LLM-enhanced RL LLMs have emerged as a powerful tool to enhance RL. Recent work has used LLMs to generate reward functions based on natural language descriptions (15; 38; 39). For instance, (10; 8; 18; 11) shape rewards by training an RL agent to learn and complete intermediate tasks guided by language, yet focusing on very different (non-multiagent) environments.

The work of (6) is the first to present a Decision-Language Model for generating reward functions for RMABs from human prompts. The model performs a form of evolutionary search to find reward functions aligned with the given prompt in two interleaving phases: generation and reflection. In the generation phase, an LLM generates a set of reward functions. Based on reward function's performances, in the reflection phase (15; 28), the LLM selects the function best aligned with the prompt. This function is then included in the prompt for the next generation phase or returned. In contrast to our work, DLM mixes generation with selection and does not explicitly account for the multi-objective nature

of the reward design problem. Furthermore, in contrast to our work, they focus on small RMAB instances ($\sim$ 20 arms). Throughout the paper, we will use a slightly modified variant of DLM adjusted to our setting (see Appendix C.2 in the full paper (33)) as a baseline.

Multi-Objective Reinforcement Learning (MORL) Research on MORL focuses on learning policies that maximize (and balance between) multiple objective functions, typically via scalarizing the objectives into a single reward function (19) or approximating the Pareto front (27; 32; 13). In the context of multiagent systems, MORL has been used as a method to ensure the fair treatment of the individual agents (12; 41; 40). Closest to ours from these lines of work are the papers by (40) and (13). (40) uses ideas from the resource allocation literature to combine multiple objectives into a singular non-linear objective function and focuses on policy learning for such non-linear objective functions. (13) focuses on finding a set of policies that approximate the Pareto front for various sequential planning problems. However, in contrast to our paper, neither considers reward design, human natural language preference prompts and LLMs.

We refer to Appendix A in the full paper (33) for additional related work.

3 Preliminaries

An instance of Restless Multi-Armed Bandits (RMAB) is defined by a set of N arms, a time horizon T, and a budget K. We also refer to arms as agents. Each arm $i \in [N]$ is an independently evolving MDP with state space $\mathcal{S}_i$, actions $\mathcal{A}_i = \{0, 1\}$, transition function $P_i : \mathcal{S}_i \times \mathcal{A}_i \times \mathcal{S}_i \to \mathbb{R}_{\geq 0}$, and reward function $R_i : \mathcal{S}_i \to \mathbb{R}$. We refer to 1 as the *active* action corresponding to pulling the arm (i.e., allocating a resource) and 0 as the *passive* action corresponding to not pulling the arm. We focus on the popular case where each MDP consists of two states, i.e., $\mathcal{S}_i = \{0, 1\}$ for all $i \in [N]$, yet our methodology applies to MDPs with arbitrary state spaces. We refer to 0 as the *bad* and 1 as the *good* state. For each step in which an agent is in the good state, they derive a *utility* of 1, while they derive a utility of 0 in the bad state. Accordingly, agents' *default reward function* R^* is $R^*(s) = s$. We assume that there is a set of categorical features. Each arm is associated with a value of each feature. A *global reward function* is a reward function defined over features, which induces a reward function for each arm by plugging in its feature values (see Example 1).

In each step within the time horizon T, the planner observes the state of all arms and decides to pull a subset of at most K arms. As solving the RMAB problem optimally is computationally intractable (23), we make use of the very popular state-dependent Whittle index (36; 22), which given arms' reward functions tries to quantify for each state of each arm the reward gain achieved from applying the active action to the arm in this state. In the Whittle index policy Π, in each step, we compute the Whittle index for each arm (based on its current state) and pull the arms with the K highest Whittle indices. We will use it as the solution strategy in the following.

For a global reward function R, we write $\Pi(R)$ to denote the Whittle index policy for R, i.e., the Whittle index policy for the instance where each agent uses the function R after plugging in their feature values as their reward. We refer to $\Pi(R^*)$ as the *default policy*. To assess the quality of a global reward function R, we often consider the *utility feature distribution* for some feature X. This distribution shows for each value of the feature, the expected utility generated by arms with this feature value under the policy $\Pi(R)$ (see Figure 2a).

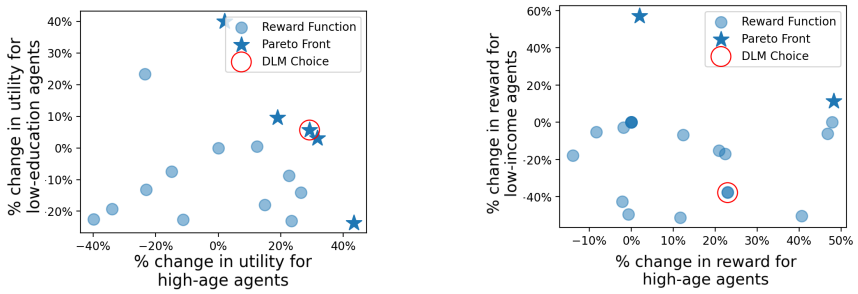
4 Problem Statement & Challenges

We assume that we are given a human-language preference prompt, concatenating one or multiple *preference clauses*. Each preference clause specifies a single optimization goal. We explicitly consider three types of preference clauses (yet our methodology extends to arbitrary ones): (i) Give priority to agents with certain feature values, i.e., increase the utility generated by these agents, (ii) do not shift the utility distribution for some feature, and (iii) maximize the summed utility generated by all agents. We mostly focus on the first type and refer to them as *prioritization clauses and prompts*. A preference prompt is a set $P = \{p_1, p_2, \dots\}$ of the involved preference clauses. We call a prompt P *singular* if $|P| = 1$ and *composite* otherwise; our focus is on the latter. We can influence the utility agents generate by selecting a single global reward function (inducing reward functions $(R_i)_{i\in[n]}$ for all agents).

