WASHINGTON UNIVERSITY IN ST. LOUIS

McKelvey School of Engineering
Department of Computer Science & Engineering

# Towards Fair Sequential Resource Allocation: Algorithmic Designs, Interventions, and Evaluations

by
Ashwin Kumar

A dissertation presented to
the McKelvey School of Engineering
of Washington University in
partial fulfillment of the
requirements for the degree
of Doctor of Philosophy

December 2025
St. Louis, Missouri

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgments

Words will never be enough to express my feelings and gratitude to all the people that made it possible to complete my PhD research adventure. But as it is customary, I shall attempt to do so here.

First, to Prof. William Yeoh, my advisor, mentor, and guide throughout this journey, thank you for everything. Your decision to accept me as your PhD student was the beginning of the biggest transformations I would see in myself. Working with you taught me how to do research, how to translate my thoughts and ideas into action, and how to effectively communicate them with the world. You supported me through uncertainties and challenges, giving me space to stumble and correct myself, and to discover what research meant for me. As much as you were my advisor, you were also a role model, showing me how to be a better human being, both within academia and outside in the real world. I am honored to have had the chance to work with you and, now, to call you my friend.

I am very grateful to my committee members: Profs. Yevgeniy Vorobeychik, Alvitta Ottley, and Chien-Ju Ho, and Dr. Imanol Arrieta Ibarra. Our interactions throughout the years pushed me to be better and helped strengthen this thesis substantially. Thank you for supporting (and criticizing) my work, and for being exemplary researchers for me to look up to. I'm also fortunate to have had an amazing cohort of fellow PhD students to share this journey with. To all my collaborators and co-authors, thank you for agreeing to work with me. I learned a great deal from you. A huge thanks goes to my wonderful YODA labmates: Stylianos Vasileiou, Ben Rachmut, Yinxu Tang, Samuel Liu, Jean Springsteen, Christabel Wayllace, Athena Tabakhi, and Khoi Hoang. You all made the lab a fun and exciting place to be, and I will always cherish our time together.

This journey would not have been possible without the unwavering support of my family. To my parents and my brother, thank you for your unconditional love and encouragement. You have always believed in me, even when I doubted myself. Your sacrifices and hard work have paved the way for me to pursue my dreams. To Urvi, my best friend and partner in crime, thank you for being my rock. Your support, understanding, and companionship have been invaluable throughout this journey. I am grateful to have you by my side.

Finally, I want to thank all my friends who have been a part of my life during this time. Your friendship and support have meant the world to me. I feel extremely lucky just considering the fact that there are too many of you to name here. You know who you are, and I am grateful for each and every one of you, especially to those who laugh at my terrible jokes. And lastly, a special shoutout to every single person who took the time to sit and play board games with me, whether in person or online.

With this thesis, I close a chapter that spanned a fifth of my life so far. I carry the lessons learned and relationships formed into the next chapter, and I look forward to what lies ahead.

Ashwin Kumar

*Washington University in St. Louis*
*December 2025*

To Mummy, Papa and Ankur.

To Urvi.

ABSTRACT OF THE DISSERTATION

Towards Fair Sequential Resource Allocation: Algorithmic Designs, Interventions, and Evaluations

by

Ashwin Kumar

Doctor of Philosophy in Computer Science

Washington University in St. Louis, 2025

Professor Yevgeniy Vorobeychik, Chair

This thesis develops a comprehensive framework for fair sequential resource allocation in multi-agent systems where a centralized allocator coordinates actions under global feasibility constraints, while satisfying preferences of different agents. Ranging from ridesharing platforms and homelessness intervention programs to power grid management, such systems play a critical role in shaping access to essential resources. Yet, existing approaches to resource allocation often prioritize aggregate utility, leading to systematic inequities across individuals and groups, particularly in sequential settings where decisions unfold over time.

To address this challenge, we introduce the Distributed Evaluation, Centralized Allocation (DECA) framework, which unifies a broad class of real-world allocation problems. DECA separates agent-side evaluation from a central allocator that must satisfy feasibility constraints while also respecting agents' preferences. Building on this framework, we develop methods to (i) detect and quantify temporal inequities through empirical studies and visualization tools, and (ii) design interventions that balance fairness and efficiency with controllable trade-offs. These interventions span post-processing corrections for deployed systems, learning-based in-processing methods that incorporate fairness during training, and data-centric pre-processing approaches that reduce downstream bias.

Our contributions include fairness analyses of real-world domains such as ridesharing, homelessness services, and power grid operations, as well as algorithmic methods that operationalize fairness under centralized feasibility constraints. Viewing online data collection as a resource allocation problem, we also develop methods to improve fairness in mobility prediction through equitable online data collection. We further extend fairness audits to contemporary AI systems by detecting biases in reward models used in reinforcement learning from human feedback (RLHF) for large language models, connecting classical fairness concerns to modern AI training pipelines.

Together, these frameworks, methodologies, and empirical studies advance the design of AI systems that allocate resources not only efficiently but also equitably. By integrating fairness into sequential resource allocation, this work contributes toward building AI systems that are more accountable, trustworthy, and socially responsible.

# Part I

# Introduction | प्रारम्भ

# Chapter 1

# Introduction

Artificial Intelligence (AI) has become deeply embedded in decision-making processes that profoundly impact human lives. From healthcare diagnosis and criminal justice to financial lending and resource allocation, AI systems increasingly determine who receives what resources, when, and how. The global AI market, projected to reach \$3.68 trillion by 2034, reflects not just economic growth but the expanding scope of AI's influence on society's most critical decisions.[1]

However, as AI systems become more sophisticated and consequential, AI algorithms are used to make decisions that impact millions of people. In such *resource allocation* problems, a fundamental challenge emerges: ensuring that these systems allocate resources fairly across different groups and individuals. There is substantial evidence that AI systems, if not carefully designed, can inadvertently perpetuate or even exacerbate existing societal biases and inequities [8, 102]. There is a growing body of work studying fairness in AI, particularly in the context of machine learning predictions [58, 35]. In this dissertation, we study the field of resource allocation, a setting where a pool of resources must be allocated among multiple individuals or groups. Typically, these decisions are optimized for overall system utility (e.g., maximizing total profit or efficiency). However, this utilitarian approach can introduce biases, making fairness an important consideration in AI-driven decision-making.

This challenge is particularly acute in *sequential resource allocation* problems, where decisions unfold over time and past allocations influence future opportunities. Consider ridesharing platforms that repeatedly match drivers with passengers, homelessness services that continuously assign housing resources, or energy grids that dynamically distribute power during crises. In each case, short-term efficiency optimizations can compound into long-term inequities, systematically disadvantaging certain groups while benefiting others.

[1] https://www.precedenceresearch.com/artificial-intelligence-market

Existing fairness research has approached fair allocation from two main angles: (1) one-shot allocation problems, where resources are allocated in a single decision round [119, 132, 126], developing ideas like maximin fairness, envy-freeness, and Nash social welfare; and (2) fair Machine Learning (ML) predictions, where models are trained to make fair predictions based on historical data [58, 35]. However, these approaches often fall short in addressing the unique challenges posed by sequential allocation problems. There is a small body of work studying fairness in multi-agent reinforcement learning (MARL) settings [138, 66], but these methods typically assume fully decentralized execution, where each agent independently selects actions. This assumption does not hold in many real-world resource allocation problems, where a central authority enforces global constraints and coordinates agent actions.

The problem of fair sequential resource allocation represents a critical gap in current AI research. While substantial work exists on fairness in one-shot ML predictions and static resource allocation problems, the temporal dynamics inherent in many real-world systems introduce complexities that existing approaches fail to address. Sequential allocation problems require reasoning about fairness across time horizons, balancing immediate needs with long-term equity, and accounting for how past decisions should affect present resource distributions. Moreover, these systems often involve multiple stakeholders with competing interests, uncertain future demands, and evolving environmental conditions—all of which traditional fairness frameworks struggle to handle.

## 1.1 Problem Statement and Significance

This thesis addresses the fundamental question: *How can we design AI systems that allocate indivisible resources fairly in sequential, multi-agent environments while maintaining system efficiency and adaptability?* This problem is both technically challenging and socially urgent for several reasons.

First, sequential resource allocation problems are ubiquitous in high-stakes domains. Transportation networks, healthcare systems, social services, and energy infrastructure all involve repeated allocation decisions that accumulate into patterns of advantage or disadvantage. When these systems operate without explicit fairness considerations, they often perpetuate or amplify existing societal inequities. For instance, algorithmic hiring systems can systematically exclude certain demographic groups, while emergency response systems may inadvertently provide slower service to underserved communities. Recognizing that biases exist in such systems is the first step, so that we can work towards mitigating them. Fairness in multi-step resource allocation is critical, as algorithmic biases can lead to disparities and decreased trust in automated systems [102]. Beyond ethical considerations, fair resource allocation may also be desirable from the perspective of the central controller; for example, governments may want to ensure welfare programs equally benefit all communities, or companies may wish to maintain diversity in their service or conform to regulatory requirements.

Second, the temporal nature of these problems creates unique fairness challenges. Unlike static scenarios where fairness can be evaluated at a single point in time, sequential allocation requires reasoning about fairness trajectories—how equitable treatment evolves over multiple decision rounds. A system might make decisions that appear fair at any given moment but lead to systematically biased outcomes over time, or conversely, might temporarily violate fairness constraints to achieve better long-term equity. This temporal complexity demands new theoretical frameworks and algorithmic approaches.

Finally, the multi-agent nature of these systems combined with the need to enforce global resource constraints introduces a structure that existing multi-agent techniques in sequential decision-making are unable to support. Ensuring fairness while maintaining efficiency requires designing new frameworks for learning and adaptation in this setting that support centralized allocations while considering diverse preferences from the agents involved in such systems.

## 1.2 Limitations of Existing Approaches

Current approaches to fairness in AI systems suffer from several key limitations when applied to sequential resource allocation problems.

**Static Fairness Metrics:** Most existing ML fairness research focuses on one-shot decisions using metrics like demographic parity, equalized odds, or individual fairness. These metrics, while valuable, fail to capture the cumulative effects of repeated decisions. A system achieving optimal demographic parity at each time step might still generate unfair long-term outcomes if it systematically assigns lower-quality resources to certain groups.

**Limited Temporal Reasoning:** Few existing frameworks explicitly model how fairness constraints should evolve over time. Should fairness be evaluated at each decision point, over fixed time windows, or across entire interaction histories? How should systems trade off short-term fairness violations against long-term equity goals? Current approaches provide little guidance on these critical temporal considerations. The few approaches that attempt to tackle fair behavior in sequential decision-making tasks often struggle to model fairness over long time horizons properly. Typical approaches optimize for fairness over the entire history, or aim to maximize fairness at a terminal time step. However, these approaches fail to account for long-horizon settings or continous tasks where there is no terminal time step, which is common in many real-world resource allocation problems.

**Expectation of Decentralized Execution:** Existing methods for learning long-term fair behavior in sequential decision-making tasks [66, 138, 176] assume full decentralizability; i.e., each agent is independently able to execute actions of their choice. This is not possible in resource allocation domains where global constraints necessitate coordination. In many such domains, this coordination is centrally enforced (e.g. by an authority). This breaks the learning algorithms used in most state-of-the-art fair MARL methods, especially algorithms based on policy optimization methods like PPO [131]. This is because the trajectories generated by the agents during training do not reflect their policy distribution, as the actions are orchestrated by a central controller.

**Lack of Controllable Fairness-Utility Tradeoffs:** Methods for fair resource allocation often optimize a singular objective, with limited or no flexibility in the fairness-utility tradeoff. This makes it difficult to adapt these methods for use in domains where online control of these tradeoffs is a desirable property.

**Lack of Cross-Domain Generality:** Many fairness interventions are highly domain-specific, relying on particular assumptions about agent preferences, resource types, or environmental dynamics. This limits their applicability across different sequential allocation problems. A more general framework is needed to systematically study fairness interventions across diverse domains.

## 1.3 Contributions and Approach

**Thesis claim.** Fair sequential allocation of indivisible resources under centralized execution and global feasibility constraints can be advanced by (i) a unifying modeling lens, (ii) case studies and evaluations that reveal temporal inequities in deployed systems, and (iii) interventions with *controllable* fairness–utility trade-offs at post-, in-, and pre-processing stages—together with audits for modern AI components that influence allocation.

**C1. A unifying modeling lens: DECA.** We formalize the *Distributed Evaluation, Centralized Allocation* (DECA) framework for sequential allocation under system-level constraints. DECA separates heterogeneous, agent-side evaluation from a centralized allocator that must satisfy feasibility (capacity, coupling, and policy) constraints each round, enabling trajectory-level fairness objectives while reflecting centralized execution in practice (Part I, Chapter 2).

**C2. Case studies and evaluations of temporal inequity.** We present empirical studies that surface disparities accumulating over time in real systems: (i) a user study showing visual explanations improve comprehension [79]; (ii) an analysis of Winter Storm Uri that quantifies group-level disparities induced by operational policies [81]; and (iii) *FairVizARD*, an interactive visualization system for examining fairness in ridesharing allocations [83] (Part II, Chapters 3, 4, 5). We also present similar evaluations in later parts: (iv) a case study of homelessness resource allocation that surfaces inequities in existing policies [73] (Part III, Chapter 7); (v) an observational analysis of geolocation prediction models that reveals demographic disparities [84] (Part IV, Chapter 11); and (vi) an audit of LLM reward models that detects *prefix bias* in RLHF training data [82] (Part V, Chapter 12).

**C3. Interventions across the pipeline with controllable trade-offs.** We design allocation mechanisms that expose explicit, monotone parameters for fairness–utility control and preserve centralized feasibility:

- *Post-processing (deployment-time) corrections:* We develop the SI and GIFF algorithms. **SI** shows that simple incentives can improve matching fairness in ridesharing [80] and extend to homelessness resource allocation [73]; **GIFF** generalizes SI to diverse fairness functions and problem settings [74] (Part III, Chapters 6, 7, 8).
- *In-processing (learning-time) methods:* We formalize **past-discounting** of collected utilities to make infinite-horizon fairness tractable [77] and introduce **DECAF**, which incorporates long-term fairness in multi-agent learning while respecting centralized feasibility [76] (Part IV, Chapters 9, 10).
- *Pre-processing (data-time) interventions:* **Fairness-aware sampling** for location prediction reduces downstream allocation bias [84] (Part IV: Chapter 11).

**C4. Auditing modern AI components that affect allocation.** We detect *prefix bias* in LLM reward models used in RLHF, connecting classical fairness concerns to contemporary training pipelines that influence sequential decision systems [82] (Part V, Chapter 12).

**Approach and validation.** The workflow proceeds from case studies/evaluations $\rightarrow$ modeling $\rightarrow$ intervention (post/in/pre) $\rightarrow$ modern-AI auditing. Validation combines user studies and system demonstrations (C2), simulation and ablation under feasibility constraints (C3), and observational analyses of real events (C2). Claims are calibrated to these modalities.

**Assumptions and scope.** We focus on indivisible resources, centralized execution with enforceable global constraints, and truthful, non-strategic or externally validated agents. Strategic manipulation and incentive compatibility are not the primary focus; extensions are discussed in the conclusion.

## 1.4 Thesis Organization

This thesis contains content that has been presented in two ICAPS papers [79, 80], one FAccT paper [82], one AAMAS extended abstract [75, 76], one ICAPS system demonstration [78, 83], two AASG workshop papers [73, 77] and one journal perspective [81]. It also contains content from three conference papers currently under submission [84, 74, 76]. The thesis is divided into six parts, with each part encapsulating thematically related chapters.

### 1.4.1 Outline

This thesis develops and evaluates computational approaches for detecting, measuring, and mitigating unfairness across AI systems. The work is organized into six parts:

- **Part I: Introduction**: Establishes foundational concepts and motivation. Chapter 1 presents the problem scope and thesis contributions. Chapter 2 introduces resource allocation theory, fairness metrics, and the DECA framework that underlies our multi-agent approaches.
- **Part II: Identifying Fairness Concerns**: Demonstrates bias detection in real-world systems through two case studies. First, Chapter 3 describes a user study establishing that users understand explanations better through visualizations, motivating their use [79]. Given this context, Chapter 4 analyzes the disproportionate impacts of power outages during the winter storm Uri [81], while Chapter 5 presents FairVizARD, an interactive visualization system for studying fairness in ridesharing [83].
- **Part III: Incentive-Based Fair Allocation**: Develops post-processing methods for fairness improvement. Chapter 6 introduces Simple Incentives (SI) for improving matching fairness in ridesharing systems [80], Chapter 7 applies these methods to homelessness resource allocation [73], and Chapter 8 presents GIFF, a generalized framework capable of handling diverse fairness functions [74].
- **Part IV: Learning and Sampling for Fairness**: Addresses fairness during model training and data collection. Chapter 9 examines temporal considerations in fairness learning [77], Chapter 10 presents DECAF for fair multi-agent learning [76], and Chapter 11 demonstrates fairness-aware data sampling in location prediction [84].
- **Part V: Bias in Language Models**: Extends fairness analysis to modern AI systems, developing methods for detecting and quantifying bias in reward models used for LLM training. Chapter 12 presents techniques for identifying prefix bias in LLM-based reward models [82].

- **Part VI: Conclusion**: Chapter 13 concludes with our contributions and broader implications, wrapping up with a discussion on potential extensions, future research directions, and ethical considerations.

# Chapter 2

# Background

The primary goal of this chapter is to establish notation and core concepts used throughout the thesis and to situate our setting relative to adjacent formalisms.

## 2.1 Preliminaries

### 2.1.1 Notation

We denote vectors in bold $(\mathbf{z}, \mathbf{w})$, and discount factors are $\gamma \in [0, 1)$. For a vector $\mathbf{z} \in \mathbb{R}^n$, $z_i$ denotes its $i$-th component and $z_{(i)}$ denotes the $i$-th order statistic (ascending). Agents are indexed by $i \in \alpha = \{1, \dots, n\}$. We will use $\mathbb{E}[\cdot]$ for expectations under the appropriate distribution, made explicit when needed.

### 2.1.2 Markov Decision Processes and Sequential Decision Making

A (discounted, infinite-horizon) Markov Decision Process (MDP) is defined as a tuple [144] $\langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition probabilities $P(s' \mid s, a)$, reward function $r(s, a)$, and discount factor $\gamma \in [0, 1)$. A policy $\pi(a \mid s)$ maps states to distributions over actions. The (state) value and action-value functions are:

$$V^\pi(s) = \mathbb{E}_\pi\Big[\sum_{t=0}^{\infty} \gamma^t r(S_t, A_t) \,\Big|\, S_0 = s\Big], \qquad Q^\pi(s, a) = r(s, a) + \gamma\, \mathbb{E}_{s' \sim P(\cdot|s,a)}\big[V^\pi(s')\big]. \quad (2.1)$$

Under standard assumptions, an optimal stationary policy $\pi^\star$ exists and can be characterized via the Bellman optimality operator.

**Partial observability.** In many applications, the agent receives an observation $o \in \mathcal{O}$ generated from the latent state $s$ by an observation function $\Omega(o \mid s)$. We will use $o_i$ to denote the (possibly partial) information available to agent $i$.

**Multi-agent settings.** A cooperative multi-agent MDP specifies a joint action space $\prod_i A_i$, a shared transition model, and (depending on the setting) either individual utilities $r_i$ or a team utility [143, 43, 125]. Policies may be centralized or factorized into per-agent policies with varying information structures [4]. We return to decentralized versus centralized execution below when introducing DECA.

### 2.1.3 Resource Allocation Problems

A one-step (static) resource allocation problem takes as input a set of agents $\alpha$, a set of resources or actions with resource requirements, and an objective. The allocator selects an assignment $x$ from a feasibility region $\mathcal{X}$ (e.g., matching, knapsack, or packing constraints) to maximize a welfare objective constructed from agent utilities $u_i$:

$$\max_{x \in \mathcal{X}} \quad W(x; \{u_i\}_{i \in \alpha}). \tag{2.2}$$

When resources are *indivisible*, $\mathcal{X}$ often yields an integer program (e.g., bipartite matching or multiple knapsack). In the sequential setting, the allocator repeats this decision at each epoch with evolving states, supplies, and demands, so that allocations accumulate over time; modeling this accumulation is central to our fairness treatment.

### 2.1.4 Fairness

Two traditions inform fairness in allocation and decision-making: (i) social choice and welfare economics, and (ii) fairness in machine learning.

*(i) Fairness via social welfare.* Classical approaches embed equity into the objective, e.g., Rawlsian (leximin/maximin) priorities [126], $\alpha$-fair utilities and related Nash social welfare variants, or Generalized Gini Functions. These choices are principled joint measures of equity and efficiency; we formalize some of them in Section 2.5.

*(ii) Fairness in prediction and classification.* In supervised learning, group metrics such as statistical parity and equality of opportunity constrain predictive behavior across sensitive groups [35, 58]. While our focus is allocation under constraints rather than classification, these metrics will inform evaluation when allocations depend on learned predictions. This is often studied through the lens of pre- [65, 168], in- [176, 66], and post-processing interventions [118, 102].

**Scope for this thesis.** Our primary interest is *sequential* or *repeated* allocation of *indivisible* resources under centralized feasibility and resource constraints, where fairness is evaluated on accumulated outcomes across time (Section 2.5). Unless noted, we assume agents provide truthful signals/utilities or externally validated estimates.[2]

### 2.1.5 Fairness in (Multi-Agent) Reinforcement Learning

A common way to impose fairness in Reinforcement Learning (RL) is to optimize a scalarized social welfare over per-agent returns (e.g., $\alpha$-fair or Gini-weighted sums), leading to multi-objective MDP formulations with a chosen scalarization [138, 176]. Recent (fair) Multi-Agent RL (MARL) methods typically assume *fully decentralized execution*: each agent selects and executes its own action, and coordination emerges via learning, often supported by a mixing architecture for team rewards.

This assumption, however, does not fit many real deployments where a central authority must enforce *global* feasibility (capacity, coupling, policy) at each decision point. The commonly used alternative to this is fully centralized execution, where a single policy selects joint actions for all agents. This is often impractical due to exponential action spaces and lack of agent autonomy. Practically, many systems lie between these extremes: agents may evaluate options locally, but a central controller enforces global constraints and selects the final actions conditioned on agent preferences and predicted utilities. This decouples *preference evaluation* (by agents) from *resource allocation* (by a central allocator). This execution structure is orthogonal to “centralized training, decentralized execution (CTDE)”, a commonly used

[2]A systematic treatment of incentives and strategic behavior is an important extension; we discuss it in the conclusion.

paradigm in MARL where agents are trained with access to global information but execute independently.

Motivated by this gap between fully decentralized execution and intractable fully centralized control, we introduce our first contribution: a general execution abstraction we call Distributed Evaluation, Centralized Allocation (DECA). In DECA, agents produce local utility evaluations of their candidate actions, while a central allocator—constrained by capacity, coupling, or policy rules—selects the single feasible joint action. This abstraction matches how many deployed systems actually operate. The separation it enforces—evaluation on the per-agent interface, allocation at the system interface—lets us capture indivisible resources under global feasibility, avoid exponential joint-action enumeration, and cleanly define fairness on accumulated outcomes over time. We next formalize DECA and use it as the common substrate for modeling, learning, and fair allocation throughout the thesis.

## 2.2 Distributed Evaluation, Centralized Allocation (DECA)

Most multi-agent RL approaches assume agents can act independently, maximizing their respective utility [43, 4]. This overlooks many real-world settings where actions must be coordinated to enforce global constraints like limited resources, rendering existing methods inapplicable and fairness difficult to enforce. We address this gap by studying fairness in a distinct paradigm: *Distributed Evaluation, Centralized Allocation (DECA).*

DECA captures a broad class of problems where agents compute local utility estimates, but do not independently execute actions. Instead, a central allocator makes final decisions over which actions are taken, subject to global constraints. This problem has been independently studied in various domains under different names, from ridesharing [135] and social resource allocation [70, 71] to satellite scheduling [110] and network routing [165]. However, prior work has been domain-specific, with different assumptions and solution methods. **For the first time, we present a single unifying framework that encapsulates all these settings under a single DECA formulation. Further, we are the first to study fairness in this general DECA setting.**

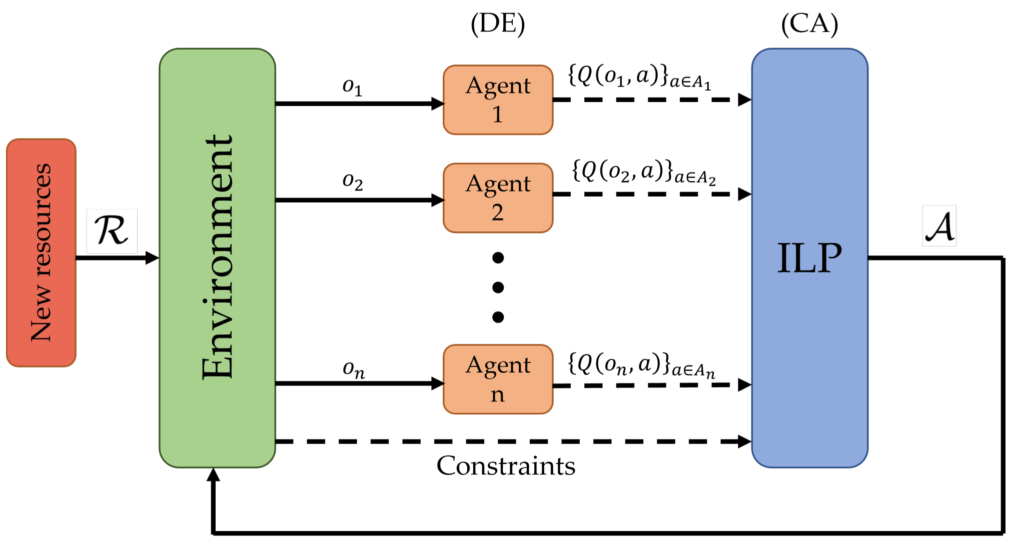


Figure 2.1: An outline of the DECA pipeline. Each agent evaluates its available actions in a decentralized manner (DE), and the ILP finds the best joint action $\mathcal{A}$ using these evaluations and resource constraints (CA).

In the context of cooperative multi-agent learning, DECA differs from the well-known Centralized Training with Decentralized Execution (CTDE) paradigm. While CTDE governs how agents learn, DECA describes how decisions are executed: agent policies evaluate candidate actions locally (Distributed Evaluation, DE), but a central module aggregates these evaluations and selects actions under resource constraints (Centralized Allocation, CA). MARL methods designed for decentralized execution struggle in this setting, motivating new approaches tailored to DECA.

We model DECA as a constrained multi-agent Markov decision process (CMMDP) [30] with partial observability, defined by the tuple:

$$\mathcal{M} = \langle \alpha, S, \mathcal{O}, \{A_i\}, T, R_u, \gamma, c \rangle \tag{2.3}$$

where $\alpha$ is the set of agents, $S$ is the global state space, and $\mathcal{O} : S \to \prod_i O_i$ maps states to agent observations. Each agent $i$ has action space $A_i$, and $T : S \times \prod_i A_i \times S \to [0, 1]$ defines joint transitions. The reward function $R_u : S \times \prod_i A_i \to \mathbb{R}^n$ returns a vector of agent utilities, and $\gamma$ is the discount factor. The resource map $c : \bigcup_i A_i \to \mathbb{R}^K$ gives per-action consumption across $K$ resource types.

In a DECA problem, illustrated by Figure 2.1, agents independently evaluate actions based on their local observations (DE), while a central controller aggregates these evaluations and optimizes resource allocation subject to constraints (CA). Agent actions include the null action and may have other effects apart from allocation of resources (e.g., moving in an environment), but only actions which consume resources are constrained.

### 2.2.1 Optimization Framework

The distributed evaluation (DE) step involves agents learning to predict the utilities of observation-action pairs using approaches such as Deep Q-learning [106, 59]. The utility estimates are computed using partially-observable post-decision states, which are estimated locally, ignoring other agents' actions due to the infeasibility of exploring the joint state space.

The central allocation (CA) step solves an integer linear program (ILP) that combines predicted utilities with resource constraints. Let $x_i(a) \in \{0, 1\}$ be a binary decision variable that indicates whether agent $i$ is assigned action $a \in A_i$. Let $\mathcal{R} \in \mathbb{R}^K$ denote the vector of available resources, with $\mathcal{R}_k$ representing the quantity of resource type $k \in \{1, 2, \ldots, K\}$. Let $c(a) \in \mathbb{R}^K$ denote the resource consumption vector for action $a$, where $c(a)_k$ is the amount of resource $k$ consumed by action $a$. The objective of the ILP is to select one action per agent that maximizes the total predicted utility $Q(o_i, a)$, subject to resource constraints. The optimization problem is defined as:

$$\max_{x_i(a)\in\{0,1\}} \quad \sum_{i\in\alpha}\sum_{a\in A_i} x_i(a)\cdot Q(o_i, a) \tag{2.4}$$

$$\text{subject to} \quad \sum_{a\in A_i} x_i(a) = 1, \quad \forall i \in \alpha \tag{2.5}$$

$$\sum_{i\in\alpha}\sum_{a\in A_i} x_i(a)\cdot c(a)_k \leq \mathcal{R}_k, \quad \forall k \in \{1, \ldots, K\} \tag{2.6}$$

The first constraint ensures that exactly one action is selected for each agent. The second constraint ensures that the total resource consumption across all selected actions does not exceed the available amount of any resource. This ILP defines the central allocation mechanism, while the predicted Q-values $Q(o_i, a)$ are learned by each agent, enabling distributed evaluation. This offers benefits over completely distributed approaches by encapsulating resource constraints, and over completely centralized approaches by reducing the complexity of the learning objective. This setup is seen in many resource allocation problems [70, 7, 135, 110, 165].

While DECA has not been formalized in prior work, it has seen application in many domains. From optimizing passenger-driver matches in ridesharing [135, 120] to efficient allocation of homelessness resources [70, 71], many real-world applications follow this general structure. It is also analogous to the predict-then-optimize (P+O) approach [159, 37], where a predictive model estimates unknown parameters that are subsequently used in optimization. However, unlike P+O, which may not specifically address resource allocation or multi-agent systems, DECA explicitly focuses on these complexities. This distinction is crucial as it allows us to restrict the problem space and tailor our research towards enhancing fairness within multi-agent resource allocation.

### 2.2.2 Distinguishing DECA from Dec-POMDP

Decentralized Partially Observable Markov Decision Processes (Dec-POMDPs) are a widely used formalism for modeling cooperative multi-agent systems under partial observability [59, 43]. Dec-POMDPs model cooperative decision-making in which agents execute their own policies in a fully decentralized manner. In contrast, DECA separates decision evaluation from allocation: agents independently predict the utility of their potential actions, but a central allocator determines which actions are ultimately executed. This central allocation step distinguishes DECA from Dec-POMDPs by shifting action selection from the agents to the allocator. This also prevents many popular algorithms for Dec-POMDPs from being used to solve DECA problems.

## 2.3 Solving DECA Problems

Solving DECA problems requires addressing two coupled challenges: (i) acquiring utilities for agent-action matches, and (ii) allocating resources subject to global constraints and reported agent utilities.

### 2.3.1 Predicting Utilities in DECA

Agents may have heterogeneous valuations of available resources, expressed through utility estimates $Q(o_i, a)$. These utilities may come from different sources: in some cases, they reflect inherent black-box preferences (e.g., human-provided rankings), while in others they are learned through interaction with the environment.

When utilities are learned, several approaches are possible:

- **Independent learning:** Agents may estimate utilities via Independent Q-Learning (IQL) [43], treating other agents as part of the environment.
- **Team-based learning:** When a shared team reward is used, cooperative multi-agent reinforcement learning methods such as Value Decomposition Networks (VDN) and QMIX can be employed [4].
- **Model-based learning:** Approximate Dynamic Programming techniques can incorporate post-decision states to guide learning more efficiently [135].

### 2.3.2 Allocation Step

The allocator combines the agents' predicted utilities with global resource constraints to determine the final resource allocation. The optimization problem in Eq. 2.4 takes the form of an integer linear program (ILP). While general-purpose ILP solvers are a common solution approach, other methods such as heuristics, relaxations, or decomposition techniques may be appropriate depending on problem scale and structure. Additionally, more efficient algorithms and distributed approaches exist for allocation problems with stricter constraints. For instance, if the constraints boil down to a bipartite matching between agents and resources, the Hungarian algorithm [72] can solve the allocation problem in polynomial time.

## 2.4 Payoff-vector (Z): Modeling Resource Distribution over Time

We are interested in resource allocation problems that have a temporal aspect to them, i.e., after each allocation, new resources may arrive in the system, and resources and agents may enter or exit the allocation pool. We consider two cases: 1) A fixed number of agents, and 2) An arbitrary number of agents that belong to a fixed number of groups. In both cases, we consider a total of $n$ groups or agents. Given this, we can construct a vector of payoffs $\mathbf{Z} = [z_1, z_2, \ldots, z_n]$ that captures the accumulated value of all resources allocated to each agent/group over time.

Unless specified otherwise, we use $z_i$ to denote the cumulative payoff for agent or group $i$ at the current time-step. When referring to a specific time, we will write $z_i^{(t)}$ to indicate the value at time-step $t$. The cumulative reward is given by:

$$z_i = \sum_{\tau=0}^{t} r_i^{(\tau)}, \tag{2.7}$$

where $r_i^{(\tau)}$ is the reward received by agent $i$ at time-step $\tau$. In some settings, an average payoff is used instead:

$$\bar{z}_i = \frac{1}{t+1} \sum_{\tau=0}^{t} r_i^{(\tau)}. \tag{2.8}$$

This payoff vector serves as a temporal record of how resources have been distributed across agents and provides a foundation for incorporating fairness criteria into future allocation decisions.

## 2.5 Fairness Concepts Used

To capture fairness, we consider a fairness function $F : \mathbb{R}^n \to \mathbb{R}$, which maps any payoff vector $\mathbf{Z} = [z_1, \ldots, z_n]$ to a numerical value such that larger values correspond to fairer distributions. In other words, for any two payoff vectors $\mathbf{Z}_1$ and $\mathbf{Z}_2$, we say that $\mathbf{Z}_1$ is fairer than $\mathbf{Z}_2$ if and only if $F(\mathbf{Z}_1) > F(\mathbf{Z}_2)$.

In the literature on fair multi-agent resource allocation, two main schools of thought emerge:

- **Social Welfare Function Approaches.** In these methods, fairness is directly embedded into a social welfare function that aggregates individual utilities [138, 176]. Such functions may be designed to satisfy three desirable properties: They exhibit **impartiality**, remaining invariant under any permutation of agents to ensure equal treatment; they promote **equity** by rewarding reallocations that transfer resources from better-off agents to those worse-off, consistent with the Pigou-Dalton principle; and they ensure **efficiency** by assigning a higher value to allocations in which every agent receives higher or equal utility compared to alternative allocations.

  These metrics usually take the form of a summation over transformations of agent utilities, for example:

  - $\alpha$**-Fairness:** For a given payoff vector $\mathbf{Z} = [z_1, \ldots, z_n]$, define the per-agent $\alpha$-fair utility as:

    $$U_\alpha(z) = \begin{cases} \frac{z^{1-\alpha}}{1-\alpha} & \text{if } \alpha \neq 1, \\ \log(z) & \text{if } \alpha = 1, \end{cases} \tag{2.9}$$

    and the overall fairness measure as: $F_\alpha(\mathbf{Z}) = \sum_{i=1}^{n} U_\alpha(z_i)$.

  - **Generalized Gini Function (GGF):** Order the components of $\mathbf{Z}$ as $z_{(1)} \leq z_{(2)} \leq \cdots \leq z_{(n)}$. Then, the GGF function is defined as:

    $$F_{GGF}(\mathbf{Z}) = \sum_{i=1}^{n} w_i \, z_{(i)}, \tag{2.10}$$

    with weights satisfying $w_1 \geq w_2 \geq \cdots \geq w_n$ and $\sum_{i=1}^{n} w_i = 1$.

By varying $\alpha$ or $w_i$, these metrics can transition between utilitarian and egalitarian fairness. Specifically, $\alpha$-fairness with $\alpha = 1$ is equivalent to the popular log Nash Welfare metric. [108]

- **Distributional Approaches.** Alternatively, fairness may be measured separately via distributional metrics—such as variance, the Gini index, or Jain's fairness index—and then combined with total utility [80, 124]. These metrics can capture non-linear relationships among agents' utilities and provide a distinct measure of fairness that is later interpolated with the overall efficiency. Often, these capture the notion of *disparity* or *inequality* in the distribution of utilities, which we denote as $D(\mathbf{Z})$, with larger values indicating greater disparity and lower fairness. Let $\mathbf{Z} = [z_1, \ldots, z_n]$ with $z_i \geq 0$ and $\bar{z} = \frac{1}{n}\sum_i z_i$. Then, common disparity measures include:

  - **Variance / Coefficient of Variation (CV).**

$$\mathrm{Var}(\mathbf{Z}) = \frac{1}{n}\sum_{i=1}^{n}(z_i - \bar{z})^2, \qquad \mathrm{CV}(\mathbf{Z}) = \frac{\sqrt{\mathrm{Var}(\mathbf{Z})}}{\bar{z}} \quad (\bar{z} > 0). \tag{2.11}$$

    Variance penalizes dispersion but is not scale-invariant; CV is scale-invariant and equals 0 iff all $z_i$ are equal.

  - **Gini index.** Using either the pairwise form or the Lorenz-sorted form (with $z_{(1)} \leq \cdots \leq z_{(n)}$):

$$\mathrm{Gini}(\mathbf{Z}) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}|z_i - z_j|}{2n\sum_{i=1}^{n} z_i} = \frac{2\sum_{i=1}^{n} i\, z_{(i)}}{n\sum_{j=1}^{n} z_j} - \frac{n+1}{n}. \tag{2.12}$$

    Gini $\in [0, 1)$ for nonnegative $\mathbf{Z}$; 0 iff all $z_i$ are equal.

  - **Jain's fairness index.**

$$J(\mathbf{Z}) = \frac{\left(\sum_{i=1}^{n} z_i\right)^2}{n\sum_{i=1}^{n} z_i^2}, \tag{2.13}$$

    with $J \in [1/n, 1]$, attaining 1 iff all $z_i$ are equal and invariant to common scaling. A convenient disparity measure is $D_J(\mathbf{Z}) := 1 - J(\mathbf{Z})$.

These dispersion metrics are permutation-invariant and quantify inequality independently of the total utility. They are typically combined with efficiency via either a *penalized objective*, for example:

$$\max_{\pi} \ \lambda \Big( \sum_{i=1}^{n} z_i^{\pi} \Big) \ - \ (1-\lambda)\, D(\mathbf{Z}^{\pi}), \qquad \lambda \in [0,1], \tag{2.14}$$

where $D$ can be Var, Gini, or $D_J$, or via a *constrained form*, such as:

$$\max_{\pi} \sum_{i=1}^{n} z_i^{\pi} \quad \text{s.t.} \quad D(\mathbf{Z}^{\pi}) \leq \tau. \tag{2.15}$$

# Part II

# Identifying Fairness Concerns

# Chapter 3

# A Planning-based Visualization Framework for Communicating Explanations to Human Users

## 3.1 Introduction & Contribution

Before diving into our exploration of fairness in AI systems, we take a step back to address a fundamental challenge that underlies all AI research: how do we effectively communicate complex algorithmic decisions to human users? This question becomes particularly critical in fairness applications, where affected stakeholders must understand not just what decisions were made, but why certain allocations or policies were chosen. While the focus will soon shift to fairness and resource allocation, this chapter develops essential methodological foundations by exploring visualizations as a medium for communicating explanations to human users. In this work, we explore the field of explainable AI planning (XAIP) [24] and propose a visualization framework that informs how we communicate fairness-related decisions in later chapters.

Generally, communicating information to human users can be typically achieved through four mediums: Verbal, non-verbal, text-based, and visual. One of the most common computer interfaces to communicate information is the graphical user interface, where information is presented graphically through a combination of visualizations and text [99]. From the perspective of communicating explanations, although text can be a natural representation for an explanation, when presented alone, it may require increased cognitive effort and possibly increase the likelihood of misunderstanding the task, especially for novice users.

In contrast, a well-established educational principle, called the *multimedia learning principle*, posits that humans learn better from visuals and words, than from words alone [100]. For example, Clark *et al.* [27] showed that accompanying text-based instructions with visuals improved students' performance on a test by a median amount of 89%. Interestingly, students got around 65% of answers correct after seeing a combination of text and visuals, compared to less than 40% of answers correct after reading a text comprised of words alone. Similar results have also been obtained in object assembly tasks [14]. Thus, there is strong evidence within the psychology community that the use of visual content has a profound effect on increasing retention and comprehension when compared to text alone.