Example 1. Consider an RMAB instance with three binary features A, B, and C. A preference prompt P could be "Prioritize agents with $A = 0$ and prioritize agents with $B = 1$", i.e., $P = \{$"prioritize agents with $A = 0$", "prioritize agents with $B = 1$" $\}$. Two possible global reward functions for the prompt are $R'(s) = s \cdot (1 - A) \cdot B$ and $R''(s) = s \cdot (1 - A) + s \cdot B$. For function R'', the reward of an agent i with $A = 0$ and $B = 1$ is $R_i(s) = 2s$, while the reward of an agent j with $A = 1$ and $B = 1$ is $R_j(s) = s$. Selecting R'', agent i is more likely to receive a resource than agent j, as the good state contributes more reward for i.

We want to design a single global reward function that is "well-aligned" with all clauses of the given human-language preference prompt. However, as clauses can contradict each other, perfect alignment with all clauses becomes impossible. For instance, if a prompt requests the prioritization of two fully disjoint subpopulations, each resource will only benefit one of the two. When picking the reward function, we need to carefully balance the interests of the two groups of agents against each other. Generally, in the presence of multiple agents and limited resources, each clause can be viewed as a separate independent objective that we want to optimize, rendering this a multi-objective problem.

To illustrate tradeoff decisions we face between different clauses when selecting reward function, in Figure 1, we show two instances from our experiments for a prompt consisting of two prioritization clauses. Every point represents LLM-designed reward function. The x and y axes represent quality of reward function from the perspective of the two prioritized subgroups where higher percentage



(a) Prompt: "Prioritize agents with old age and agents with low education"

(b) Prompt: "Prioritize agents with high age and agents with low income"

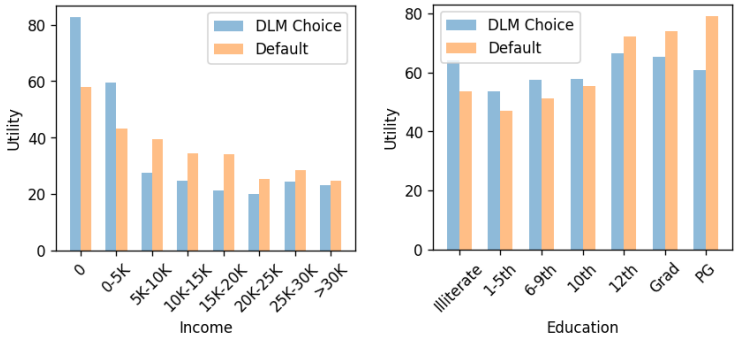
Fig. 1: Tradeoffs between prioritization clauses.

values indicate more benefits . Reward functions marked with stars lie on the Pareto fronts (no other available function dominates them).

In our experiments, we observe that the DLM model from previous work picks functions from very different parts of the Pareto frontier, potentially clearly prioritizing one subgroup over another (see Figure 1a for an example). In many other instances, it also picks suboptimal functions, i.e., functions that do not lie on the frontier, that may even harm one of the subgroups while strongly benefiting the other (see Figure 1b). This highlights the risks (and shortcomings of DLM) in not accounting for the multi-objective nature of the problem, as it picks reward functions that are inefficient (i.e., dominated) and unfair (i.e., heavily favoring one clause over the other).

Another shortcoming of DLM are unintended utility shifts. Moving from the default reward function to a reward function aligned with a given (prioritization) prompt causes shifts within the distribution of resources and utility. Due to correlations between features, this change might lead to unintended utility shifts for features not specified in the prompt. Figure 2 shows an example of this from our experiments. We present the utility feature distribution for the two features *income* and *education* for two reward functions: The reward function selected by DLM for the prompt "Prioritize agents with low income" (orange) and the default reward (blue). While the utility generated by low-income agents increases when moving from the default to the customized reward function, the utility generated by highly educated agents decreases, a side-effect the end-user might be unaware of and that might conflict with their allocation goals. In our proposed approach, we are able to account for this issue by incorporating the prevention of unintended utility shifts as a tradeoff dimension.

Thus, through the Social Choice Language Model, *our goal is to create a model that handles multiple tradeoffs posed by composite "multi-objective" prompts in a principled, transparent, and customizable fashion and outputs a single effective and fairly aligned global reward function.*



(a) Income feature (b) Education feature

Fig. 2: Utility feature distributions for default reward function (orange) and reward function returned for prompt "Prioritize agents with low income" (blue) by DLM baseline. x-axis depicts feature value and y-axis total utility generated by agents with this value.

5 Social Choice Language Model (SCLM)

We propose a Social Choice Language Model to generate rewards from human language composite preference prompts (see Figure 3 for a visualization). Separating the generation and selection of reward functions, the model consists of two sequential components. The LLM-powered *generator* generates a set of candidate reward functions. Subsequently, taking a social-choice-inspired viewpoint, the *adjudicator* selects a reward function from the pool to be returned to the user in two steps: First, a scorer model computes an alignment score for each reward function with each prioritization clause (i.e., we judge each reward function from the perspective of all relevant "objectives"). Second, a user-defined social welfare function aggregates these scores into a "winning" candidate reward function. By selecting the social welfare function, the user can control the area of the Pareto frontier from which reward functions get selected. While we remark that our model can also be used to tackle multi-objective issues arising when designing rewards in single-agent RL, the details of our components (e.g., the reflection in the generator and the computation of alignment scores) are specific to the multiagent nature of the RMAB problem.

5.1 Generator

Given a prompt, our generator creates a set of candidate reward functions (as Python code) via a variant of evolutionary search following (6): We proceed in multiple steps. First, inputting the problem description, feature descriptions and the preference prompt, we ask an LLM to generate code for a reward function. We repeat this query n_p times to obtain a set $\mathcal{R}$ of n_p candidate reward functions.

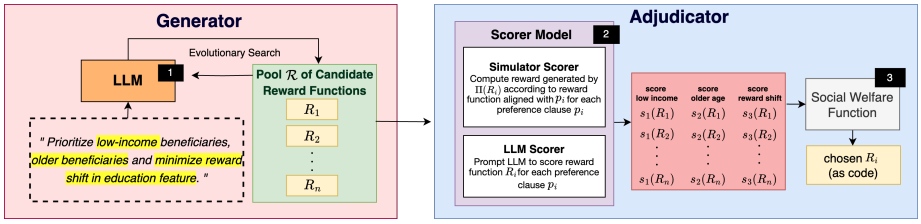


Fig. 3: In step 1, preference prompt is passed to the generator, which performs an evolutionary search to create a pool $\mathcal{R}$ of candidate reward functions. In step 2, these functions are passed to the adjudicator where a scorer model computes the alignment scores. In step 3, a user-defined social welfare function selects a reward function based on the alignment scores.

Afterwards, for each function $R \in \mathcal{R}$ we compute the utility feature distributions of the policy $\Pi(R)$ induced by the reward function R on the given RMAB instance (via repeatedly simulating the policy on the instance). Then, the prompt and the set of candidate reward functions together with the associated utility feature distributions are passed to an LLM, which is asked to select the reward function R' from $\mathcal{R}$ best aligned with the prompt (15; 28). Now, we repeat the whole process, this time including the selected policy R' as a seed in the reward function generation prompts. Once we have executed the process n_r times, we add all generated $n_p \cdot n_r$ candidate reward functions R to the pool $\mathcal{R}$ (see Appendix C.2 in the full paper (33) for details).

5.2 Adjudicator

The adjudicator selects a reward function from a given pool of candidate reward functions returned by the generator. To handle complex tradeoffs arising within composite prompts and resulting multi-objective optimization problem, the adjudicator follows a social choice approach. Social choice is a discipline at the intersection of economics, philosophy, and mathematics and concerned with aggregating potentially contradicting preferences of set of voters into fair compromise alternative from a given candidate set (4; 20). It thus provides a theoretically grounded and methodology for balancing competing interests. In our problem, we interpret reward functions as candidates and preference clauses in the prompt as voters with their preferences over the candidates reflecting reward function's alignment with the clause. This view gives rise to the following strategy: Given a prompt $P = \{p_1, p_2, \ldots, p_\ell\}$, we evaluate each reward function $R \in \mathcal{R}$ from the perspective of each preference clause p_i by computing an (alignment) score $s_i(R)$. $s_i(R)$ measures the alignment of $\Pi(R)$ with preference clause p_i, i.e., how much the voter representing p_i "likes" the candidate R.

In the following, in Section 5.2, we describe how the adjudicator selects the reward function given the scores; in Section 5.2, we describe how scores get computed; and in Section 5.2, we present a guarantee on the quality of the selected reward function if computed scores are noisy.

Selection via Social Welfare Function Social welfare functions select an alternative based on input preferences of voters. The pros and cons of individual social welfare functions have been extensively researched and debated in social choice (4; 26). The *generalized p-mean*, $f_p(\cdot) : \mathbb{R}^l_{>0} \to \mathbb{R}_{>0}$, is a rich class of social welfare functions which we consider in this work. It is defined for a given $p \in (-\infty, 1]$ and strictly positive vector $\mathbf{s}(R) = (s_1(R), \ldots, s_l(R)) \in \mathbb{R}^l_{>0}$ as follows:

$$f_p(\mathbf{s}(R)) = \begin{cases} \min_{i\in[l]} s_i(R) & \text{if } p = -\infty, \\ \left(\frac{1}{l}\sum_{i\in[l]} s_i(R)^p\right)^{1/p} & \text{if } p \notin \{-\infty, 0\}, \\ \left(\prod_{i\in[l]} s_i(R)\right)^{1/l} & \text{if } p = 0. \end{cases} \tag{1}$$

In our experiments, we consider the three arguably most popular social welfare functions, which can be written as a generalized p-mean:

Utilitarian ($p = 1$) Return the reward function maximizing the sum of its scores, i.e., $\arg\max_{R\in\mathcal{R}} \sum_{i\in[\ell]} s_i(R)$.

Nash ($p = 0$) Return the reward function maximizing the product of its scores, i.e., $\arg\max_{R\in\mathcal{R}} \prod_{i\in[\ell]} s_i(R)$.

Egalitarian ($p = -\infty$) Return the reward function maximizing its minimum score, i.e., $\arg\max_{R\in\mathcal{R}} \min_{i\in[\ell]} s_i(R)$.

Selecting the social welfare function gives us control over the tradeoffs between objectives: By picking the Egalitarian function, we ensure that one clause will not get prioritized over another. In contrast, the Utilitarian function prioritizes the summed alignment, allowing for mismatches between clauses; the Nash function strikes a balance between the two functions.[3] The adjudicator makes the selection process more transparent, as the different objectives, the selection criterion, and the performance of the candidate reward functions regarding the objectives become explicit.

Computing Alignment Scores It remains to describe how the alignment scores $s_i(R)$ are computed. We present two general methods to compute alignment scores, which we will use for prioritization clauses. Subsequently, we discuss two more customized methods for the prevention of unintended utility shifts or drops in total generated utility.