In this chapter, we turn to the multimedia learning principle and explore the effectiveness of visualizations in conveying explanations to human users. Drawing inspiration from work on visualizing classical planning problems, we propose a visualization framework that can represent the action-space and state-space of planning problems, serving as a medium for communicating explanations between an AI agent and a human user. We introduce two taxonomies of explanations that our framework can visualize: (1) *Domain-based explanations*, addressing discrepancies in action models, and (2) *Problem-based explanations*, arising from differences in initial or goal states. Our proposed framework is agnostic to the method of explanation generation.

**Contributions:** The main contributions of this chapter are:

1. A visualization system for explanations in planning settings.
2. A taxonomy of explanations: domain-based and problem-based.
3. An empirical demonstration of our system's efficacy through a comprehensive user study, comparing it against a text-based baseline.

By developing this visualization framework, we aim to bridge the gap between the generation of logical explanations and their effective communication to users, potentially enhancing understanding and reducing misinterpretations in human-AI interactions.

## 3.2 Background

### 3.2.1 Related Work

The fundamental problem we addressed in this chapter is formulated around the *model reconciliation problem* (MRP) [22]. Existing work has mostly focused on developing algorithms for generating explanations [157, 156, 158], and not on how they are to be conveyed to a human user; a common thread is that the explanations are communicated to users through text messages.

There has been some effort by the planning and scheduling community to create user interfaces for planning and scheduling problems [46]. While some work aims to show users the space of alternate plans [54, 97, 23], others aim to create systems to aid users in the creation of plans (e.g., Planimation [26]) or for assistance with domain modeling (e.g., Conductor [16]).[3] These kinds of systems are essential steps towards the creation of a unified planning interface, especially when humans are involved in the loop. For a system aiming to provide the complete planning pipeline to a user, a key requirement for the XAIP community is the creation of systems to deliver explanations to users in an interactive and intuitive manner.

Towards this goal, researchers have created systems using explanations for human-in-the-loop planning. For example, RADAR [133, 56] and RADAR-X [154] make use of contrastive explanations in addition to plan suggestions to develop decision-support systems for interactive explanatory dialogue with users.

Another recent system [36] discusses the design of an iterative planning interface that takes user preferences into account while helping them create plans via plan property dependencies. While these systems make use of interactive user interfaces, and the latter system uses a visualization to show plan execution, they all present explanations in text, and do not focus on how effectively the explanations are delivered. To the best of our knowledge, this work is the first attempt to investigate to what extent visualizations are an effective medium for conveying explanations to users in an MRP setting.

[3]We use both Planimation and Conductor as inspiration for the VizXP framework and discuss the details in a later section.

### 3.2.2 Classical Planning

A *classical planning* problem, typically represented in PDDL [52], is a tuple $\Pi = \langle D, I, G \rangle$, which consists of the domain $D = \langle F, A \rangle$ – where $F$ is a finite set of fluents representing the world states ($s \in F$) and $A$ a set of actions – and the initial and goal states $I, G \subseteq F$. An action $a$ is a tuple $\langle pre_a, \mathit{eff}_a^{\pm} \rangle$, where $pre_a$ are the preconditions of $a$ – conditions that must hold for the action to be applied; and $\mathit{eff}_a^{\pm}$ are the addition $(+)$ and deletion $(-)$ effects of $a$ – conditions that must hold after the action is applied. The solution to a planning problem $\Pi$ is a plan $\pi = \langle a_1, \ldots, a_n \rangle$ such that $\delta_\Pi(I, \pi) = G$, where $\delta_\Pi(\cdot)$ is the transition function of problem $\Pi$. The cost of a plan $\pi$ is given by $C(\pi, \Pi) = |\pi|$. Finally, a cost-minimal plan $\pi^* = \text{argmin}_{\pi \in \{\pi' | \delta_\Pi(I, \pi') = G\}} C(\pi, \Pi)$ is called an optimal plan.

### 3.2.3 Model Reconciliation Problem

A *model reconciliation problem* (MRP) [22] is defined by the tuple $\Psi = \langle \Phi, \pi \rangle$, where $\Phi = \langle M^R, M_H^R \rangle$ is a tuple of the agent's model $M^R = \langle D^R, I^R, G^R \rangle$ and the human's approximation of the agent's model $M_H^R = \langle D_H^R, I_H^R, G_H^R \rangle$ , and $\pi$ is the optimal plan in $M^R$. For brevity, we will refer to $M_R$ and $M_H^R$ as the "agent" and "human", respectively. A solution to an MRP is an explanation $\epsilon$ (e.g., a set of model information) such that when it is used to update the human's model $M_H^R$ to $\widehat{M}_H^{R,\epsilon}$, the plan $\pi$ is optimal in both the agent's model $M^R$ and the updated human model $\widehat{M}_H^{R,\epsilon}$. The goal is to find a cost-minimal explanation, where the cost of an explanation is defined as the length of the explanation.

In addition to adding information to the user's model, an explanation might also involve the removal of information from a user's model such that it is consistent with the agent's explanation [155]. Therefore, our notion of explanation is defined as follows:

**Definition 1** (Explanation)**.** *Given an agent $M^R$, a user $M_H^R$, and an optimal plan $\pi$, assume that $\pi$ is only optimal in $M^R$. Then, $\epsilon = \{\epsilon^+, \epsilon^-\}$ is an explanation from $M^R$ to $M_H^R$ for $\pi$ if $\pi$ is optimal in $\widehat{M}_H^{R,\epsilon} = (M_H^R \cup \epsilon^+) \setminus \epsilon^-$, where $\epsilon^+ \subseteq M^R$ and $\epsilon^- \subseteq M_H^R$,*

As such, $\epsilon^+$ is the addition of model information to the user's model from the agent's model and $\epsilon^-$ is the removal of information from the user's model. The latter is important in order to account for any inconsistencies arising when adding new information to the human's model.

## 3.3 Taxonomy of Explanations

Most MRP algorithms look at explaining either optimal or valid plans to human users [22]. Towards that end, such explanations, using insights from social sciences [103], are considered according to three main properties: *Social explanations* for modeling the expectations of the explainee; *selective explanations* for choosing the explanations among several competing hypotheses; and *contrastive explanations* for differentiating the properties of two competing hypotheses. Among these properties, contrastive explanations have received a lot of attention [60]. However, all explanations share two common elements; They either express discrepancies between the domain-action models of the agent and the user (i.e., *domain-based* explanations) or involve differences in the initial and/or goal state assumptions of the planning problems of the agent and the user (i.e., *problem-based* explanations). Below, we formalize these two notions as characteristics of explanations stemming from MRP scenarios.

**Domain-based Explanations:** Assume an agent $M^R$, a user $M^R_H$, and a plan $\pi$ that is optimal in $M^R$ but not $M^R_H$. We say that an explanation from $M^R$ to $M^R_H$ for $\pi$ is a *domain-based* explanation, denoted by $\epsilon_d$, if all of its elements involve the action dynamics in $M^R$ and/or $M^R_H$. In other words, the elements of the explanation must involve addition (or removal) of actions, preconditions of actions, or effects of actions to (or from) $M^R_H$. Note that we make the assumption that explanations involving the addition or removal of an entire action can be specified as a set of preconditions and/or effects accompanied by the name of the action.

**Problem-based Explanations:** Assume an agent $M^R$, a user $M^R_H$, and a plan $\pi$ that is optimal only in $M^R$ but not $M^R_H$. We say that an explanation is a *problem-based* explanation, denoted by $\epsilon_p$, if all of its elements involve the addition (or removal) of initial and/or goal states to (or from) $M^R_H$.

These categories make intuitive sense as any planner takes, as input, a domain file and a problem file, which fully specify the planning problem $\Pi$. We will utilize these two types of explanations while evaluating our visualization framework. The information conveyed in domain-based explanations would translate across problem instances and would be something more commonly needed for novices learning about the domain, while problem-based explanations could arise regardless of expertise level due to misunderstanding the problem.

We also note that these two kinds of explanations are not isolated, and some MRPs can have solutions that include both types of explanations.

There is one other scenario in which explanations are needed: when the user makes mistakes in creating the plan even after having knowledge of the model. We posit our visualization framework to be able to aid humans in this regard as well, but for the purpose of analysis, we focus on the two categories discussed above.

Further, the information provided in all explanations discussed above falls into one of the following two categories:

- **Action-Space Information:** Given a planning problem $\Pi$, and an associated domain $D$ containing actions $A = \langle pre_A, eff_A^{\pm} \rangle$, the action-space information corresponds to information about the preconditions and effects for each action in $D$. Domain-based explanations will contain this kind of information.
- **State-Space Information:** Given a planning problem $\Pi$, a plan $\pi$, and a sequence of states $S$ involved in the execution of $\pi$, the state-space information corresponds to information about the predicates in each state in $S$. Problem-based explanations, which address errors in the initial and goal state, contain this kind of information.

It is easy to see the parallels between the two kinds of explanations and the two kinds of information discussed above. These categories are also interlinked vis-a-vis the fact that the state influences what actions are possible, and the action dynamics decide what the next state will be. Thus, an ideal system for presenting explanations should be able to convey both types of information to the users, while maintaining the link between them. In the next section, we discuss two existing visualization systems that present action-space and state-space information and use those ideas to motivate the design of a framework capable of visualizing plans and their execution as well as presenting explanations, using both state-space and action-space information.

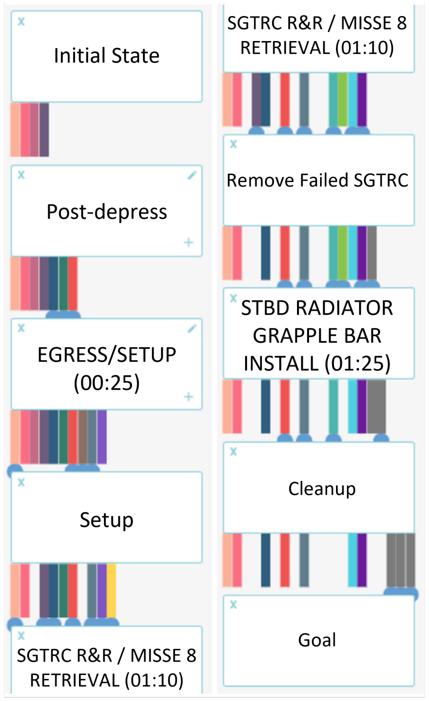

Initial State
Post-depress
EGRESS/SETUP (00:25)
Setup
SGTRC R&R / MISSE 8 RETRIEVAL (01:10)
SGTRC R&R / MISSE 8 RETRIEVAL (01:10)
Remove Failed SGTRC
STBD RADIATOR GRAPPLE BAR INSTALL (01:25)
Cleanup
Goal

Figure 3.1: Illustration of Conductor, one of the approaches used as an inspiration for the action space visualization [16].

## 3.4 Visualization Framework

In this section, we describe a framework that can be utilized to design and deploy visualizations for presenting explanations to users. Borrowing elements from existing work in plan visualization, such a framework should be able to show all kinds of explanations discussed in the previous section. Given an explanation based on the user's plan and the agent's plan, it should support the visualization of the following information for the human user's model: (1) Plan length; (2) Wrong/missing initial/goal state; (3) Wrong/missing preconditions; (4) Wrong/missing effects; and (5) Wrong/missing actions.

As noted earlier, most MRP-based explanations are *contrastive* and, typically, involve a *foil* provided by the human user in terms of an alternative plan [141]. In addition, the context of the user's own plan may help them in better understanding the agent's explanation. Hence, the user's plan provides an excellent window for presenting explanations, a fact that is useful for the visualization techniques proposed in the following sections.

**Action-Space Visualization: Fact Flows**

“Fact routes” from Conductor [16] provide an easy way to visually represent preconditions as “stations” on each action that need to be filled, and effects as routes originating from that action. Conductor, combined with Marshal [15], is aimed at helping users create domains and plans concurrently. Using fact routes to show the evolution of facts over time, it aims to use interactions with users to facilitate creating correct plans and domains. We found that Conductor’s framework was limited by the fact that the length of the plan, as well as the number of predicates involved, increases the number of fact routes to the extent that it might overwhelm users (see Figure 3.1). This makes it unsuitable for all but the simplest of domains that contain few predicates and have short plans.

To remedy this, we introduce a simplification to Conductor. Instead of tracking all fact routes as individual columns, we visualize just the routes moving into and going out of each action as *fact flows*. Optionally, a user interaction like a click may show the history of the fact route for any particular action, thus retaining all relevant information for users who require it. This reduces clutter and allows us to present longer plans with domains that can contain larger number of predicates within a limited space. For example, consider a fact flow `in(truck1, city2)`, and an action that does not use truck location as precondition; Conductor would show this fact flow before the action, while in our simplification, this unnecessary fact flow would be hidden.

In order to visualize explanations, we propose two methods: (1) *Highlight-based* and (2) *Port-based* methods. Using the former, the precondition/effect flows are highlighted based on whether they are unaffected (colored grey), wrong (colored red), required/missing (colored yellow), or required/present (colored green). The latter method employs “ports” for the preconditions and effect of each action, an extension of the “stations” used in Conductor. Ports can be colored based on whether they are unaffected (colored blue), wrong (colored red) or required (colored green), and fact flows can be either missing (not plugged in) or present (plugged in). Figure 3.2 shows an example of the same information conveyed using both methods. Note that we present this just as a stylistic choice.

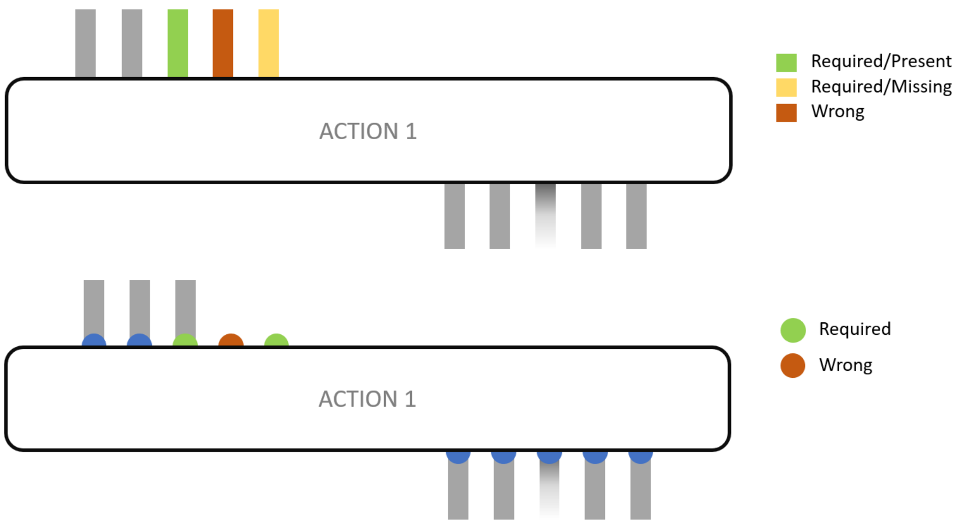


Figure 3.2: An action-space visualization example, with preconditions at the top of each action, and effects at the bottom. Deletion effects are represented as a flow fading out. Top: The highlight-based explanation visualization; Bottom: The port-based explanation visualization.

One additional modification we make to Conductor's design is the introduction of the fact flows to the initial state. This allows us to visualize any predicates affected by a problem-based explanation by treating the initial state as a pseudo-action for which ”preconditions” need to be added or removed (i.e., modifications to the initial state).

### 3.4.1 State-Space Visualization: Abstraction

While the action-space framework is sufficient to visualize all explanations, it fails to show information about the state of the world at certain times throughout the execution of the plan. Many planning domains contain features that can enable humans to think about them in terms of physical abstractions. Simple classical domains like BlocksWorld and Logistics naturally lend themselves to the physical space, presenting users the ability to keep track of the current state of the world by tracking their positions in their mental space. Moreover, planning visualization interfaces like Planimation [26] and WebPlanner [97] utilize state-space visualizations to assist in planning and display plan execution. Planimation, in particular, allows users to create visualizations for plans using an animation profile to specify how different elements are visualized.

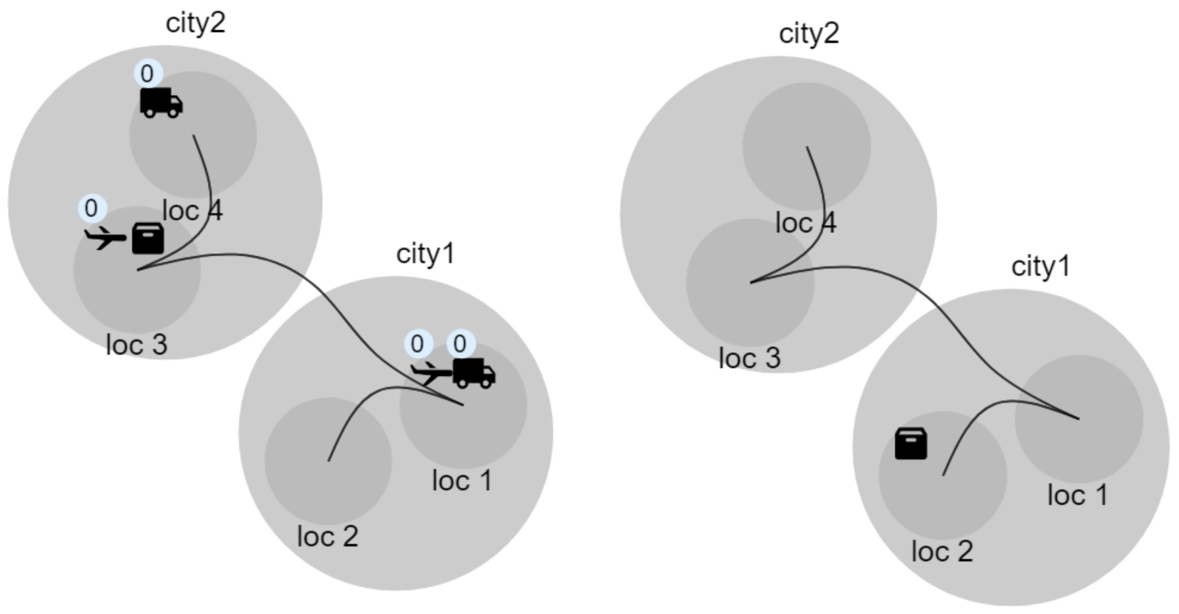


Figure 3.3: A state-space visualization example. Left: The initial state; Right: The goal state.

Inspired by such systems, we propose an abstraction-based plan visualization which we extend to display explanations as well.

We describe states and transitions between states using *containers* (objects in the world that can "contain" others), contents (objects that can be "contained" in others), and links (ways for contents to move between containers). We note that state-space visualizations like Planimation also fall within the framework described here.

**Abstract Space:** The positional relationships between various objects (e.g., `On`, `In`, etc.) and the motion of objects between *containers* form the basis of the state-space abstraction visualization. We present one hierarchy-based approach for visualizing state-space information for planning domains that possess these kinds of relationships. Concretely, this approach requires the following properties:

- Domain objects are classified as either *containers*, *contents*, or both.
- Domain objects are either *movable* objects or *immovable* objects.
- All domain actions must move items between containers.

- Predicates must identify and fully specify the relationship between objects for any state.

Note that in some cases it might be necessary to introduce pseudo-predicates to allow for the last property. For example, in BlocksWorld, `onTable(a)` can be "reified" to `onTable(a, table)` with "table" being a dummy object created to represent the implicit table. This is only needed for the visualization, and need not change the planning process.

It is also possible to relax the requirement for all actions to move items between containers, if they modify "properties" of objects. However, this can only be done if the property lends itself to visualization, e.g. color or on/off status, which can be easily shown using a change in color or an added status indicator. However, this is highly domain dependent, and there might be properties that cannot be visualized.[4]

Any planning domain satisfying the above properties can be used to create a visual representation of the state of the world at any given step. We can visualize a network of containers connected by edges that movable contents or containers can traverse, with each edge-type represented by certain actions (e.g., in Logistics, the `move-airplane` action moves an airplane between two locations), with each action causing an object to move across one of these edges, with optional animations.

This is a basic setup and may be specialized and modified for each domain. For example, Figure 3.3 shows the initial/goal states for a Logistics problem.

Within the state-space visualization, it is much easier to see "why" some positional relationships are not true. Simple preconditions like the requirement for different actions to have objects "in" certain locations are intuitively shown in the state if true, and effects of an action can be clearly seen with the motion of objects across these edges. This can help users during plan creation.

For presenting explanations within the state-space visualization, we employ the highlighting technique discussed in the action-space visualization. For each state in the execution of

[4]It is possible to broaden the scope of the container-based visualization via augmentations, to generalize it to more planning domains. For example, it is not trivial to fit object properties like color or capacity (e.g. fuellevel in NoMystery) into the container framework. However, not all information needs to be captured in the abstraction. With simple augmentations to denote properties (like a blip with current fuel), even such properties can be described in the state space visualization. However, that is domain-specific, and thus will need to be done on a case-by-case basis.

the plan, starting from the initial state, we display the current state with respect to the actions that are executed in the human's plan, using the agent's domain. We assume if an action's preconditions are not met, none of its effects take place. Each object involved in a missing/wrong precondition or effect is shown similarly to the highlight-based approach in the action-space visualization, and a tooltip can further reveal information about the explanation.

### 3.4.2 Integrated Action- and State-Space Visualization

We now present *Visualizations for eXplainable Planning* (VizXP), a visualization framework that combines the action-space and state-space elements discussed previously. It can visualize plans and their execution as well as present explanations to human users, using both state-space and action-space information. The inclusion of the action-space information also conveniently presents a simple way for users to select and view different states after the execution of each action. Highlights in the action-space visualization provide an overview of the steps where the users' plan went wrong, with the state-space visualization providing more detail about what exactly went wrong. In addition, VizXP also allows users to debug and correct their plans during the creation phase.

We can see how problem-based explanations would be better represented in the state-space visualization, showing the complete start and goal state, in addition to any errors. Similarly, the action-space visualization would be better for understanding domain-based errors, making it clear which conditions were wrong, for which action. However, both visualizations work together in synergy, augmenting the information provided by the other with context.

Finally, we note that depending on the application, an additional visualization might present the agent's correct plan alongside the human's plan, similar to contrastive explanation methods. This can then be used to display the 'required' information with the human's plan only visualizing the missing and wrong information. This is required for domain-based explanations that involve actions not in the user's plan.

The visualization is an augmentation to explanations, and should be used as an aid in helping users comprehend them. Here we also provide some intuition for extending the framework to methods of explanation not included in our evaluation with the prototype:

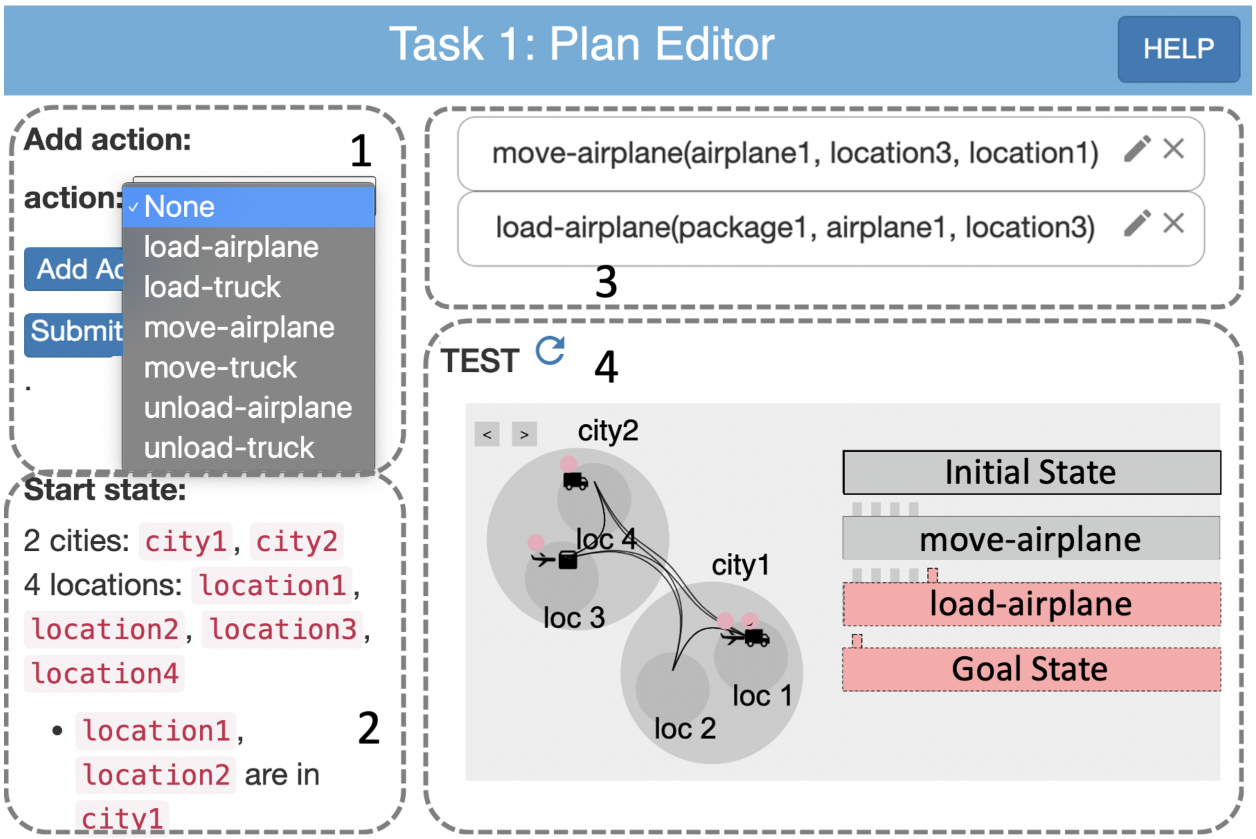
- For contrastive explanations with foils that are not complete user plans, we present two alternatives. The first is to create the corresponding hypothetical plan and compare the agent's plan with it. Otherwise, it is also possible to just visualize the portion of the two plans that differs, omitting the common initial part of the plan and tail, if any.
- In our analysis, we provide one-shot explanations, however it is possible to present sequential explanations using the same framework, if an algorithm exists to generate such explanations.

## 3.5 Human-Subject Study

We now discuss the setup for our evaluation, where we compared VizXP against a text-based benchmark, an approach commonly used by current state-of-the-art systems [36, 154], through a user study conducted on the online crowdsourcing platform Prolific [116]. The goal of the evaluation is to investigate to what degree MRP explanations presented by VizXP are effective and easily understood by humans compared to the text-based benchmark. Based on insights from other research communities, such as the multimedia learning principled described in the Introduction section, we hypothesize that *participants will perform significantly better with VizXP compared to the text-based baseline.*

**Knowledge of the Human Model:** Existing MRP solvers require knowledge of the human model $M_H^R$, which is a difficult assumption to satisfy in practice. To combat this, in the user study, we first described a tweaked "wrong" model to the user and asked them to create a plan using that model. We used users' ability to create a plan that is valid in the provided model as a proxy for them having the said model, and only considered participants who succeeded in doing this as candidates for the study. We realize that this has the potential to bias the study towards people who can create plans in the first place, but we make this trade-off to satisfy the conditions for model reconciliation.

Once users created a plan with the wrong model, they were provided MRP explanations and were asked to answer a series of questions as well as correct their plans based on those explanations. The users' answers to those questions as well as their ability to correct their plans reflect their understanding of the explanations provided.



move-airplane(airplane1, location3, location1)

load-airplane(package1, airplane1, location3)

Start state:

2 cities: `city1`, `city2`
4 locations: `location1`, `location2`, `location3`, `location4`

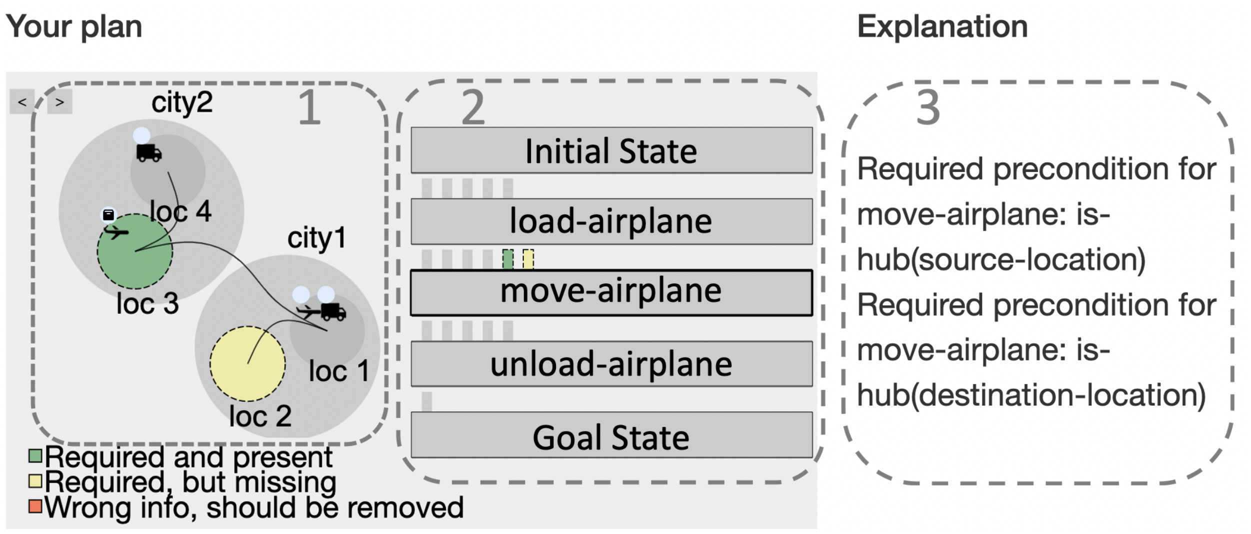
- `location1`, `location2` are in `city1`

Figure 3.4: A view of the plan editor for the user study. (1) Action selection; (2) The initial and goal states; (3) User's current plan; (4) Test visualization showing validity of the plan.

**Domain and Problem:** Our choice of domain was the Logistics domain [101], which we simplified to make it less complex for people with no background in planning.

Predicates `in-city`, `in`, and `at` were combined into one `in` predicate to avoid confusion. We renamed `airports` to `hubs` and changed the corresponding predicates to allow for some ambiguity to introduce errors in the domain. We created a simple problem with two cities containing two locations each. One location within each city is a hub. Figure 3.3 shows the initial and goal states for this problem. There are two airplanes and two trucks distributed across the locations, and one package that needs to be transported to the goal city. We considered two changes for the "wrong" model of the user:

- **C1:** We modified the action `move-airplane` by removing its precondition that the source and destination location must be hubs. Therefore, a domain-based explanation is needed to correct this error.



Required precondition for move-airplane: is-hub(source-location)
Required precondition for move-airplane: is-hub(destination-location)

Figure 3.5: A view of the explanation visualization in the user study. (1) The state-space visualization; (2) The action-space visualization; (3) The text-based explanation.

- **C2:** We changed the initial location of the package, thereby requiring a problem-based explanation to correct this error.

**Prototype Implementation:** We used elements from VizXP to create a visualization system for the selected domain. For the state-space visualization, we used circles to mark cities and locations and icons for trucks, airplanes, and packages. There were two types of links: Transporting objects between locations (visible); and loading and unloading packages from trucks or airplanes (invisible). An alternate design could further separate these out by type, having different edges for the `move-airplane` action and for the `move-truck` action, but we chose to use only two kinds for the sake of simplicity. We used animations to show objects moving between containers. For the action-space visualization, we used a limited version of the system where only the flows into the current action (preconditions) are visualized. Since none of the explanations would involve any change to the effects of actions, we decided to omit effect flows from the visualization. Figure 3.5 shows a view of the visualization presenting an explanation. We created the implementation to run on a browser, using Flask and Python as back-end, and D3.js and JavaScript for the front-end.

As VizXP is agnostic to the choice of algorithm to generate MRP explanations, we used one of the existing state-of-the-art solvers to generate the explanations. To display the explanations, we chose the highlight-based approach, with tooltips providing additional information.

|  |  | Pop. Size | Correction Ratio | Comp. Score |
|---|---|---|---|---|
| VizXP | all users | 87 | 0.701 | 5.402 |
|  | CS users | 30 | 0.800 | 5.400 |
|  | domain-based exp. | 44 | 0.681 | 5.091 |
|  | problem-based exp. | 43 | 0.721 | 5.720 |
| Text | all users | 83 | 0.627 | 4.759 |
|  | CS users | 41 | 0.585 | 4.340 |
|  | domain-based exp. | 40 | 0.625 | 4.475 |
|  | problem-based exp. | 43 | 0.628 | 5.023 |

Table 3.1: User study results.

**Study Design:** The study was designed to have two groups: The experimental group using VizXP and the control group using text. Each group was tested on two types of "wrong" models, modified using changes **C1** and **C2** (see "Domain and Problem" paragraph), each requiring a different type of explanation. Therefore, we have **four scenarios** in total, which we tested independently. We created two tasks for each user as follows:

- **Task 1:** Participants were asked to create a plan based on the modified domain and problem information provided to them using a simple plan editing interface. This interface also allows users to "test" their plans, which will provide information about the errors in their plans due to their misunderstanding of the provided domain and problem information. Depending on the scenario, this interface might be either VizXP[5] (shown in Figure 3.4) or a sequence of steps with markers for incorrect actions. A participant succeeded in Task 1 if they created and submitted a valid plan given their domain and problem. Users that succeeded in Task 1 continue to Task 2, and users that failed in Task 1 were filtered out and ignored. This is important since MRP explanation-generation algorithms assume that the user's model is known.
- **Task 2:** We informed the participants that the initial domain and problem information provided to them contained errors and presented explanations for those errors using either VizXP or text based on the group of the participant. They were then asked a series of questions to evaluate their understanding of the explanation provided (**Task 2a**). Then, they were shown the plan editor again and asked to correct their plan, this time without

[5]Users in the experimental group are shown VizXP in Task 1 to ensure that they are familiar with the system before receiving an explanation using that interface to eliminate any learning effects.

the ability to "test" their plans for correctness (**Task 2b**). A participant succeeded in Task 2b if their corrected plan is valid in the agent's model.

To incentivize participants to provide answers to the best of their ability, we provided a bonus to participants who succeeded in Task 1 and an additional bonus to participants who also succeeded in Task 2b. Further, we also included two questions for attention checks in the study, where participants were asked to type a particular string or select a particular answer in a multiple-choice question. Participants who wrongly answered both questions were removed.

Each participant had the following interactions in the study: (1) They arrive at the webpage following the link from Prolific, where they enter their demographics and some information on their educational background. (2) To ensure that they have the background necessary to solve the tasks, they are given tutorials on classical planning, the logistics domain, and the plan editing interface. (3) Following the tutorials, they are asked to complete Task 1. (4) If they succeeded in Task 1, they are asked to complete Tasks 2a and 2b. (5) All participants, including those who failed Task 1, are then asked to provide feedback on the system's usability [61] and are informed of their payments before being redirected back to Prolific.

**Participants:** We conducted the study with 200 participants (66 female, 132 male, 2 non-binary) with each of the four scenarios getting 50 random participants. Out of the 200 participants, only results from 170 participants were used as 30 participants failed Task 1 and/or wrongly answered the questions on attention checks.

**Measures:** To measure comprehension of explanations provided, we used the following measures:

- **Correction Ratio:** Proportion of users who succeeded in Task 1 who also succeeded in Task 2b.
- **Comprehension Score:** Number of questions users answered correctly in Task 2a.

## 3.6 Experimental Results

We now discuss the results of our evaluations using the measures above to evaluate the performance of VizXP in aiding users understand explanations provided. [6] For statistical significance, we used a $p$-value of 0.05 as a threshold.

Table 3.1 summarizes our results for four different groups of users who succeeded in Task 1: *all users*, the subgroup of users with a *computer science* (CS) background, the subgroup of users who were given the model with change **C1** and *domain-based explanations*, and the subgroup of users who were given the model with change **C2** and *problem-based explanations*. For each group of users, we report the population size of that group and our three measures. We now discuss the results for each of those measures:

- **Correction Ratio:** More users were able to accurately correct their plans with VizXP (= 70.1%) than with the text-based baseline (= 62.7%). This difference is not statistically significant with a two-proportion $z$-test ($p = 0.303$), but the difference is similar for both domain-based and problem-based explanations. Among the subgroup of users with a CS background, the difference is larger – 80.0% of users succeeded in correcting their plans with VizXP compared to 58.5% with the text-based baseline. The likely reason is that a fraction of users without a CS background failed to sufficiently understand the planning problem and succeeded in Task 1 due to the aid of the "test" functionality in the plan editing interface. It is also possible that CS users performed better with VizXP because of their familiarity with graph-like visualizations.
- **Comprehension Score:** Similar to the previous two measures, users scored better on this measure with VizXP (= 5.402 out of 7 questions answered correctly on average) compared to with the text-based baseline (= 4.759). This difference is statistically significant ($\chi^2(1, N = 170) = 5.2252$, $p = 0.0223$, and $\epsilon^2 = 0.0371$ with Kruskal-Wallis H non-parametric tests). This difference and statistical significance is further amplified among the subgroup of users with a CS background. Figure 3.6 plots the distribution of comprehension scores for all users.

[6]The results for the study and the code for the prototype implementation can be found at https://github.com/YODA-Lab/VizXP-User-Study

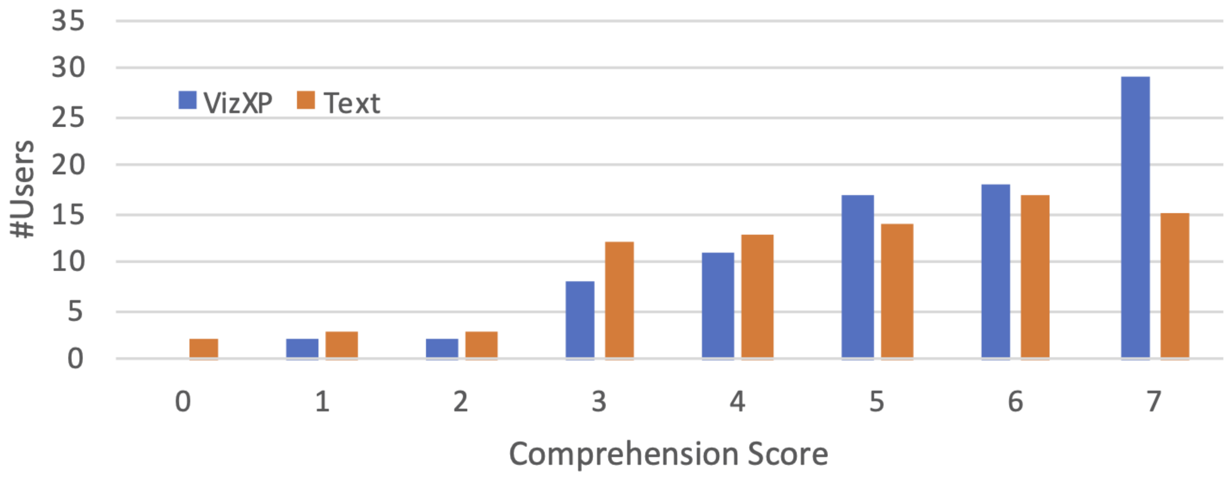


Figure 3.6: Comprehension score distribution for all users.

The trends above generally apply for the two subgroups who were given domain- and problem-based explanations also. However, there are not much noticeable differences between the two subgroups, indicating that VizXP performed equally well for both subgroups, as did the text-based baseline.

While the statistically significant results with the comprehension score measure are consistent with our expectations, we were surprised by the lack of statistical significance on the results with the correction ratio measures. We suspect the reason is that a non-trivial number of users succeeded in Task 1 despite not understanding the planning problem well due to the aid of the "test" functionality.

Additionally, we were surprised to find that 11 users answered at least 6 of the 7 comprehension questions correctly, implying that they understood the explanations well, but failed to accurately correct their plans. This observation implied that their error is due to typos and not misunderstanding of the explanations. This observation thus hints that the comprehension score measure, for which VizXP is statistically better than the text-based baseline, is more accurate at measuring how well users understand the explanations provided than the correction ratio measure.

Finally, we note some limitations of this work as well. In the user study, we require users to create full alternate plans, but in many contrastive explanation systems, users are also able to provide partial foils. It is possible to envision a system designed using VizXP that can use partial foils by splitting the plan at the point of interest and comparing the partial plans, but further work is required to test that ability and its applicability to real systems. Further, we design the user study to ensure the users have a model similar to the one we desire. In practice, obtaining the human model is an open problem in XAIP, which we hope future work will address.

## 3.7 Conclusion

This chapter focused on the effective communication of explanations to human users. We introduced VizXP, a visualization framework designed to convey explanations generated through model reconciliation processes in planning. By leveraging a combination of state-space and action-space visualizations, VizXP can effectively represent both domain-based and problem-based explanations. This work lays some foundations for the successful deployment of explainable AI planning systems in real-world scenarios, where the ability to convey complex reasoning in an accessible manner is paramount.