[3] Note that social welfare functions also allow for assigning a different importance to clauses: The user could submit an importance score w_i for each clause p_i, which can be easily incorporated in the social welfare function, e.g., the Utilitarian welfare function becomes $\arg\max_{R\in\mathcal{R}} \sum_{i\in[\ell]} w_i \cdot s_i(R)$.

Simulator Scorer Model (SCLM-SIM) For each preference clause $p_i \in P$, we compute a reward function R_i aligned with p_i by casting it as a singular prompt to the DLM pipeline (see Appendix C.2 in the full paper (33)). For each $R \in \mathcal{R}$, we compute as $s_i(R)$ the expected reward according to reward function R_i produced by policy $\Pi(R)$ (again, we approximate this quantity by running multiple simulations). Accordingly, $s_i(R)$ quantifies the quality of the policy induced by the candidate reward function R from the perspective of p_i (as captured by R_i). As the scale of the reward functions can vary significantly among preference clauses, we normalize the scores by the performance of the default policy, i.e., we compute $\frac{s_i(R)-s_i(R^*)}{s_i(R^*)}$.

LLM Scorer Model (SCLM-LLM) The Simulator Scorer Model assumes access to reward functions capturing individual preference clauses well. If no well-aligned reward functions can be obtained, the performance of SCLM-SIM can deteriorate because it can become noisy. Another disadvantage of SCLM-SIM is that the scores in SCLM-SIM are all computed via simulation, which can become computationally costly. Motivated by this, we propose a quicker and more flexible LLM-based approach, where we prompt an LLM to rate the alignment of a candidate reward function with a preference clause. In particular, for each $R \in \mathcal{R}$ and $p_i \in P$, we use a prompt that includes R, p_i, and the utility feature distributions produced by policy $\Pi(R)$. We ask the LLM to rank how well R aligns with the preference clause p_i on a scale from 1 to 5 (see **??** for prompt texts).

Preventing Unintended Utility Shifts and Utility Drop Aligning reward functions to a prioritization prompt may cause (unintended) utility shifts in other features (e.g., due to feature correlations, shifting utility to low-income beneficiaries might shift it away from more educated ones). See Figure 2 for a concrete example. SCLM offers users the option to explicitly prevent these shifts by adding additional clauses ("objectives") to the prompt: Given a prompt P (e.g., the prompt from Example 1), for each feature not referenced in the prompt, the user can add a new preference clause requesting a minimum shift in the utility distribution of this feature (e.g., for Example 1 they could add "do not change the utility distribution for feature C"). To compute the alignment score $s_i(R)$ between a reward function R and a clause p_i="minimize utility shift for feature X", we compare feature X's utility distribution under the default policy with its utility distribution under the policy $\Pi(R)$. Specifically, we quantify the difference using the Earth mover's distance (EMD) between the two distributions. Afterward, we apply 0-1 normalization to all scores $s_i(R)_{R\in\mathcal{R}}$ for prompt p_i, which are input to the social welfare function (along with the alignment scores for the other clauses).

Another potential risk of aligning a reward function with a prioritization prompt is that it can sharply decrease the summed utility generated by all agents: The user might request the prioritization of a subpopulation that does not benefit much from receiving a resource, leading to severe drops in the summed utility generated by all agents. Users can address this issue in our model by adding a

clause p_i="maximize the total generated utility" to the prompt. As the alignment score $s_i(R)$ of p_i with some reward function R we compute the summed utility, i.e., the total number of steps in which arms are in an active state, generated by all agents under the policy $\Pi(R)$ (computed via multiple simulations of the policy on the given instance). We again apply 0-1 normalization to all scores $s_i(R)_{R\in\mathcal{R}}$ for prompt p_i.

Error Bounds for Adjudicator's Selection Even though we observe in our experiments that the scorer models produce mostly accurate scores, the output scores are oftentimes still a bit noisy. To measure how errors propagate through the Social Choice Language Model and how they affect the final reward function selection, we consider the following setup.

Suppose instead of observing the true score vector $\mathbf{s}(R_j) = (s_1(R_j), s_2(R_j), ..., s_l(R_j))$, the Scorer Model (SCLM-SIM or SCLM-LLM) returns a noisy score estimate $\tilde{\mathbf{s}}(R_j)$ with multiplicative noise $\alpha \in (0, 1]$ satisfying

$$\alpha \cdot \mathbf{s}(R_j) \leq \tilde{\mathbf{s}}(R_j) \leq \frac{1}{\alpha} \cdot \mathbf{s}(R_j), \quad \forall R_j \in \mathcal{R}. \tag{2}$$

Let $\tilde{R}^* = \arg\max_{R_j\in\mathcal{R}} f_p(\tilde{\mathbf{s}}(R_j))$ be the best reward function under the generalized p-mean function for the observed, noisy scores ($\tilde{R}^*$ will be returned by SCLM); and $R^* = \arg\max_{R_j\in\mathcal{R}} f_p(\mathbf{s}(R_j))$ be the best reward function under the generalized p-mean function for the true, latent score. We define the (relative) regret we encounter by choosing $\tilde{R}^*$ instead of R^* as:

$$\textit{Relative Regret} = \frac{f_p(\mathbf{s}(R^*)) - f_p(\mathbf{s}(\tilde{R}^*))}{f_p(\mathbf{s}(R^*))} \tag{3}$$

The relative regret measures the relative drop in p-mean welfare of the reward function chosen by the adjudicator as compared to the optimal reward function. We show that the relative regret degrades gracefully in the multiplicative error parameter, highlighting that even in the presence of noise, SCLM selects good reward functions with guarantees.

Proposition 1. *The relative regret is bounded by* $1 - \alpha^2$.