Our human-subject study yielded the following key findings: (1) Users, on average, demonstrated better understanding of explanations when using VizXP compared to a traditional text-based baseline; and (2) The improvement was particularly pronounced among users with a computer science background, indicating VizXP's potential for enhancing explanation comprehension even among experienced users.

These findings underscore the importance of considering the medium of explanation delivery, an aspect often overlooked in the pursuit of more sophisticated explanation generation algorithms. VizXP complements the explanation techniques developed in previous chapters by providing an effective means of communicating these tailored explanations to users.

Looking ahead, several promising avenues for future research emerge. We could explore the effectiveness of VizXP across a broader range of domains and user demographics to assess its generalizability and identify potential areas for improvement. The development of dynamic visualization techniques that adapt in real-time to user feedback and understanding is another exciting direction, potentially incorporating machine learning algorithms to personalize the visual representation based on individual user preferences and comprehension patterns.

Additionally, integrating VizXP with other modalities of explanation, such as natural language generation or interactive dialogues, could create multi-modal explanation systems that cater to diverse learning styles and preferences. Lastly, investigating the long-term impact of visualization-based explanations on user trust, decision-making, and overall satisfaction with AI systems in real-world applications would provide valuable insights for the field of explainable AI.

While not directly involved in the upcoming chapters, we extend the learning that humans understand better with visualizations to develop methods for detecting fairness concerns in real-world systems for resource allocation. The insights gained from this work have influenced research in other areas of explainable AI planning, especially in the design of user studies and the evaluation of explanation effectiveness. The designed framework has been used in work studying logic-based explanations in classical and hybrid planning [157].

# Chapter 4

# Identifying Fairness Concerns I: Energy Disruptions During Winter Storm Uri

## 4.1 Introduction & Contribution

Having established methodological foundations for communicating complex AI decisions, we now turn our attention to the first of two case studies that demonstrate why fairness in algorithmic systems demands urgent attention. This chapter dives into an empirical investigation of real-world fairness violations, examining how infrastructural decisions during crisis events can perpetuate systematic inequalities. While brief, this analysis provides critical context and motivation for the algorithmic contributions that follow in later chapters.

Winter Storm Uri, which devastated Texas in February 2021, provides a stark illustration of how algorithmic and operational decisions in critical infrastructure can disproportionately impact vulnerable populations. While the storm's immediate causes were meteorological, the distribution of power outages across different communities reveals patterns that raise fundamental questions about fairness in resource allocation during emergencies.

Through our investigation of power outage patterns during Winter Storm Uri, this chapter accomplishes several objectives that directly inform the subsequent algorithmic contributions in this dissertation. First, it demonstrates the critical importance of disaggregated analysis in fairness evaluation, showing how aggregate statistics can mask significant disparities. Second, it reveals how operational decisions made under pressure can have profound equity implications, establishing the real-world stakes for the fair resource allocation algorithms we develop in later chapters. Finally, it illustrates the complex interplay between geographical, demographic, and infrastructural factors that fair algorithmic systems must account for.

This empirical foundation motivates the theoretical and algorithmic work that follows, providing concrete evidence for why we cannot treat fairness as an afterthought in system design, but must instead build equity considerations into the core of our decision-making frameworks from the outset.

## 4.2 Background

Modern infrastructure heavily relies on access to energy to improve the quality of life and provide essential services to citizens. Disruptions in the provision of energy services can have far-reaching consequences, impacting access to safe food, health services, and communication. Winter Storm Uri, which impacted Texas in February 2021, serves as a stark example of the detrimental effects of energy infrastructure failure [18]. A lack of winterization of certain infrastructure necessary for successful power system operations led to widespread power outages, leaving a significant portion of the state's population without heating, water, internet, and cellular service [33]. Tragically, this resulted in numerous deaths and exposed the vulnerability of communities when energy delivery is compromised. Fairness in energy distribution, an often overlooked aspect of energy access, becomes crucial in such scenarios.

This analysis aims to investigate the impacts of disparate energy access during Winter Storm Uri, particularly focusing on the potential injustice along ethnic lines in the distribution of power outages across the state. By analyzing data from a recent study [42], we explore the hourly rate of power outages aggregated by counties in Texas. Our analysis reveals that among rural populations, Hispanic households faced substantially higher outage rates throughout the storm compared to other racial and ethnic groups.

## 4.3 Analysis

The dataset [42] encompasses outage information between the 10th and 24th of February 2021, providing hourly power outage rates for 252 counties (>99% of the counties in Texas), alongside demographic information for each county. This county-level information includes the number of customers (households), the fraction of customers without power during any given hour (outage rate), and the percentage of the population in each county that is Black and Hispanic. Additionally, counties are classified as urban or rural. For our analysis, we concentrate on the period from the 15th through the 19th of February, encompassing the peak of the outages. We examine the hour-by-hour outages among Hispanic, Black (non-Hispanic), and White (non-Hispanic) populations. We combine the county-wise outage rates from the study by Flores et al. [42] with data about race and Latino origin from the 2021 Texas census data [152] to get the outage rates by group. Since we lack granularity of outage data below the county level, we assume all populations within a county suffered equal outage rates.

Upon initial inspection, the distribution of power outages across groups appears reasonably equitable when analyzed from a state-wide aggregate, as shown in Figure 4.1(a). However, over 70% of the population in Texas is concentrated in the 20 most populated urban counties. On the surface, the county-wide outage rates were similar between the urban and rural counties with a 24.8% average outage rate on the 16th and 17th of February in urban counties and a 25.2% average outage rate in rural counties. Since we only have county-level outage rates, we are unable to disambiguate between differences across groups at an intra-county level. Thus, the large urban counties have a dominating effect on the overall statistics, as large swathes of each group's population are assumed to suffer equal outage rates. To adjust for this urban county bias, we remove the urban counties from the analysis, considering only the 170 rural counties. The outage rates for this subset of the data are shown in Figure 4.1(c). Figures 4.1(b) and (d) provide a closer look at this discrepancy, looking at data from the 16th and 17th of February when the highest number of outages were observed. We can see a clear difference in the distribution of outages for rural Hispanic households and other groups.

A clear trend emerges: **within rural counties, Hispanic households consistently experienced higher outage rates compared to other households**. During peak outages, when temperatures were as low as 6°F, rural Hispanic households faced outage rates up to 34% higher than rural White households and 50% higher than rural Black households.

To further explore this disparity, we examined the geographical distribution of the rural Hispanic population as shown in Figure 4.2. We see that many of the counties with a high

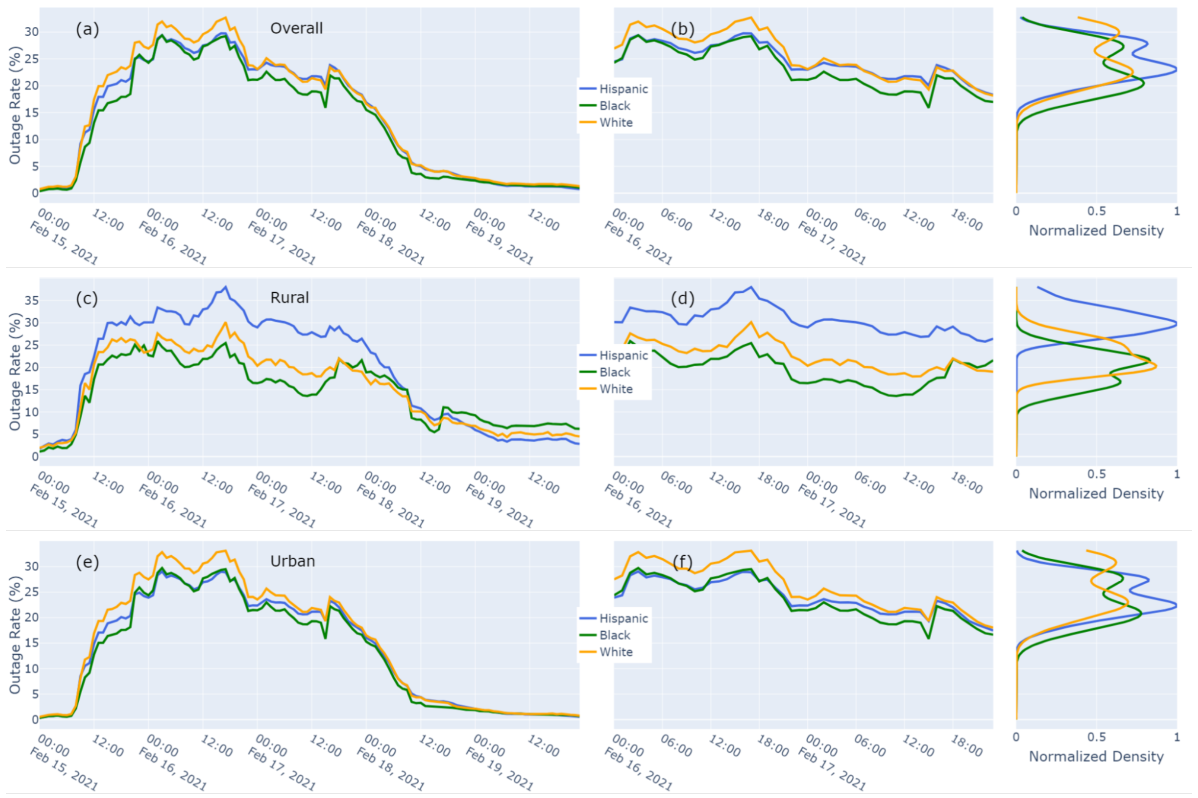


Figure 4.1: Outage rates and their distributions for all Texas counties combined (panels a and b), rural populations (panels c and d), and urban populations (panels e and f) during the 2 days of peak outages (16 & 17 February 2021). For the rural population, the mean relative difference between Hispanic and Black populations was 36.2%, and the mean relative difference between Hispanic and White populations was 26.7%. When aggregated across all Texas counties, these differences were 6.58% and -3.66%, respectively.

fraction of Hispanics also observed a higher outage rate (along the southern border of Texas). This contributes towards the increased average outage rate of the Hispanic rural population over other populations. Specifically examining the number of Hispanic households affected, some counties with a large concentration of Hispanics contribute significantly towards the total number of the Hispanic population suffering outages (Figure 4.2, right).

We also note that the primary causes of outages were issues with electricity production [18], and malfunctions at power plants [89, 40]. ERCOT, the organization operating Texas's electrical grid, also employs Transmission and/or Distribution Service Providers (TDSPs), that control which regions to turn off [89]. In addition, reports describe the blackouts as 'rotating' [40]. Further, only 1.9 GW out of 46 GW of capacity loss was caused by "transmission and substation outages" [89]. This strongly suggests that the regions being

| Compared Groups | Rural | Urban | Overall |
|---|---|---|---|
| Hisp. vs Black | 36.04% | 2.94% | 6.42% |
| Hisp. vs White | 26.29% | -8.57% | -3.77% |
| Hisp. vs Black (Max) | 50.35% | 18.16% | 20.78% |
| Hisp. vs White (Max) | 34.36% | 0.12% | 3.69% |

Table 4.1: Mean and maximum relative difference for different populations, comparing the Hispanic population to the Black and White population.

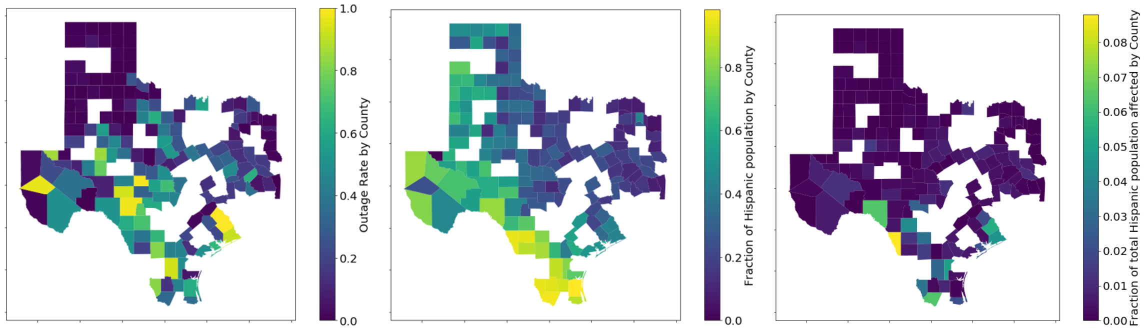

Outage Rate by County
0.0
0.2
0.4
0.6
0.8
1.0
Fraction of Hispanic population by County
0.0
0.2
0.4
0.6
0.8
Fraction of total Hispanic population affected by County
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08

Figure 4.2: Demographic and outage information for rural counties in Texas. (left) Outage distribution across the counties during the two days of peak outages. (middle) Fraction of Hispanic population in each county. (right) Contribution of each county towards the total rural Hispanic households seeing outages.

affected by blackouts were the result of deliberate choices, as opposed to resulting from environmental causes like transmission line failures eliminating power to certain regions.

Seven different TDSPs control and manage power distribution across different areas of Texas, each independently making decisions. Based on the above findings, it is possible to suggest that some TDSPs took the decision to divert power away from certain areas with higher Hispanic populations, leading to the observed disparity. If power was diverted from some other regions, or if there was guidance on equitable energy distribution, a fairer distribution of energy resources could have taken place.

While the observed effect may not have been intentional, the outcome remains unjust. Rural Hispanic households encountered inequitable energy access during a potentially lethal event due to decisions made by the operators. This highlights the urgency of addressing energy justice—an issue that has been overlooked for far too long. Our analysis also underscores the necessity of adopting a more nuanced perspective when evaluating energy equity. Simply examining summary statistics at the overall population level could mistakenly portray the system as fair. Instead, careful consideration is essential to identify different groups that may be disproportionately affected by such events.

## 4.4 Viewing Uri Through the DECA Lens

While our empirical analysis here is constrained by county-level data and limited visibility into operational decisions, it is nonetheless useful to view Winter Storm Uri as an instance of a **DECA** problem, as defined in Chapter 2.

**Agents and evaluation.** In the Uri setting, local service areas (e.g., substations or transmission districts) can be thought of as "agents" producing implicit utility signals. These utilities reflect the benefit of continued service to their customers, such as maintaining heating, powering medical equipment, or preserving food and water access. Although not explicitly modeled in ERCOT's operations, these signals align with the distributed evaluation stage in DECA.

**Central allocation.** Ultimately, ERCOT and the Transmission/Distribution Service Providers (TDSPs) acted as central allocators. Confronted with strict global constraints—insufficient generation capacity to serve all load—they determined which areas to curtail. This matches the centralized allocation step in DECA, in which actions must satisfy system-wide feasibility (e.g., keeping total demand within available supply).

**Fairness implications.** Our findings show that, absent fairness-aware objectives, allocation outcomes can disproportionately burden certain demographic groups. In DECA terms, the allocator solved a feasibility problem but did not optimize for equity over the payoff vector $\mathbf{Z}$ that tracked service received by each group. Embedding fairness into the central allocation—whether as an explicit constraint or as part of a fairness–efficiency trade-off like in DECAF—could mitigate such disparities.

**Opportunities for future work.** With more granular operational data (e.g., feeder-level outage schedules, critical load prioritizations, and demographic overlays), one could instantiate the full DECA framework and directly evaluate fairness functions such as $\alpha$-fairness or the Generalized Gini Function over $\mathbf{Z}$. Such an analysis would quantify how different allocation strategies trade off efficiency and fairness, offering a principled way to design interventions for future crises.

## 4.5 Conclusion

This chapter examined Winter Storm Uri as a real-world instance of a Distributed Evaluation, Centralized Allocation (DECA) problem under extreme scarcity. Our analysis revealed that rural Hispanic households faced substantially higher outage rates than other groups, disparities that were hidden in statewide aggregates. These outcomes underscore the danger of treating feasibility alone as sufficient: without fairness objectives, central allocation can systematically disadvantage vulnerable populations.

Although limited by county-level information and lack of power generation data, this study illustrates the stakes of fairness in resource allocation and motivates the algorithmic frameworks that follow. In particular, it provides empirical grounding for fairness in DECA problems, showing why fairness must be evaluated on disaggregated payoff vectors and embedded directly into allocation mechanisms. The next chapters extend this perspective with further case studies and formal methods for achieving equitable allocations.

# Chapter 5

# Identifying Fairness Concerns II: Using Visualizations to Understand Perceptions of Fairness in Large-scale Ridesharing

## 5.1 Introduction & Contribution

In the previous chapters, we discussed how visualizations can help users understand complex algorithmic decisions, and discussed a real-life case study of fairness violations in energy infrastructure. In this chapter, we combine these ideas to design tools that can help stakeholders understand and evaluate fairness in deployed algorithmic systems, forming the second case study in this dissertation.

The ridesharing domain is an ideal testbed for this investigation: it involves multiple parties with inherently competing interests—passengers seeking reliable and affordable service, drivers pursuing fair compensation, and platforms optimizing efficiency and revenue. Unlike the post-hoc analysis of Winter Storm Uri, here we study systems that make real-time algorithmic decisions that directly shape fairness outcomes across these stakeholders. This creates an urgent need for tools that help algorithm designers, policymakers, and affected communities understand and evaluate the fairness implications of different matching strategies.

On-demand ridesharing is a large-scale, real-time allocation problem with competing objectives. Unlike single-passenger dispatch, drivers carry multiple riders with distinct origins and destinations; matching choices propagate as vehicles move, inducing spatio-temporal

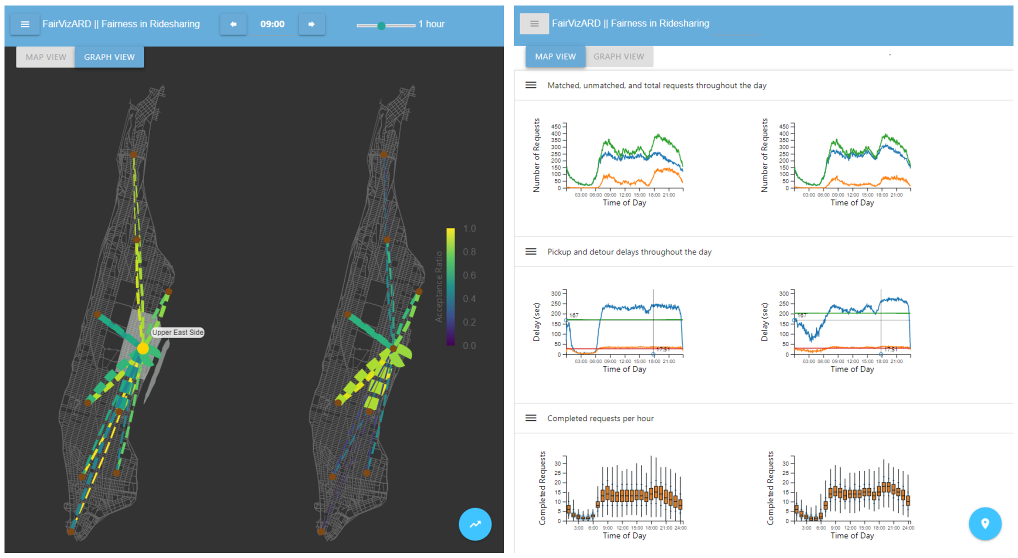

FairVizARD || Fairness in Ridesharing
09:00
1 hour
MAP VIEW
GRAPH VIEW
Upper East Side
Acceptance Ratio
1.0
0.8
0.6
0.4
0.2
0.0
FairVizARD || Fairness in Ridesharing
MAP VIEW
GRAPH VIEW
Matched, unmatched, and total requests throughout the day
Number of Requests
Time of Day
Pickup and detour delays throughout the day
Delay (sec)
Time of Day
Completed requests per hour
Completed Requests
Time of Day

Figure 5.1: From left to right: (i) *Map View* of FairVizARD. (ii) *Graph View* of FairVizARD.

dependencies and neighborhood-level impacts. This makes ridesharing a concrete instance of *Distributed Evaluation, Centralized Allocation (DECA)*: local evaluations of feasible trips feed a central allocator that enforces global capacity and timing constraints.

A key limitation of existing ridesharing matching algorithms is that they primarily optimize monetary or satisfaction proxies (e.g., platform/driver revenue, average delay), so equity only improves when it coincides with those goals. In practice, this can produce systematic disparities—for example, policies that keep drivers in dense zones to maximize throughput while underserving lower-demand neighborhoods. Consequently, there is growing interest in measuring and improving fairness in ridesharing [53, 86, 107]. Yet most approaches distill fairness to a single metric, leaving open the questions of *which* fairness notions stakeholders actually care about and *how* to trade them off across parties (passengers, drivers, platforms).

To tackle this, we introduce **FairVizARD**—a **Fair**ness **Viz**ualization for **A**nalysis of **R**idesharing **D**ata (Figure 5.1). FairVizARD combines map- and graph-based views to expose spatio-temporal patterns in requests, driver activity, and outcomes; supports side-by-side comparison of matching algorithms under the same inputs; and uses aggregation/buffering to keep city-scale data legible without overwhelming users. Rather than prescribing a single fairness objective, the system surfaces stakeholder-specific signals (e.g., acceptance ratios, delays, inter-zone flows) so users can interrogate fairness under their own criteria. Because ridesharing allocations are selected centrally from locally evaluated options under shared constraints, we treat this domain as a concrete *Distributed Evaluation, Centralized Allocation (DECA)* instance and use visualization to audit its fairness.

**Contributions:** The main contributions of this chapter are:

1. **Visualization system:** Design and implementation of *FairVizARD*, an interactive, city-scale visualization for exploratory, stakeholder-centric analysis of ridesharing allocations with split-screen algorithm comparison.
2. **Zonal Fairness:** A geography-aware diagnostic (minimum inter-zone acceptance ratio) that highlights underserved flows and offers a simple and novel lens on spatial equity.
3. **User study and downstream use:** A detailed user study and expert interview benchmark FairVizARD against a numeric-metrics dashboard. Participants matched the dashboard on overall comprehension while *surfacing additional fairness-relevant spatial patterns and stakeholder-specific concerns* that aggregates missed. We carry the top rider- and driver-side concerns forward and instantiate them as objectives/-constraints in the next chapter (Chapter 6).

## 5.2 Background

### 5.2.1 Ridesharing Matching Problems:

While there are many variants of *ridesharing matching problems* (RMPs), in a typical problem definition, a matching algorithm receives as inputs a continuous stream of batches of requests from passengers $\mathcal{U}$ containing their pickup and dropoff locations and the current locations of all the taxis $\mathcal{D}$. It then needs to match as many requests to taxis as possible in a way that optimizes some objective function (e.g., maximizes the number of requests matched) subject to physical constraints $\kappa$ (e.g., maximum number of passengers that can share a taxi) and time-related constraints $\mathcal{T}$ (e.g., passengers with matched requests must be picked up within a certain time).

More formally, a typical RMP is defined by a tuple $\langle \mathcal{G}, \mathcal{C}, \mathcal{D}, \mathcal{U}, \kappa, \mathcal{T}, \Delta \rangle$, where:

- $\mathcal{G} = \langle L, E \rangle$ is a graph representing a road network, where each node $l \in L$ is a location in the city and each edge $e \in E$ connects two neighboring nodes in the graph.
- $\mathcal{C} : E \to \mathbb{R}^+$ is a function that defines the cost $\mathcal{C}(e)$ of each edge $e \in E$ in the graph.
- $\mathcal{D}$ is a set of taxi drivers in the problem.
- $\mathcal{U}$ is the set of requests from passengers, with each request $u_i \in \mathcal{U}$ defined by the tuple $\langle s_i, g_i, t_i, f_i \rangle$ containing the pickup location $s_i \in L$, the dropoff location $g_i \in L$, the arrival time $t_i$ of the request, and the fare $f_i$ that will be paid if the request is served.
- $\kappa$ is the maximum number of passengers that can share one taxi.
- $\mathcal{T}$ is the set of time-related constraints imposed on the problem. Examples include the *pickup delay*, defined as the difference between the arrival time of a request and the time the passenger is picked up, or the *detour delay*, defined as the extra time taken to reach a dropoff location due to ridesharing, being within some user-defined upper bound. Both of these delays are computed using the edge costs $\mathcal{C}(e)$ of the relevant edges taken by the taxi.

A *solution* $M = \{(u_i, d_j), \ldots\}$ is a set of matches between request $u_i \in \mathcal{U}$ and taxi driver $d_j \in \mathcal{D}$ such that all the constraints in $\kappa$ and $\mathcal{T}$ are satisfied. A solution is *optimal* if it optimizes its objective function, which could differ across problem definitions.

RMPs has been extensively studied, where researchers have introduced methods that improve the quality of the matches made in terms of increasing the number of requests matched [94, 95, 96], reducing the pickup and detour delays [7, 64, 96], and increasing the revenue of the drivers [86]. The complexity of RMP algorithms and, as a result, the time taken by algorithms to match drivers to passenger requests, increases with the increase in the value of $\kappa$. As the runtime of the RMP algorithms need to be relatively small, most existing works have either considered assigning one request at a time (sequentially) to available drivers for high value of $\kappa$ [96, 146, 64] or assigning all active requests together in a batch for a small value of $\kappa$ [167, 172]. The sequential solution is faster to compute but the solution quality is typically poor [151]. Researchers have proposed integer optimization approaches for assigning all active requests together for a high value of $\kappa$ [7], and further improved them by including information about anticipated future requests while matching the current batch of requests [134, 95]. This last class of approaches fit neatly into the DECA framework, where the agents (i.e., drivers) evaluate their own payoffs for serving requests (resources), and a central planner (i.e., the RHC) allocates requests to drivers based on these payoffs.

### 5.2.2 Fairness in Ridesharing:

Early work in this area has focused on fairness from the perspective of passengers, where researchers assume that each passenger will get a discount on the fare of their ride depending on the type of *partners* that share the taxi. With this assumption, researchers investigated how the discounts should be distributed across passengers [53]. Later work more formally defined the concept of ridesharing fairness while tackling the issue of transparency by allowing passengers to indicate their preferences for the type of partners they wish to share their rides with [163]. When such preferences are available, researchers can then apply game-theoretic concepts such a stable pairings [44]. More recent work has focused on fairness from the perspective of drivers as well as two-sided fairness for both passengers and drivers [145]. There is also work on group-based fairness that aims at identifying and balancing bias against certain groups of passengers or drivers on grounds like location, race, and gender [107, 164].

The literature thus far has focused only on defining and quantifying fairness in such a way that they can be used as objective functions or additional constraints of RMP algorithms. In contrast, we are interested in developing a visualization system that can be used to better understand the implicit notions of fairness of laypersons using ridesharing systems. None of the works discussed thus far have visualized the fairness of RMP algorithms.

## 5.3 FairVizARD Design

We now discuss the design of FairVizARD, including the design challenges, the various components of the visualization system and the new fairness metric we propose, *Zonal Fairness*. Our goal was to build an interactive visual interface that allows users to explore different notions of fairness and, as a bonus, also allows algorithm developers to analyze the performance of their algorithms across common metrics. To that end, we had the following design challenges:

- **Identifying key points of information to visualize:** With so many possible factors affecting notions of fairness, one of the key challenges was to identify and prioritize the information to visualize. Since the data is primarily spatio-temporal data, we decided to use a map-based visualization (*map view*) to describe the spatial distribution of the data and a graph-based visualization (*graph view*) to describe the temporal distribution of the data.
- **Handling a large volume of data:** Another key challenge was finding out suitable aggregation techniques to handle the large volume of data points that needed to be visualized. It was more of a challenge on the map view, with over over 200,000 requests coming in over a day, and 1,000 drivers active at any time based on the algorithms used to simulate ridesharing. One of the primary ways we tackled this issue was by using binning. Data is aggregated spatially (e.g., within zip codes or neighborhoods) and temporally before being presented to the user. In addition, when multiple layers of visualization are active at the same time, we reduce the information density on the screen by suppressing the visibility of some of the features until users interact with them.
- **Accounting for multiple parties or stakeholders of the system:** Since the goal of FairVizARD is to allow the exploration of fairness across multiple stakeholders (i.e., passengers, drivers, and RHCs), it was important to visualize data relevant to all stakeholders. Based on metrics found in the literature for ridesharing fairness, we identified important pieces of information to display for the different stakeholders, which we describe in the following subsections.

FairVizARD shows data for a single day (24 hours) with one-minute resolutions. This resolution is decided based on the decision epochs considered by the algorithms we run, and can be scaled as required. We demonstrate FairVizARD using Manhattan, NY as the network graph. All algorithms were run using demand data from the NY Yellow Taxi Dataset [112], with 1000 drivers operating across 24 hours. FairVizARD can be used to look at a single algorithm at a time, or to compare two or more algorithms side by side in a split-screen view.

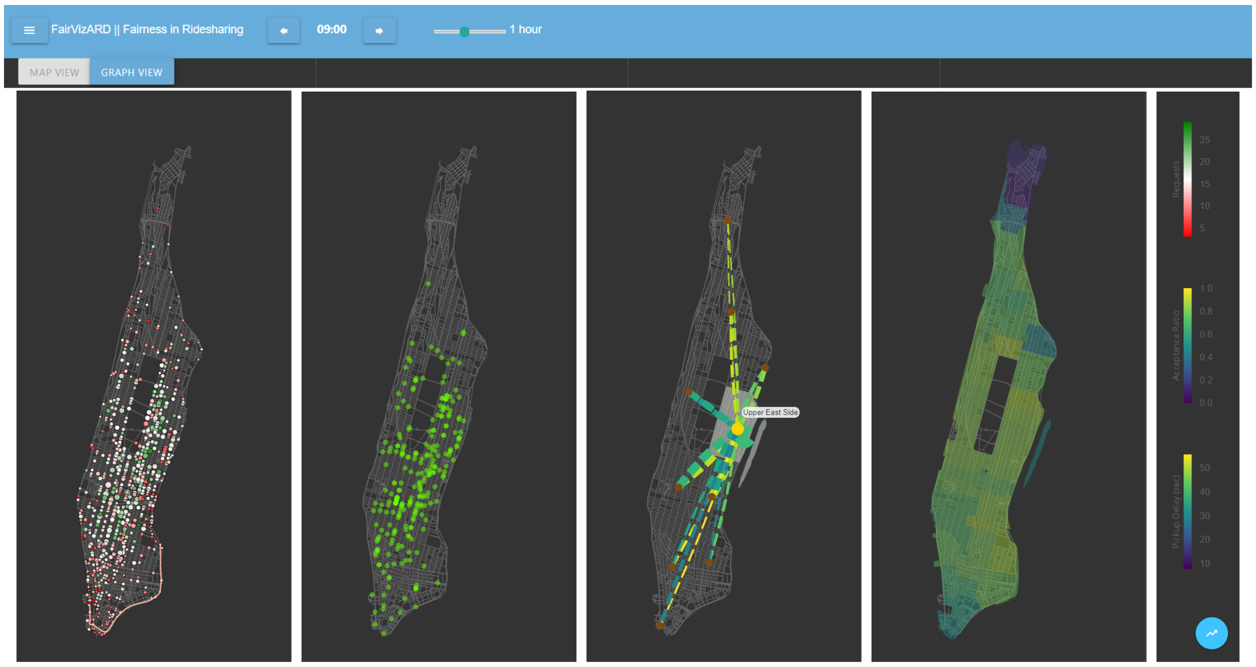


Figure 5.2: Possible Map Views. From left to right: (i) Taxi Data; (ii) Request Data; (iii) Inter-Zone Data; (iv) Individual Zone Data; (v) Legend

### 5.3.1 Map View

Recall that the data to visualize is primary spatio-temporal data. Therefore, FairVizARD is composed of two views: A *map view* that uses a map-based visualization to visualize the spatial distribution of the data and a *graph view* that describe the temporal distribution of the data. We now describe the map view and discuss the graph view later.

Figure 5.1(i) shows the map view displaying a specific type of information, which we will discuss later. In general, this view shows data at a particular time of day distributed across the map. When aggregated data is shown, then users can also adjust the *time window* over which the data is being aggregated by using a slider (Figure 5.1(i)). Users can select different kinds of information to display on this view through the use of a sidebar. Each of these sub-views can be stacked, allowing users to look for correlations across different aggregation methods and data. Figure 5.2 illustrates the four possible sub-views:

- **Taxi Data:** Figure 5.2(i) shows the locations of all taxis at the current time, where each individual circle corresponds to a taxi. The size of a circle grows proportionally to the number of active requests of the taxi at the current time step. The color of a circle indicates the number of requests matched to the taxi over the previous time window, where the legend is shown on the top of Figure 5.2(v). We use a diverging color scale of red-white-green because it may be of interest to spot outliers that are matched with too few requests (colored red) or too many requests (colored green) compared to the average. Clicking on a circle highlights the path of the taxi (colored orange), showing the pickup and dropoff locations of its active requests along the way. This sub-view was intended to give users a quick overview of the distribution of taxis over the city, and to help understand the spread of requests over drivers.

- **Request Data:** Figure 5.2(ii) shows the pickup locations of all the requests received at the current time step as green circles. Users can also toggle on the dropoff locations of those requests, which will be shown as red circles. Additionally, users can also filter the data to just show the matched requests, unmatched requests, or all requests. This sub-view thus give users a sense of where passengers are currently located, where they want to travel to, and whether requests from some location or requests to some location are consistently unmatched by the algorithm.

- **Inter-Zone Data:** Figure 5.2(iii) shows the relationship between zones in the city, where zones are contiguous regions that are either defined by zip codes or by user-defined neighborhoods. Each zone is visualized on the map as a node located at the centroid of the zone. A directional edge (shown by a flow animation) from zone $z_1$ to zone $z_2$ indicates the number of matched requests, whose pickup locations are in zone $z_1$ and dropoff locations are in zone $z_2$, in the previous time window. The thickness of the edge is proportional to the square root of the number of matched requests. The color of an edge from zones $z_i$ to $z_j$ can be used to indicate one of two possible metrics:

    - *Acceptance ratio*, which is the ratio between the number of matched requests $|\mathcal{M}(z_i, z_j)|$ from zone $z_i$ to zone $z_j$ and the total number of incoming requests $|\mathcal{U}(z_i, z_j)|$ from zone $z_i$ to zone $z_j$. Figure 5.2(iii) colors the edges using this metric, where the legend is shown in the middle of Figure 5.2(v).
    - *Average detour delay*, which is the average detour delays for all matched requests from zones $z_i$ to $z_j$.

  Users can hover over a zone and the visualization will only show the incoming and outgoing edges of that zone and the geographical boundary of that zone. For example, Figure 5.2(iii) shows the visualization when the user is hovering over the "Upper East

Side" zone. This sub-view was intended to allow users to analyze the amount of traffic between zones and the degree in which requests from/to a zone is not matched or have disproportionately long delays.

- **Individual Zone Data:** Figure 5.2(iv) illustrates data related to zones, which are either defined by zip codes or user-defined neighborhoods. Each zone is colored according to its *average pickup delay*, which is the average difference between the arrival time of matched requests from the zone and the time the passenger is picked up. This sub-view allows users to identify neighborhoods with disproportionately long delays.

Through the design of the map view, we also discover a new fairness metric, called *Zonal Fairness*, that is fairly intuitive. Using the same notion of zones as described above, the zonal fairness value $Z_f$ is defined as:

$$Z_f = \min_{(z_i,z_j)} AR(z_i, z_j) = \min_{(z_i,z_j)} \frac{|\mathcal{M}(z_i, z_j)|}{|\mathcal{U}(z_i, z_j)|} \tag{5.1}$$

where $(z_i, z_j)$ is an arbitrary pair of zone nodes, and $AR(z_i, z_j)$, $\mathcal{M}(z_i, z_j)$, and $\mathcal{U}(z_i, z_j)$ are the acceptance ratio, number of matched requests, and number of incoming requests from zone $z_i$ to zone $z_j$, respectively. Intuitively, the value of $Z_f$ indicates the ratio of the requests matched for the pair of zone nodes that is most underserved. The value of $Z_f$ can be extracted from the Inter-Zone Data sub-view if the user chooses to visualize the acceptance ratio metric. A hypothetical matching algorithm could also use this metric as its objective function and seek to maximize $Z_f$.

### 5.3.2 Graph View

We now discuss the graph view of FairVizARD, which visualizes temporal distribution of the data. Figure 5.3 displays the four possible graphs that are shown in graph view of FairVizARD, where each graph is under its own collapsible header. The types of graphs are as follows:

- **Incoming Request Graphs:** The graphs at the top of Figure 5.1(ii) show examples, where they plot the number of matched, unmatched, and total requests received against the time of the day that the requests were received. Users can use these graphs

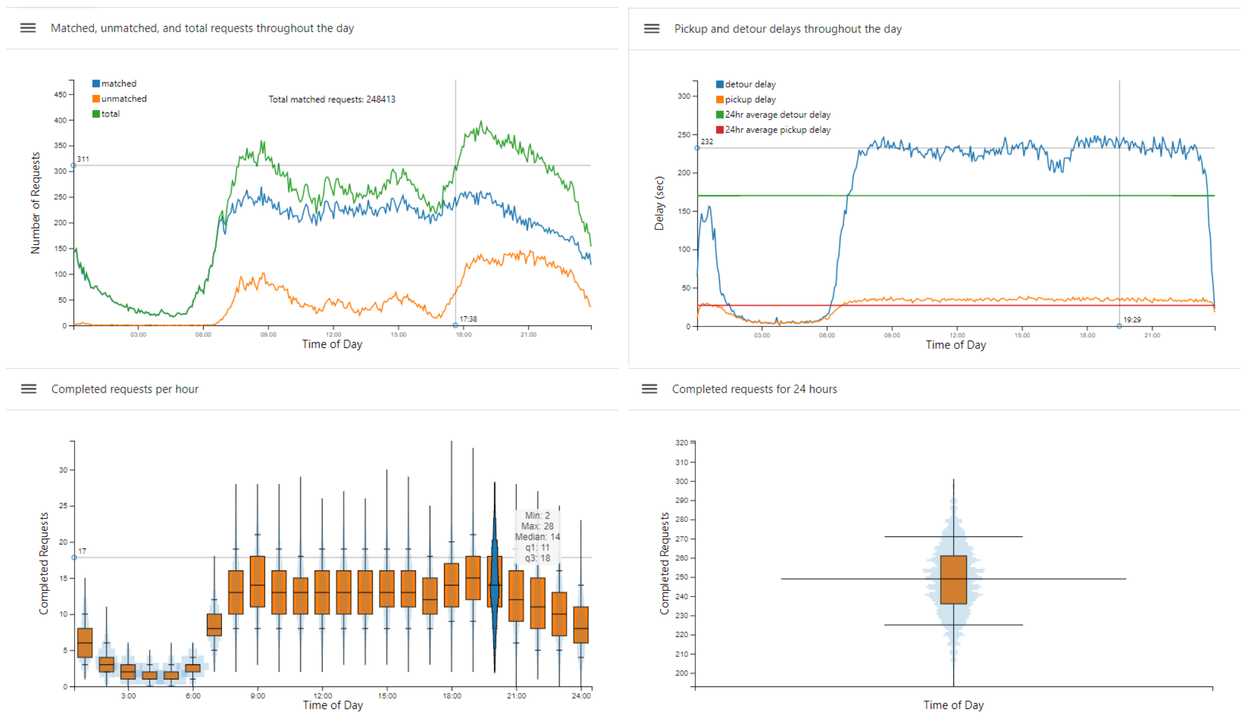


Figure 5.3: Examples of various plots in the graph view. Clockwise, from top left: (i) Incoming Request Graphs; (ii) Pickup and Detour Delay Graphs; (iii) Completed Request Graphs (per hour); (iv) Completed Request Graphs (per day)

to identify peak times in the day and to get a sense for when requests are usually unmatched.

- **Pickup and Detour Delay Graphs:** The graphs in the middle of Figure 5.1(ii) show examples, where they plot the average pickup and detour delays for requests arriving at each time step as well as their 24-hour averages. Users can use these graphs to evaluate algorithms based on metrics that are important to passengers. Using these graphs in conjunction with request graphs will also allow users to identify correlations between the number of requests received and the delays, if they exist.

- **Completed Request Graphs:** The graphs at the bottom of Figure 5.1(ii) show examples, where they plot box plots of the distribution of the number of requests completed by taxi drivers throughout the day. Each box plot provides the range of requests completed (i.e., their minimum and maximum values), the median (shown by the horizontal line in the orange rectangle), the second and third quartiles (the ranges corresponding to the bottom and top portions of the rectangle, respectively), and the

10th and 90th percentiles (shown using ticks on the vertical line). Users can use these graphs to evaluate algorithms based on metrics that are important to drivers. For example, if the box plots show a large variance during some time of the day, then it means that some drivers are disproportionately completing fewer requests compared to others.

## 5.4 Evaluation Setup

We now discuss the setup for our evaluation, where we compared FairVizARD against a benchmark visualization of numeric metrics through exploratory user studies and an expert interview. The goal of the evaluation is to examine how FairVizARD impacts data comprehension and exploration of ridesharing matching algorithms. We had the following goals in our user study:

- **Goal 1 (User Understanding of Data):** Understand if FairVizARD allows users understand the same data as effectively as a set of contextually-relevant numeric metrics used in literature to quantify fairness.
- **Goal 2 (Facilitation of Exploration):** Discern if users could make fairness-related observations through FairVizARD that they could not make through numeric metrics alone.
- **Goal 3 (Accommodation of Different Notions of Fairness):** Better understand the diversity of opinions on fairness and see if FairVizARD allows users to explore their notions of fairness.

### 5.4.1 User Study Design

We ran an exploratory user study with 18 participants, where each participant used both FairVizARD and a dashboard of numeric metric values. Participants completed a survey where they needed to rank two ridesharing matching algorithms in terms of their fairness for each of the three parties (i.e., passenger, driver, and RHC) in the system. In the survey, the

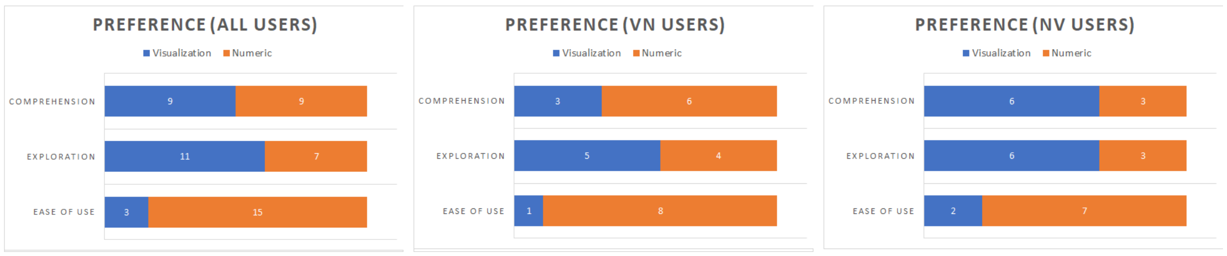


Figure 5.4: System preference across users, on the basis of ease of use, exploration, and comprehension of fairness. From left to right: (i) Preferences of all users; (ii) Preference of users in the VN ordering; (iii) Preference of users in the NV ordering

participants were also asked to describe their own notions of fairness that they were using to rank the algorithms and describe how they used the information shown in each system to measure the two algorithms based on their notions of fairness.

**Numeric Metrics as Benchmark Comparison:** The current state of the art in evaluating the fairness of matching algorithms is through numeric values of different fairness metrics. For example, researchers have seeked to maximize the minimum revenue of taxi drivers [86] and to improve the equitability in the distribution of revenue of taxi drivers [145]. We thus developed a dashboard that presents relevant statistical information of common fairness metrics for the algorithms being evaluated. Specifically, it presents information for the following metrics: (1) Statistical information on the distribution of completed requests across all taxi drivers; (2) Total number of completed requests; (3) Statistical information on the distribution of acceptance ratio across all minutes of a day; (4) Statistical information on the distribution of inter-zone acceptance ratio across all pairs of zones; (5) Statistical information on the pickup delay across all matched requests; and (6) Statistical information on the detour delay across all matched requests.

**Accounting for Ordering Bias and Learning Effects:** Since all users will use both FairVizARD and the dashboard of numeric metrics to compare the matching algorithms, the ordering in which they use the systems may influence their perceptions of the systems. For example, users may have more difficulty using the first system compared to the second system since they are already more familiar with the data by the time they use the second system. Therefore, we randomly split our pool of participants into halves, where one half uses FairVizARD before the dashboard and vice versa for the other half, to better understand and account for ordering biases. We hypothesize that the ranking of the two algorithms for a user should remain unchanged across both FairVizARD and the dashboard of numeric metrics. However, to ensure that users did not carry over their observations and conclusions from using the first system to the second system, we swapped the positions of the algorithms and used different input data for the two systems.

### 5.4.2 Participants

We recruited 18 volunteers (13 men and 5 women), whose ages are between 23 and 45 years, with an average age of 28 years. All participants had some graduate education and had taken an algorithms course at some point in time. 17 participants reported familiarity with ridesharing systems. We use the identifiers VN1 to VN9 to refer to the 9 participants that use FairVizARD first before using the dashboard of numeric metrics, and we use identifiers NV1 to NV9 to refer to other 9 participants who used the dashboard first.[7]

### 5.4.3 Ridesharing Algorithms

We evaluated FairVizARD using two ridesharing matching algorithms (RewardPlusDelay and NeurADP), whose objective is to maximize the number of requests matched and both enforce the same capacity and time-related constraints. *RewardPlusDelay* [7] is a myopic algorithm that does not consider the future value of matches while making matches in the current time step. After generating the feasible groups of requests, it matches taxis to these groups using an integer linear program. Aside from the objective of maximizing the number of requests matched, it also adds a component to the objective function to prefer matches that minimize detour delays.

*Neural Approximate Dynamic Programming* (NeurADP) [134] is a non-myopic approach that consider the future value of matches while making matches in the current time step. It does so by using a neural network to learn the expected future value (i.e., the expected future effect of current matches of taxis to feasible groups of requests) and incorporates those learned expected future values in its integer linear program. Aside from the objectives of RewardPlusDelay, it also seeks to maximize the expected future value in its objective function.

[7]We used 'N' for *numeric metrics* and 'V' for *visualization*.

Table 5.1: Measures used by users to define their notions of fairness; some users provided more than one measure

| Party | Fairness Measures Defined by Users | # Users |
|---|---|---|
| Drivers | Equal revenue | 7 |
| | High average revenue | 7 |
| | All drivers in same area/time should get the same number/type of matched requests | 3 |
| | Certainty that they will get matched throughout the day/Consistency in getting matches | 2 |
| | Travel time to pick up passenger should be small | 2 |
| | Minimize idle hours | 1 |
| | Guaranteed revenue (i.e., a base salary) | 1 |
| | Maximize distribution of taxis across different locations | 1 |
| | Getting matched based on their preference (e.g., preferred destinations) | 1 |
| Passengers | Minimize average detour delay | 14 |
| | Uniform acceptance ratio across locations | 11 |
| | High acceptance rates/Low rejection rates for all locations | 4 |
| | Uniform detour delay across locations | 3 |
| | Requests from the same pickup location and time should have the same likelihood of getting matched | 1 |
| | Identical detour delays for requests with the same pickup and dropoff locations | 1 |
| | Minimize maximum detour delay | 1 |
| Ride-Hailing Companies | Maximize profit/requests matched | 15 |
| | Balance between driver-side fairness and passenger-side fairness | 9 |
| | Smaller detour delay | 2 |
| | Maximize number of shared rides | 2 |
| | Matching requests across different locations | 1 |
| | Maximize passenger satisfaction | 1 |
| | Reputation | 1 |

## 5.5 Experimental Results

We now discuss the results of our evaluations, first, in the context of the three goals of our study.

**Goal 1 (User Understanding of Data):** To evaluate the degree to which a user understands the data presented to quantify fairness, we count how many of the following four key easily-observable differences between the two algorithms they observed: (1) NeurADP matched more requests compared to RewardPlusDelay, which corresponds to more revenue for RHCs. (2) The variance in the number of matched requests per taxi driver is smaller with NeurADP than with RewardPlusDelay. (3) RewardPlusDelay matched more requests from zones with smaller number of requests. (4) RewardPlusDelay's matched requests have smaller average delays than NeurADP's matched requests.

Overall, users using FairVizARD and the dashboard with numeric metrics observed on average 3.38 and 3.22 differences, respectively. There was no ordering bias observed since users using FairVizARD observed at least as many differences as when they use the dashboard in both orderings. Therefore, we conclude that FairVizARD does allow users to understand data that is relevant for fairness as easily as, if not better than, numeric metrics.

**Goal 2 (Facilitation of Exploration):** In general, users were able to make fairness-related observations using FairVizARD that they did not make using the dashboard with numeric metrics. We provide an anecdotal observation below of user NV1, who used the dashboard first before using FairVizARD. When using the dashboard, they noted that "Neighbourhoods are more equally served" for RewardPlusDelay. When using FairVizARD, they noticed underserved neighborhoods with low acceptance ratios for NeurADP. Overall, they concluded that RewardPlusDelay is fairer:

> "*To me fairness would mean serving all neighborhoods and all people equally. It is more important than maximizing income for drivers or companies. Algorithm 2 underserves Inwood and Harlem, where there are more people of color and black people. For this reason I think Algorithm 1 is better.*"

where Algorithms 1 and 2 correspond to RewardPlusDelay and NeurADP, respectively. The fact that the user knew the population demographics in the different areas of Manhattan, coupled with having the data spatially represented on a map, allowed them to make more nuanced fairness-related observations than if they used numeric metrics alone.

**Goal 3 (Accommodation of Different Notions of Fairness):** The diversity of opinions on fairness among our users is apparent, as shown by Table 5.1, where most users defined multiple notions of fairness for a single party. Given the myriad notions of fairness, even for a single application domain of ridesharing, it becomes even clearer that it is unlikely that a single fairness metric can capture all the underlying intricacies involved in measuring the fairness of a matching algorithm. Therefore, there is a strong need for exploratory systems that allow users to interact with data across different dimensions to come to a more well-informed conclusion.

### 5.5.1 Ordering Effects

We also observed that the order in which users used the systems influenced their preferences and time spent on each system (see Figures 5.4 and 5.5).

**Ordering Effect on Preference:** Overall, both FairVizARD and the dashboard with numeric metrics are identical in terms of their *ease of comprehension*. However, users preferred

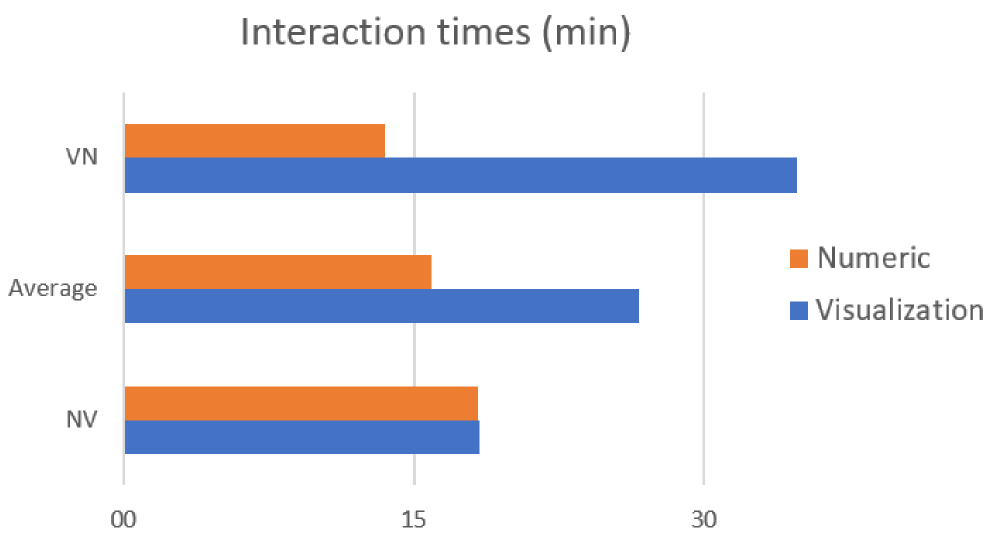


Figure 5.5: Average user interaction times

whichever system they used second, probably because they were already more familiar with the data. Unsurprisingly, users also thought that it is *easier to explore* the data using FairVizARD, though the ratio of users who thought so is higher in the group that used the dashboard with numeric metrics first compared to the group that used FairVizARD first. Again, we suspect that users were not able to as easily explore the data because they were overwhelmed with FairVizARD as their first system. In contrast, when users are already familiar with the broad trends in the data after using the dashboard, they are able to better investigate finer trends through FairVizARD. Finally, also unsurprisingly, users also strongly preferred the dashboard of numeric metrics in terms of the *ease of use*. This preference is visible in both groups, likely because the dashboard requires significantly less effort to find relevant data, almost giving users direct answers to the questions.

**Ordering Effect on Time Spent:** On both orderings, users spent more time on FairVizARD than on the dashboard with numeric metrics (see Figure 5.5). However, the difference is marginal when users used the dashboard first and is substantially larger when they use FairVizARD first. As described above, once users are familiar with the data after using the dashboard, they are able to more easily use FairVizARD. The NV group of users finished the study 9 minutes faster than users in the VN group, which is a significant difference.

### 5.5.2 Qualitative Comments

We now discuss qualitative comments provided by users, where we group them thematically and provided representative samples from each group:

**Quick vs. In-Depth Analysis:** The biggest difference between FairVizARD and the dashboard with numeric metrics is the volume of information presented to the user. Given that the dashboard provide more concise aggregate information, many users found it easier to use:

> "*Numeric fairness metrics provide less information at once but make it easier for viewer to have a idea with less time.*" – VN6

Two users commented that the curation of the metrics meant that statistics that were hard to see in FairVizARD were easier to make sense of with the dashboard; they both used FairVizARD first before using the dashboard. On the flip side, users believe that FairVizARD allows for a more in-depth analysis through better exploration of the data and the creation of a 'more informed decision':

> "*The visualization revealed more information. It took some time to understand all the functions but once I understood, I felt like I was able to investigate more and better.*" – NV1

The consensus is that numeric metrics concisely convey 'relevant' information for a quick analysis while FairVizARD allows for in-depth analyses. This suggests that the difference in preference may fall roughly along the lines of how users prefer to interact with their data – summary vs in-depth. This, in turn, could be a reflection of the 'need for cognition' of different users [19]. This position is nicely summarised by the following quote:

> "*Visualization provided far more information and nuance, but at the expense of having to critically think about what it means to benefit a stakeholder.*" – VN5

**Kinds of Information:** Another theme in the comments was about the utility of different types of information in the two systems. An important feature of numeric metrics was the presence of curated aggregate information. The absence of this in FairVizARD is a con for many users:

> "*There was not a clear way to see the total zone-wise acceptance rate of rides because the visualization required me to put in a specific time. If there was a visualization that summarized that information for the whole day, that would have been a lot more useful.*" – VN2

On the other hand, the presence of spatio-temporal information was important to many users:

> "*The visualizations highlighted the potential disparate treatment among the different neighborhoods, which I didn't think of during Task A.*" – NV4

As a result, we believe that a combination of both systems is the way forward. We posit, however, that while the FairVizARD could easily be modified to include the aggregate information, it would be difficult to include that volume of spatio-temporal information numerically.

### 5.5.3 Expert Interview

In addition to the user study, we also conducted a semi-structured interview with an expert who has worked in the ridesharing domain for 12 years in areas like fleet optimization, driver behavior analysis, agent-based simulation of taxis, labor economics (of drivers), and impact of ride-hailing innovation. We thus leveraged their experience to get better insights on the utility of FairVizARD, by asking them some analytical questions. Their responses to the various questions are discussed below.

They were able to identify all four key easily-observable differences discussed in Goal 1 above. Similar to a majority of users, they also felt that the dashboard of numeric metrics is easier to use:

"*For the purpose of reaching high-level judgement (e.g., fairness), numeric fairness metrics allows well-trained decision makers to quickly arrive at conclusions.*"

Both for ease of comprehension and ease of exploration, they ranked the visualization as the better system, where FairVizARD allows people to:

"*Actually explore and spot microscopic trends (particularly spatial-temporal patterns on the map); the true value of the visualization is [that] you don't need to know beforehand what exactly you're looking for.*"

An improvement for FairVizARD can be to include the numeric metrics, since they are useful in different scenarios:

"*For a high level decision maker, numeric metrics are what they are more used to particularly if they are transportation expert. Visualization is good, but they probably only need those numbers, the key performance statistics, then they can make decisions.*"

"*Let's say there is a major change, in demand pattern, in driver population, or the fleet size, then I think the visualization will help them to think about, for example, new metrics they should come up with to better manage or better regulate.*"

### 5.5.4 Utility of FairVizARD for Algorithm Developers

During the course of development of the system, we also observed that FairVizARD can help ridesharing algorithm developers in identifying and resolving issues in their systems.

Ridesharing systems are complex large scale spatio-temporal systems. There are thousands of taxis in the system and hundreds of customers arrive at each decision epoch. The matching decision taken by algorithms at one decision epoch has an impact on the decisions at future decision epochs. Due to such complications, the systems are hard to debug and silent errors are common. The visualization can provide aid in debugging such systems by showing the locations of taxis and customers and their movement across time.

While analyzing fairness of NeurADP algorithm using FairVizARD, we observed that after reaching certain intersections, some taxis were not moving from their location. We communicated this observation to NeurADP developers and they found that the issue occurred due to a bug in their simulator where if an empty taxi reaches an intersection from where the time taken to move to any neighboring intersection is more than decision epoch duration, it was not moving. The visualization helped NeurADP developers uncover this bug in their simulator and it improved the performance of their algorithm by nearly 5%.

### 5.5.5 Discussion

Overall, we found that FairVizARD performed well on all the three goals of the user study. The results show that it allowed users to (Goal 1) make fairness-related observations from the data; (Goal 2) not only better explore the data, but also integrate their background knowledge, to arrive at more nuanced fairness-related conclusions; and (Goal 3) have their diverse notions of fairness accommodated. An analysis of the fairness metrics that users used highlighted the diversity in opinions on fairness in ridesharing (see Table 5.1). Additionally, users focused more on average-case statistics instead of the worst case, which is important because the literature on fair algorithms often seek to improve the worst case. Finally, when users are balancing the fairness of different parties, they tend to try to be fair to the majority.

A natural question that arises then is the following: 'If FairVizARD met all its goals, why did the user preferences not reflect it?' We believe that the answer is two-fold: (1) Users have an inherent preference on the amount of effort necessary to reach conclusions from the data. The qualitative comments suggest that the difference in preference between the visualization and numeric metrics is more a reflection of this than about how each system allowed users to process fairness. (2) The current iteration of FairVizARD isn't perfect. Although the visualization provides spatio-temporal information, it does not provide aggregate information that frames the problem using statistics initially shown to the user (in the case of NV participants) and it does not allow for macro-level comparisons between different algorithms. From these observations and expert user recommendations, we conclude that an ideal system would involve a combination of both high-level statistics from the dashboard of numeric metrics and in-depth information provided by FairVizARD. Further, from the ordering effects observed, presenting the high-level statistics first is the way to go.

This leads us to our final point of discussion – the use of FairVizARD by algorithm developers. Though we did not highlight this in the paper, the visualization is a powerful tool that can also be used to find silent errors related to the matching algorithm analyzed. Our expert user went a step further and suggested that algorithm developers and even high-level decision makers can use such a system during anomalous events when metrics calibrated for the day-to-day may no longer be as relevant.

## 5.6 Conclusion

This chapter highlights how it is important to understand the impact of sophisticated algorithms in terms of fairness for the different parties in systems like ridesharing. Existing work has often distilled fairness into a single optimizable metric. However, users hold diverse views about what should count as "fair," as illustrated by Table 5.1. We proposed *FairVizARD* to let stakeholders analyze matching outcomes according to their own notions of fairness. In our user study, participants were able to explore data more deeply with FairVizARD than with a dashboard of numeric metrics; the two approaches were comparable in overall comprehension, while the dashboard was easier to use. Users generally agreed that FairVizARD provided richer, more detailed information, albeit with the cost of thinking more carefully about how that information mapped to fairness.

Taken together, these results suggest that visualization complements high-level metrics: summaries quickly orient decision-makers, while spatio-temporal exploration reveals fairness-relevant patterns (e.g., neighborhood disparities, inter-zone flows) that aggregates may miss. This chapter also introduced *Zonal Fairness* as a simple geographic diagnostic for underserved flows.

Insights from the study surface concrete stakeholder priorities as perceived by participants in our study, e.g., rider-side acceptance and delay disparities and driver-side earnings/match consistency. In the next chapter (Chapter 6), we carry these concerns forward along with our geographic fairness metric and operationalize them as objectives and constraints within our DECA-based allocation formulations, providing algorithmic mechanisms that directly target the fairness issues identified here.

# Part III

# An Incentive-Based Approach to Fairness

# Chapter 6

# Simple Incentives for Fair Matching in Ridesharing Systems

## 6.1 Introduction & Contribution

So far, we've explored two case studies, demonstrating that fairness in resource allocation is a real-world concern that hasn't received significant attention. Building on our findings from FairVizARD [83] in the previous chapter, we now take our first steps toward improving fairness in DECA systems, targeting the top measures of fairness for passengers (uniform acceptance ratio) and drivers (equal revenue) found in the user study.

As we discussed in the previous chapter, on-demand ridesharing has been gaining traction over the past few years as a solution to the growing need for urban mobility [7, 95]. With recent approaches to this problem, using dynamic programming and deep reinforcement learning for matching passenger requests to available drivers has led to significant improvements in service rate (percentage of passenger requests served) as well as total passenger delay [7, 134, 94, 87].

However, as our user study revealed, focusing on efficiency can lead to reduced fairness for both drivers and passengers in this system. While fairness in ridesharing has been a subject of some prior discussion [107, 164], the particular issue of *geographic disparities* has received less attention. Raman et al. [124] consider balancing overall efficiency and *either* geographic or income fairness, but this approach loses a great deal of overall efficiency, and requires complete retraining of the deep reinforcement learning model for any change in hyperparameters, such as the relative importance of fairness.

Instead of proposing yet another algorithm-specific approach to address fairness in ridesharing, we seek to develop a general framework that can be applied orthogonally to a large class of existing ridesharing approach. Specifically, we leverage the **DECA-like** two-stage approach that many existing ridesharing algorithms [7, 134, 95] rely on: (1) Identifying or learning good value functions for matches (DE), and (2) Optimizing over those value functions to find a good match (CA). While one could incorporate fairness in the value function learning stage, doing this would undermine our goal of generality since different ridesharing approaches employ different ways of identifying or learning their value functions. However, since all approaches employ similar ILP-based optimization approaches to find good matches, incorporating fairness in the second stage would result in the generality that we seek.

Therefore, we propose *Simple Incentives* (SI) [80], a general framework for including fairness in ILP-based matching, using any off-the-shelf value function approximations. The key ingredient in SI is a novel linear measure of relative disparity that can be associated with individuals based on their group membership (e.g, passengers' origin and destination pair or drivers' relative income level).

**Contributions:** The main contributions of this chapter are:

1. We show how we can derive a general form for this measure from the lens of minimizing variance in a metric of interest, following which we derive instantiations of passenger- and driver-side fairness functions.
2. We theoretically demonstrate that our approaches, under mild assumptions, provably improve the service rate of passengers from the most historically disadvantaged region as well as improve the income of the driver with the least historical income.
3. We empirically demonstrate the generality of SI by showing that it significantly improves fairness for a variety of ridesharing algorithms with minimal compromises on service rate. Additionally, through experiments, we show that the best SI variant achieves greater fairness than the state-of-the-art fair ridesharing approach, while at the same time yielding overall system efficiency (measured by the service rate) that is nearly inline with a state-of-the-art approach that maximizes efficiency and ignores fairness.

4. We show that our passenger- and driver-side approaches can be combined to improve two-sided fairness in such systems.

5. Finally, unlike the existing state of the art [124], our approach is completely online, allowing ridesharing operators or policy makers to tune the tradeoff between fairness and efficiency in real time during execution.

## 6.2 Background

### 6.2.1 Order Dispatching in Ridesharing

Rideshare-matching has been extensively studied, and researchers have introduced methods that improve the quality of the matches made in terms of increasing the number of requests matched [95, 96], reducing the pickup and detour delays [7, 64], and increasing drivers' earnings [86]. The complexity of ridesharing algorithms increases with the increase in vehicle capacity and fleet size. As the runtime of real-time algorithms need to be relatively small, most existing work has either considered assigning one request at a time (sequentially) to available drivers for high capacities [96, 146] or assigning all active requests together in a batch for a small capacity [167, 172]. The sequential solution is faster to compute but the solution quality is typically poor [151]. Alonso-Mora et al. [7] proposes ILP optimization approaches for assigning all active requests together for high-capacity ridesharing. Shah et al. [134] and Lowalekar et al. [95] further improve these approaches by including information about anticipated future requests while matching current batch of requests to available drivers.

### 6.2.2 Fairness in Ridesharing:

Researchers have evaluated ridesharing fairness from many viewpoints. For passengers, there has been work on addressing lack of transparency [163], using game-theoretic approaches to fairness [44], and benefit sharing by ensuring non-increasing disutility [53]. Driver-side fairness has also been explored from the economic perspective, by using a max-min approach to fairness to balance efficiency and fairness [86], and by looking at fairness over longer periods of time by equalizing driver income proportional to the number of hours spent on the platform [145]. Fairness isn't restricted to monetary benefits, however. Motivated by demographic and geographic fairness concerns, recent work formulates a matching problem with parameters to trade profit for fairness in terms of discriminatory cancellations, looking at factors like start/end locations, race, gender, or age of passengers [107] and drivers [164].

A work that is closest to ours is by Raman et al. [124], which looks at disparate treatment of passengers and income disparity amongst drivers. While they also look at geographic zones to quantify fairness for passengers, their approach requires the training of a neural network based value function to include the fairness term in the objective, making it costly to change parameters for fairness. Our approach presents an online way to address this problem, without retraining existing value functions. Further, our approach offers better tradeoffs between efficiency and fairness as compared to the existing approach, and we show this in our empirical evaluation.

## 6.3 Dynamic Matching in Ridesharing Settings

A matching algorithm for ridesharing receives as inputs a continuous stream of batches of requests from passengers $\mathcal{R}$ and the current state of all the taxis $\mathcal{V}$ operating in a street network $\mathcal{G}$. The street network $\mathcal{G} = \langle \mathcal{L}, \mathcal{E}, c(e) \rangle$ is a graph containing locations $\mathcal{L}$ connected by roads $\mathcal{E}$, with a cost function $c : \mathcal{E} \rightarrow \mathbb{R}^+$ that defines the cost $c(e)$ of each edge $e \in \mathcal{E}$ in the graph, which commonly corresponds to the time needed by a taxi to traverse $e$. A request is a tuple $r = \langle q, d, t \rangle$ that contains the pickup location $q$, dropoff location $d$, and the request arrival time $t$. The vehicle $i$ is associated with a state $v_i = \langle l_i, p_i, c_i, U_i \rangle$ that includes its location $l_i$, current path $p_i$, capacity $c_i$, and the set of requests $U_i$ it is currently serving.

State-of-the-art approaches to this dynamic matching problem take a discrete-time multi-agent perspective [7, 134]. In this framework, a decision-maker is faced in each time step with a set of outstanding requests, along with the state of vehicles at that point in time. The set of actions of this decision-maker is the set of all feasible matches of requests to vehicles. Since any matching impacts the distribution of future requests (unmatched requests can spill over into the next time step) and vehicle states, the associated expected future value is captured by a value function. This allows the decision-maker to make non-myopic

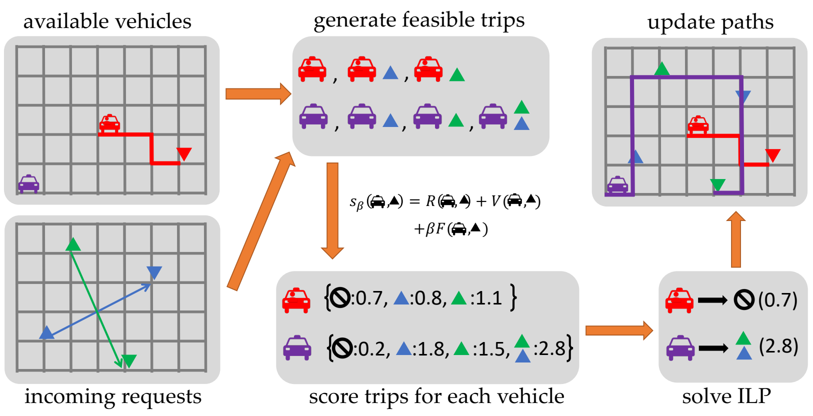


Figure 6.1: The rideshare-matching pipeline. Every time step, based on incoming requests and available vehicles, feasible trips for each vehicle are generated. These are assigned values using a score function, and an ILP is solved to optimize for total score.

decisions by considering expected rewards in addition to immediate rewards. To manage problem complexity, a value function is associated with individual vehicles and assumed to be identical for all vehicles, conditional on vehicle state; furthermore, it is approximated by a *Value Function Approximator* (VFA) – most recently, by a deep neural network in the NeurADP framework [134].

A key workhorse in such approaches is an integer linear program (ILP) for computing the optimal matching of requests to vehicles in a given time step, given a value function. We now formalize the general form of this ILP, which is central to our proposed approach below. For each vehicle $i \in \mathcal{V}$, we define an *action* $a \subseteq \mathcal{R}$ as a subset of requests that could be matched to it. Each action $a$ of vehicle $i$ is associated with a (discounted) value $V(i, a)$ obtained using VFA, and an immediate reward:

$$R(i, a) = \sum_{r \in a} R(i, r) \tag{6.1}$$

where $R(i, r)$ is an immediate reward to vehicle $i$ for servicing request $r$. Let $A_i$ be the set of feasible actions of vehicle $i$ (i.e., all feasible subsets of requests that can be matched to $i$). Then, for any feasible action $a \in A_i$ of vehicle $i$, we define the score of vehicle $i$ associated with action $a$ as:

$$s(i, a) = V(i, a) + R(i, a). \tag{6.2}$$

Let $\mathcal{A} = \{a_i\}_{i \in \mathcal{V}}$, $a_i \cap a_j = \emptyset$ for all $i \neq j$ be a matching of requests to vehicles. We say that this matching is optimal if it maximizes the total score $\sum_i s(i, a_i)$. We can compute such an optimal matching by solving the following ILP:

$$\max_{x_i(a) \in \{0,1\}} \sum_{i \in \mathcal{V}} \sum_{a \in A_i} x_i(a) s(i, a) \quad \text{s.t.} \tag{6.3}$$

$$\sum_{a \in A_i} x_i(a) = 1 \qquad \forall i \in \mathcal{V} \tag{6.4}$$

$$\sum_{i \in \mathcal{V}} \sum_{a \in A_i | r \in a} x_i(a) \leq 1 \qquad \forall r \in \mathcal{R} \tag{6.5}$$

where $x_i(a)$ is an indicator variable associated with vehicle $i$ and its action $a \in A_i$. The constraints ensure that each vehicle is assigned exactly one action (Eq. 6.4) and no request is assigned to more than one vehicle (Eq. 6.5). In each vehicle's set of available actions, there is always the null action (i.e., accepting no new requests), so that there is always a solution. The final assignment $\mathcal{A}$ is a concatenation of all vehicle assignments. In existing approaches [134], this ILP serves to maximize the total expected number of requests served (i.e., the *service rate* when normalized by the total number of requests), and it can be readily generalized to maximize total expected profit. Figure 6.1 shows an illustration of the ILP matching process.

Eq. 6.3–6.5 can be directly mapped as a DECA optimization as described in Eq. 2.4–2.6, making ridesharing a concrete instance of the DECA framework. The agents are the vehicles, with resources being passengers, actions being trips (containing sets of passenger requests), and the score function $s(i, a)$ being the utility or Q-function. Constraint 6.4 is the "one action per agent" constraint; and Constraint 6.5 is a resource constraint when each request $r$ is treated as a resource with unit inventory ($\mathcal{R}_r = 1$) and consumption $c(a)_r = \mathbf{1}\{r \in a\}$, yielding $\sum_i \sum_a x_i(a)\, c(a)_r \leq \mathcal{R}_r$. Vehicle capacity and timing limits are encoded via the feasible action sets $A_i$ (and, in DECA terms, could also be expressed as additional resource constraints). Thus, the ridesharing ILP is a domain-specific instance of the general DECA formulation.

## 6.4 SI: A General Fairness Framework

By focusing on maximizing measures of efficiency such as service rate, traditional matching algorithms for ridesharing fail to account for the emergence of fairness issues for both the passengers and drivers. Additionally, since matching algorithms in ridesharing are often trained on real-world data, they can further exacerbate historical inequalities. For example, from a passenger's perspective, it is conceivable that while maximizing efficiency, matching algorithms focus resources towards high-traffic areas (e.g., downtown locations) while reducing service in low-traffic areas (e.g., suburbs). If such algorithms are deployed in practice, it may also lead to a feedback loop that continually reduces service to regions with low demand, which causes fewer passengers to make requests from such regions and so on. From a driver's perspective, the matching algorithms may prioritize drivers in high-traffic areas to continually service requests from those areas over drivers in low-traffic areas, leading to disparity in driver income. In this work, our goal is to tackle this kind of unfairness with minimal impact to the efficiency vis-a-vis the service rate.

We consider *statistical parity* [41, 2] as the notion of fairness in this work, which defines that the expected value of a given metric $z$ over a group $g$ is the same as $\bar{z} = \mathbb{E}_{g' \in G}\left[z(g')\right]$ the expected value of that same metric over all groups $g' \in G$:

$$z(g) = \bar{z} \tag{6.6}$$

More generally, we can write it as:

$$|\bar{z} - z(g)| \leq \epsilon \qquad \forall g \in G \tag{6.7}$$

where $\epsilon$ is a slack parameter.

As an example, let's say that we want parity in service rates for passenger groups defined by their origin-destination pairs and in normalized income for drivers. If a matching algorithm achieves statistical parity for both passengers and drivers, then the probability of a passenger receiving a ride is the same regardless of their origin and destination locations and the income of drivers are the same for all drivers.

While the goal of achieving parity is noble, achieving it through the matching of a single time step is rarely possible, especially when there is a large disparity. Instead, it is often better to look at amortized parity over a longer period of time [145]. To do this, we aim for a matching that "moves closer" towards parity with the goal of achieving parity in the near future. Towards that end, our framework uses variance $\text{var}(\mathbf{Z})$, where $\mathbf{Z} = \{z(g), \forall g \in G\}$ is the set of metric values for all groups, as a proxy measure for fairness and, at each time step, it takes a gradient step in the solution space, moving in the direction that minimizes variance. Recall that $\mathbf{Z}$ here is the payoff vector as defined in Chapter 2, collecting historical utilities in an averaged manner for each group.

If we assume that the average of the metric over all groups is stable (i.e., $\frac{\partial}{\partial \mathcal{A}} \bar{z} \simeq 0$, a reasonable assumption if a long enough history is included), then we can find an assignment for a modified score function that accounts for the gradient of the variance with respect to the assignment $\mathcal{A}$:

$$s'(i,a) = s(i,a) - \lambda \frac{\partial}{\partial \mathcal{A}} \text{var}(\mathbf{Z}) \tag{6.8}$$

$$= s(i,a) - \frac{1}{|\mathbf{Z}|} \lambda \frac{\partial}{\partial \mathcal{A}} \sum_{z_j \in \mathbf{Z}} (z_j - \bar{z})^2 \tag{6.9}$$

$$= s(i,a) + \frac{2}{|\mathbf{Z}|} \lambda \sum_{z_j \in \mathbf{Z}} (\bar{z} - z_j) \frac{\partial z_j}{\partial \mathcal{A}} \tag{6.10}$$

where $\lambda$ is a hyperparameter. The general form above for the second term is our *incentive* score: A constant (weight) multiplied by the disparity of group $j$, scaled by a derivative term. We show later that the sum can usually be simplified within the context of a given action $a$ and the derivative can be approximated for specific metrics, including our two passenger- and driver-side fairness metrics of interest.

The "Simple Incentives" idea is that, for each group involved in an action, provide them with an incentive (or penalty) proportional to how disadvantaged (or advantaged) their group has been historically. Given the recent abundance of black-box algorithms, we find this simplicity helpful from a transparency perspective, making it easy to explain to any stakeholder how this score is calculated.

**SI(+):** While there is generally a consensus for applying incentives to help disadvantaged groups, it may be controversial to impose penalties for advantaged groups in some applications. With this in mind, we also present a modification to SI, where we clip the incentive term to be larger than 0, resulting in the SI(+) variant, where the "+" indicates that we include only positive incentives.

In the following sections, we discuss how we specify the incentive score (see Eq. 6.10) for two use cases, one for passengers and one for drivers.

### 6.4.1 SI for Passengers (SIP): Geographic Fairness

Our notion of passenger-side fairness is defined by the idea that the probability of receiving a ride should not depend on your origin or destination. To do this, we divide the geographical area served by the fleet into a collection of areas $C$. Recall that each request $r$ contains both the origin $o$ and destination $d$. Thus, we can map each passenger to one of $C \times C$ groups, uniquely identified by the origin-destination area pair based on $(o, d)$. Computing the service rate for each of these groups gives us the *passenger*-side metric set $\mathbf{Z}_p$, where $z_i \in \mathbf{Z}_p$ denotes the *historical* service rate for passenger group $i$. For a given request $r$, let $g(r)$ denote the group to which $r$ belongs, and let $z(r) = z_{g(r)}$ (i.e., the historical service rate of the group $g(r) \in C \times C$). These historical service rates are updated every time step after an assignment is made. The goal is thus to achieve parity in the service rates for all geographic groups.

We now derive the fairness incentive for this metric, based on Eq. 6.10:

$$
\begin{aligned}
s'(i,a) &= s(i,a) + \frac{2}{|\mathbf{Z}_p|}\lambda \sum_{z_j \in \mathbf{Z}_p} (\bar{z} - z_j)\frac{\partial z_j}{\partial \mathcal{A}} && (6.11)\\
&\simeq s(i,a) + \frac{2}{|\mathbf{Z}_p|}\lambda \sum_{z_j \in \mathbf{Z}_p} (\bar{z} - z_j)\frac{\partial z_j}{\partial a} && (6.12)\\
&= s(i,a) + \frac{2}{|\mathbf{Z}_p|}\lambda \sum_{r \in a}(\bar{z} - z(r))\frac{\partial z(r)}{\partial a} && (6.13)\\
&\simeq s(i,a) + \frac{2}{|\mathbf{Z}_p|}\lambda \sum_{r \in a}(\bar{z} - z(r))\,\lambda' && (6.14)\\
&= s(i,a) + \beta \sum_{r \in a}(\bar{z} - z(r)) && (6.15)
\end{aligned}
$$

Notice that computing $\frac{\partial z_j}{\partial \mathcal{A}}$ is difficult due to circular dependencies: The matching $\mathcal{A}$ depends on the score function, which depends on the service rates, which in turn depend on the matching. As such, computing it precisely will require a global optimization procedure. Therefore, we approximate it with $\frac{\partial z_j}{\partial a}$ (from Eqs. 6.11 to 6.12) based on the assumption that the change in the service rate of zone $z_j$ that is based on action $a$ is independent of the actions of other vehicles. Additionally, as serving a single request $r$ can only make small changes to the service rate $z(r)$ because service rates are aggregated over a reasonably long time period, we assume that the change is a (positive) constant $\lambda'$ for all requests (from Eqs. 6.13 to 6.14). We also introduce a *fairness incentive* term $F_p(i,a)$ to represent the summation in the second term of Eq. 6.15 and use it in the new approximated score function $s_\beta$:

$$F_p(i,a) = \sum_{r \in a} (\bar{z} - z(r)) \tag{6.16}$$

$$s_\beta(i,a) = s(i,a) + \beta\, F_p(i,a) \tag{6.17}$$

Here, $\beta$ is a hyperparameter that controls the relative importance of geographic fairness for passengers. In summary, for our SIP score function $s_\beta$, we include a fairness incentive term $F_p(i,a)$ that increases (or decreases) the score of actions proportional to how disadvantaged (or advantaged) their respective passenger groups have been according to our geographic fairness metric.

As described earlier, we also have the corresponding SI(+) version of this passenger-side incentive, which takes into account only requests from groups with below-average service rates:

$$F_p(i,a) = \sum_{r \in a} \max\{\bar{z} - z(r), 0\} \tag{6.18}$$

We call this variant of the framework SIP(+).

### 6.4.2 SI for Drivers (SID): Income Fairness

Our notion of driver-side fairness is defined by the idea that all drivers should earn similar incomes. This makes our driver-side fairness metric $\mathbf{Z}_d$ the set of all $z_i$, where $z_i$ now denotes *historical* driver incomes for each driver $i$. We scale the driver incomes by the largest driver income to restrict it to within $[0,1]$ and avoid scaling issues. The goal here is thus to achieve parity in the scaled income for all drivers.