Proof sketch. We observe the monotonicity and positive homogeneity of the generalized p-mean function applied to the input values. These properties allow the application of function f to both sides of Inequality 2. Subsequently, using the definition of R^* and $\tilde{R}^*$, we derive the regret bound. For a complete proof, see Appendix D in the full paper (33).

6 Experiments

We describe our testset (Section 6.1), the compared methods (Section 6.2), and our experimental results both for dealing with composite prioritization prompts

(Section 6.3) and additionally minimizing unintended side effects (Section 6.4). Following the work of (6), which constitutes our most important baseline, we use Gemini Pro (2) as the LLM in our experiments.

6.1 Dataset Description

ARMMAN (3) is a non-profit in India that operates large-scale Maternal and Child Care Mobile Health programs for underserved communities. One of their programs disseminates critical health information via weekly automated voice messages. The goal of the NGO is to maximize beneficiaries' engagement, i.e., the number of messages they listen to. A limited number of beneficiaries are called by health workers every week to boost engagement. The problem of planning which beneficiaries to call has been modeled and solved as an RMAB, where the good/bad state corresponds to a high/low weekly engagement of the beneficiary. We use anonymized data from a quality improvement study conducted in January 2022 (34). For each beneficiary, we have access to their income, education, and age level, which we use as our three features. Beneficiaries' historic listenership values are used to estimate their transition probabilities under the passive action (17). One problem in estimating transition probabilities under the active action is that due to the limited number of service calls made, such transitions are rare. Thus, active transition probability estimates are noisy. To alleviate this issue, we use the features and passive transition probabilities from ARMMAN together with synthetically generated active transition probabilities. Finally, we create three datasets, each consisting of five sampled RMAB instances with $N = 2100$ arms, a budget of $B = 210$ and a time horizon of $T = 12$. The three datasets differ in how much each feature impacts the effect of applying an active action. In addition to the real-world domain, we also create three completely synthetic domain datasets (see Appendix B.3 and B.4 in the full paper (33) for more details on dataset generation).

Problem Instances Instances of our problem consist of two parts: A preference prompt and an RMAB instance. We initially focus on prioritization prompts. Specifically, for each feature X, we consider two different prioritization clauses "Prioritize agents with low/high value of feature X". This gives rise to 6 singular prompts consisting of one prioritization clause, two for each feature. For composite prompts, we take all combinations of two features and the two prioritization clauses for each feature (e.g. "Prioritize agents with high value of feature A and also prioritize agents with low value of feature B"). This results in $3 \cdot 4 = 12$ composite prompts. For each domain, we run each prompt on the 15 RMAB instances from the three datasets.

6.2 Models & Baselines

We analyze six different variants of SCLM differing in the used social welfare function (Utilitarian, Egalitarian, Nash) and scorer model (Simulator or LLM), e.g., we denote as *SCLM-SIM-Egalitarian* SCLM with the Simulator Scorer

Model and the Egalitarian social welfare function. In our generator, we generate 4 candidate reward functions in each step and run 5 iterations to generate a total of 20 candidate reward functions. In addition, we consider several LLM-focused baselines (see Appendix F in the full paper (33) for detailed descriptions):

LLM-Zeroshot This baseline only queries the LLM once. It asks to return a reward function aligned with the given preference prompt and provides the problem and feature description as additional context in the prompt.

DLM This baseline implements the Decision-Language Model by Behari et al. (6) (see Appendix C.2 in the full paper (33)).

DLM-PromptEngg This is a modified version of DLM where within the reflection prompt, we include examples for singular queries of how the LLM should reason over the different reward function choices (see Appendix F in the full paper (33)).

6.3 Results: Composite Prioritization Prompts

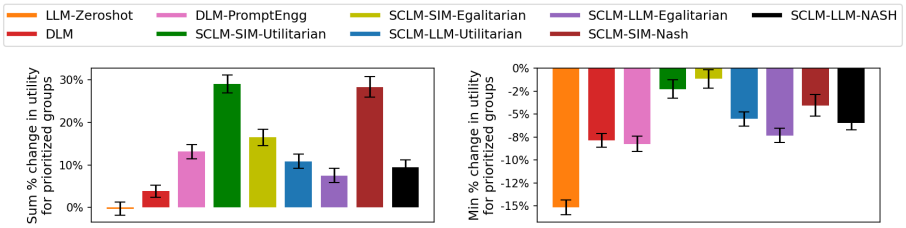
We analyze the performance on the 12 composite prompts described above which request the prioritization of two subpopulations (see Appendix E.1 in the full paper (33) for additional results).

Evaluation Metrics As our goal is to fulfill the preferences specified by the user (in contrast to the classic goal of maximizing total utility), we need to quantify the alignment of the returned reward function with the given prompt P to evaluate our models. Due to the composite, multi-objective nature of our prompts, we start by measuring the alignment of the returned reward function R with each prioritization clause $p_i \in P$ in a separate evaluation score $e_i(R)$. For this, we need to quantify how well a given reward function prioritizes the subpopulation specified in the prompt. However, as our prompts are written in human language, these subpopulations are not precisely defined (as the prompts only speak of agents with "high"/"low" value of some feature X). Notably, one could think that the scores $s_i(R)$ computed in our adjudicator could be used as our evaluation scores $e_i(R)$, as they measure how well a reward R aligns with a prioritization clause p_i. However, this would create an unfair competitive advantage for the SCLM compared to our baselines who do not have access to these scores.