We now derive the fairness incentive for this metric, based on Eq. 6.10, where we model the income of a driver to be proportional to the value of the request they serve $R(i, r)$:

$$s'(i,a) = s(i,a) + \frac{2}{|\mathbf{Z}_d|}\lambda \sum_{z_j \in \mathbf{Z}_d} (\bar{z} - z_j)\frac{\partial z_j}{\partial \mathcal{A}} \tag{6.19}$$

$$= s(i,a) + \frac{2}{|\mathbf{Z}_d|}\lambda \sum_{z_j \in \mathbf{Z}_d} (\bar{z} - z_j)\frac{\partial z_j}{\partial a} \tag{6.20}$$

$$= s(i,a) + \frac{2}{|\mathbf{Z}_d|}\lambda\,(\bar{z} - z_i)\frac{\partial z_i}{\partial a} \tag{6.21}$$

$$= s(i,a) + \delta\,(\bar{z} - z_i)\,R(i,a) \tag{6.22}$$

Eqs. 6.19 and 6.20 are equivalent because, unlike the case in passenger-side fairness, computing $\frac{\partial z_j}{\partial \mathcal{A}}$ when the metric is driver income is straightforward as the income of a driver depends solely on their actions. Next, Eqs. 6.20 to 6.22 are equivalent because $\frac{\partial z_j}{\partial a} = R(i,a)$ when $j = i$ and is 0 otherwise.

Analogous to the case for passenger-side fairness, we also introduce a similar fairness incentive term $F_d(i,a)$ and use it in the new score function $s_\delta$:

$$F_d(i,a) = (\bar{z} - z_i)\,R(i,a) \tag{6.23}$$

$$s_\delta(i,a) = s(i,a) + \delta\,F_d(i,a) \tag{6.24}$$

Here, $\delta$ is a hyperparameter that controls the relative importance of income fairness for drivers.

We also have the variant that we call SID(+), where the fairness incentive is applied only to drivers with below-average income:

$$F_d(i,a) = \sum_{r \in a} \max\{\bar{z} - z_i, 0\}\,R(i,r) \tag{6.25}$$

Table 6.1: Notation reference for proofs. Note that here we have used $r$ and $a$ interchangeably, as per the assumption for the proofs

| **Symbol** | **Meaning** |
|---|---|
| $A_i$ | Set of available actions for vehicle $i$ |
| $R(i,r)$ | Immediate reward if vehicle $i$ takes action $a = \{r\}$ |
| $V(i,r)$ | VFA prediction of future value if vehicle $i$ takes action $a = \{r\}$ |
| $Q(i,r)$ | Score for vehicle $i$ taking action $\{r\}$, as used in NeurADP ($= R(i,r) + V(i,r)$) |
| $F_p(i,r)$ | Passenger-side fairness score for assigning request $r$ to vehicle $i$ |
| $F_d(i,r)$ | Driver-side fairness score for assigning request $r$ to vehicle $i$ |
| $s_\beta(i,a)$ | Modified score function that takes passenger fairness into account ($= Q(i,r) + \beta F_p(i,r)$) |
| $\beta$ | Passenger fairness hyperparameter. Defines the relative weight of fairness |
| $\mathcal{A}(w)$ | Assignment; a matching of all vehicles to valid actions using fairness weight $w \in \{\beta, \delta\}$. |
| $s_\delta(i,a)$ | Modified score function that takes driver fairness into account ($= Q(i,r) + \delta F_d(i,r)$) |
| $\delta$ | Driver fairness hyperparameter. Defines the relative weight of fairness |
| $z_j$ | The historical metric for group $j$ |
| $z'_j(\beta)$ | The updated metric for the group $j$ after assignment $\mathcal{A}(\beta)$ |
| $z'_j(\delta)$ | The updated metric for the group $j$ after assignment $\mathcal{A}(\delta)$ |
| $g(r)$ | The group that the request $r$ belongs to |
| $g_{\min}$ | The group with the worst metric value |
| $\mathcal{R}_f$ | The set of requests from group $g_{\min}$ |

## 6.5 Theoretical Properties

While our approach directly works to minimize variance in the selected metric across groups, *max-min* fairness is another popular notion of fairness, where we want to maximize the worst-off group. Towards that end, given a sufficient large weight ($\beta$ for passenger-side fairness and $\delta$ for driver-side fairness), our approaches provide guarantees for improving worse-off groups. We provide the theorem statement, followed by brief proof sketches for building intuition, and finally include the complete proofs. Both proofs assume that vehicles can only accept a single request at any time step (including the null action). With this assumption, each action $a$ contains only one request $r$, allowing us to use them interchangeably.

Let $\mathcal{A}(w)$ be the matching solution produced by the ILP with fairness weight $w \in \{\beta, \delta\}$. Further, let $z'_j(w)$ denote the updated metric value of group $j$ after an assignment $\mathcal{A}(w)$; and $g_{\min} = \text{argmin}_j\, z_j$ denote the group with the smallest metric value. We now describe the theoretical properties for each of our two fairness metrics below.

### 6.5.1 Passenger-side Fairness Properties

We show that if vehicles can only service a single request at a time, and if there is at least one request from the group with the worst current service rate that is not served when $\beta = 0$, our approaches improve the worst service rate for a sufficiently high $\beta$ (see Theorem 1). Note that the fairness score for a single request can written as

$$f(r) = \bar{z} - z(r), \tag{6.26}$$

Formally, suppose that each action $a$ contains at most one request $r$, allowing us to use both $a$ and $r$ interchangeably. Under this assumption, the assignment $\mathcal{A}$ will contain at most one request per vehicle. For actions which contain only one request, $F_p(i, a) = f(r), a = \{r\}$.

**Improving service rate:** Recall that $z_j$ are *historical* service rates for groups $j$, where groups are passenger origin-destination neighborhood pairs. We assume that $z_j \neq z_k$ whenever $j \neq k$ (an assumption that is nearly always true in practice). Let $T_j^h$ be the total number of requests involving group $j$ prior to the current matching round, and let $T_j$ be the *current* number of requests involving $j$. All requests that are still outstanding (i.e., requests that have neither been canceled nor serviced yet) are counted as current rather than historical. Let $\mathcal{A}(\beta)$ be the matching solution produced by the ILP with fairness weight $\beta$, and let $S_j(\beta)$ be the number of requests from group $j$ that are serviced in the matching $\mathcal{A}(\beta)$. We define $z'_j(\beta)$ as the service rate for group $j$ *after* the matching computed by the ILP with fairness weight $\beta$. Since $z'_j(\beta) = \frac{z_j \cdot T_j^h + S_j(\beta)}{T_j^h + T_j}$, it is evident that $z'_j(\beta)$ is increasing in $S_j(\beta)$ for each group $j$ i.e., serving more requests from a group at any time leads to higher improvement in service rate for that group.

We now introduce some additional notation. Let $g_{\min}$ be the group with the lowest service rate, i.e., $g_{\min} = \operatorname{argmin}_j z_j$, and let $\mathcal{R}_f = \{r\}_{g(r)=g_{\min}}$ be the set of current requests corresponding to group $g_{\min}$.

Any request $r$ with $g(r) = g_{\min}$ will have the highest score (Eq. 6.26). Further, since $F_p(\cdot, r)$ only depends on the group service rate $z(r)$, every request from the same group has the same $F_p(\cdot, r)$. More formally:

**Property 1.** *All requests $r_f \in \mathcal{R}_f$ have the same fairness score $F_p(\cdot, r_f)$, which is larger than the fairness score $F_p(\cdot, r)$ of all other requests $r \notin \mathcal{R}_f$.*

In what follows, we show that as long as it is possible to improve the service rate of $z_{g_{\min}}$ relative to $\mathcal{A}(0)$ – a condition which we formalize as *passenger-min-unfairness* in the following definition – we can do so for a sufficiently high $\beta$. For ease of notation, let $Q(i,r) = V(i,r) + R(i,r)$.

**Definition 2** (passenger-min-unfair)**.** *A matching $\mathcal{A}$ is* passenger-min-unfair *if there exists a request $r_f \in \mathcal{R}_f$ not served in $\mathcal{A}$, there is a vehicle $i \in \mathcal{V}$ such that $r_f \in A_i$, but the request assigned to $i$ is $r_i \notin \mathcal{R}_f$.*

**Theorem 1.** *If $\mathcal{A}(0)$ is passenger-min-unfair, then there exists $\beta > 0$ such that $z'_{g_{\min}}(\beta) > z'_{g_{\min}}(0)$.*

**Proof Sketch:** It can be shown that:

1. The *highest-preferred* request of vehicle $i$, $r_i^* = \text{argmax}_{r \in A_i} s_\beta(i,r)$ will always be assigned to some vehicle in the optimal assignment. This follows from the nature of the ILP, which maximizes the total score.

2. Any vehicle that prefers (see (1)) $r_f \in \mathcal{R}_f$ also prefers it for higher $\beta$ values. This can be proven using the fact that $r_f$ has the largest fairness incentive, and increasing $\beta$ only increases the contribution of the fairness incentive towards $s_\beta(i,a)$.

3. There is a threshold $\bar{\beta}$ value such that for higher $\beta$ values, the $\beta F_p(\cdot)$ term in $s_\beta(i,a)$ dominates the other factors (i.e., any action $a_1$ with $F_p(i,a_1) > F_p(i,a_2)$ will be preferred by vehicle $i$ over action $a_2$). This can be shown if we assume some bounds on $R(\cdot)$ and $V(\cdot)$.

4. Any request $r_f$ that was assigned in $\mathcal{A}(0)$ will also be assigned in $\mathcal{A}(\beta)$, where $\beta > \bar{\beta} \geq 0$. This follows from a combination of the previous points.

If $\mathcal{A}(0)$ is passenger-min-unfair, then we can conclude that there is some request $r \in \mathcal{R}_f$ that was not preferred in $\mathcal{A}(0)$ but will now be preferred (3) and assigned to some vehicle (1). Thus, $\mathcal{A}(\beta)$ assigns all requests in $\mathcal{R}_f$ that were assigned in $\mathcal{A}(0)$ and at least one more, thus increasing the service rate for $g_{\min}$ as compared to $\mathcal{A}(0)$. $\square$

We now give the full proof of this theorem below. We begin by first proving several lemmas that serve as useful building blocks.

**Lemma 1.** *If $s_\beta(i,r) > s_\beta(i,r')$ for all $r' \in A_i \setminus \{r\}$ of a vehicle $i$ with available actions $A_i$, then request $r$ will always be assigned to some vehicle.*

*Proof.* If another vehicle services request $r$, the result clearly holds. So, suppose that no other vehicle does. Assume that there is an optimal matching in which vehicle $i$ also does not take request $r$, with the resulting total objective value $\sum_k s_\beta(k, a_k^*)$. However, since $s_\beta(i,r) > s_\beta(i,r_k^*)$, this solution cannot be optimal. □

Let $Q_{\max} = \max_{i,r} Q(i,r)$; $Q_{\min} = \min_{i,r} Q(i,r)$; and $\epsilon_{\min} = \min_{j \neq k} |z_j - z_k|$. Then, we define a *threshold* $\bar{\beta}$:

$$\bar{\beta} = \frac{Q_{\max} - Q_{\min}}{\epsilon_{\min}}. \tag{6.27}$$

**Lemma 2.** *If $r_f \in \mathcal{R}_f$ is optimal for vehicle $i$ for some $\beta$, then it is optimal for all $\beta' > \beta$ values.*

*Proof.* We prove this lemma by contradiction. Assume there exists a $\beta' > \beta$ such that, for some $r'$:

$$\begin{aligned} s_{\beta'}(i,r_f) &\leq s_{\beta'}(i,r') \\ Q(i,r_f) + \beta' F_p(i,r_f) &\leq Q(i,r') + \beta' F_p(i,r') \\ \beta'(F_p(i,r_f) - F_p(i,r')) &\leq Q(i,r') - Q(i,r_f) \end{aligned} \tag{6.28}$$

Since $r_f$ is optimal for $\beta$, we similarly have:

$$\begin{aligned} Q(i,r_f) + \beta F_p(i,r_f) &\geq Q(i,r') + \beta F_p(i,r') \\ Q(i,r') - Q(i,r_f) &\leq \beta(F_p(i,r_f) - F_p(i,r')) \end{aligned} \tag{6.29}$$

Combining Eqs. (6.28) and (6.29), we get:

$$\beta'(F_p(i,r_f) - F_p(i,r')) \leq \beta(F_p(i,r_f) - F_p(i,r')) \tag{6.30}$$

If $F_p(i, r_f) - F_p(i, r') = 0$, then they are tied for both $\beta$ and $\beta'$. As $F_p(i, r_f) - F_p(i, r') \geq 0$ (Property 1), Eq. (6.30) implies that $\beta' \leq \beta$, which contradicts our assumption. □

**Lemma 3.** *If $\beta > \bar{\beta}$, then $s_\beta(i, r) > s_\beta(i, r')$ if $F_p(i, r) > F_p(i, r')$, for all vehicles $i \in \mathcal{V}$ and requests $r \in \mathcal{R}$.*

*Proof.* Assume, for a proof by contradiction, that there exists a pair of requests $r$ and $r'$ for vehicle $i$ with $F_p(i, r) > F_p(i, r')$ such that $s_\beta(i, r) \leq s_\beta(i, r')$. Then:

$$\begin{aligned} Q(i, r) + \beta F_p(i, r) &\leq Q(i, r') + \beta F_p(i, r') \\ \bar{\beta} < \beta &\leq \frac{Q(i, r') - Q(i, r)}{F_p(i, r) - F_p(i, r')} \\ \frac{Q_{\max} - Q_{\min}}{\epsilon_{\min}} &< \frac{Q(i, r') - Q(i, r)}{F_p(i, r) - F_p(i, r')} \\ \frac{Q_{\max} - Q_{\min}}{Q(i, r') - Q(i, r)} &< \frac{\epsilon_{\min}}{F_p(i, r) - F_p(i, r')} \end{aligned}$$

However, this last inequality cannot hold because the left side $\geq 1$ and the right side $\leq 1$. □

**Lemma 4.** *Every request $r_f \in \mathcal{R}_f$ served in $\mathcal{A}(0)$ is also served in $\mathcal{A}(\beta)$ when $\beta > \bar{\beta} \geq 0$.*

*Proof.* Let $i$ denote the vehicle that served request $r_f$ in $\mathcal{A}(0)$; $\hat{r}$ denote an arbitrary request with $Q(i, \hat{r}) > Q(i, r_f)$, which was assigned to another vehicle $j$ in $\mathcal{A}(0)$; and $\check{r}$ denote an arbitrary request with $Q(i, \check{r}) < Q(i, r_f)$. There are the following two cases: **(Case 1)** Request $r_f$ is optimal for vehicle $i$ when $\beta = 0$. In this case, the request is still optimal for vehicle $i$ when $\beta > \bar{\beta} \geq 0$ (Lemma 2) and it will thus be served in $\mathcal{A}(\beta)$ (Lemma 1). **(Case 2)** Request $r_f$ is not optimal for vehicle $i$ when $\beta = 0$. We first show why request $r_f$ remains preferred over request $\check{r}$ when $\beta > \bar{\beta}$. $s_\beta(i, r_f) = Q(i, r_f) + \beta F_p(i, r_f) > Q(i, \check{r}) + \beta F_p(i, \check{r}) = s_\beta(i, \check{r})$. The inequality is because $Q(i, \check{r}) < Q(i, r_f)$ and $r_f$ has the largest fairness score $F_p(\cdot, r_f)$ because it is in $\mathcal{R}_f$ (Property 1).

We now show that all requests $\hat{r}$ will not be served by vehicle $i$ through the following two cases:

- **Case 1.** Request $\hat{r} \notin \mathcal{R}_f$. As $r_f \in \mathcal{R}_f$, when $\beta > \bar{\beta}$, $F_p(i, r_f) > F_p(i, \hat{r})$ (Property 1) and $s_\beta(i, r_f) > s_\beta(i, \hat{r})$ (Lemma 3). Therefore, vehicle $i$ prefers $r_f$ over $\hat{r}$.

- **Case 2.** Request $\hat{r} \in \mathcal{R}_f$. There are the following two subcases:
  - Request $\hat{r}$ is assigned to a vehicle $j$ in $\mathcal{A}(0)$ for which the request is optimal. The request remains optimal when $\beta > \bar{\beta} \geq 0$ (Lemma 2) and it will thus be served in $\mathcal{A}(\beta)$ (Lemma 1). Further, since $\hat{r}$ was assigned to vehicle $j$ instead of vehicle $i$ in $\mathcal{A}(0)$, we know that $Q(j, \hat{r}) = s_0(j, \hat{r}) \geq s_0(i, \hat{r}) = Q(i, \hat{r})$. Combining this inequality and the fact that $\hat{r}$ has the largest fairness score $F_p(\cdot, \hat{r})$ when $\beta > \bar{\beta} \geq 0$ because it is in $\mathcal{R}_f$ (Property 1), we get $s_\beta(j, \hat{r}) = Q(j, \hat{r}) + \beta F_p(j, \hat{r}) \geq Q(i, \hat{r}) + \beta F_p(i, \hat{r}) = s_\beta(i, \hat{r})$. Therefore, it is still better to assign the request to vehicle $j$ instead of vehicle $i$ when $\beta > \bar{\beta} \geq 0$.
  - Request $\hat{r}$ is assigned to a vehicle $j$ in $\mathcal{A}(0)$ for which the request is not optimal; there exists another request $r'$ with $Q(j, r') = s_0(j, r') > s_0(j, \hat{r}) = Q(j, \hat{r})$. If $r'$ is assigned to a vehicle $k$ in $\mathcal{A}(0)$ for which the request is optimal, then the arguments from the previous subcase holds, and $\hat{r}$ is assigned to vehicle $j$ because the more preferred $r'$ is assigned to a different vehicle. If $r'$ is assigned to a vehicle $k$ in $\mathcal{A}(0)$ for which the request is not optimal, then one can recursively apply the same reasoning until the more preferred request is assigned to a vehicle for which the request is optimal.

  Consequently, all requests $\hat{r}$ with higher scores than $r_f$ will be served by other vehicles, and $r_f$ has the highest score among the remaining available requests for vehicle $i$. Therefore, $r_f$ will be assigned to vehicle $i$ or some other vehicle $k$ for which $s_\beta(k, r_f) \geq s_\beta(i, r_f)$.

Thus, requests in $\mathcal{R}_f$ served in $\mathcal{A}(0)$ are served in $\mathcal{A}(\beta)$. □

We are now ready to prove Theorem 1.

*Proof of Theorem 1.* We show that $z'_{g_{\min}}(\beta) > z'_{g_{\min}}(0)$ because every request in $\mathcal{R}_f$ served in $\mathcal{A}(0)$ is also served in $\mathcal{A}(\beta)$ (Lemma 4) and there is one other request in $\mathcal{R}_f$ not served in $\mathcal{A}(0)$ but is served in $\mathcal{A}(\beta)$. Since $\mathcal{A}(0)$ is passenger-min-unfair, there exists a vehicle $k$ that could serve some request $r_f \in \mathcal{R}_f$ that was dropped, instead serving $r' \notin \mathcal{R}_f$. However, when $\beta > \bar{\beta}$, requests $\mathcal{R}_f \cap A_k$ have the largest fairness score (Property 1) and will thus have larger $s_\beta$ scores compared to other requests (Lemma 3). As $Q(k, r_f) > Q(k, r_{f'})$ for all

$r_{f'} \in \mathcal{R}_f \cap A_k \setminus \{r_f\}$ (Definition 2), $r_f$ is now optimal for vehicle $k$ and is served in $\mathcal{A}(\beta)$ (Lemma 1). $\square$

The above proof holds for both SIP and SIP(+) methods. For SIP(+), Eq. 6.26 is written as:

$$f(r) = \max(\bar{z} - z(r), 0) \tag{6.31}$$

The rest of the proof remains the same.

While we focus on service rate as a way to measure group-specific performance $z_j$, our results apply for any performance measure that increases linearly with the number of requests served for a group $j$. Specifically, these groups do not need to be defined based on geographic areas.

However, our results are only for a particular matching round and do not consider service rates over a series of many time steps. We discuss this further in the next section, where our empirical results demonstrate that the proposed approaches do, in fact, significantly improve fairness over time on multiple metrics of inequality and can do so without significantly reducing overall efficiency.

### 6.5.2 Driver-side Fairness Properties

We show that when using SID(+), worse-off drivers are guaranteed to get the request that benefits them the most, as long as no fellow worse-off driver competes for that request. As a further generalization, we also show that each disadvantaged driver gets their highest-ranked request that does not benefit another driver more, for a high enough delta.

We show that if any driver with worse-off income has a higher-preferred request they are not assigned when $\delta = 0$, such that no other worse-off driver can serve that request, our approaches improve the income of that driver by assigning their most-preferred request for a sufficiently high $\delta$ (see Theorem 2).

Note that the fairness score for a driver $j$ can be written as:

$$f(j) = (\bar{z} - z_j), \tag{6.32}$$

Formally, suppose that each action $a$ contains at most one request $r$, allowing us to use both $a$ and $r$ interchangeably. Under this assumption, the assignment $\mathcal{A}$ will contain at most one request per vehicle. For actions which contain only one request, $F_d(j,a) = f(j)R(j,r), a = \{r\}$. For actions with zero requests, $F_d(j,a) = 0$. For the following proofs, we add the assumption that each request provides the same income to all drivers (i.e., $R(j,r) = R(r)$ for all drivers $j$). This assumption is reasonable as customers are not expected to pay different amounts when matched to different drivers.

**Improving driver income:** Recall that $z_j$ is *historical* income for driver $j$, and each driver can be thought of as their own group. We assume that $z_j \neq z_k$ whenever $j \neq k$ (an assumption that is nearly always true in practice). Let $\mathcal{A}(\delta)$ be the matching solution produced by the ILP with fairness weight $\delta$. We define $z'_j(\delta)$ as the updated income estimate for driver $j$ *after* the matching computed by the ILP with fairness weight $\delta$. The action which maximizes $z'_j$ is the one with the largest $R(j,r)$, leading to the highest increase in income.

Consider the *ranking* of requests elicited by the reward function $R(j,a)$ for each driver. Let $r^*_j = \text{argmax}_r\, R(j,r)$ denote the request that maximizes the reward for driver $j$. Note that this order is not affected by $\delta$. We call $r^*_j$ the preferred request for driver $j$. Being assigned the preferred request maximizes $z'_j$ for driver $j$. Finally, we also assume that $r^*_j$ is unique (i.e., $R(r^*_j) > R(r')$ for all $r' \in A_j \setminus \{r^*_j\}$).

We say a driver $j$ is *worse-off* or disadvantaged when $z_j < \bar{z}$, or when $f(j) > 0$. Similarly, we say a driver $j$ is *better-off* or advantaged when $z_j \geq \bar{z}$, or when $f(j) \leq 0$.

In what follows, we show that for a large enough $\delta$, each worse-off driver $j$ is assigned their highest preference $r^*_j$ as long as no other worse-off driver can serve $r^*_j$. This is especially impactful if, in the initial matching $\mathcal{A}(0)$, $j$ was not matched to $r^*_j$, a condition we formalize as *driver-min-unfairness* in the following definition. Just like the previous proof, let $Q(i,r) = V(i,r) + R(i,r)$.

**Definition 3** (driver-min-unfair). *A matching $\mathcal{A}$ is* driver-min-unfair *if any worse-off driver $j$ is assigned a request $r \neq r^*_j$ in $\mathcal{A}$ and there exists no other worse-off driver $k$ that can serve $r^*_j$.*

**Theorem 2.** *If $\mathcal{A}(0)$ is driver-min-unfair for any driver $j$, then there exists $\delta > 0$ such that $z'_j(\delta) > z'_j(0)$, when using SID(+).*

**Proof Sketch:** It can be shown that:

1. For a worse-off driver $j$, their highest-preferred request $r^*_j$ will have the highest score for some value of $\delta$, and will be assigned in $\mathcal{A}(\delta)$ to some driver.
2. Any better-off driver that can serve $r^*_j$ will not serve it for some large value of $\delta$.

If $\mathcal{A}(0)$ is driver-min-unfair, then we can conclude that $r^*_j$ will be assigned to $j$ for some large value of $\delta$ because no other worse-off driver can serve it; no better-off driver can serve it (2); and it must be served by some driver (1). Thus, $z'_j(\delta) > z'_j(0)$ because $j$ is getting their highest-preferred request $r^*_j$ for some large $\delta > 0$, but not when $\delta = 0$. □

We build some helpful intuition before proving this theorem. Note that Lemma 1 still holds for this case, as it is a property of the ILP. Let $j$ be the worse-off driver with preferred request $r^*_j$. Since $\mathcal{A}(0)$ is min-unfair, we know $r^*_j$ was not assigned to $j$. First, we show that for some large value of $\delta$, $s_\delta(j, r^*_j) \geq s_\delta(j, r')$ for all $r' \in A_j \setminus \{r^*_j\}$.

**Lemma 5.** *For any worse-off driver $j$ with $f(j) > 0$, there exists a threshold weight $\delta'$ such that for any $\delta > \delta'$, $s_\delta(j, r^*_j) \geq s_\delta(j, r')$ for all $r' \in A_j \setminus \{r^*_j\}$.*

*Proof.* The proof for this is constructive. Take any $r' \in A_j \setminus \{r^*_j\}$. Since $R(r^*_j) > R(r')$, we have, for some $\delta$:

$$\begin{aligned}
s_\delta(j, r^*_j) &\geq s_\delta(j, r') \\
Q(j, r^*_j) + \delta f(j) R(r^*_j) &\geq Q(j, r') + \delta f(j) R(r') \\
\delta f(j) \left( R(r^*_j) - R(r') \right) &\geq Q(j, r') - Q(j, r^*_j) \\
\delta &\geq \frac{Q(j, r') - Q(j, r^*_j)}{f(j) \left( R(r^*_j) - R(r') \right)} \\
\delta \geq \delta' &= \frac{Q_{\max} - Q_{\min}}{f(j) \left( R(r^*_j) - R(r') \right)}
\end{aligned}$$

Since $j$ is a worse-off driver, $f(j)$ is positive, and thus, the denominator is positive. If $Q(j, r_j^*) > Q(j, r')$, the numerator is negative, and the property holds even at $\delta = 0$. Otherwise, there exists some positive threshold $\delta'$ above which $r_j^*$ is guaranteed to have the highest score for vehicle $j$. □

By Lemma 1, we also know that for such a $\delta$, $r_j^*$ is guaranteed to be assigned to some driver. Next, we prove that under the conditions of driver-min-unfairness, this driver is guaranteed to be driver $j$. To show this, consider another driver $k$ that is competing for $r_j^*$. Let $r_j$ and $r_k$ be the corresponding requests that $j$ and $k$ will be assigned respectively if they fail to get $r_j^*$. Then, to favor the scenario where $j$ gets $r_j^*$, we need the score for that assignment to be larger than the score for when $k$ gets $r_j^*$:

$$\begin{aligned} s_\delta(j, r_j^*) + s_\delta(k, r_k) &\geq s_\delta(j, r_j) + s_\delta(k, r_j^*) \\ s_\delta(j, r_j^*) - s_\delta(j, r_j) &\geq s_\delta(k, r_j^*) - s_\delta(k, r_k) \end{aligned} \tag{6.33}$$

Solving the left-hand side (LHS) first, we have the following:

$$\text{LHS} = Q(j, r_j^*) - Q(j, r_j) + \delta f(j) \left(R(r_j^*) - R(r_j)\right)$$

Similarly, the right-hand side (RHS) becomes:

$$\text{RHS} = Q(k, r_j^*) - Q(k, r_k) + \delta f(k) \left(R(r_j^*) - R(r_k)\right)$$

We use the following shorthand notations for ease of comprehension:

$$\begin{aligned} \Delta Q_j &= Q(j, r_j^*) - Q(j, r_j) \\ \Delta Q_k &= Q(k, r_j^*) - Q(k, r_k) \\ \Delta R_j &= R(r_j^*) - R(r_j) \\ \Delta R_k &= R(r_j^*) - R(r_k) \end{aligned}$$

Substituting these into Eq. 6.33, we get:

$$\Delta Q_j + \delta f(j) \Delta R_j \geq \Delta Q_k + \delta f(k) \Delta R_k$$

$$\begin{aligned} \delta f(j)\Delta R_j &\geq \delta f(k)\Delta R_k + \Delta Q_k - \Delta Q_j \\ f(j)\Delta R_j x &\geq f(k)\Delta R_k + \frac{1}{\delta}\left(\Delta Q_k - \Delta Q_j\right) \end{aligned} \tag{6.34}$$

Note that the effect of the $\Delta Q$ terms goes to zero as $\delta \to \infty$. We are now ready to prove Theorem 2.

*Proof.* Let $j$ be the worse-off driver with preferred request $r_j^*$, such that $\mathcal{A}(0)$ is driver-min-unfair for $j$. We know $r_j^*$ was not assigned to $j$ in $\mathcal{A}(0)$, and $j$ was matched to some other request $r_0$. By Lemma 5, we know that for any $\delta$ larger than a threshold value $\delta'$, $r_j^*$ has the largest score $s_\delta(j, r_j^*)$ among all actions available to $j$. By Lemma 1, we can say that $r_j^*$ is assigned to some driver in $\mathcal{A}(\delta)$.

Further, since $\mathcal{A}(0)$ is driver-min-unfair, no other worse-off driver can serve $r_j^*$. Thus, any drivers competing for $r_j^*$ are better-off drivers. For SID(+), recall that we clip the fairness score to be positive. So, $f(k) = 0$ for all better-off drivers. Plugging this into Eq. 6.34, we see $j$ is guaranteed to be a preferred match for $r_j^*$ over $k$ if the following condition is satisfied:

$$f(j)\Delta R_j \geq \frac{1}{\delta}\left(\Delta Q_k - \Delta Q_j\right)$$

$\Delta R_j$ is guaranteed to be positive since $r_j^*$ is the highest preference. $f(j)$ is also positive since $j$ is a worse-off driver. Thus, we can always find a large enough $\delta > \delta'$ such that this inequality holds, and thus guarantee that $j$ gets their highest preferred request. Therefore, since $R(r_j^*) > R(r_0)$, it must be the case that $z_j'(\delta) > z_j'(0)$ for all drivers $j$ for which $\mathcal{A}(0)$ is driver-min-unfair. □

If we interpret $\Delta R_x$ to be the *benefit* that driver $x$ gets from being assigned $r_j^*$, Eq. 6.34 tells us that any worse-off driver $j$ will get $r_j^*$ as long as they benefit from it more than any other driver, weighted by their fairness score. This also leads to an interesting behavior when $f(k)$ is negative. In this case, the matching may prefer to give $r_j^*$ to $k$ iff $R(r_j^*) < R(r_k)$, i.e., this assignment harms $k$. So, for SID, it prefers pulling down better-off drivers as much as pulling up worse-off drivers, and may forego improving disadvantaged drivers as long as it can worsen the better off groups more, pulling them towards the mean.

This observation also lets us make an additional, looser corollary statement about SID(+):

**Corollary 1.** *All worse-off drivers $j$ that do not receive their highest priority request $r_j^*$ in $\mathcal{A}(0)$ are assigned that request using SID(+) for some large enough $\delta$, as long as no other worse-off driver $k$ can benefit more from receiving it, i.e. $f(j)\Delta R_j > f(k)\Delta R_k$.*

### 6.5.3 Addendum for Theoretical Properties

We note that Theorems 1 and 2 guarantee that the *current* $z_{g_{\min}}$ sees the most improvement, but it is possible that $\text{argmin}_i(z'_i) \neq g_{\min}$. This is due to the fact that the fairness scores are calculated using previous values of $Z$, but due to batch allocation, the order might change. However, at the next time step, the new minimum would be improved, and so on, and thus we expect $\min(Z)$ to rise over time. This is backed by our experimental results.

For the theoretical analysis in the previous section, we assume that an action contains only one request at a time. We provide an example here to illustrate why this assumption is necessary for Theorem 1.

**Example 1.** *Consider a vehicle with two available actions $a_1$ and $a_2$. Let $a_1$ contain one action from the worst off group $g_{\min}$, and $a_2$ contain 2 requests from some other group $g$ such that $z_g < (z_{g_{\min}} + \bar{z})/2$. We can see that $F(\cdot, a_2) > F(\cdot, a_1)$, thus it is possible that for some choice of value function, $a_2$ will be selected over $a_1$, thus voiding the property proved in Theorem 1.*

However, in our experiments and algorithm setup, we relax this assumption to allow a vehicle to pick up multiple requests at one time. While this does violate the conditions for Theorem 1 to hold, we note that this decision will still improve fairness for groups other than $z_{g_{\min}}$. Since the other group in the above example needs to lie between $z_{g_{\min}}$ and $\bar{z}$, it is one of the under-served groups, and thus even this alternate decision still helps improve some disadvantaged group, even if it doesn't improve $g_{\min}$.

## 6.6 Experimental Results

To comprehensively evaluate our SI framework, we ran three sets of experiments. First, to demonstrate the generality of the framework, we evaluated it in combination with various

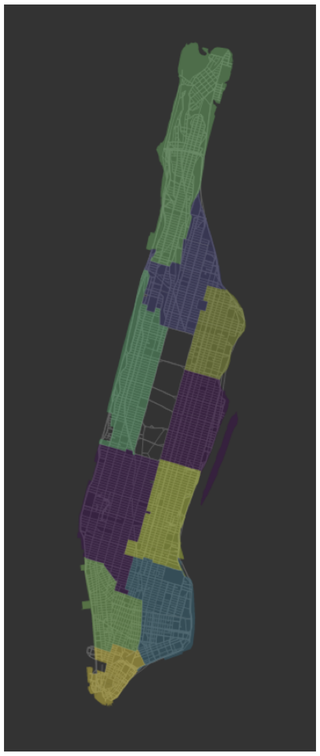


Figure 6.2: A map of Manhattan island, with each independent colored region representing one area used for our passenger grouping. There are 10 areas, meaning there are 100 passenger groups.

ridesharing algorithms, which use different value function approximations, from the literature. Second, to demonstrate the competitiveness of the framework, we evaluated it against a state-of-the-art ridesharing fairness approach. Finally, to demonstrate the flexibility of the framework, we evaluated it on two-sided fairness with both of our passenger- and driver-side fairness approaches combined.

We evaluate the performance and fairness metrics after running the matching algorithms over a 24-hour period on the island of Manhattan using demand data from the NY Yellow Taxi dataset [112]. The locations in the road network correspond to street intersections, with edges as roads connecting them. We define areas with respect to a standard partition of Manhattan into neighborhoods, where passenger groups correspond to pairs of areas in which passengers are to be picked up and dropped off, illustrated in Figure 6.2. Our efficiency objective is to maximize service rate, and thus, correspondingly, we set the value of each request to 1. For each hyperparameter $\beta$ and $\delta$, we performed a logarithmic search in the $[0.5, 20]$ range to capture a wide range of behaviors. Consistent with literature [134, 95, 124],

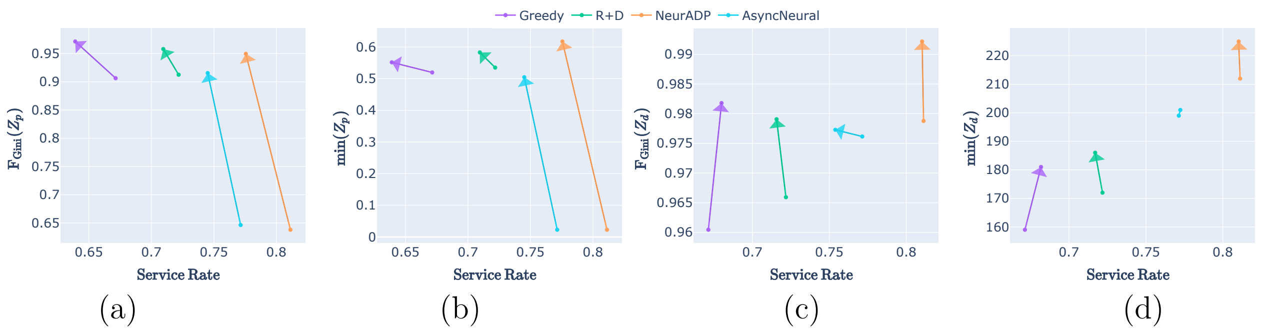


Figure 6.3: Change in efficiency and passenger-side fairness (with (a) $F_{\text{Gini}}$ and (b) min) as well as driver-side fairness (with (c) $F_{\text{Gini}}$ and (d) min) when Simple Incentives are used. The arrows show the best improvement for each algorithm metric while limiting SR to remain above 95% of the initial value. Optimal point is to the top-right (high fairness & high service rate).

we use a fleet size of 1000 vehicles, with a maximum request waiting time of 300 seconds. In all approaches, any request not assigned in the current assignment (one minute window) is dropped. All experiments were run on a Ryzen 3700x CPU, RTX2080 Super GPU, and 32GB RAM. [8]

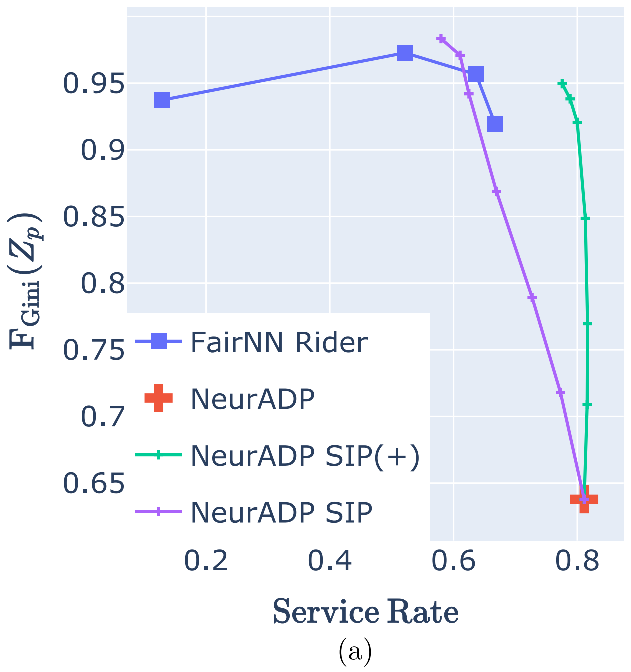
We consider two standard measures of equity: The *Gini* coefficient $\text{Gini}(\mathbf{Z})$ and the minimum metric value $\min(\mathbf{Z})$[9] for a set $\mathbf{Z}$ of service rates for passenger-side fairness or income for driver-side fairness. For ease of comprehension, we plot $F_{\text{Gini}}(\mathbf{Z}) = 1 - \text{Gini}(\mathbf{Z})$ so that our goal is to maximize each of these metrics. Our efficiency measure is the overall service rate, defined as the fraction of all passenger requests served.

### 6.6.1 Generality: Evaluation on Benchmark Algorithms

We tested the efficacy of our framework on a variety of natural and state-of-the-art ridesharing algorithms:

- **Greedy:** A baseline algorithm that considers only current rewards $R(i, a)$ as the score in the ILP optimization.

[8]The code can be found at: `https://github.com/YODA-Lab/Simple-Incentives-For-Ridesharing`
[9]For drivers, we plot the minimum income (unscaled), to prevent bias because of scaling by the max.