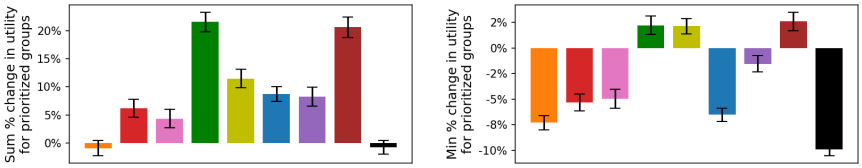
Instead, we assume that the terms "low" and "high" in the input prompts refer to the most extreme feature values. Let p_i be some prompt prioritizing agents with a high/low value of some feature X. As the evaluation score $e_i(R)$, we compute the summed utility generated by the agents with highest/lowest value of X under the policy $\Pi(R)$ normalized by the utility generated by these agents under the default policy $\Pi(R^*)$.[4] Reflecting the multi-objective nature of our problem, we consider two metrics for measuring the alignment of a reward R with a full composite prompt: sum and minimum of % change of the utility

[4] In Appendix E.1 in the full paper (33), we also check how the results change if we instead interpret "low"/"high" to refer to the lowest/highest two or three values. We observe very similar trends.

generated by the two prioritized groups under policy $\Pi(R)$ compared to the default policy, i.e., the sum (resp. minimum) of the evaluation scores for R.



(a) Synthetic domain: sum % change (left) and minimum of % change (right) in utility for the two groups prioritized.



(b) Real-world domain: sum % change (left) and minimum of % change in utility for the two groups prioritized.

Fig. 4: Results comparing the quality of reward design methods for composite prioritization prompts. Results are averaged across $180 = 12 \cdot 15$ values: 12 composite prompts on 15 RMAB instances (from 3 datasets). Error bars represent std-error.

Results In Figure 4, we show the averaged results from the synthetic and real-world domain. We depict the average summed and minimum alignment with the two clauses of the composite prompt, i.e., the minimum/summed change in the utility generated by the prioritized group of agents.

We start by focusing on SCLM with Simulator Scorer *SCLM-SIM* (green-shaded bars), our strongest method. On both domains, *SCLM-SIM* significantly outperforms all baselines for both minimum and summed % change independent of whether the Utilitarian, Egalitarian or Nash social welfare function is chosen. *SCLM-SIM-Utilitarian* outperforming the baselines for the minimum change and *SCLM-SIM-Egalitarian* outperforming them for the summed change highlights the advantages of the SCLM, as these objectives are not explicitly optimized by the respective models, e.g., *SCLM-SIM-Utilitarian* aims at maximizing the summed change and not the minimum one, but still performs well regarding the minimum change. This indicates that SCLM independent of the chosen welfare function does a better job at picking effective and aligned reward func-

tions (on the Pareto front). Comparing *SCLM-SIM-Utilitarian* and *SCLM-SIM-Egalitarian*, the two methods exhibit a big difference under the summed change criterion, while the difference regarding the minimum change is much smaller. Examining individual instances we find in Appendix E.1 in the full paper (33), that both functions lead to very different selections on the instance level; unsurprisingly, the Egalitarian method creates rewards that benefit both groups in a more balanced fashion. For the Nash welfare function, we found that the performance was similar, yet slightly inferior to the Utilitarian welfare function in all relevant evaluation dimensions.

If we replace Simulation Scorer with LLM Scorer, performance of SCLM decreases, but is better than all of our three baselines. The difference between LLM and Simulation Scorer highlights the advantage of the additional information acquired through more complex and computationally expensive simulation method. Regarding the performance of baselines, our DLM baseline with prompt engineering *DLM-PromptEngg* improves slightly upon the results of DLM in the synthetic domain, while in real-world domain, their performance is similar. This suggests that prompt engineering itself is not sufficient to adequately deal with the multi-objective nature of composite prompts; an external component (like our adjudicator) is needed. Finally, LLM-zeroshot consistently performs the worst, which highlights the non-triviality of reward design problem and the need for a guided extensive search within reward function search space.

Table 1: Results comparing different reward function selection strategies, aggregated across three real-world datasets. Higher summed % change in desired feature(s) implies better alignment with prioritization clauses, whereas less unintended shift and less % drop in utility are better.

Method	**Summed % Change in Desired Feature(s)**	**Unintended Shift**
DLM-PrioritizationOnly	6.809±0.86	0.302±0.02
DLM-ExtendedPrompt-Fair	-0.254±0.73	0.276±0.02
SCLM-PrioritizationOnly	13.131±0.86	0.316±0.02
SCLM-ExtendedPrompt-Fair	15.364±0.94	0.099±0.01

6.4 Addressing Fairness and Biases

As discussed in Section 5.2, we can also use our pipeline to prevent unintended side-effects of aligning reward functions with prioritization clauses, i.e., (i) shifts in the utility feature distribution of features not included in the prompt, (ii) drops in the total generated utility, and (iii) arbitrary weighted combinations of the above two goals and additional preference clauses. We focus on 1 here and relegate the results for 2 and 3, which paint a very similar picture, to Appendix D.3 in the full paper (33).

We analyze all 6 singular and 12 composite prioritization prompts (see Section 6.1), where we add additional clauses to prevent shifts in the utility distribution of all features not referenced in the prompt. We use the simulator scorer with a Utilitarian social welfare function and call the resulting model SCLM-ExtendedPrompt-Fair. As baselines, we consider DLM only prompted with the prioritization clause(s) (called DLM-PrioritizationOnly) and DLM prompted with the prioritization clause(s) and clause(s) together for a request for minimizing of utility shifts for the other features (called DLM-ExtendedPrompt-Fair). We also consider SCLM-SIM-Utilaterian only prompted with the prioritization clause(s) (called SCLM-PrioritizationOnly). See Appendix E in the full paper (33) for more details on the prompts.

To compute the alignment with prioritization clauses, similar to Section 6.3, we compute average change in utility generated by prioritized subpopulations. To quantify unintended utility shifts, we compute the average Earth mover's distance between utility feature distribution under the candidate and default reward function for each feature not included in one of the prioritization clauses.