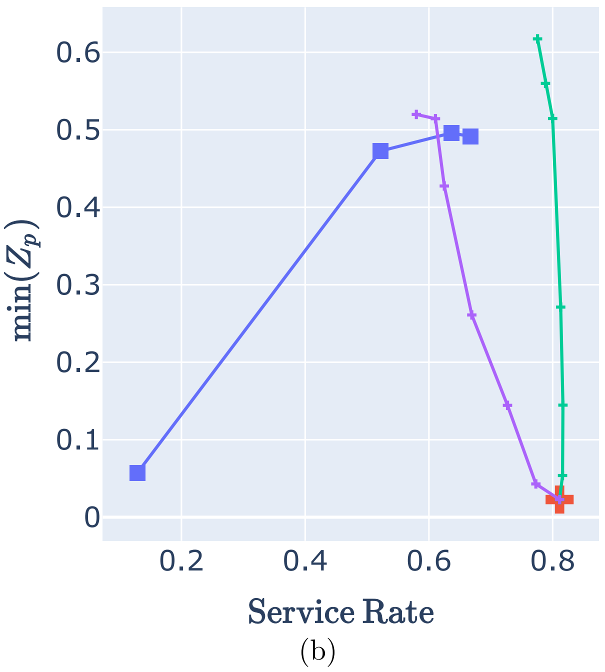
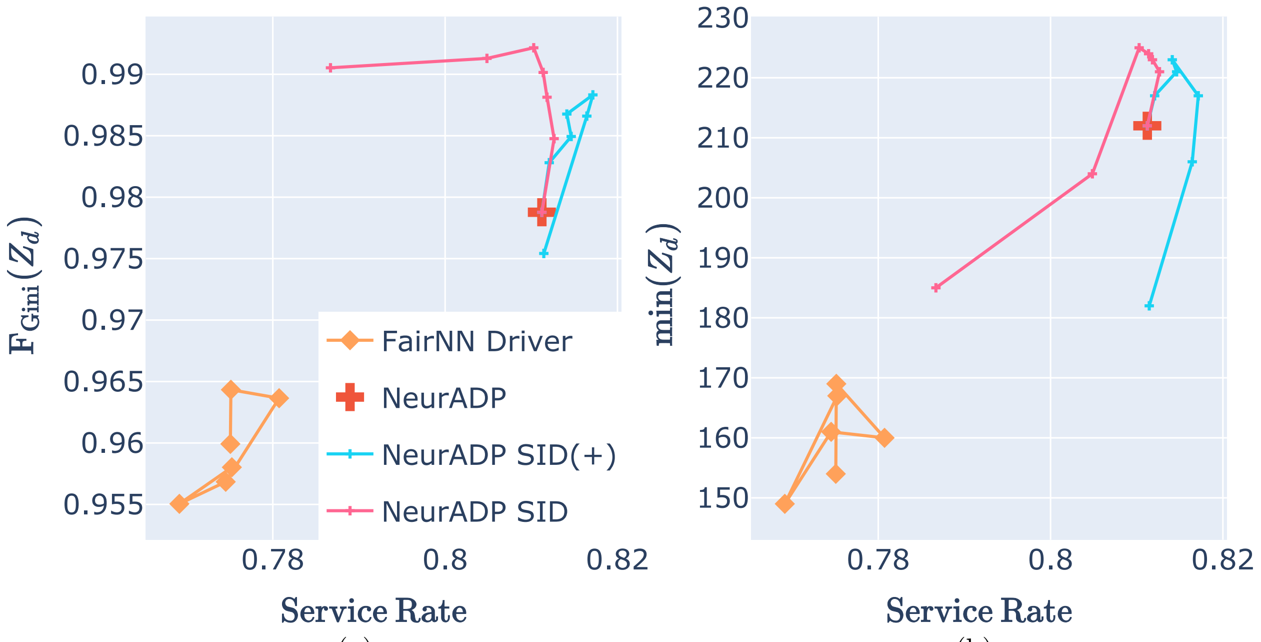


Figure 6.4: Comparison of SIP & SIP(+) against NeurADP & FairNN (line plotted in the order of hyperparameter values) for passenger-side fairness with (a) $F_{\text{Gini}}$ and (b) min. Optimal point is to the top-right (high fairness & service rate).

- **R+D:** A pioneering approach proposed by Alonso-Mora et al. [7] that estimates future values by using delays for passengers, as part of the score function in the ILP optimization.
- **NeurADP**: A state-of-the-art approach by Shah et al. [134] that uses deep reinforcement learning to approximate the value function, as part of the score function in the ILP optimization. We use a pre-trained model trained with 1000 vehicles.
- **AsyncNeural:** An asynchronous and distributed baseline algorithm that uses the approximated value function from NeurADP, but each vehicle greedily chooses its action and ties in vehicle order are broken randomly. (No ILP.)

Figure 6.3 show the impact of SI on our four ridesharing algorithms. The origin of each arrow corresponds to the performance of an algorithm without SI and the arrowhead corresponds to the best performance with SI while restricting the service rate to be above 95% of the service rate without SI. Figures 6.3(a) and 6.3(b) show the results for passenger-side fairness



(a) (b)

Figure 6.5: Comparison of SID & SID(+) against NeurADP & FairNN (line plotted in the order of hyperparameter values) for driver-side fairness with (a) $F_{\mathrm{Gini}}$ and (b) min. Optimal point is to the top-right (high fairness & service rate).

for the $F_{\mathrm{Gini}}$ and min metrics, respectively; while Figures 6.3(c) and 6.3(d) show the results for driver-side fairness for those two metrics as well.

We observed that our methods can provide significant improvement in fairness at a marginal impact to service rates. Even with varying degrees of initial fairness, consistent improvement for both passengers and drivers can be seen. The one outlier is that SI failed to improve driver fairness for AsyncNeural. The reason is that, for each driver, their fairness incentive scaling term in $F_d(\cdot)$ for all actions are identical. Therefore, the incentive does not affect their preference ordering of requests to serve, and the outcome of the algorithm remains unchanged with and without the fairness incentive term. On the other hand, with the other centralized approaches, the ILP is able to arbitrate between different vehicles and prioritize low-income drivers.

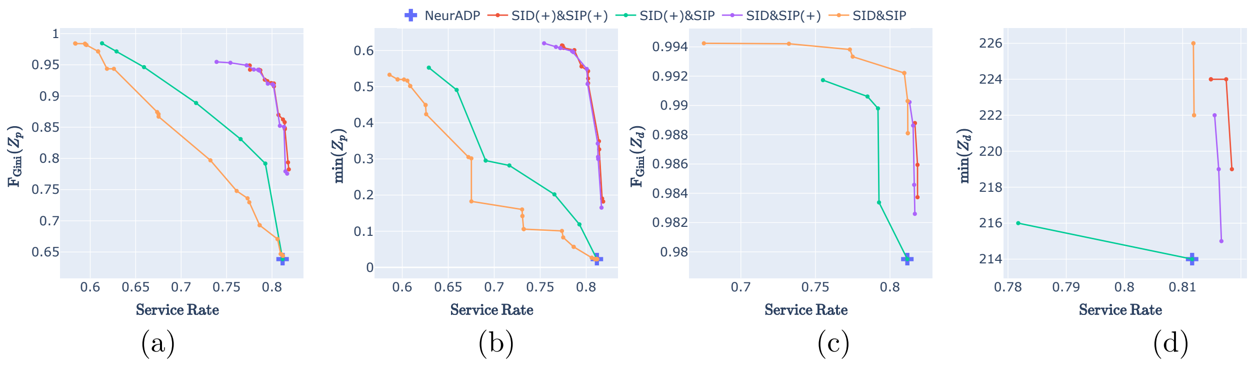


Figure 6.6: Comparison of Pareto frontiers of our combined variants on efficiency vs passenger-side fairness (with (a) $F_{\text{Gini}}$ and (b) min) and driver-side fairness (with (c) $F_{\text{Gini}}$ and (d) min). Optimal point is to the top-right (high fairness & service rate)

### 6.6.2 Competitiveness: Comparisons with FairNN

As our fairness baseline, we use *FairNN*, a recent fairness extension of NeurADP [124]. Like our SI framework, FairNN also considers geographic fairness for passengers *or* income fairness for drivers, but unlike SI, it cannot consider both together. It does this by minimizing the variance in service rates or incomes, learning a neural network-based VFA to do so. As it follows a similar ILP formulation to solve the optimization problem, it is well suited for comparison with SI. It also has a hyperparameter $\lambda$ that controls the scale of the variance term in the objective, similar to $\beta$ or $\delta$ in our formulation.

We note below some key differences between our approach and FairNN: (1) FairNN directly includes variance in the optimization, but as history size increases, the change in variance per action diminishes, reducing the impact of the fairness term if a constant weight is used. SI computes the scores based on historical inequalities rather than marginal contribution, which, combined with metric scaling, allows our approach to work well even with large histories. (2) FairNN applies the fairness uniformly across all actions, and we can improve *both* overall efficiency and fairness by adding further flexibility to the objective (as shown below with SI(+)). (3) Possibly most important of all, this approach requires full retraining of VFA for even a small change in the tradeoff weight, or a change in any other problem parameters (such as the particular measure of fairness used), making it difficult to scale in practice. Our approach is completely online, and can be used with any pre-existing value function, and with any hyperparameter value.

We trained FairNN using their provided code, suggested parameter values, and suggested hyperparameter values $\lambda$ from $10^7 - 10^{10}$ for passengers and $\frac{1}{6} - \frac{6}{6}$ for drivers, each of which requires costly retraining. For SI, we use the pre-trained NeurADP value function described in the previous section. NeurADP (without fairness) also acts as our efficiency baseline.

**Baseline Comparisons for Passenger-side Fairness:** Figures 6.4(a) and 6.4(b) compare SIP and SIP(+), with different $\beta$ values, against both baselines. FairNN outperforms SIP when the service rate is small ($\sim$0.65), but SIP(+) significantly outperforms FairNN by achieving similar fairness with much higher service rates. Consistent with results in Figure 6.3, SIP and SIP(+) improves the fairness of NeurADP as $\beta$ increases.

**Baseline Comparisons for Driver-side Fairness:** Figures 6.5(a) and 6.5(b) compare SID and SID(+), with different $\delta$ values, against NeurADP. FairNN for drivers had poor efficiency compared to NeurADP, which is consistent with observations by the authors [124]. Performing better than FairNN, SID and SID(+) improve the fairness of NeurADP as $\delta$ increases. Analogous to the passenger-side variants, SID(+) outperforms SID.

**Relative Performance of our Variants:** For both drivers and passengers, we observed the SI(+) variants provided better tradeoffs between fairness and efficiency. The reason is that these variants improve fairness without significantly sacrificing efficiency since high-efficiency groups are not penalized. The base variants achieved much better fairness in the extreme, albeit at a higher cost to service rate. The reason is that if extreme fairness is required, then there may be no choice but to penalize high-efficiency groups.

### 6.6.3 Ablation Experiments: Two-Sided Fairness

Finally, we evaluate SI with NeurADP's value function on two-sided fairness with both passenger- and driver-side fairness approaches combined. Specifically, we ran ablation experiments, running all combinations of $\beta$ and $\delta$ values in a logarithmic grid search for the score function:

$$s_{\beta,\delta}(i,a) = s(i,a) + \beta F_p(i,a) + \delta F_d(i,a) \tag{6.35}$$

Figure 6.6 shows the Pareto frontier for each passenger- and driver-side fairness combination. In general, we observe the trend that using SIP leads to much lower service rates as compared to SIP(+). SID and SID(+) with SIP(+) are almost equivalent, Pareto dominating the other approaches, with the exception of extremely high fairness regions. SIP and SID combined resulted in the fairest algorithm, albeit at a high cost to service rate.

Typically, one would expect a tradeoff between fairness and service rate. However, out of 175 combinations of $\beta$ and $\delta$ hyperparameter values as well as SIP/SIP(+) and SID/SID(+) pairs, 15 combinations ($\sim$8.5%) outperformed NeurADP on *all* five metrics (four fairness metrics and service rate). A commonality across all 15 combinations is that they all have small hyperparameter values with $\beta \leq 2$ and $\delta \leq 2$. Further, 41 out of 175 combinations ($\sim$23.5%) outperformed NeurADP on four out of five metrics, with $\beta \leq 2$ and $\delta \leq 10$. This observation reinforces the idea that *fairness does not have to be a trade off* and, in many cases, improving fairness and efficiency can go hand-in-hand.

**Limitations:** We do not prescribe "best" hyperparameters as that decision is subjective and will depend on the use-case. Instead, we provide an easy way to tune the importance of fairness using the hyperparameters. We selected geographic fairness because of the lack of other protected demographic information of passengers in public datasets. However, our formulation is general enough to allow any group-based division of passengers (e.g., by race and/or gender). Our income fairness does not take into account the number of hours worked by drivers as we assume that all drivers work all 24 hours, consistent with the literature.

## 6.7 Conclusion

This chapter introduced *Simple Incentives (SI)*, a post-processing framework for improving fairness in large-scale ride-sharing without retraining underlying models that generate value estimates. Within the DECA setting—local evaluation with a centralized ILP allocator—SI modifies only score function used in the objective via small, targeted terms derived from observed disparities in passenger service rates and driver incomes. Under mild assumptions, we showed that SI increases the worst-served passenger group's service rate and the minimum driver income. Empirically, across multiple benchmarks SI improved fairness (e.g., Gini, minimum service rate, minimum income) while remaining competitive on efficiency, and in some regimes improved both.

Since SI acts as a post-processing approach, only modifying utility estimates after agents compute them. We designed it for the ridesharing domain, but the underlying ideas are broadly applicable to other DECA instances. The key insight is that, by using historical disparities in the metric of interest, we can compute simple, targeted incentives that nudge the allocator towards fairer outcomes. This approach is appealing because it requires no re-training, is transparent, and can be applied to any underlying value function approximation.

The next chapter explores how SI can be applied to other DECA instances. Specifically, we consider homelessness prevention, a distinct DECA instance in which the decision maker has to allocate a set of support interventions to individuals at risk of homelessness.

# Chapter 7

# Fairness in Scarce Societal Resource Allocation: A Case Study in Homelessness Applications

## 7.1 Introduction & Contribution

In the previous chapter, we presented Simple Incentives (SI), a post-processing approach to improve group fairness in ridesharing allocation. This chapter studies the transferability of SI to a different, high-stakes resource allocation setting: homelessness intervention assignment.

Real-world social programs routinely face the problem of allocating scarce, indivisible resources to many people in need. Examples range from refugee housing and food aid to homelessness services and child-protection interventions. Because needs are heterogeneous and capacity is limited, purely utilitarian allocations can leave some groups consistently worse off, motivating methods that explicitly reason about fairness alongside efficiency.

This chapter studies homelessness intervention assignment as a concrete, high-stakes instance of fair resource allocation. We model the setting as *Distributed Evaluation, Centralized Allocation (DECA)*: the counterfactually constructed utility function provides local estimates of household utility (here, re-entry risk) for each feasible intervention, and a central allocator solves a constrained ILP to select assignments under capacity limits. Building on the previous chapter's incentive-based approach for ridesharing, we ask: can a similarly lightweight, *post-processing* mechanism improve group fairness here—without retraining predictive models or redesigning the allocator?

Our approach adapts *Simple Incentives (SI)* to this domain [73]. We view group fairness through the lens of statistical parity over historical outcomes and derive a small, closed-form adjustment to the allocator's scores that moves group metrics toward parity over time. We also study two targeted variants—SI(+) and SI(–)—that selectively apply positive or negative incentives. We evaluate on a real homelessness dataset with counterfactual re-entry probabilities and batched, time-ordered assignment, measuring the trade-off between fairness (Gini over group re-entry risks) and efficiency (aggregate re-entry risk).

**Contributions:** The main contributions of this chapter are:

1. **DECA formulation for homelessness allocation:** We cast homelessness intervention assignment as a DECA problem with capacity constraints and black-box VFAs, enabling principled fairness interventions at the allocation layer.
2. **Incentive-based fairness for statistical parity:** We derive a simple, differentiable proxy that adjusts per-household scores using historical group disparities, yielding SI and two targeted variants (SI(+), SI(–)) that require no retraining and integrate directly into the ILP.
3. **Empirical case study with efficiency–fairness trade-offs:** Across multiple demographic features, using these SI variants yields substantial fairness improvements with minimal efficiency loss. Notably, SI(+) achieves up to 39% reduction in Gini with only 3% increase in re-entry risk, while SI(–) can achieve even larger fairness gains when negative incentives are acceptable.

Together, these results demonstrate that fairness-aware post-processing at the allocator can transfer beyond mobility into social service allocation—preserving deployability (no model changes), exposing explicit trade-offs, and informing when targeted incentives are preferable to blanket adjustments.

## 7.2 Background

### 7.2.1 Homelessness Services

Homelessness is a long-standing social issue that affects over half a million people annually in the US [109]. Congress allocates funds to support homeless people using different kinds of interventions, and individuals or families are assigned to these interventions on a case-by-case basis. Recent work [69, 111] looks at using a data-driven approach to improve the quality of these assignments, by estimating counterfactual probabilities of *re-entry* into homelessness. The objective is to create an alternative assignment that reduces the overall probability of re-entry across all people availing homeless services.

In this section, we instantiate a resource allocation problem on a homelessness prevention dataset [69]. We formulate the homelessness-intervention allocation problem as a DECA problem, represented by the tuple $\langle \alpha, S, \{A_i\}_{i\in\alpha}, T, R, \gamma, c\rangle$. In this model, the interventions $I$ are treated as the resources, with limited capacity $C_i$ that may become available at future times. The agents $h \in \alpha$ correspond to the various households looking for interventions, where each household $h$ is associated with a state $S_h$ that captures some demographic information about the household, as well as their current status.

An action $a_h^i \in A_h$ corresponds to matching a household to an available intervention $i \in I$. $R(h, i)$ is the benefit of assigning households $h$ to intervention $i \in I$. To avoid discriminating between individuals, we keep this a unit reward for all allocations. The utility of an assignment, as described in existing work [69] is just the probability of re-entry into homelessness when household $h$ is matched to intervention $i$, $\Pr(h, i)$.

Thus, we can write the action-value for this problem as:

$$Q(a_h^i) = R(s_h, i) - \Pr(h, i) \tag{7.1}$$

$$= 1 - \Pr(h, i) \tag{7.2}$$

An allocation $\mathcal{A}$ is a concatenation of interventions assigned to each household, with $\mathcal{A}_h$ denoting the intervention allocated to household $h$. The consumption function $c(\mathcal{A}_h)$ returns a vector of size $|I|$, showing which intervention was used by an action. The optimization problem can then be written as follows, following Eq.2.4. $x_h(i)$ is an indicator variable denoting whether household $h$ was matched to intervention $i$.

$$\max_{x_h(i)\in\{0,1\}} \sum_{h\in\alpha}\sum_{i\in I} x_h(i) Q(a_h^i) \quad \text{s.t.} \tag{7.3}$$

$$\sum_{i\in I, x_h(i)\in\{0,1\}} x_h(i) = 1, \quad \forall h \in \alpha \tag{7.4}$$

$$\sum_{a\in\mathcal{A}} c(a)_i \leq C_i, \quad \forall i \in I \tag{7.5}$$

Equations 7.4 and 7.5 show that each household must be allocated one intervention, and the total capacity for each intervention must not be violated. We assume that we have exactly as many interventions as households. Since the dataset (from [69] contains the actual intervention that was assigned to each household, we can use that to construct the capacities of each intervention. The ILP we formulated here is equivalent to the one proposed by Kube et al. [69].

We also note that there is an online nature to this problem. The original dataset spans a few years, but in real life, we cannot wait for years before allocating resources to homeless people in need. Thus, we use information about the arrival dates of various households and batch them into groups over discrete time windows of length 30 days. Each time window, we accumulate the arrived households and solve the optimization problem stated above to allocate interventions. For our experiments here, we do not model the exit dates of the households, and thus, consumed interventions do not become available at a later date again.

## 7.3 Simple Incentives and Homelessness

The households in the previous setup may belong to various different groups, as described by features in their state description $s_h$. Let $G : \alpha \rightarrow \{1, 2, \ldots, g\}$ be a general function that maps the group membership of household $h$ to group $j$ uniquely. Further, let $z_j$ denote the historical average probability of re-entry for group $j$, and $\bar{z}$ denote the average of all group re-entry probabilities. Further, let $z(h) = z_{G(h)}$ map households to their group metric value. Then, it is desirable to have similar probabilities of re-entry across groups while having a low overall re-entry probability. To incorporate the incentive-based fairness function as described in Eq.6.10, we construct the following modified score function.

$$\begin{aligned} s'(a) &= Q(a) + \beta \sum_{z_j \in \mathbf{Z}} (\bar{z} - z_j) \frac{\partial z_j}{\partial \mathcal{A}} \\ &= Q(a) + \beta \sum_{z_j \in \mathbf{Z}} (\bar{z} - z_j) \frac{\partial z_j}{\partial a} \end{aligned} \tag{7.6}$$

Any action $a$ serves to modify only the connected household's group probability. So, for all $z_j \neq z(h)$, the gradient term is zero. Thus, we can simplify Eq.7.6 as:

$$s'(a) = Q(a) + \beta \left(\bar{z} - z(h)\right) \frac{\partial z(h)}{\partial a} \tag{7.7}$$

The change in $z(h)$ depends on the history being considered. To not bias this by group size, we instead just approximate $\frac{\partial z(h)}{\partial a} = Pr(h,i) - z(h)$. Any action that has a larger probability than the current group metric increases it, and vice-versa. The magnitude of change is also proportional to the difference, so this serves as a good substitute for the gradient. Note that $z_j$ here is the probability of re-entry, which needs to be minimized. In the equations derived for ridesharing in the previous chapter, we approximated the change in $z$ given an action $a$ only as a positive term, as the actions corresponded only to positive rewards (getting service for passengers, getting rides for drivers). Here, since it is possible to have both good and bad actions (in terms of re-entry probability), we need to be aware of the sign of the change.

Given this context, we can write a fairness score $F(a)$ as follows:

$$\begin{aligned} F(a) &= group\_adv \times action\_adv \\ group\_adv &= \bar{z} - z(h) \\ action\_adv &= \Pr(h,i) - z(h) \end{aligned} \tag{7.8}$$

The sign of the group advantage tells us whether the group $G(h)$ is better than the average group or not, and the action advantage tells us whether the intervention $i$ leads to a reduction (or increase) in the group metric. This leads to a more nuanced fairness score compared to the ones used in ridesharing. A summary is shown in Table 7.1.

Table 7.1: The effect of the fairness function $F(a_h^i)$ for $\beta > 0$.

|  | **Better-off group** $\bar{z} - z(h) > 0$ | **Worse-off group** $\bar{z} - z(h) < 0$ |
|---|---|---|
| **Bad action** $\Pr(h, i) - z(h) > 0$ | Make this action seem less bad | Make this action seem worse |
| **Good action** $\Pr(h, i) - z(h) < 0$ | Make this action seem less good | Make this action seem better |

Thus, the final form of the score is as follows:

$$s_\beta(a) = Q(a) + \beta\, F(a) \tag{7.9}$$

Here, $\beta$ is a hyperparameter that controls the value of fairness. Table 7.1 shows the effects of this fairness score on various types of groups and actions. In general, this has the effect of making actions for worse-off groups seem much more consequential, thus incentivizing picking of a better option over a worse one, while it makes better-off groups' choices seem less consequential, so it is okay to trade-off more of their value to serve the worse off groups.

Additionally, we also explore the effect of only performing a subset of the score modifications.

- SI(+): We only use the bonus when it makes actions seem better than they are, i.e. when $F(a)$ is positive.

$$s_\beta(a) = Q(a) + \beta\, \max(F(a), 0) \tag{7.10}$$

- SI(-): We only use the bonus when it makes actions seem worse i.e. when $F(a)$ is negative.

$$s_\beta(a) = Q(a) + \beta\, \min(F(a), 0) \tag{7.11}$$

We expect both these approaches to have different effects, as they make different combinations of actions more appealing to the decision maker (ILP). As a reminder, the ILP maximizes the action-values. Since the action value contains the negative of the probability of re-entry, it minimizes the cumulative probability of re-entry across all assignments.

## 7.4 Experimental Results

In this section, we describe the experimental setup and results of our method on the homelessness prevention dataset. We present both quantitative and qualitative results, first describing the dataset and experimental setup, followed by the results on various group divisions, followed by an empirical analysis using one of the group divisions.

### 7.4.1 Experimental Setup

We used the counterfactual probabilities generated using Bayesian Additive Regression Trees (BART) as presented by Kube et al. [69] as our estimates of re-entry probability for each household when matched to one of four interventions: Emergency Shelter (ES), Transitional Housing (TH), Rapid Re-housing (RRH) and Homelessness Prevention measures (Prev). Descriptions of these interventions can be found in the original paper [69]. Each household has a heterogenous order of utility over the available interventions. In total, there were 13,940 households. There were differing counts of each intervention available: Prev: 6202, ES: 4441, TH: 2451, RRH: 846. As stated earlier, we assumed each intervention slot could only be assigned once and would not re-enter the system, and similarly, each household was also considered for entry only once.

Each household was associated with close to 50 features. Of these, 38 features had at least two and at most twenty unique groups, excluding groups with less than 50 members. We considered each of these groups as potential groups of interest, and ran experiments to see if our method could improve fairness on arbitrary group divisions.

We ran experiments for a variety of $\beta$ values in a logarithmic grid search, starting at $\beta = 10$ and ending at $\beta = 10000$. For each feature used as a group identifier, we ran 3 experiments, one with the basic SI, one with SI (+) and one with SI (-).

As a metric for fairness, we use the Gini coefficient, a popular metric used to measure income inequality in a population. This is an instance of a distributional metric, just like variance (Chapter 2). We evaluate $Gini(\mathbf{Z})$ over all group metrics. A Gini coefficient of 0 implies perfect equality, so we want to minimize this metric. Our metric for efficiency is the overall probability of re-entry, which is just the aggregate re-entry probability $\Pr(h,i)$ for all allocated household-intervention pairs. We want this to be low as well.

In our experiments, we compare our fair solutions with the fairness incentives to a baseline ILP matching (Eq.7.3. To do this, we report the Price of Fairness (PoF) and the Benefit of Fairness (BoF). The PoF compares the loss in efficiency vis-a-vis the re-entry probability. Let $PRE_{opt}$ be the re-entry probability with the optimal ILP solution, and $PRE_{SI}$ be the re-entry probability with one of our fairness methods. Then,

$$\text{PoF} = \frac{PRE_{SI}}{PRE_{opt}} \tag{7.12}$$

Here, we expect suboptimal solutions to have PoF>1. Similarly, if $Gini_{opt}$ results from the optimal ILP and $Gini_{SI}$ results from our methods,

$$\text{BoF} = \frac{Gini_{SI}}{Gini_{opt}} \tag{7.13}$$

Here, we expect our solutions to have BoF<1.

### 7.4.2 Results: Various Group Divisions

For each feature used as group information, we report the BoF for the best $Gini(\mathbf{Z})$ obtained by our method for any $\beta$ value, while maintaining the PoF to be below 1.05. We also report the corresponding $\beta$ value that led to that assignment. The results are presented in Table 7.2.

We observe that we are generally able to significantly improve the fairness using any of our methods for most group divisions. There are some instances (like PrimaryRace) where using SI-based fairness doesn't improve the distribution at all, but barring these exceptions, we observe a few trends:

- SI(-) performs strongly, getting the best BoF with low PoF for 25/38 features.
- SI is the next strongest method, being the best in 9/38 features.
- SI(+) is the poorest performing method.

Table 7.2: Fairness improvement for group divisions based on various features. For each method, we show the best Benefit of Fairness (BoF) achieved while restricting the Price of Fairness (PoF) to be below 1.05.

| **Group** | **Original** | | **SI** | | | **SI(+)** | | | **SI(-)** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Prob** | **Gini** | **Beta** | **PoF** | **BoF** | **Beta** | **PoF** | **BoF** | **Beta** | **PoF** | **BoF** |
| PrimaryRace | 0.2443 | 0.1114 | 0 | – | – | 0 | – | – | 25 | 1.0021 | **0.9982** |
| Gender | 0.2443 | 0.0184 | 100 | 1.0129 | 0.8723 | 500 | 1.0457 | 0.3977 | 2500 | 1.0227 | **0.0421** |
| Ethnicity | 0.2443 | 0.1752 | 25 | 1.0025 | 0.7231 | 25 | 1.0023 | **0.7210** | 1000 | 1.0031 | 0.8536 |
| VeteranStatus | 0.2443 | 0.0579 | 100 | 1.0066 | 0.9405 | 100 | 1.0431 | **0.6369** | 750 | 1.0146 | 0.6500 |
| DisablingCondition | 0.2443 | 0.1223 | 25 | 1.0489 | 0.9288 | 25 | 1.0231 | 0.9400 | 75 | 1.0484 | **0.8764** |
| HousingStatusAtEntry | 0.2443 | 0.1939 | 75 | 1.0410 | 0.5712 | 100 | 1.0436 | 0.6977 | 2500 | 1.0390 | **0.5273** |
| HUDChronicHomeless | 0.2443 | 0.0941 | 25 | 1.0288 | 0.8808 | 25 | 1.0199 | 0.8884 | 100 | 1.0330 | **0.5964** |
| PhysicalDisability | 0.2443 | 0.0580 | 250 | 1.0497 | **0.4089** | 250 | 1.0115 | 0.7993 | 10000 | 1.0475 | 0.5091 |
| ReceivePhysicalDisabilityServices | 0.2443 | 0.0372 | 100 | 1.0087 | 0.9510 | 250 | 1.0381 | 0.8608 | 2500 | 1.0204 | **0.6670** |
| HasDevelopmentalDisability | 0.2443 | 0.1386 | 10 | 1.0000 | 0.9988 | 10 | 1.0004 | 0.9983 | 1000 | 1.0476 | **0.4891** |
| ReceiveDevelopmentalDisabilityServices | 0.2443 | 0.0963 | 0 | – | – | 25 | 1.0437 | 0.7385 | 7500 | 1.0449 | **0.3466** |
| HasChronicHealthCondition | 0.2443 | 0.0264 | 0 | – | – | 250 | 1.0445 | 0.8548 | 2500 | 1.0373 | **0.6426** |
| ReceiveChronicHealthServices | 0.2443 | 0.0051 | 0 | – | – | 0 | – | – | 0 | – | – |
| HasHIVAIDS | 0.2443 | 0.0372 | 1000 | 1.0132 | **0.4644** | 100 | 1.0103 | 0.6428 | 1000 | 0.9990 | 0.6677 |
| HasMentalHealthProblem | 0.2443 | 0.0863 | 50 | 1.0361 | 0.8037 | 75 | 1.0474 | 0.7868 | 500 | 1.0448 | **0.5828** |
| ReceiveMentalHealthServices | 0.2443 | 0.0793 | 50 | 1.0345 | 0.8205 | 50 | 1.0209 | 0.873 | 250 | 1.0379 | **0.6614** |
| HasSubstanceAbuseProblem | 0.2443 | 0.0940 | 50 | 1.0468 | **0.6092** | 75 | 1.0472 | 0.822 | 10000 | 1.0087 | 0.8402 |
| ReceiveSubstanceAbuseServices | 0.2443 | 0.0196 | 2500 | 1.0371 | 0.0899 | 7500 | 1.0422 | 0.4135 | 10000 | 1.0118 | **0.3769** |
| DomesticViolenceSurvivor | 0.2443 | 0.0659 | 50 | 1.0458 | **0.5524** | 25 | 1 | 0.9999 | 25 | – | – |
| proj_type_ent | 0.2443 | 0.1210 | 50 | 1.0488 | 0.6118 | 25 | 1.0285 | 0.8845 | 750 | 1.0471 | **0.5172** |
| proj_type_exit | 0.2443 | 0.1026 | 25 | 1.0427 | 0.6990 | 25 | 1.0354 | 0.8607 | 250 | 1.0446 | **0.5511** |
| numProj_before_exit | 0.2443 | 0.0449 | 0 | – | – | 0 | – | – | 250 | 1.0432 | **0.6672** |
| reentered | 0.2443 | 0.0981 | 50 | 1.0436 | 0.5927 | 75 | 1.0445 | 0.6915 | 1000 | 1.0332 | **0.4766** |
| reenteredNotStable | 0.2443 | 0.0739 | 75 | 1.0466 | **0.3994** | 100 | 1.0438 | 0.6735 | 1000 | 1.0285 | 0.4562 |
| reenteredFedDef | 0.2443 | 0.0982 | 50 | 1.0436 | 0.5930 | 75 | 1.0446 | 0.6914 | 1000 | 1.0332 | **0.4770** |
| reenterType | 0.2443 | 0.0891 | 25 | 1.0250 | 0.8621 | 50 | 1.0417 | 0.7413 | 1000 | 1.0489 | **0.4649** |
| HousingStatusNextCall | 0.2443 | 0.1419 | 25 | 1.0245 | **0.6551** | 50 | 1.0489 | 0.7788 | 750 | 1.0246 | 0.6690 |
| SpousePresent | 0.2443 | 0.0947 | 500 | 1.0415 | 0.4416 | 1000 | 1.0367 | 0.4943 | 5000 | 1.0436 | **0.4001** |
| Children | 0.2443 | 0.1871 | 100 | 1.0466 | **0.7330** | 100 | 1.0277 | 0.7512 | 2500 | 1.0339 | 0.9339 |
| Children0_2 | 0.2443 | 0.1977 | 50 | 1.0444 | 0.9373 | 100 | 1.0492 | 0.9192 | 250 | 1.0415 | **0.9162** |
| Children3_5 | 0.2443 | 0.0924 | 0 | – | – | 0 | – | – | 0 | – | – |
| Children6_10 | 0.2443 | 0.1434 | 25 | 1.0002 | 0.9638 | 25 | 1.0003 | 0.9648 | 500 | 1.0122 | **0.9558** |
| Children11_14 | 0.2443 | 0.1550 | 75 | 1.0024 | 0.9159 | 100 | 1.0269 | 0.9013 | 1000 | 1.0421 | **0.8860** |
| Children15_17 | 0.2443 | 0.2197 | 75 | 1.0032 | 0.8554 | 750 | 1.0136 | 0.8590 | 10000 | 1.003 | **0.8450** |
| num_members | 0.2443 | 0.1543 | 500 | 1.0487 | **0.7437** | 250 | 1.0440 | 0.7509 | 500 | 1.0109 | 0.8111 |
| UnrelatedChildren | 0.2443 | 0.1288 | 25 | 1.0138 | 0.7532 | 50 | 1.0341 | 0.7388 | 250 | 1.0422 | **0.6620** |
| UnrelatedAdults | 0.2443 | 0.2619 | 500 | 1.0175 | **0.8881** | 100 | 1.0053 | 0.9073 | 10000 | 1.0246 | 0.8912 |
| proj | 0.2443 | 0.1209 | 50 | 1.0487 | 0.6081 | 25 | 1.0285 | 0.8838 | 750 | 1.0473 | **0.5140** |

This suggests that only improving the probabilities of actions is not a good strategy, while typically, only reducing the probabilities (making bad options worse for worse-off groups and good options seem less good for advantaged groups) is successful in allowing for better fairness over repeated allocations over time. We also observe that there is a high variability in this value across features, indicating that different features have different responses to the trade-off weight.

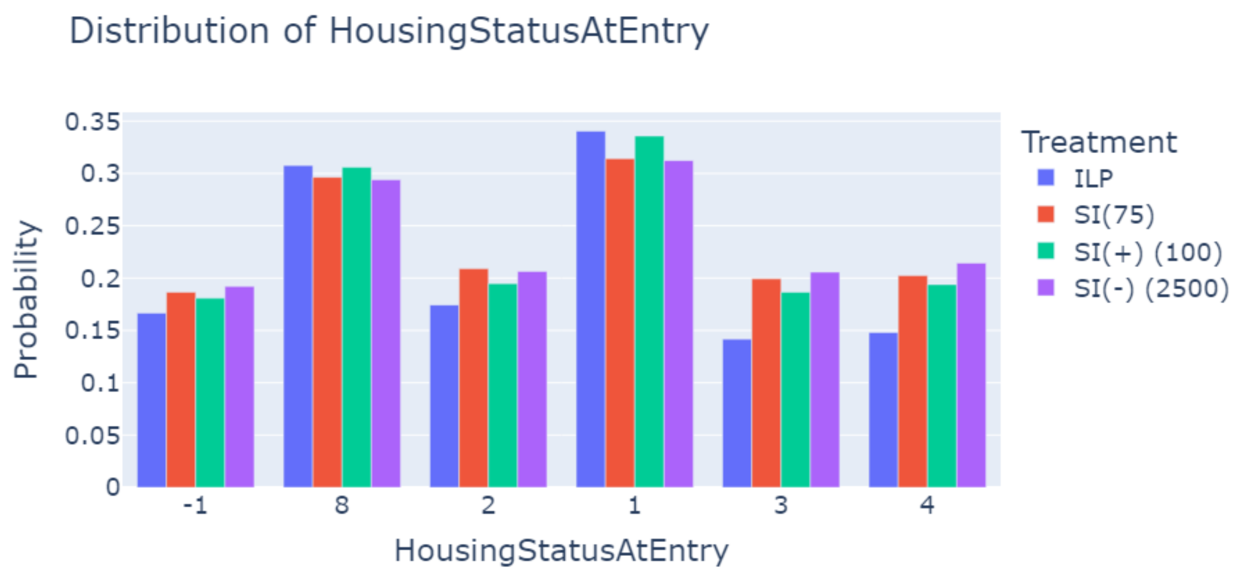


Figure 7.1: The distribution of average probabilities across different groups by solution method for the feature "HousingStatusAtEntry".

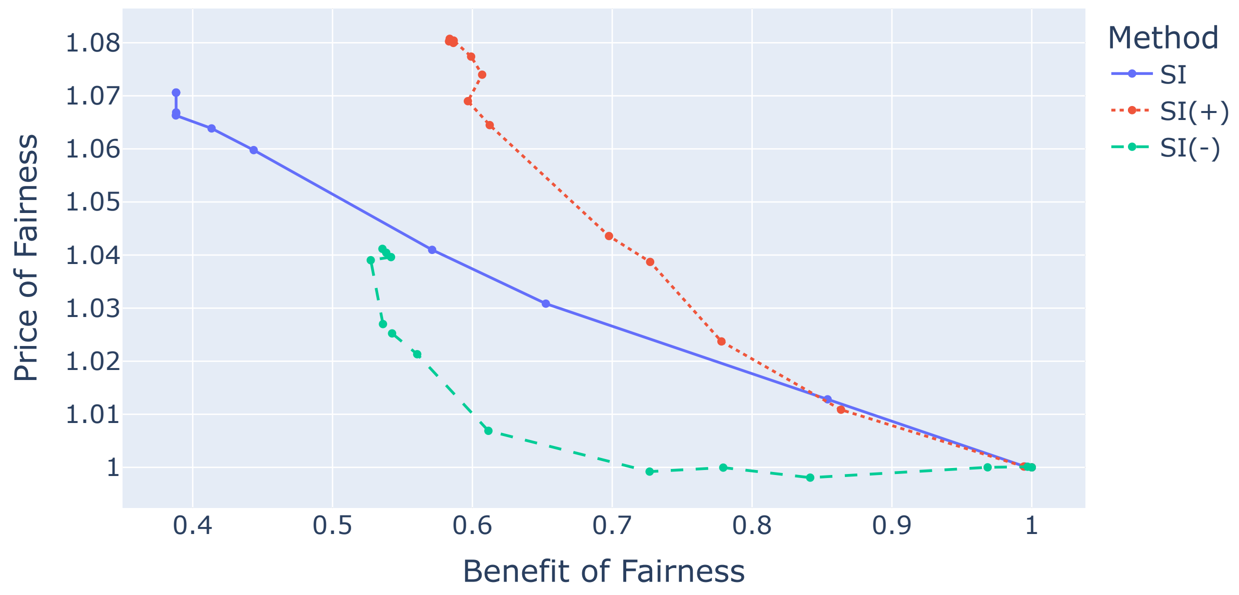


Figure 7.2: The trade-offs in PoF and BoF for the feature "HousingStatusAtEntry". The optimal point is to the bottom-left.

### 7.4.3 Empirical Observations: Selected Results

Here, we pick one representative group feature and analyze it in detail, to provide readers with a better understanding of our results. We select the feature "HousingStatusAtEntry", which captures the current housing status of a household before intervention. Using this feature to inform group division, we get the following groups: 1: Homeless (1156 households), 2: at imminent risk of losing housing (3551 households), 3: at risk of homelessness (1242 households), 4: stably housed (402 households), 8: unknown status (6326 households) and -1: missing information (1263 households).

Figure 7.1 shows the distribution of average groupwise probabilities of re-entry for the four treatments (ILP, SI, SI(+), SI(-)) of the corresponding row in Table 7.2. We can see that for all SI methods, the group probabilities are shifted towards the mean. Empirically, this means we see an improvement in fairness, while the overall probability of re-entry is bounded to be within 5% of the original. This visually shows the intended effect of our incentive score: to move group scores closer to the mean.

Looking at Figure 7.2, we can more directly compare the performance of the three SI variants when $\beta$ is varied. With increasing $\beta$, we see an improvement in fairness, but at a cost to overall utility. Qualitatively, we see that the SI(-) method pareto dominates the others. For larger BoF ratios, we see that it is possible to improve fairness without significant increase in PoF. However, if much lower Gini values are required, SI seems to be the better approach, achieving much lower BoF albeit at a higher cost. On the other hand, SI(+) still seems to improve fairness, but at a much higher PoF. The general trend seems to favor SI and SI(-) over SI(+), as we observe similar behavior for most other features.

We note that the trends observed are not universal, and there are various features that, when used as group membership, do not work well with our framework to improve fairness. We hypothesize that the relationship between features and the utility estimation process plays a crucial role in deciding which method works well for that feature. Further experiments are needed to tease out the exact cause of the variable behavior.

## 7.5 Conclusion

In this chapter, we presented a general framework for modelling resource allocation problems with repeated matching, and using demographic parity as a starting point, derived a simple incentive-based scheme for continually improving fairness in an online setting. We showed three variants of the previously developed SI framework, and showed their efficacy through evaluation on a real-world dataset. This confirms that SI is a flexible and general framework that can be adapted to a variety of settings. However, this method still has some limitations, which we try to overcome in the upcoming chapters, in addition to trying to explain the empirical behavior of the different variants of SI.

# Chapter 8

# GIFF: A General Incentives-based Framework for Fairness

## 8.1 Introduction & Contribution

SI is a method that looks at accumulated historical utility across groups or individuals (**Z**) and computes modifications to the online utility function to compensate for any historical disparities. Further, SI instantiates a very specific kind of fairness, derived from a lens of improving statistical parity and minimizing the variance in utility across agents/groups. In this chapter, we ask ourselves the following questions:

1. Can we identify potential fairness concerns that might result from our actions instead of just reacting to them?
2. Can a post-processing method like SI be created to improve diverse fairness metrics beyond statistical parity?
3. SI used clipping heuristics (SI(+) and SI(-)), which work well in practice. Can we develop a more general optimization framework that can encompass them?

Through a first principles analysis of the SI framework, we develop a more general incentives-based framework for fairness, which we call GIFF (General Incentives-based Fairness Framework). The core idea behind GIFF is the fact that the predicted Q-values are estimates of the expected change in the group metric **Z** if a particular action is taken. Thus, instead of approximating the change in fairness as done in SI, we can compute the exact change in fairness if a particular action is taken, and use this to compute incentives. This leads to a more general framework that can be used to derive incentives for a variety of fairness metrics, and can also be used to understand the empirical behavior of SI. We compare GIFF to SI in the domain of ridesharing, and show that it performs comparably to SI, and even matches SI(+), indicating that it is a stronger variant of SI that can work well without arbitrary heuristics. We also derive how GIFF can be used to optimize fairness functions beyond var($\mathbf{Z}$), like $\alpha - fairness$ and Generalized Gini Functions (GGF), with theoretical guarantees on the fairness improvement. Finally, we show that GIFF can be used to improve fairness by considering inter-agent interactions through a novel advantage correction mechanism.

**Contributions:** The main contributions of this chapter are:

1. We develop a general incentives-based framework for fairness (GIFF) that can be used to improve diverse fairness metrics in DECA problems without the need for additional learning.
2. We derive instantiations of GIFF for variance, $\alpha$-fairness and GGF.
3. We provide theoretical bounds on the actual fairness improvement in relation to the locally estimated fairness gain when using GIFF for multiple fairness functions.
4. We evaluate GIFF on the ridesharing and homelessness problems and show that it performs comparably to SI, and even matches SI(+) without the need for arbitrary heuristics.
5. We show experiments outlining the need for advantage correction in GIFF, and how it can be used to improve fairness by considering inter-agent interactions.

## 8.2 Background

### 8.2.1 Problem Setup

We revisit some concepts from Chapter 2 that will be useful in this chapter. First, we restate the optimization problem from Eq.2.4, which is the general form of the online resource allocation problem we are trying to solve. This DECA framework is described by the tuple

$\mathcal{M}$ with the following components:

$$\mathcal{M} = \langle \alpha, S, \mathcal{O}, \{A_i\}_{i\in\alpha}, T, R, \gamma, c\rangle \tag{8.1}$$

- $\alpha$ is the set of agents indexed by $i$ ($n$ agents).
- $S$ is the global state space.
- $\mathcal{O} : S \to O_1 \times O_2 \times \ldots \times O_n$ is the observation function that maps the true state to agent observations.
- $A_i$ is the action space for agent $i$, where an action $a$ includes allocation of a set of resources.
- $T : S \times A_1 \times A_2 \times \ldots \times A_n \times S \to [0, 1]$ represents the joint transition probabilities.
- $R : S \times A_1 \times A_2 \times \ldots \times A_n \to \mathbb{R}^n$ denotes the (utility) reward function, which returns a vector of rewards, one for each agent.
- $\gamma$ is the discount factor for future rewards.
- $c : A_1 \cup A_2 \cup \ldots \cup A_n \to \mathbb{R}^K$ maps each action to its resource consumption for $K$ types of resources.

Each agent can then learn an action-value function $Q(o_i, a)$, which estimates the expected long-term return from taking action $a$ given the local observation $o_i$, without knowledge of other agents' current actions. Formally, this is defined as:

$$Q(o_i, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_i^{(t)} \,\middle|\, o_i^{(0)} = o_i, a_i^{(0)} = a\right], \tag{8.2}$$

where $r_i^{(t)}$ is the reward received by agent $i$ at time-step $t$, and $\gamma \in [0, 1)$ is the discount factor. The expectation is taken over the trajectories induced by the environment dynamics and policies of all agents, conditioned only on agent $i$'s observation and action. In any state $s$, agents compute the Q-values for all of their available actions and communicate them to the central arbitrator, who can then use them to compute an allocation.

Let $\mathcal{A}$ denote the allocation of actions decided by the central allocator such that $\mathcal{A}_i$ is the action assigned to agent $i$. Let $x_i(a) \in \{0, 1\}$ be a binary decision variable that indicates whether agent $i$ is assigned action $a \in A_i$. Let $\mathcal{R} \in \mathbb{R}^K$ denote the vector of available resources, with $\mathcal{R}_k$ representing the quantity of resource type $k \in \{1, 2, \ldots, K\}$. Let $c(a) \in \mathbb{R}^K$ denote the resource consumption vector for action $a$, where $c(a)_k$ is the amount of resource $k$ consumed by action $a$. This gives us the following optimization:

$$\max_{x_i(a) \in \{0,1\}} \quad \sum_{i \in \alpha} \sum_{a \in A_i} x_i(a) \cdot Q(o_i, a) \tag{8.3}$$

$$\text{subject to} \quad \sum_{a \in A_i} x_i(a) = 1, \quad \forall i \in \alpha \tag{8.4}$$

$$\sum_{i \in \alpha} \sum_{a \in A_i} x_i(a) \cdot c(a)_k \leq \mathcal{R}_k, \quad \forall k \in \{1, \ldots, K\} \tag{8.5}$$

As discussed earlier, this can be solved as an Integer Linear Program (ILP) at each time step to get the optimal allocation, as we do in other approaches in this dissertation. Alternatively, some systems assume agents act independently without a central arbitrator or consensus-based decision making. In this case, methods like first-come-first-served or random tie breaks are used to decide who gets contested resources. This approach is much more commonly seen in multi-agent RL [176, 66, 143]. When using a policy optimization based approach, agents express preferences over actions which maximize their chances of getting good resources in the form of a policy $\pi$ rather than expressing valuations over bundles of resources. This does not work in the DECA setting, as agents do not have control over the resources they get, and the central arbitrator needs to make decisions based on the Q-values. We will focus on the ILP-based approach in this chapter.

We also revisit the payoff vector $\mathbf{Z}$ from Chapter 2, which is a temporal record of how resources have been distributed across agents and provides a foundation for incorporating fairness criteria into future allocation decisions.

**Payoff-vector (Z):** We are interested in resource allocation problems that have a temporal aspect to them, i.e., after each allocation, new resources may arrive in the system, and resources and agents may enter or exit the allocation pool. We consider two cases: 1) A fixed number of agents, and 2) An arbitrary number of agents that belong to a fixed number of groups. In both cases, we consider a total of $n$ groups or agents. Given this, we can construct a vector of payoffs $\mathbf{Z} = [z_1, z_2, \ldots, z_n]$ that captures the accumulated value of all resources allocated to each agent/group over time.

Unless specified otherwise, we use $z_i$ to denote the cumulative payoff for agent or group $i$ at the current time-step. When referring to a specific time, we will write $z_i^{(t)}$ to indicate the value at time-step $t$. The cumulative reward is given by:

$$z_i = \sum_{\tau=0}^{t} r_i^{(\tau)}, \tag{8.6}$$

where $r_i^{(\tau)}$ is the reward received by agent $i$ at time-step $\tau$. In some settings, an average payoff is used instead:

$$\bar{z}_i = \frac{1}{t+1} \sum_{\tau=0}^{t} r_i^{(\tau)}. \tag{8.7}$$

### 8.2.2 Fairness Concepts

To formalize fairness, we define a fairness function $F : \mathbb{R}^n \to \mathbb{R}$ that maps any payoff vector $\mathbf{Z} = [z_1, \dots, z_n]$ to a scalar value, where higher values indicate fairer distributions. We revisit common fairness functions, that can be SWF approaches (like $\alpha$-fairness and GGF) or distributional approaches (like variance).

- **$\alpha$-Fairness:** For $\alpha \geq 0$, the per-agent utility is

$$U_\alpha(z) = \begin{cases} \frac{z^{1-\alpha}}{1-\alpha} & \text{if } \alpha \neq 1, \\ \log(z) & \text{if } \alpha = 1, \end{cases} \tag{8.8}$$

  and the overall fairness is $F_\alpha(\mathbf{Z}) = \sum_{i=1}^{n} U_\alpha(z_i)$.

- **Generalized Gini Function (GGF):** Sorting $\mathbf{Z}$ as $z_{(1)} \leq \cdots \leq z_{(n)}$, define

$$F_{GGF}(\mathbf{Z}) = \sum_{i=1}^{n} w_i \, z_{(i)}, \tag{8.9}$$

  where $w_1 \geq w_2 \geq \cdots \geq w_n$ and $\sum_{i=1}^{n} w_i = 1$.

- **Variance:** Measures disparity as $F_{var}(\mathbf{Z}) = -\mathrm{Var}(\mathbf{Z})$.

Having outlined these approaches, we aim to balance efficiency and fairness via a joint objective. Specifically, for a given time horizon $T$ with payoff vector $\mathbf{Z}_T$, we seek an allocation policy that maximizes:

$$\max \quad (1-\beta)U_T + \beta F(\mathbf{Z}_T), \tag{8.10}$$

$$U_T = \sum_{i=1}^{n} z_i, \tag{8.11}$$

Here, $\beta \in [0,1]$ is a trade-off parameter that controls the relative importance of efficiency (total utility) versus fairness.

## 8.3 A General Incentives-based Fairness Framework (GIFF)

Here we describe the formulation for GIFF, inspired by SI, but more general and principled. There are two key departures from SI in GIFF: 1. we use the fact that the Q-function, which captures the long-term utilitarian effects of actions, can also be used to guide the decision maker towards a fair allocation, and 2. we can explicitly calculate the local post-allocation payoff vector to get the change in fairness instead of approximating it. Previous work has considered directly learning to optimize for fairness [176, 66] or using knowledge of the true reward function to myopically improve variance using domain-specific post-processing [80]. Instead, we provide a General Incentives-based Framework for Fairness (GIFF) that improves fairness without the need for additional learning for a variety of domains and fairness functions.

To achieve this, we develop an approach that takes advantage of the instantaneous pre-decision payoffs $\mathbf{Z}_t$ at the current time $t$ to guide the present decision towards a fairer outcome by improving the perceived value of fairer allocations. We do this by computing the estimated improvement to fairness, the *fairness gain* of actions, and augmenting the pre-trained Q-values to reflect our objective (Eq. 8.10).

First, we assume an idealized scenario. Let $\mathcal{A} = [a_i]_{i\in\alpha}$ denote an allocation that contains one action for each agent. We overload the notation for the reward function to let $R(\mathcal{A}_i)$

be a shorthand for the true reward received by agent $i$ under allocation $\mathcal{A}$. The updated payoff vector $\mathbf{Z}_{t+1}|\mathcal{A}$ can be computed using $\mathbf{Z}_t$ and $\mathcal{A}$, by updating the payoffs using the true rewards. Then, the fairness gain for any allocation can be defined as:

$$\Delta F(\mathcal{A}) = F(\mathbf{Z}_{t+1}) - F(\mathbf{Z}_t) \tag{8.12}$$

$$z_i^{t+1} = z_i^t + R(\mathcal{A}_i) \qquad \forall i \tag{8.13}$$

$$\mathcal{A}_f^* = \operatorname*{argmax}_{\mathcal{A}} \Delta F(\mathcal{A}) \tag{8.14}$$

Here, $\mathcal{A}_f^*$ is the allocation that improves fairness the most in the current step. We can also conceive of a similar search which maximizes over an entire sequence of allocations to maximize long-term fairness. However, even for the one-step allocation, the search space is combinatorial in the number of agents and their respective action spaces, as we have to consider all possible joint actions. This makes the global optimization intractable, necessitating alternate methods for computing a fair allocation.

### 8.3.1 Using Q-values to Estimate Fairness

We observe that it is much easier to reason about fair actions if we can decompose the fairness gain across agents. To achieve this, we reason only over the locally conditioned updated payoff vector $\mathbf{Z}_t^{a^i}$, updating all accumulated utilities based only on a single agent's action $a^i$, keeping everything else unaffected.

$$z_j^{t+1} = z_j^t + \mathbb{I}\{j = i\}\, R(a^i) \qquad \forall j \tag{8.15}$$

In many real problems, having access to the true reward function $R(a)$ is unlikely. Further, in dynamic environments, agents may take a critical action earlier, which leads to a payoff after multiple steps; however, a critical decision towards getting to it may happen much earlier. If the agent is a Q-learner, we can leverage the fact that the Q-values encode the long-term value of taking certain actions, and the difference in Q-values will be small in states where all routes lead to similar payoffs. Thus, if we use the Q-value as a proxy for the reward function, we can account for long-term returns without knowing the reward function or the environment dynamics.

$$\Delta F(a^i) = F(\mathbf{Z}_{t+1}^{a^i}) - F(\mathbf{Z}_t) \tag{8.16}$$

$$z_j^{t+1} = z_j^t + \mathbb{I}\{j = i\}\, Q(o_i, a^i) \qquad \forall j \tag{8.17}$$

This quantity $\Delta F(a^i)$, termed the *fairness gain* measures the marginal (local) impact of agent $i$'s action on the overall fairness of the allocation given the current distribution of resources $\mathbf{Z}_t$. Note that computing this for all agent actions is linear in the size of the action space for each agent. This has two benefits. First, we do not need to have access to the true reward function (which is not available in many cases) and, second, we capture more than just the immediate return, allowing us to capture long-term effects of certain allocations. However, this has the drawback that it does not capture inter-agent interactions very well. Thus, we introduce an additional mechanism that can provide this information.

### 8.3.2 Advantage Correction: Incorporating Counterfactual Fairness Gains

Fairness concerns in dynamic resource allocation may also require that agents who are already well-off (i.e., have a high accumulated utility) are discouraged from taking resources that can help worse-off agents improve their return, thus allowing disadvantaged agents to catch up. This is also a desirable property of social welfare functions, termed **equity**, used to capture the notion of accumulated wealth: Moving resources from better-off agents to worse-off agents should improve the fairness function's value. This is also known as the Pigou-Dalton principle [176] and has been discussed in previous works in fair multi-agent RL.

However, in practice, the local fairness gain $\Delta F(a^i)$—which reflects only the immediate change in an agent's own utility—does not capture the broader altruistic impact of reallocating resources. In many fairness metrics, this counterfactual update considers solely the acting agent's payoff, thereby ignoring the significant improvement in fairness that would occur if the resource were allocated to a disadvantaged agent. Consequently, even when an agent's decision to forgo an action yields $\Delta F = 0$, it may still lead to substantial overall fairness gains if the resource were reallocated. This limitation motivates the inclusion of a method to more accurately account for these counterfactual benefits.

To this end, we introduce an *advantage correction* term that incentivizes better-off agents to give up their top preferences to benefit other disadvantaged agents. Recall that in our setting, when agent $i$ takes action $a$, its local fairness gain $\Delta F(a)$ is computed by updating the agent's payoff $z_i$ with $Q(o_i, a)$ (Eqs. 8.16 and 8.17). To measure the counterfactual benefit of this action, we consider allocating this resource to any other agent that can take this action $j \neq i, a \in A_j$, and compute the counterfactual benefit $\Delta F^{(j)}$, using $Q(o_j, a)$ to update agent $j$'s payoff:

$$\Delta F^{(j)} = F\big(\mathbf{Z}_{t+1}^{(j)}\big) - F(\mathbf{Z}_t), \tag{8.18}$$

$$z_j^{t+1} = z_j^t + Q(o_j, a) \tag{8.19}$$

Let us define the set of these candidate counterfactual agents as $\alpha_c(a) = \{j \in \alpha : j \neq i, a \in A_j\}$. We can then compute the average counterfactual fairness improvement:

$$\Delta F_{\text{avg}}(a) = \frac{1}{|\alpha_c(a)|} \sum_{j \in \alpha_c(a)} \Delta F^{(j)} \tag{8.20}$$

To capture the benefit of allocating this resource to agent $i$, we can calculate the advantage function:

$$F_{\text{adv}}(a) = \Delta F(a) - \Delta F_{\text{avg}}(a) \tag{8.21}$$

A negative $F_{\text{adv}}$ suggests that another agent would benefit more from the resource allocation, whereas a positive $F_{\text{adv}}$ indicates that the current agent can better improve fairness. If the fairness metric follows the principle of equity, then we expect agents with higher fairness gain to be the disadvantaged agents. Instead of $\Delta F_{\text{avg}}$, alternate baselines like the maximum fairness improvement may also be considered.

To integrate this counterfactual fairness measure with the action's inherent quality, we weigh the fairness advantage by the relative Q-value gap:

$$\Delta Q(a) = Q(o_i, a) - \min_{a' \in A_i} Q(o_i, a'), \tag{8.22}$$

which reflects how much better action $a$ is compared to the worst option for agent $i$. This is helpful in preventing disproportional changes because of Q-value overestimation.

Finally, we define the counterfactual advantage correction term as:

$$\Delta Q_{\text{adv}}(a) = F_{\text{adv}}(a)\,\Delta Q(a). \tag{8.23}$$

This formulation has the following intuitive implications:

- If the fairness gain $\Delta F(a)$ is lower than the mean counterfactual gain $\Delta F_{\text{avg}}(a)$, then $F_{\text{adv}}(a)$ is negative, leading to a negative $\Delta Q_{\text{adv}}(a)$. This reduces the attractiveness of action $a$, discouraging further accumulation by already advantaged agents.
- Conversely, if $\Delta F(a)$ exceeds $\Delta F_{\text{avg}}(a)$, then $F_{\text{adv}}(a)$ is positive, and $\Delta Q_{\text{adv}}(a)$ is positive. This boosts the value of actions that help a disadvantaged agent catch up.

### 8.3.3 GIFF-modified Q-values

We combine the original Q-value estimate with the fairness gain and the counterfactual advantage correction to obtain the GIFF-modified Q-value, which we can use in the optimization in Equation 8.3 to compute allocations:

$$Q_f(a) = \Delta F(a) + \delta\,\Delta Q_{\text{adv}}(a)$$

$$\boxed{Q^{\text{GIFF}}(o_i, a, \beta, \delta) = (1-\beta)\,Q(o_i, a) + \beta\,Q_f(a)} \tag{8.24}$$

- $\beta \in [0,1]$ controls the trade-off between efficiency (standard Q-values) and fairness, with $\beta = 1$ leading to allocations based purely off the fairness gain.
- $\delta \geq 0$ controls the degree of advantage correction. Empirically, we observed that small positive values ($< 0.5$) lead to good results, but this is dependent on the environment and fairness function being used.

In practice, by adjusting $\beta$ and $\delta$, the central optimizer can be nudged toward actions that balance both overall system utility and fairness, as measured by $F(\mathbf{Z})$.

With this, we are ready to present the GIFF optimization and problem statement. It is identical to the original ILP (Eqs. 8.3–8.5), except that we replace the Q-values with the GIFF-modified Q-values from Eq. 8.24. A DECA-GIFF problem is defined as the tuple $\mathcal{M}_{\text{GIFF}} = \langle \alpha, S, \mathcal{O}, \{A_i\}_{i\in\alpha}, T, R, \gamma, c, F, \beta, \delta \rangle$, where $F$ is the fairness function and $\beta, \delta$ are the GIFF parameters. The goal is to solve:

$$\max_{x_i(a)\in\{0,1\}} \quad \sum_{i\in\alpha}\sum_{a\in A_i} x_i(a) \cdot Q^{\text{GIFF}}(o_i, a, \beta, \delta) \tag{8.25}$$

$$\text{subject to} \quad \sum_{a\in A_i} x_i(a) = 1, \quad \forall i \in \alpha \tag{8.26}$$

$$\sum_{i\in\alpha}\sum_{a\in A_i} x_i(a) \cdot c(a)_k \leq \mathcal{R}_k, \quad \forall k \in \{1, \ldots, K\} \tag{8.27}$$

**Approximation for Distributed Computation:** Computing the counterfactual fairness gains requires access to the Q-values of all agents. However, in many practical settings, collecting the true Q-values from every agent at each time-step may be infeasible due to communication constraints or scalability concerns.

To reduce this overhead, we adopt an approximation where each agent assumes that all other agents evaluate actions similarly to themselves. That is, agent $i$ approximates the Q-values of other agents $j \in \alpha_c$ as:

$$Q(o_i, a) \simeq Q(o_j, a) \quad \forall j \in \alpha_c, \tag{8.28}$$

where $\alpha_c \subseteq \alpha$ is the subset of agents competing for $a$.

This assumption enables each agent to use their own Q-value estimates as stand-ins for those of others when computing fairness-aware updates. As a result, the modified Q-values can be computed using only the current utility vector $\mathbf{Z}_t$ and the agent's local Q-values, avoiding the need for global Q-value communication.

**A Note on Computational Overhead:** Assuming that evaluating the fairness function on a payoff vector takes constant time, computing the GIFF-modified Q-value for one agent involves two main steps for each available action. First, the fairness gain for an action is computed in constant time. Second, the advantage correction term requires evaluating the counterfactual fairness gain for up to $n$ candidate agents, resulting in $O(n)$ time per action.

With an average of $m$ actions available per agent, the overall computational overhead for calculating the modified Q-values for one agent is $O(m \cdot n)$. The total time for all agents, then, is $O(m \cdot n^2)$, which is much smaller than evaluating fairness over the total combinatorial search space of $O(m^n)$ joint actions.

## 8.4 Theoretical Results

In this section, we provide theoretical guarantees for the core mechanism of GIFF. The algorithm's fairness-aware Q-value, $Q^{\text{GIFF}}$, relies on a tractable surrogate for the true, combinatorial fairness improvement of a joint allocation. Specifically, GIFF approximates the true joint gain by summing the individual, local fairness gains of each agent's action (or Q-value), a quantity we formally define as the surrogate $S$. We now prove three key properties of this design: (1) This surrogate $S$ is a principled lower bound on the true fairness improvement for several canonical fairness functions; (2) Maximizing this surrogate is guaranteed to improve as the fairness weight $\beta$ increases; and (3) How these guarantees on our surrogate translate to guarantees on the realized, real-world fairness.

Throughout this section, let $\mathbf{Z} = (z_1, \dots, z_n) \in \mathbb{R}^n$ denote the current payoff vector, and let $y = (y_1, \dots, y_n) \in \mathbb{R}^n_{\geq 0}$ denote the increments from a feasible allocation in the current round. For a fairness function $F : \mathbb{R}^n \to \mathbb{R}$ we define:

$$\Delta_{\text{joint}} := F(\mathbf{Z} + y) - F(\mathbf{Z}), \qquad \Delta_i^{\text{local}} := F(\mathbf{Z} + y_i e_i) - F(\mathbf{Z}), \qquad S := \sum_{i=1}^{n} \Delta_i^{\text{local}},$$

where $e_i$ is the $i$-th unit vector. We call $\Delta_{\text{joint}}$ the *realized fairness improvement*, $\Delta_i^{\text{local}}$ the *local fairness gain* for agent $i$, and $S$ the *sum of local gains*, which we term as GIFF's **surrogate**.

Our proofs rely on the following assumption:

**Assumption 1** (Nonnegative increments)**.** *Any coordinate's change in utility is nonnegative for any allocation: $y_i \geq 0$ for all $i$.*

**Assumption 2** (Q-value correctness)**.** *For every agent $i$ and action $a$, the estimated Q-value equals the true utility increment:*

$$Q(o_i, a) \ = \ \Delta z_i(a),$$

*where $\Delta z_i(a)$ denotes the change in payoff $z_i$ that results from assigning action $a$ to agent $i$. In other words, Q-values are perfectly accurate predictors of payoff increments.*

Our results fall into three families:

1. A *Local–Gain Lower Bound*, showing that the surrogate never overstates realized fairness improvements.
2. A *Monotonicity in $\beta$*, showing that surrogate fairness is nondecreasing as the fairness weight $\beta$ increases.
3. *Slack bounds*, quantifying the gap between surrogate and realized fairness to turn surrogate guarantees into realized guarantees.

### 8.4.1 Local–Gain Lower Bound

Our first main result establishes that GIFF's per-agent local gains form a certified lower bound on the realized joint fairness change. The result covers four canonical metrics: $\alpha$-fairness, negative variance, generalized Gini (GGF), and maximin.

**Theorem 3** (Local–Gain Lower Bound)**.** *Let $\mathbf{Z} = (z_1, \ldots, z_n) \in \mathbb{R}^n$ be a payoff vector and let $y = (y_1, \ldots, y_n) \in \mathbb{R}^n_{\geq 0}$ be a nonnegative increment vector. For each fairness function $F$ below, the realized joint gain dominates the sum of local gains:*

$$\Delta_{\text{joint}} := F(\mathbf{Z} + y) - F(\mathbf{Z}) \ \ \geq \ \sum_{i=1}^{n} \Delta_i^{\text{local}}, \qquad \Delta_i^{\text{local}} := F(\mathbf{Z} + y_i e_i) - F(\mathbf{Z}),$$

*under the stated assumptions:*

1. *$F = F_\alpha$ ($\alpha$-fairness): $\alpha \geq 0$ and $z_i, \ z_i + y_i$ lie in the domain of $U_\alpha$ (so $z_i > 0$ and $z_i + y_i > 0$ when $\alpha = 1$).*

*2. $F = F_{\mathrm{var}}$ (negative variance): no further assumptions (beyond $y \geq 0$).*

*3. $F = F_{\mathrm{GGF}}$ (generalized Gini) with nonincreasing weights $w_1 \geq \cdots \geq w_n$.*

*4. $F = F_{\min}$ (maximin): no further assumptions (beyond $y \geq 0$).*

*Moreover, equality holds in (1); in (2) equality holds iff at most one $y_i > 0$; and in (4) equality conditions are given in Lemma 9.*

**Lemma 6** (α-fairness: exact additivity)**.** *Let $F_\alpha(\mathbf{Z}) = \sum_{i=1}^n U_\alpha(z_i)$ with $U_\alpha(t) = \frac{t^{1-\alpha}}{1-\alpha}$ for $\alpha \neq 1$ (domain $t \geq 0$) and $U_1(t) = \log t$ (domain $t > 0$). If $z_i$ and $z_i + y_i$ are in the domain for all $i$, then*

$$F_\alpha(\mathbf{Z} + y) - F_\alpha(\mathbf{Z}) = \sum_{i=1}^n \big[F_\alpha(\mathbf{Z} + y_i e_i) - F_\alpha(\mathbf{Z})\big].$$

*Proof.* By separability, $F_\alpha(\mathbf{Z} + y) - F_\alpha(\mathbf{Z}) = \sum_i [U_\alpha(z_i + y_i) - U_\alpha(z_i)]$, and for a single-coordinate update $F_\alpha(\mathbf{Z} + y_i e_i) - F_\alpha(\mathbf{Z}) = U_\alpha(z_i + y_i) - U_\alpha(z_i)$. Summing over $i$ yields the identity. □

**Lemma 7** (Negative variance: nonnegative synergy)**.** *Let $F_{\mathrm{var}}(\mathbf{Z}) = -\mathrm{Var}(\mathbf{Z})$ with $\mathrm{Var}(\mathbf{Z}) = \frac{1}{n}\sum_{i=1}^n (z_i - \mu)^2$ and $\mu = \frac{1}{n}\sum_i z_i$. If $y \geq 0$, then*

$$F_{\mathrm{var}}(\mathbf{Z} + y) - F_{\mathrm{var}}(\mathbf{Z}) \;\; \geq \;\; \sum_{i=1}^n \big[F_{\mathrm{var}}(\mathbf{Z} + y_i e_i) - F_{\mathrm{var}}(\mathbf{Z})\big],$$

*with equality iff at most one $y_i > 0$.*

*Proof.* Use $F_{\mathrm{var}}(\mathbf{Z}) = \mu^2 - \frac{1}{n}\sum_i z_i^2$. Let $S = \sum_i z_i$ and $Y = \sum_i y_i$. Then

$$F_{\mathrm{var}}(\mathbf{Z} + y) - F_{\mathrm{var}}(\mathbf{Z}) = \frac{2SY + Y^2}{n^2} - \frac{2}{n}\sum_i z_i y_i - \frac{1}{n}\sum_i y_i^2.$$

For a single update $y_i$, $F_{\mathrm{var}}(\mathbf{Z} + y_i e_i) - F_{\mathrm{var}}(\mathbf{Z}) = \frac{2Sy_i + y_i^2}{n^2} - \frac{2}{n} z_i y_i - \frac{1}{n} y_i^2$. Summing over $i$ gives

$$\sum_i \Delta_i^{\mathrm{local}} = \frac{2SY}{n^2} - \frac{2}{n}\sum_i z_i y_i + \Big(\frac{1}{n^2} - \frac{1}{n}\Big)\sum_i y_i^2.$$

Subtracting yields $\Delta_{\mathrm{joint}} - \sum_i \Delta_i^{\mathrm{local}} = \frac{Y^2 - \sum_i y_i^2}{n^2} = \frac{2}{n^2}\sum_{i<j} y_i y_j \geq 0$, with equality iff at most one $y_i > 0$. □

**Lemma 8** (GGF: joint gain dominates locals)**.** *Let* $F_{\mathrm{GGF}}(\mathbf{Z}) = \sum_{k=1}^{n} w_k\, z_{(k)}$ *with nonincreasing weights* $w_1 \geq \cdots \geq w_n$ *(and* $w_{n+1} := 0$*), where* $z_{(1)} \leq \cdots \leq z_{(n)}$ *are the sorted entries of* $\mathbf{Z}$*. If* $y \in \mathbb{R}^n_{\geq 0}$*, then*

$$F_{\mathrm{GGF}}(\mathbf{Z}+y) - F_{\mathrm{GGF}}(\mathbf{Z}) \;\geq\; \sum_{i=1}^{n} \big[ F_{\mathrm{GGF}}(\mathbf{Z}+y_i e_i) - F_{\mathrm{GGF}}(\mathbf{Z}) \big].$$

*Proof.* Use the equivalent form $F_{\mathrm{GGF}}(\mathbf{Z}) = \sum_{k=1}^{n} v_k\, S_k(\mathbf{Z})$ with $v_k := w_k - w_{k+1} \geq 0$ and $S_k(\mathbf{Z}) = \sum_{j=1}^{k} z_{(j)}$. Since $v_k \geq 0$, it is enough to prove the claim for each $S_k$ and sum with weights $v_k$.

Fix $k$ and let $\tau$ be the $k$-th smallest value in $\mathbf{Z}$. Set $L = \{i : \; z_i < \tau\}$, $T = \{i : \; z_i = \tau\}$, $c = k - |L|$. Any $k$-subset achieving $S_k(\mathbf{Z})$ must include all of $L$ and exactly $c$ indices from $T$.

*Local updates.* For a single-coordinate increase $y_i \geq 0$,

$$S_k(\mathbf{Z}+y_i e_i) - S_k(\mathbf{Z}) = \begin{cases} y_i, & i \in L, \\ y_i, & i \in T \text{ and } c = |T|, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore

$$\sum_{i=1}^{n} \big[ S_k(\mathbf{Z}+y_i e_i) - S_k(\mathbf{Z}) \big] = \sum_{i\in L} y_i \;+\; \mathbf{1}_{\{c=|T|\}} \sum_{i \in T} y_i. \tag{*}$$

*Joint update.* For the full increment $y \geq 0$,

$$\begin{aligned} S_k(\mathbf{Z}+y) = \min_{|J|=k} \sum_{j\in J} (z_j + y_j) \;&\geq\; \min_{\substack{C \subseteq T \\ |C|=c}} \left[ \sum_{j \in L} (z_j + y_j) + \sum_{j\in C} (z_j + y_j) \right] \\ &= S_k(\mathbf{Z}) + \sum_{j\in L} y_j + \min_{\substack{C\subseteq T \\ |C|=c}} \sum_{j \in C} y_j. \end{aligned}$$

If $c = |T|$, the minimum over $C$ equals $\sum_{j\in T} y_j$; otherwise it is $\geq 0$. Thus

$$S_k(\mathbf{Z}+y) - S_k(\mathbf{Z}) \;\geq\; \sum_{j\in L} y_j \;+\; \mathbf{1}_{\{c=|T|\}} \sum_{j\in T} y_j, \tag{**}$$

which matches or exceeds the sum in (*). This proves the claim for $S_k$. Multiplying by $v_k \geq 0$ and summing over $k$ yields the inequality for $F_{\text{GGF}}$. □

**Lemma 9** (Maximin: joint never underestimates locals)**.** *Let $F_{\min}(\mathbf{Z}) = \min_i z_i$, let $m = \min_i z_i$, and let $S = \{i : z_i = m\}$ be the set of minimizers. Let $\sigma$ be the second-smallest baseline value (or $+\infty$ if none), and let*

$$\sigma' := \min_{j \neq i^\star}(z_j + y_j)$$

*be the post-update second-smallest value when $|S| = 1$ with $S = \{i^\star\}$. If $y \geq 0$, then*

$$F_{\min}(\mathbf{Z} + y) - F_{\min}(\mathbf{Z}) \;\geq\; \sum_{i=1}^{n}\big[F_{\min}(\mathbf{Z} + y_i e_i) - F_{\min}(\mathbf{Z})\big].$$

***Equality conditions.***

- *If $|S| \geq 2$ (multiple minima), equality holds iff $\min_{i \in S} y_i = 0$. Otherwise the inequality is strict.*
- *If $|S| = 1$ with $S = \{i^\star\}$ (unique minimum), write*

  $$\textit{local sum} \;=\; \min\{y_{i^\star},\, \sigma - m\}, \qquad \textit{joint} \;=\; \min\{y_{i^\star},\, \sigma' - m\}.$$

  *Equality holds iff either*

  1. *$y_{i^\star} \leq \sigma - m$ (then both sides equal $y_{i^\star}$), or*
  2. *$\sigma' = \sigma$ (then both sides equal $\min\{y_{i^\star}, \sigma - m\}$).*

  *If $y_{i^\star} > \sigma - m$ and $\sigma' > \sigma$, the inequality is strict.*

*Proof. Local updates.* If $i \notin S$, the minimum stays $m$, so the local change is 0. If $|S| \geq 2$ and $i \in S$, raising a single tied minimum still leaves some index at $m$, so the local change is 0. If $|S| = 1$ with $S = \{i^\star\}$, then

$$F_{\min}(\mathbf{Z} + y_{i^\star} e_{i^\star}) - F_{\min}(\mathbf{Z}) = \min\{\, m + y_{i^\star},\, \sigma \,\} - m = \min\{\, y_{i^\star},\, \sigma - m \,\}.$$

Summing over $i$ gives the stated "local sum."

*Joint update.* If $|S| \geq 2$,

$$F_{\min}(\mathbf{Z}+y) - F_{\min}(\mathbf{Z}) = \min\{\, \min_{i \in S} y_i,\, \sigma - m \,\} \;\geq\; 0,$$

so the inequality holds; equality iff $\min_{i \in S} y_i = 0$. If $|S| = 1$ with $S = \{i^\star\}$,

$$F_{\min}(\mathbf{Z}+y) - F_{\min}(\mathbf{Z}) = \min\{\, y_{i^\star},\, \sigma' - m \,\}.$$

Since $\sigma' \geq \sigma$ (because $y \geq 0$ on non-min entries), we have $\min\{y_{i^\star}, \sigma' - m\} \geq \min\{y_{i^\star}, \sigma - m\}$ with equality exactly under (1) or (2) above. $\square$

Now we are ready to prove the main result.

*Proof of Theorem 3.* Each case follows from the corresponding lemma, which establishes $\Delta_{\text{joint}} \geq \sum_i \Delta_i^{\text{local}}$ under the stated assumptions, and provides the equality conditions. $\square$

This theorem implies that GIFF's surrogate never *over-promises*: the realized fairness gain is guaranteed to be at least as large as the surrogate, and is exact for $\alpha$-fairness. Equality conditions for variance and maximin are given in Lemmas 7 and 9. See Subsection 8.4.5 for a detailed discussion of implications.

### 8.4.2 Monotone Surrogate Fairness under $\beta$

Next, we show that as the fairness weight $\beta$ increases, the allocation's sum of local fairness gains is nondecreasing. This gives a monotonic control knob: tuning $\beta$ cannot reduce surrogate fairness, and under uniqueness, any allocation switch yields a strict increase.

**Theorem 4** (Monotone increase of the sum of local fairness gains)**.** *Fix a decision round with baseline payoff vector $\mathbf{Z} \in \mathbb{R}^n$. Let $\mathcal{A}$ be the (finite) set of feasible joint allocations $A = (a_1, \ldots, a_n)$. For each $A \in \mathcal{A}$, define*

$$U(A) \;:=\; \sum_{i=1}^{n} Q(o_i, a_i), \qquad S(A) \;:=\; \sum_{i=1}^{n} \Delta F_i(a_i), \quad \Delta F_i(a_i) \;:=\; F(\mathbf{Z} + y_i(a_i)\, e_i) - F(\mathbf{Z}),$$

*where $y_i(a_i) \geq 0$ is the accounted increment added to $z_i$ if agent $i$ takes $a_i$, and $e_i$ is the $i$-th unit vector. (All $\Delta F_i(\cdot)$ use the same baseline $\mathbf{Z}$.)*

*For $\beta \in [0,1)$, write $\theta = \beta/(1-\beta)$ and consider*

$$G_\theta(A) \;=\; U(A) \;+\; \theta\, S(A).$$

*Let $A^*(\theta) \in \arg\max_{A\in\mathcal{A}} G_\theta(A)$ be any maximizer. Then for any $0 \leq \theta_1 < \theta_2 < \infty$ and any choices $A_1 \in \arg\max G_{\theta_1}$, $A_2 \in \arg\max G_{\theta_2}$,*

$$S(A_2) \;\geq\; S(A_1).$$

*Equivalently, as $\beta$ increases, the chosen allocation's sum of local fairness gains is nondecreasing.*

*Proof.* By optimality of $A_1$ at $\theta_1$ and $A_2$ at $\theta_2$,

$$U(A_1) + \theta_1 S(A_1) \;\geq\; U(A_2) + \theta_1 S(A_2), \tag{1}$$
$$U(A_2) + \theta_2 S(A_2) \;\geq\; U(A_1) + \theta_2 S(A_1). \tag{2}$$

Subtract (1) from (2) to cancel the $U$ terms:

$$(\theta_2 - \theta_1)\big(S(A_2) - S(A_1)\big) \;\geq\; 0.$$

Since $\theta_2 > \theta_1$, we conclude $S(A_2) \geq S(A_1)$. □

**Corollary 2** (Strict increase at a true switch under uniqueness)**.** *If the maximizer is unique at $\theta_1$ and at $\theta_2$, let $A_1 = A^*(\theta_1)$ and $A_2 = A^*(\theta_2)$. If $A_1 \neq A_2$, then*

$$S(A_2) \;>\; S(A_1).$$

*Proof.* From Theorem 4, $S(A_2) \geq S(A_1)$. If equality held, then (1) and (2) above would force $U(A_1) = U(A_2)$ as well, so both $A_1$ and $A_2$ would maximize $G_{\theta_1}$ and $G_{\theta_2}$—contradicting uniqueness. □

**Scope, tie-breaking, and the $\beta \to 1$ endpoint.** Theorem 4 requires only that (i) the feasible set $\mathcal{A}$ for the round is *finite*, and (ii) all $\Delta F_i(\cdot)$ are computed against the *same* baseline $\mathbf{Z}$ for that round. Changing the feasible set (e.g., constraints) or the baseline mid-sweep can break monotonicity; otherwise the result is agnostic to the choice of $F$ and to how $Q$ is obtained, provided $S(A)$ is well defined. Deterministic tie-breaking is *not* required for nondecreasing $S(A^*(\theta))$, but if you want strict improvement at switches without assuming uniqueness, adopt a consistent rule such as: among $G_\theta$-maximizers, first maximize $S(A)$, then $U(A)$. With this rule, any change in the selected allocation across $\theta_1 < \theta_2$ implies $S(A_2) > S(A_1)$. Finally, interpreting $\beta \to 1^-$ as $\theta = \beta/(1-\beta) \to \infty$, the maximizers converge to $\arg\max_{A\in\mathcal{A}} S(A)$, so the monotonicity statement extends continuously to the endpoint $\beta = 1$.

This monotonicity applies to the surrogate $S(A)$, not necessarily the realized fairness $F(\mathbf{Z})$. To connect the two, we require slack bounds.

### 8.4.3 Slack Bounds Between Surrogate and Realized Fairness

Define the *slack* of an allocation as

$$\text{slack} \;:=\; \Delta_{\text{joint}} - S \quad \geq 0.$$

By Theorem 3, slack $\geq 0$ for the given fairness metrics. Here we provide per-metric exact formulas or bounds, so that we can certify not just lower bounds but also two-sided guarantees.