Table 1 shows the results. Comparing *DLM-PrioritizationOnly* and *DLM-ExtendedPrompt-Fair*, we find that adding additional objective to the prompt does not result in a better performance for real-world domains. In contrast, *SCLM-ExtendedPrompt-Fair* which incorporates unintended shifts in the selection chooses reward functions resulting in significantly higher utility increases for prioritized subpopulations and significantly fewer unintended utility shifts. The fact that SCLM performs advantageously for both (conflicting) objectives highlights the quality of the pipeline and its capabilities to effectively address multiple objectives (of different types). We see similar results in synthetic domain (see Table 5 in the appendix in the full paper (33)).

7 Discussion

We present a customizable Social Choice Language Model to handle the multi-objective nature of preference prompts in reward design for RMABs. We showcase how methods from social choice can be used to improve the quality and transparency of decision-making of LLM-based frameworks, as we present an adjudicator component that makes the final decision from options generated by the LLM. SCLM significantly improves the quality of the chosen reward functions. We demonstrate that SCLM can not only handle composite prioritization prompts but arbitrary prompts containing multiple objectives, e.g., balancing the prioritization of subpopulations with the total utility generated by all agents. For future work, SCLM can be applied to other problems from multiagent planning and reinforcement learning. Further, SCLM can easily be extended to handle multiple preference prompts specified by different users.

Bibliography

[1] Abbou, A., Makis, V.: Group maintenance: A restless bandits approach. INFORMS J. Comput. **31**(4), 719–731 (2019). https://doi.org/10.1287/IJOC.2018.0863, https://doi.org/10.1287/ijoc.2018.0863

[2] Anil et al., R.: Gemini: A family of highly capable multimodal models. CoRR **abs/2312.11805** (2023). https://doi.org/10.48550/ARXIV.2312.11805, https://doi.org/10.48550/arXiv.2312.11805

[3] ARMMAN: ARMMAN: Advancing Reduction in Mortality and Morbidity of Mothers, Children, and Neonates (2024), https://armman.org/

[4] Arrow, K.J., Sen, A., Suzumura, K.: Handbook of social choice and welfare. Elsevier (2010)

[5] Ayer, T., Zhang, C., Bonifonte, A., Spaulding, A.C., Chhatwal, J.: Prioritizing hepatitis c treatment in us prisons. Operations Research **67**(3), 853–873 (2019)

[6] Behari, N., Zhang, E., Zhao, Y., Taneja, A., Mysore Nagaraj, D., Tambe, M.: A decision-language model (DLM) for dynamic restless multi-armed bandit tasks in public health. In: The Thirty-eighth Annual Conference on Neural Information Processing Systems (2024), https://openreview.net/forum?id=UiQkFXLfbu

[7] Cao, Y., Zhao, H., Cheng, Y., Shu, T., Liu, G., Liang, G., Zhao, J., Li, Y.: Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods. CoRR **abs/2404.00282** (2024). https://doi.org/10.48550/ARXIV.2404.00282, https://doi.org/10.48550/arXiv.2404.00282

[8] Carta, T., Oudeyer, P.Y., Sigaud, O., Lamprier, S.: Eager: Asking and answering questions for automatic reward shaping in language-guided rl. Advances in Neural Information Processing Systems **35**, 12478–12490 (2022)

[9] Deardorff, K.V., Rubin Means, A., Ásbjörnsdóttir, K.H., Walson, J.: Strategies to improve treatment coverage in community-based public health programs: a systematic review of the literature. PLoS neglected tropical diseases **12**(2), e0006211 (2018)

[10] Goyal, P., Niekum, S., Mooney, R.J.: Using natural language for reward shaping in reinforcement learning. arXiv preprint arXiv:1903.02020 (2019)

[11] Hazra, R., Sygkounas, A., Persson, A., Loutfi, A., Martires, P.Z.D.: Revolve: Reward evolution with large language models for autonomous driving. CoRR **abs/2406.01309** (2024). https://doi.org/10.48550/ARXIV.2406.01309, https://doi.org/10.48550/arXiv.2406.01309

[12] Jiang, J., Lu, Z.: Learning fairness in multi-agent systems. Advances in Neural Information Processing Systems **32** (2019)

[13] Kim, C.W., Moondra, J., Verma, S., Pollack, M., Kong, L., Tambe, M., Gupta, S.: Navigating the social welfare frontier: Portfolios for multi-objective reinforcement learning. arXiv preprint arXiv:2502.09724 (2025)

[14] Kwon, M., Xie, S.M., Bullard, K., Sadigh, D.: Reward design with language models. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023), https://openreview.net/forum?id=10uNUgI5Kl

[15] Ma, Y.J., Liang, W., Wang, G., Huang, D., Bastani, O., Jayaraman, D., Zhu, Y., Fan, L., Anandkumar, A.: Eureka: Human-level reward design via coding large language models. CoRR **abs/2310.12931** (2023). https://doi.org/10.48550/ARXIV.2310.12931, https://doi.org/10.48550/arXiv.2310.12931

[16] Ma, Y.J., Liang, W., Wang, H., Wang, S., Zhu, Y., Fan, L., Bastani, O., Jayaraman, D.: Dreureka: Language model guided sim-to-real transfer. CoRR **abs/2406.01967** (2024). https://doi.org/10.48550/ARXIV.2406.01967, https://doi.org/10.48550/arXiv.2406.01967

[17] Mate, A., Madaan, L., Taneja, A., Madhiwalla, N., Verma, S., Singh, G., Hegde, A., Varakantham, P., Tambe, M.: Field study in deploying restless multi-armed bandits: Assisting non-profits in improving maternal and child health. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 12017–12025 (2022)

[18] Mirchandani, S., Karamcheti, S., Sadigh, D.: Ella: Exploration through learned language abstraction. Advances in neural information processing systems **34**, 29529–29540 (2021)

[19] Moffaert, K.V., Drugan, M.M., Nowé, A.: Hypervolume-based multi-objective reinforcement learning. In: Purshouse, R.C., Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J. (eds.) 7th International Conference on Evolutionary Multi-Criterion Optimization. Lecture Notes in Computer Science, vol. 7811, pp. 352–366. Springer (2013)

[20] Moulin, H.: Fair division and collective welfare. MIT press (2004)

[21] Nelson, L.A., Mulvaney, S.A., Gebretsadik, T., Ho, Y.X., Johnson, K.B., Osborn, C.Y.: Disparities in the use of a mhealth medication adherence promotion intervention for low-income adults with type 2 diabetes. Journal of the American Medical Informatics Association **23**(1), 12–18 (2016)

[22] Niño-Mora, J.: Markovian restless bandits and index policies: A review. Mathematics **11**(7), 1639 (2023)

[23] Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of optimal queueing network control. In: Proceedings of IEEE 9th annual conference on structure in complexity Theory. pp. 318–322. IEEE (1994)

[24] Pita, J., Jain, M., Marecki, J., Ordóñez, F., Portway, C., Tambe, M., Western, C., Paruchuri, P., Kraus, S.: Deployed armor protection: the application of a game theoretic model for security at the los angeles international airport. In: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: industrial track. pp. 125–132 (2008)

[25] Qian, Y., Zhang, C., Krishnamachari, B., Tambe, M.: Restless poachers: Handling exploration-exploitation tradeoffs in security domains. In: Jonker, C.M., Marsella, S., Thangarajah, J., Tuyls, K. (eds.) Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016. pp. 123–131. ACM (2016), http://dl.acm.org/citation.cfm?id=2936946

[26] Rawls, J.: A theory of justice. In: Applied ethics, pp. 21–29. Routledge (2017)

[27] Roijers, D.M., Vamplew, P., Whiteson, S., Dazeley, R.: A survey of multi-objective sequential decision-making. Journal of Artificial Intelligence Research **48**, 67–113 (2013)

[28] Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., Yao, S.: Reflexion: language agents with verbal reinforcement learning. In: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Annual Conference on Neural Information Processing Systems 2023 (2023), http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html

[29] Syed, S.T., Gerber, B.S., Sharp, L.K.: Traveling towards disease: transportation barriers to health care access. Journal of community health **38**, 976–993 (2013)

[30] Tambe, M.: Security and game theory: algorithms, deployed systems, lessons learned. Cambridge university press (2011)

[31] Tsai, J., Rathi, S., Kiekintveld, C., Ordonez, F., Tambe, M.: Iris-a tool for strategic security allocation in transportation networks. AAMAS (Industry Track) pp. 37–44 (2009)

[32] Van Moffaert, K., Nowé, A.: Multi-objective reinforcement learning using sets of pareto dominating policies. The Journal of Machine Learning Research **15**(1), 3483–3512 (2014)

[33] Verma, S., Boehmer, N., Kong, L., Tambe, M.: Balancing act: Prioritization strategies for llm-designed restless bandit rewards. arXiv preprint arXiv:2408.12112 (2024)

[34] Verma, S., Singh, G., Mate, A., Verma, P., Gorantla, S., Madhiwalla, N., Hegde, A., Thakkar, D., Jain, M., Tambe, M., Taneja, A.: Expanding impact of mobile health programs: SAHELI for maternal and child care. AI Mag. **44**(4), 363–376 (2023). https://doi.org/10.1002/AAAI.12126, https://doi.org/10.1002/aaai.12126

[35] Verma, S., Zhao, Y., Sanket Shah, N.B., Taneja, A., Tambe, M.: Group fairness in predict-then-optimize settings for restless bandits. openreview.net/pdf?id=GJlZbpLWX3 (2024)

[36] Whittle, P.: Restless bandits: Activity allocation in a changing world. Journal of applied probability **25**(A), 287–298 (1988)

[37] Xie, T., Zhao, S., Wu, C.H., Liu, Y., Luo, Q., Zhong, V., Yang, Y., Yu, T.: Text2reward: Reward shaping with language models for reinforcement learning. In: The Twelfth International Conference on Learning Representations, ICLR 2024. OpenReview.net (2024), https://openreview.net/forum?id=tUM39YTRxH

[38] Xie, T., Zhao, S., Wu, C.H., Liu, Y., Luo, Q., Zhong, V., Yang, Y., Yu, T.: Text2reward: Reward shaping with language models for reinforcement learning. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=tUM39YTRxH

[39] Yu, W., Gileadi, N., Fu, C., Kirmani, S., Lee, K., Arenas, M.G., Chiang, H.L., Erez, T., Hasenclever, L., Humplik, J., Ichter, B., Xiao, T., Xu, P., Zeng, A., Zhang, T., Heess, N., Sadigh, D., Tan, J., Tassa, Y., Xia, F.: Language to rewards for robotic skill synthesis. In: Tan, J., Toussaint, M., Darvish, K. (eds.) Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 229, pp. 374–404. PMLR (2023), https://proceedings.mlr.press/v229/yu23a.html

[40] Zimeng, F., Nianli, P., Muhang, T., Brandon, F.: Welfare and fairness in multi-objective reinforcement learning. In: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems. pp. 1991–1999. ACM (2023)

[41] Zimmer, M., Glanois, C., Siddique, U., Weng, P.: Learning fair policies in decentralized cooperative multi-agent reinforcement learning. In: International Conference on Machine Learning. pp. 12967–12978. PMLR (2021)