**Lemma 10** (Slack for $\alpha$-fairness). *Let $F_\alpha(\mathbf{Z}) = \sum_i U_\alpha(z_i)$ with $U_\alpha(t) = \frac{t^{1-\alpha}}{1-\alpha}$ for $\alpha \neq 1$ (domain $t \geq 0$) and $U_1(t) = \log t$ (domain $t > 0$). Assume all arguments lie in the domain. Then*

$$\Delta_{\text{joint}} \;=\; S \quad \textit{and} \quad \text{slack} = 0.$$

Interpretation. *The surrogate is exact for any nonnegative increment profile.*

*Proof.* By separability, $F_\alpha(\mathbf{Z}+y) - F_\alpha(\mathbf{Z}) = \sum_i \big[U_\alpha(z_i+y_i) - U_\alpha(z_i)\big]$, while for a single-coordinate update $F_\alpha(\mathbf{Z}+y_i e_i) - F_\alpha(\mathbf{Z}) = U_\alpha(z_i+y_i) - U_\alpha(z_i)$. Summing over $i$ gives $\Delta_{\text{joint}} = S$. □

**Lemma 11** (Slack for negative variance)**.** *Let* $F_{\text{var}}(\mathbf{Z}) = -\text{Var}(\mathbf{Z})$*, and set* $Y := \sum_i y_i$*. Then*

$$\text{slack} \;=\; \frac{Y^2 - \sum_i y_i^2}{n^2} \;=\; \frac{2}{n^2}\sum_{i<j} y_i y_j \;\in\; \Big[0,\; \frac{Y^2}{n^2}\Big(1 - \tfrac{1}{m}\Big)\Big],$$

*where* $m := |\{i : y_i > 0\}|$. Interpretation. *The gap is a pure "synergy" term that grows when gains are spread across more agents; it vanishes when at most one coordinate increases.*

*Proof.* Using $F_{\text{var}}(\mathbf{Z}) = \mu^2 - \frac{1}{n}\sum_i z_i^2$ with $\mu = \frac{1}{n}\sum_i z_i$, the calculation in Lemma 7 gives

$$\Delta_{\text{joint}} - \sum_i \Delta_i^{\text{local}} = \frac{Y^2 - \sum_i y_i^2}{n^2} = \frac{2}{n^2}\sum_{i<j} y_i y_j \;\geq 0.$$

For the upper bound, by Cauchy–Schwarz, with $m$ positive entries, $\sum_i y_i^2 \geq Y^2/m$, hence $Y^2 - \sum_i y_i^2 \leq Y^2(1 - 1/m)$. □

**Lemma 12** (Slack for GGF (nonincreasing weights))**.** *Let* $F_{\text{GGF}}(\mathbf{Z}) = \sum_{k=1}^n w_k\, z_{(k)}$ *with nonincreasing weights* $w_1 \geq \cdots \geq w_n$ *and* $w_{n+1} := 0$*, and let the increments* $y$ *be nonnegative. Set* $Y := \sum_i y_i$*,* $m := |\{i : y_i > 0\}|$*, and* $y_{\max} := \max_i y_i$*. Define*

$$q := \min\Big(m,\; \lfloor Y/y_{\max} \rfloor\Big), \qquad r := Y - q\, y_{\max} \in [0, y_{\max}).$$

*Then*

$$\Delta_{\text{joint}} \;\leq\; y_{\max}\sum_{k=1}^{q} w_k \;+\; r\, w_{q+1}, \qquad \textit{and hence} \qquad \text{slack} \;\leq\; \Big(y_{\max}\sum_{k=1}^{q} w_k + r\, w_{q+1}\Big) - S.$$

*If the baseline order of* $\mathbf{Z}$ *is strict and preserved after the update, writing* $y_{(1)} \leq \cdots \leq y_{(n)}$ *aligned with that order,*

$$\Delta_{\text{joint}} \;=\; \sum_{k=1}^{n} w_k\, y_{(k)} \;=\; S, \quad \textit{so}\ \text{slack} = 0.$$

Interpretation. *The* $y_{\max}$*-cap is a simple data-dependent envelope (depending only on* $m$*,* $Y$*,* $y_{\max}$ *and the weights); it is typically tight when mass is spread.*

*Proof.* Let $y_{[1]} \geq \cdots \geq y_{[n]}$ be the increments sorted descending. By rearrangement,

$$\Delta_{\text{joint}} \;\leq\; \sum_{k=1}^{n} w_k \, y_{[k]}.$$

Maximizing the right-hand side under $0 \leq y_{[k]} \leq y_{\max}$, $\sum_k y_{[k]} = Y$, and at most $m$ positives fills the top $q$ slots with $y_{\max}$ and places the remainder $r$ in slot $q{+}1$, yielding $y_{\max} \sum_{k=1}^{q} w_k + r \, w_{q+1}$. Since slack $= \Delta_{\text{joint}} - S$, the stated slack bound follows. In the no-rank-crossing regime, each baseline rank-$k$ index remains at rank $k$, so the joint change equals $\sum_k w_k y_{(k)} = S$. $\square$

**Lemma 13** (Slack for maximin)**.** *Let* $F_{\min}(\mathbf{Z}) = \min_i z_i$*. Denote* $m^\star := \min_i z_i$ *and* $S := \{i : z_i = m^\star\}$*. Let* $\sigma := \min_{j \notin S} z_j$ *(or* $+\infty$ *if* $S = \{1, \ldots, n\}$*), and define* $y_{\max} := \max_i y_i$*.*

1. *If* $|S| \geq 2$*,*
$$\text{slack} \;=\; \min\bigl\{ \min_{i \in S} y_i, \; \sigma - m^\star \bigr\} \;\in\; [0, \, y_{\max}],$$
*with* slack $= 0$ *iff* $\min_{i \in S} y_i = 0$*.*

2. *If* $|S| = 1$ *with* $S = \{i^\star\}$*, let* $\sigma' := \min_{j \neq i^\star}(z_j + y_j) \geq \sigma$*. Then*
$$\text{slack} \;=\; \min\{ y_{i^\star}, \, \sigma' - m^\star \} \;-\; \min\{ y_{i^\star}, \, \sigma - m^\star \} \;\in\; \bigl[0, \, \min\{y_{i^\star}, \, \sigma' - \sigma\} \bigr].$$
*In particular,* slack $= 0$ *if either* $y_{i^\star} \leq \sigma - m^\star$ *or* $\sigma' = \sigma$*.*

*Proof.* Let $\Delta_{\text{joint}} = F_{\min}(\mathbf{Z} + y) - F_{\min}(\mathbf{Z})$ and $\Delta_i^{\text{local}} = F_{\min}(\mathbf{Z} + y_i e_i) - F_{\min}(\mathbf{Z})$.

(1) If $|S| \geq 2$, every single-coordinate update leaves some entry at $m^\star$, so $\Delta_i^{\text{local}} = 0$ for all $i$ and $S = 0$. The new minimum after the joint update is $\min\{ m^\star + \min_{i \in S} y_i, \; \sigma \}$, hence $\Delta_{\text{joint}} = \min\{\min_{i \in S} y_i, \; \sigma - m^\star\}$ and the stated bounds follow.

(2) If $|S| = 1$ with $S = \{i^\star\}$, then $\Delta_{i^\star}^{\text{local}} = \min\{y_{i^\star}, \sigma - m^\star\}$ and $\Delta_i^{\text{local}} = 0$ for $i \neq i^\star$, so $S = \min\{y_{i^\star}, \sigma - m^\star\}$. After the joint update, the second-smallest value becomes $\sigma' = \min_{j \neq i^\star}(z_j + y_j) \geq \sigma$, hence $\Delta_{\text{joint}} = \min\{y_{i^\star}, \sigma' - m^\star\}$. Subtracting gives the claim and bounds. $\square$

These results allow us to strengthen the monotonicity guarantee: if $S$ increases by more than an upper bound on slack, then realized fairness must also strictly increase. For $\alpha$-fairness, this is immediate because slack $= 0$.

### 8.4.4 Corollary: Strict Realized Fairness at $\beta$-Driven Switches

Combining Theorem 4 with the slack bounds above yields a useful corollary: when reallocations occur as $\beta$ increases, and the increase in surrogate fairness exceeds the slack bound of the previous allocation, realized fairness must strictly improve.

This gives a precise, testable condition:

- For $\alpha$-fairness, every true switch increases realized fairness.
- For variance, a computable quadratic bound applies (Lemma 11).
- For GGF, a $y_{\max}$-cap yields a data-dependent bound, with slack vanishing when no rank crossings occur (Lemma 12).
- For maximin, the gap depends on whether the minimum is unique (Lemma 13).

In practice, this means $\beta$ can be tuned with confidence that surrogate monotonicity is preserved, and—in many metrics—realized fairness improves as well.

### 8.4.5 Practical Implications of the Theoretical Results

The preceding theorems are not only of theoretical interest but also provide practical tools for deploying GIFF in real systems:

**Certified lower bounds.** The Local–Gain Lower Bound guarantees that GIFF's surrogate never overstates realized fairness improvement. This turns the surrogate into a safe proxy: if an allocation is predicted to achieve at least $\varepsilon$ fairness gain, the realized gain is provably $\geq \varepsilon$. For $\alpha$-fairness, the surrogate is exact.

**Safe tuning of $\beta$.** The monotonicity theorem ensures that increasing the fairness weight $\beta$ cannot reduce surrogate fairness. Practitioners can therefore adjust $\beta$ to explore efficiency–fairness trade-offs without fear of hidden regressions.

**Two-sided certificates.** Slack bounds provide upper bounds on how much realized fairness can exceed the surrogate. Together with the lower bound, this yields a sandwich:

$$S \;\leq\; \Delta_{\text{joint}} \;\leq\; S + \text{slack}_{\max}.$$

This makes GIFF auditable: both the minimum guaranteed improvement and the potential gap are known at each round.

**Fairness floors as constraints.** Because $S$ is conservative, one can impose hard constraints of the form $S \geq \varepsilon$ in the allocation ILP, guaranteeing that realized fairness improvement is at least $\varepsilon$ each round.

**Telescoping guarantees.** Summing the Local–Gain Lower Bound over time gives a cumulative guarantee:

$$F(\mathbf{Z}_T) - F(\mathbf{Z}_0) \;\;\geq\;\; \sum_{t=0}^{T-1} S^{(t)}.$$

This yields an auditable trajectory of fairness progress over a deployment horizon.

**Monitoring and debugging.** The bounds enable runtime checks. If the observed joint change $\Delta_{\text{joint}}$ ever falls below the computed $S$, then assumptions (such as nonnegative increments) have been violated, or an implementation error is present.

In short, the theorems make GIFF *deployable*: the system's fairness behavior becomes predictable, auditable, and tunable in ways that are both theoretically grounded and operationally meaningful.

## 8.5 Experimental Results

We show results from two experiments. First, we compare GIFF to an existing domain-specific method for ridesharing, optimizing variance (a distributional metric). Then, we look at SWF-based metrics in a domain highlighting the need for the counterfactual advantage correction.

### 8.5.1 Baseline Comparison in Real-World Domains: Ridesharing

We test our approach in the complicated ridesharing domain [135, 80], where passengers are allocated to drivers in a dynamic matching environment. Part of the complexity also arises from the fact that more than one passenger can be allocated to the same driver, sharing the trip, leading to a huge combinatorial search space that is difficult to directly optimize. Each vehicle estimates the Q-values based on groups of passengers, and the central allocation maximizes this Q-value for all drivers, subject to passenger constraints.

We compare our results to SI [80], designed for myopic fairness in the ridesharing application. SI's fairness objective is to reduce variance in groups of passengers (measured in terms of service rate) and in drivers (measured as differences in trips assigned). In our experiments, we compare both the original SI and its heuristic variant SI(+), which clips negative fairness incentives to focus solely on improvements. We also implement and test a corresponding heuristic in our method, denoted GIFF(+).

Both GIFF and SI start from the same base model that predicts raw Q-values (indicated by the red X's in Figure 8.1, which represents the baseline with no fairness adjustments). Our goal is to demonstrate that GIFF not only improves fairness over SI, but also avoids the drawbacks of SI(+) at high fairness weights. For passengers, fairness is measured as the variance in service rate for groups traveling between source and destination neighborhoods ($\mathbf{Z}_p$). For drivers, the objective is to minimize variance in driver income, measured by the number of trips per driver ($\mathbf{Z}_d$). In our simulation, 1000 vehicles are deployed on the island of Manhattan, using a real-world dataset from New York City, capturing passenger requests between 8am and 12pm during the busy morning hours.

Our results, shown in Figure 8.1, reveal the following observations:

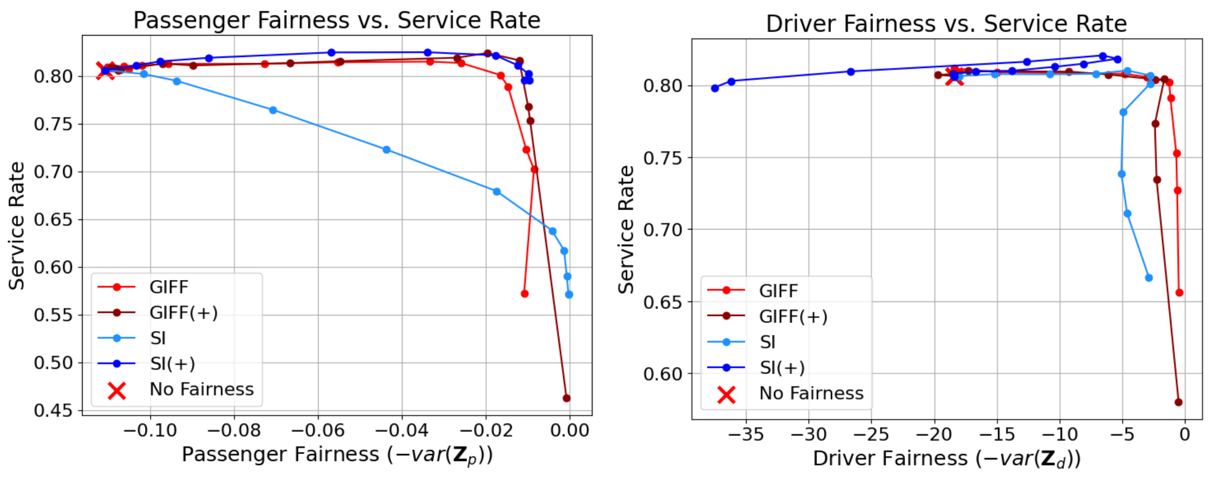
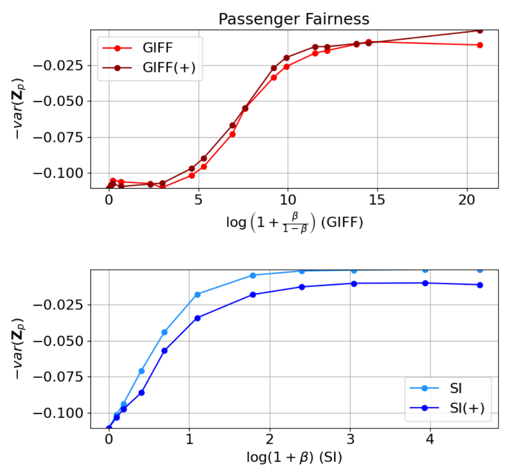


Figure 8.1: Comparison of fairness versus system utility. Each line is plotted in order of increasing fairness tradeoff weight $\beta$, starting from the red X ($\beta = 0$) Left: Passenger fairness (measured as $-var(\mathbf{Z}_p)$) versus overall service rate. Right: Driver fairness (measured as $-var(\mathbf{Z}_d)$) versus overall service rate. The red X indicates the baseline performance (raw Q-values without fairness adjustments). GIFF consistently improves the fairness-utilty tradeoff over SI. While SI(+) provides a marginal improvement for passengers, incorporating the same heuristic into GIFF (GIFF(+)) yields performance on par with SI(+). For drivers, however, SI(+) starts to degrade fairness at high fairness weights, whereas GIFF maintains a stable fairness–efficiency tradeoff.

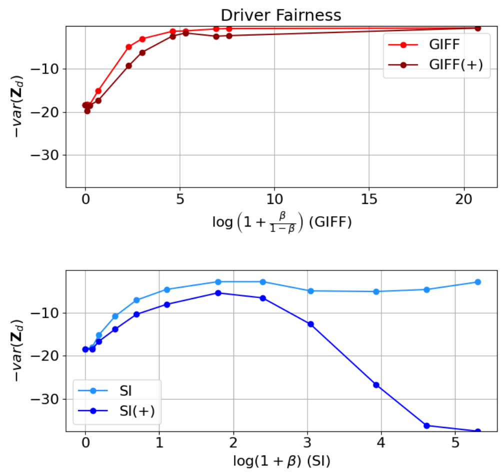
1. **Overall Superiority of GIFF:** GIFF outperforms SI for both drivers and passengers. For passengers, while SI(+) seems to be a good heuristic—yielding slightly better fairness than SI—adding the same clipping heuristic to GIFF (GIFF(+)) brings its performance in line with SI(+). This indicates that GIFF, by itself, already provides significant fairness improvements, and simple heuristics can further boost its performance when long-term Q-values are not available.

2. **Tradeoff at High Fairness Weights:** For drivers, GIFF maintains a favorable fairness–efficiency tradeoff even as the fairness weight approaches 1, while SI(+) begins to make fairness worse. In fact, SI(+) eventually results in lower fairness than the starting base model (indicated by the red X), highlighting a critical drawback of the heuristic fairness approach when overemphasized.

3. **Differences in Reward Availability:** In the passenger scenario, where we lack Q-values and rely on a binary 0–1 reward for assigning a passenger, the heuristic appears



(a) Passenger Fairness



(b) Driver Fairness

Figure 8.2: Variance vs. Fairness Weight. Top row: GIFF results using $\log\left(1+\frac{\beta}{1-\beta}\right)$. Bottom row: SI results using $\log(1+\beta)$.

to capture long-term behavior reasonably well. However, for drivers—where true Q-values are available—the baseline GIFF approach delivers more stable and consistent improvements, making GIFF(+) less effective than GIFF alone.

4. **Robustness of GIFF:** Overall, these results underscore the strength of GIFF: It works well out of the box for both drivers and passengers with a non-linear fairness metric, and reasonable heuristic adjustments can further improve performance in scenarios with limited long-term value information.

**Fairness with changing $\beta$:** In SI, the objective is additive ($U + \beta F$), as opposed to GIFF's weighted combination (Eq. 8.10). To compare the effect of this tradeoff weight on fairness, we transform the weights for GIFF as $\frac{\beta}{1+\beta}$, and plot the change in fairness for both SI and GIFF with this hyperparameter on a logarithmic scale. Figure 8.2 shows the tradeoff between fairness weight and the variance in utilities $\mathbf{Z}_p$ and $\mathbf{Z}_d$.

For passengers, both GIFF and SI keep improving fairness as $\beta$ is increased, as shown in the left panels. In the top right panel, GIFF continues to lower the variance for drivers as the fairness weight increases. However, SI(+) (bottom right) eventually worsens fairness, even falling below the baseline performance. This demonstrates that GIFF achieves a more stable fairness–utilty tradeoff.

### 8.5.2 Generalization to Homelessness Prevention

To demonstrate the versatility of our approach beyond dynamic, Q-value-based environments, we test GIFF in the domain of homelessness prevention using a real-world dataset [70]. Here, the task is to assign households to one of four interventions to minimize the total probability of re-entry into homelessness. The "cost" of assigning household $h$ to intervention $a$ is given by a pre-computed counterfactual probability, $\Pr(h, a)$. We refer readers to Chapter 7 for more details on the dataset and problem setup.

Our framework is adapted to this new context by treating it as a cost-minimization problem. We use the negative of the re-entry probabilities as utility values, so that $Q^{\text{GIFF}}(h, a, \beta, \delta) = (1 - \beta)\,(-\Pr(h, a)) + \beta\, Q_f(a)$. To further highlight GIFF's flexibility, we move beyond variance reduction and adopt the **Gini coefficient** as the fairness metric, $F_{\text{gini}} = -\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}|z_i - z_j|}{2n\sum_{k=1}^{n} z_k}$, where $z_i \in [0, 1]$ is the average re-entry probability for a given demographic group.

The dataset includes 38 household features (e.g., race, gender, family size), and we run 38 independent experiments, each defining fairness groups based on one feature. This allows us to assess the robustness of each method across a wide variety of fairness definitions. We call the adapted version of SI used in this setting SI-X, as it optimizes variance in re-entry probabilities, which is a different objective from the original SI method.

To evaluate performance across these 38 experiments, we introduce two metrics:

- **Price of Fairness (PoF):** The ratio of the total re-entry probability with fairness adjustments to the baseline (fairness-unaware) total probability. A PoF of 1.05 means a 5% increase in the overall re-entry rate.
- **Benefit of Fairness (BoF):** The percentage reduction in the Gini coefficient compared to the baseline, calculated as $1 - \frac{\text{Gini(new)}}{\text{Gini(base)}}$.

Figure 8.3 summarizes the results by plotting the distribution of BoF achieved for a given PoF threshold. Each vertical slice represents the BoF distribution over all 38 feature-based groupings. The top panel shows that GIFF is a more effective and reliable method. On 90% of the features, GIFF is able to get 60% improvement in $F_{gini}$ compared to the baseline. GIFF consistently yields a higher **mean** BoF. More importantly, GIFF demonstrates

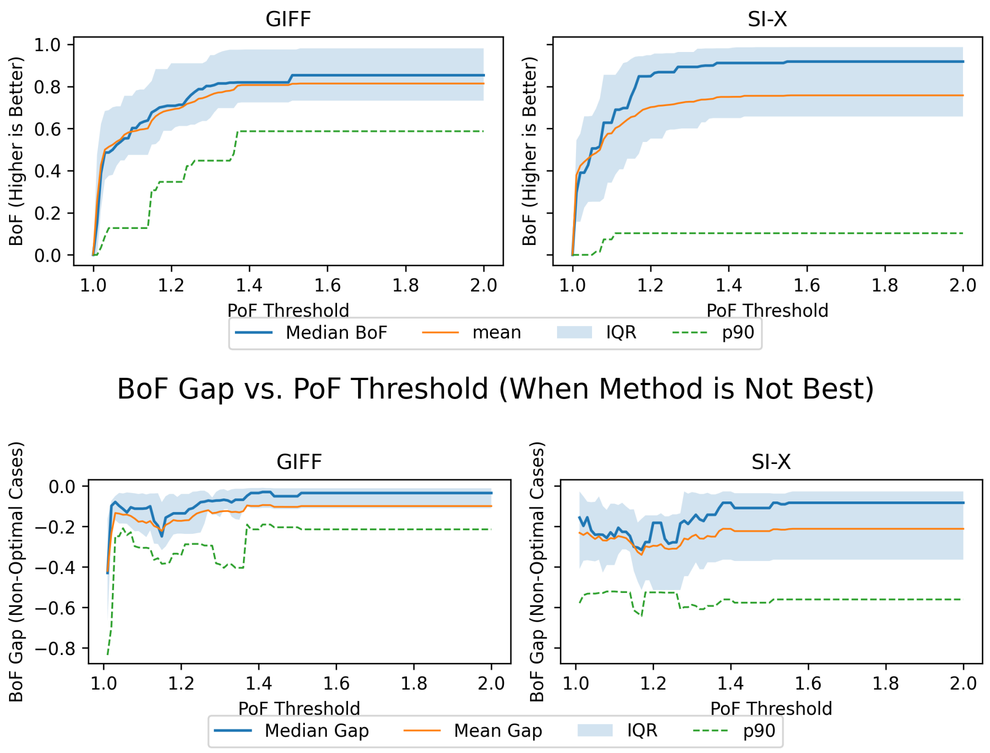


Figure 8.3: Results for the homelessness dataset. Top: Comparison of the benefit of fairness (BoF) distribution as the price of fairness (PoF) threshold is increased. Bottom: The BoF gap compared to the best method, excluding BoF=0. Each vertical slice corresponds to the distribution over all 38 features.

superior worst-case performance; its 90th percentile is substantially higher than that of SI-X, indicating that GIFF avoids the severe fairness failures that SI-X is prone to on certain groups. The bottom panel reinforces this conclusion by analyzing the **BoF Gap**, which measures how much a method underperforms *only on the tasks where it was not the best*. The gap for GIFF is minimal and concentrated near zero, showing that even when it is not the top performer, it is a close second. In contrast, SI-X exhibits a wide gap distribution, with its 90th percentile exceeding 0.4. This means that when SI-X fails, it fails badly, achieving fairness outcomes that are dramatically worse than what is possible with GIFF.

Overall, these results in a distinct problem domain with a non-linear fairness metric confirm that GIFF is a robust and broadly applicable framework for integrating fairness into resource allocation systems.

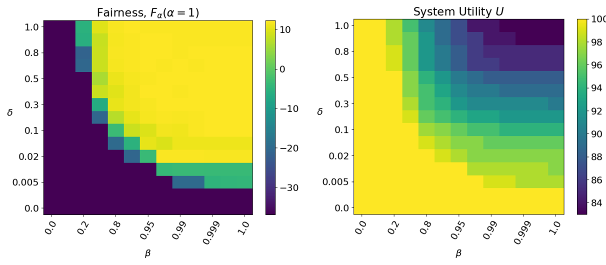
### 8.5.3 Job Allocation Environment and Advantage Correction

In this section, we evaluate our method in the challenging Job Allocation environment. The environment consists of one job that yields a reward of 1 when occupied and 4 agents that can either `occupy` or `forfeit` the job. Only one agent can occupy the job at any given time. When an agent occupies the job, it earns a reward; however, to allow another agent to take over, the current agent must forfeit—resulting in a step with no reward. This process is repeated over 100 time steps.

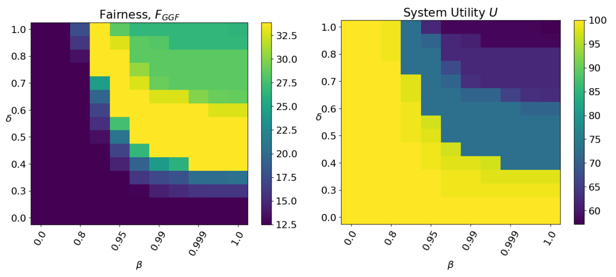
If a single agent occupies the job continuously, it can accumulate up to 100 reward. In contrast, a simple turn-taking (myopic) strategy—where an agent occupies for one step and then forfeits—yields a total of 50 system utility, which is shared roughly equally among the 4 agents (approximately 12.5 utility per agent). By comparison, an oracle solution that plans over the full horizon can achieve 96 total reward, with each agent receiving 24 utility. This oracle solution is hard to compute as the optimal strategy is highly dependent on the number of time steps. We want to test if our method can find this hard fairness optima by just varying $\beta$ and $\delta$ without learning.

We study two fairness metrics: $\alpha$-fair and GGF, which are both additive functions of reweighted agent utilities. Since both metrics are **Efficient**, the change in fairness $\Delta F$ is always positive for any action. Further, without the advantage correction term $\Delta Q_{\text{adv}}$, the forfeit action is never favored for any agent, as the total fair-efficient value for that allocation is always less than the value with the agent continuing to occupy the job. Our experiments show that without the advantage correction term, a policy with $\beta = 1$ results in outcomes that are as unfair as those with $\beta = 0$. The advantage correction term adjusts the Q-values by reducing the value of actions that offer less-than-average fairness gains for agents already better off, thereby encouraging more equitable decision-making even when the agents lack full information about the remaining time steps.

Figure 8.4a shows the performance using the $\alpha$-fair metric (Eq. 2.9) with $\alpha = 1$, equivalent to log Nash welfare. With a moderate $\delta$ (around 0.1), the model achieves near-optimal performance. As $\delta$ increases, fair behavior emerges at lower $\beta$ values, though with a slight drop in utility when $\beta = 1$. Figure 8.4b presents the results for the GGF metric (Eq. 2.10), with weights $w_i = 2^{-i}$. Here, the impact of advantage correction becomes noticeable around $\delta \approx 0.3$, and by $\delta \approx 0.8$, the policy converges to a myopic turn-taking solution at $\beta = 1$. As



(a) $\alpha$-fair results



(b) GGF results

Figure 8.4: Fairness and utility for the Job Allocation environment as functions of $\beta$ and $\delta$.

opposed to $\alpha$-fairness, GGF has a distinct band with noticeably higher fairness and utility before moving into the myopic fairness behavior. $F_\alpha$ is less sensitive to changes in utility when all agents are doing better, since it uses the log of agent utilities.

For both cases, as $\delta \cdot \beta$ increases, the fairness correction relies less on the long-term value and more on trying to be equitable. In the limit, this leads to the turn-taking solution. We also observe the general pattern that as $\delta$ increases, the transition from utilitarian to fair behavior happens at smaller $\beta$. Interestingly, in both cases we are able to identify the optimal strategy with no information about the time horizon, with a simple evaluation-only grid search.

Overall, these results demonstrate that the advantage correction term is crucial for balancing total utility and fairness in environments where long-term planning is challenging. The correction mechanism effectively nudges the policy towards more equitable outcomes in the Job Allocation task.

## 8.6 Conclusion

In this chapter, we introduced GIFF—a framework that leverages existing Q-values to infer fairness gains and adjust multi-agent resource allocations without any additional training. GIFF serves as a generalization of SI, decomposing the long-term effect of actions into a fairness gain and a counterfactual advantage correction, enabling a principled tradeoff between efficiency and equity. Empirical evaluations in ridesharing and job allocation domains demonstrate that GIFF can achieve competitive fairness–utility trade-offs and allow elicitation of expressive and equitable policies.

A key strength of GIFF lies in its simplicity and minimal overhead. Because GIFF operates as a modification step applied to pre-computed Q-values, it is straightforward to integrate into existing RL pipelines. The framework is lightweight—requiring only the maintenance of the accumulated utility vector, with a $O(M \cdot n^2)$ overhead—and introduces just two hyperparameters, $\beta$ and $\delta$, to balance efficiency and fairness during deployment without needing any additional training. GIFF also allows an expressive interpolation between utilitarian and fair behavior. Moreover, the counterfactual advantage correction term effectively simulates altruistic behavior by encouraging better-off agents to forgo top preferences in favor of actions that yield greater fairness improvements, a property that is particularly beneficial when equitable distributions are desired.

Despite these advantages, some limitations warrant further discussion. In environments with sparse rewards or highly stochastic dynamics, the direct correspondence between actions and utility may be less pronounced, which can challenge the reliability of the fairness estimation. In such cases, relying solely on current Q-values may not capture the long-term benefits accurately, suggesting a need for incorporating milestone-based planning or trajectory evaluations.

With this chapter, we conclude Part III. In the previouos 3 chapters, we discussed and developed a completely online method for post-processing Q-values to achieve fair resource allocation, without any additional training. SI and GIFF surface a simple and elegant idea: thinking about the immediate fairness implications of actions and keeping track of historic metrics is often sufficient to achieve significantly improved fairness outcomes. The 'incentives' that SI and GIFF rely on can be applied either centrally by the allocator after receiving agent Q-values, or locally by agents themselves. In the next part, we will look at a different problem: how to make agents learn the long-term fairness impact of actions during training, and how to make them cooperate to achieve fair outcomes. This will be the focus of Part IV.

# Part IV

# Learning and Sampling for Fairer Models

# Chapter 9

# Learning with Fairness through Past-Discounting

## 9.1 Introduction & Contribution

So far we've seen multiple real-world instances of resource allocation systems that align with the Distributed Evaluation, Centralized Allocation framework, and developed methods to improve fairness using post-hoc incentives. However, these methods assume that the value estimates used to compute the incentives are accurate, and do not consider how these values are learned in the first place. In many real-world settings, these values are learned through historical data or online learning, and may be biased or inaccurate.

In the next few chapters, we will study fairness during the learning stage of resource allocation and other machine learning tasks. However, to lay the foundation for this, we first dive into a previously unexplored aspect of fairness in dynamic settings: how to learn in the presence of long-term fairness objectives.

Recent work in multi-agent learning addresses temporal fairness by evaluating cumulative or terminal utilities [66, 176, 3]. Yet, these approaches either assume perfect recall—giving equal weight to all past allocations—or assess each allocation independently. In contrast, insights from behavioral economics and moral psychology reveal that human perceptions of fairness evolve over time. For example, events in the distant past tend to be perceived more abstractly [149], and individuals naturally devalue outcomes that are further removed in time [45]. Empirical studies also indicate that forgiveness increases as the temporal distance from a transgression grows [88, 162]. Additionally, to maintain the Markov property in sequential decision-making, the payoff vector must be included within the state, which can lead to an unboundedly growing state space if all past utilities are retained.

Motivated by these findings, we propose incorporating **past discounting mechanisms** into dynamic resource allocation. By discounting historical utilities, our approach offers a principled compromise between instantaneous and perfect-recall fairness. This method not only aligns more closely with observed human behavior but also ensures that the augmented state space remains bounded—a critical property for the convergence and scalability of reinforcement learning algorithms.

**Contributions:** The main contributions of this chapter are:

1. We identify and formalize a critical failure mode in long-term fairness methods [3, 176]: **perfect-recall causes unbounded state-space growth**, rendering learning intractable over long horizons.
2. We propose **past-discounted fairness memories**, a novel, behaviorally-grounded framework that balances historical context with computational feasibility by applying a geometrically decaying weight to past utilities.
3. We provide a **rigorous theoretical proof** that our past-discounting mechanism guarantees a bounded and horizon-independent augmented state space, retaining the Markov property while enabling tractable learning.
4. We empirically **demonstrate the necessity of discounting the past**, showing that an RL agent successfully learns fair policies with our method while systematically failing with perfect recall as the time horizon increases.

By integrating theoretical analysis with behavioral insights, we aim to provide a strong argument for using past discounts for fair resource allocation in dynamic environments. This also forms a foundation for the subsequent chapters, where we explore learning mechanisms that can effectively incorporate long-term fairness objectives. We note that the implications of this work extend beyond the DECA framework, and are relevant to a wide range of sequential decision-making problems involving fairness.

## 9.2 Background

### 9.2.1 Related Work

Fairness in resource allocation has a rich history, but traditional literature has focused on static, one-shot problems. Concepts like proportionality, envy-freeness, and maximin share guarantees [17, 119] provide robust solutions for individual decision points. However, their myopic nature makes them insufficient for dynamic settings like taxi matching or aid distribution, where long-term equity necessitates reasoning over multiple time steps. Even in the sequential setting, many approaches consider fairness over only the current valuations of resources or considering future value, but do not consider past memories [139, 63]. We call this **myopic** or **instantaneous fairness**.

Recognizing this, recent research has explored temporal fairness across various domains. In multi-agent reinforcement learning (MARL), some approaches constrain per-step allocations or evaluate cumulative utilities at a terminal stage [66, 176]. More advanced mechanisms include forecasting for ride-hailing applications [80] and hierarchical frameworks to mediate between efficiency and fairness [66]. Similar challenges are addressed in sequential voting [85] and online fair division [5], where a series of interdependent choices necessitates a long-term perspective. These works establish a clear consensus: achieving meaningful fairness in sequential settings requires memory of the past.

The dominant paradigm for incorporating this memory is **perfect recall**, where agents track the complete sum of historical utilities [176, 138, 3, 66, 28, 91]. The recent work by Alamdari et al. [3] formalizes this by treating history-dependent fairness as a non-Markovian problem. They show that by augmenting the state with a memory of past utilities (i.e., perfect recall), one can restore the Markov property. However, this approach introduces a critical, unaddressed challenge: for a perfect memory, the augmented state space grows unboundedly with the time horizon. This makes learning computationally intractable, especially in long-horizon tasks. Our work directly addresses this limitation: **we seek a memory mechanism that is not only Markovian but also improves tractability.**

This leaves the field caught between two extremes: myopic fairness, which is computationally simple but ignores long-term equity (especially over past allocations), and perfect-recall fairness, which is equitable in theory but can be computationally infeasible, as we show later. Our work introduces a third paradigm that occupies the practical middle ground: **past-discounted fairness**. This approach is motivated by strong evidence from behavioral economics and moral psychology, which shows that human fairness judgments are sensitive to temporal distance. People naturally devalue outcomes that are further in the past [149, 45] and are more forgiving of older transgressions [88, 162]. This concept is analogous to future-discounting in standard reinforcement learning [106, 144], but its application to past-oriented fairness memories to ensure tractability remains unexplored.

In summary, while prior work has correctly identified the need for memory in temporal fairness, it has not provided a computationally viable solution for long-horizon problems. Our contribution is to formalize a behaviorally-grounded, past-discounted memory that is (i) sufficient to restore the Markov property with a bounded, horizon-independent state, and (ii) provides a practical mechanism for learning fair policies in complex, long-running systems. To the best of our knowledge, no existing work has considered using past discounts to learn fair behavior.

### 9.2.2 Preliminaries

Social welfare functions provide a mathematical formulation to evaluate both fairness and efficiency in resource allocation. Given a utility vector

$$\mathbf{Z} = (z_1, z_2, \ldots, z_n) \tag{9.1}$$

representing the utilities received by $n$ agents, many social welfare functions have been considered in the literature, including:

- **Utilitarian Welfare**, which maximizes the total utility without explicit fairness considerations [132].
$$W_U(\mathbf{Z}) = \sum_{i=1}^{n} z_i, \tag{9.2}$$
- **Egalitarian Welfare**, which prioritizes the well-being of the worst-off agent. This is also known as Rawlsian or maximin fairness [126].
$$W_{MMF}(\mathbf{Z}) = \min_i z_i, \tag{9.3}$$

- **Nash Welfare**, which balances fairness and efficiency. This measure is rooted in Nash's bargaining solution [108] and has been influential in fair division research [20].

$$W_N(\mathbf{Z}) = \prod_{i=1}^{n} z_i, \tag{9.4}$$

- **Generalized Gini Welfare**, which is a family of functions that applies rank-based weights to the agent utilities, offering a flexible approach to balancing equity and efficiency [9, 93, 176].

Traditionally, these functions evaluate fairness at a single point in time, thereby ignoring the history of past allocations and expectations for future ones. In dynamic settings like DECA, several approaches extend these welfare functions by incorporating historical and predictive elements. For example, some works cast the allocation problem as a Multi-Agent Reinforcement Learning (MARL) task that optimizes fairness at an intermediate or terminal state [66, 176, 138, 76], while others employ Non-Markovian Decision Processes that explicitly account for the entire past trajectory of allocations [3].

In many formulations, the allocation at time $t$, denoted by $\mathcal{A}^t$, is defined as a mapping of resources to agents such that $\mathcal{A}_i^t$ represents the resources allocated to agent $i$. The welfare corresponding to the post-allocation utility vector $\mathbf{Z}^t|\mathcal{A}^t$ is then given by $W(\mathbf{Z}^t|\mathcal{A}^t)$, and the optimal allocation is defined as:

$$\mathcal{A}^{t*} = \underset{\mathcal{A}}{\operatorname{argmax}}\, W(\mathbf{Z}^t|\mathcal{A}). \tag{9.5}$$

Here, we denote by $u_i^{\mathcal{A}}$ the utility derived by agent $i$ from allocation $\mathcal{A}$, and by $u_i^t$ the utility actually received by agent $i$ at time $t$.

There exist various methods to compute the utility vector $\mathbf{Z}^t|\mathcal{A}$, and these choices influence the resulting allocation. We discuss some popular approaches below, motivating and building up to past-discounted fairness.

## 9.3 Temporal Fairness in Resource Allocation

In dynamic resource allocation, fairness must be evaluated not only on the basis of the current decision but also by considering past allocations and future expectations. In this section, we present three paradigms for temporal fairness: *Instantaneous fairness*, *perfect-recall historical fairness*, and *discounted-recall historical fairness.* We define each approach, illustrate them with examples, and discuss their inherent limitations.

### 9.3.1 Instantaneous Fairness

Instantaneous fairness considers only the current allocation decision. Formally:

$$Z_i^t \mid \mathcal{A} = u_i^{\mathcal{A}}, \tag{9.6}$$

which implies that fairness is assessed solely on the utility $u_i^{\mathcal{A}}$ derived from the current allocation. In this formulation, the welfare function $W$ is optimized based solely on the immediate utilities, yielding an allocation that is deemed fair at that specific time step.

**Definition 4** (Instantaneous Fairness)**.** *An allocation exhibits* instantaneous fairness *at time $t$ if fairness is evaluated solely on the one-step utilities. For any candidate allocation $\mathcal{A}$, define*

$$\mathbf{Z}^t \mid \mathcal{A} = \big(u_1^{\mathcal{A}}, \dots, u_n^{\mathcal{A}}\big). \tag{9.7}$$

*Then an instantaneous-fair allocation at time $t$ is any optimizer*

$$\mathcal{A}^{t*} = \underset{\mathcal{A}}{\operatorname{argmax}}\; W\big(\mathbf{Z}^t \mid \mathcal{A}\big), \tag{9.8}$$

*i.e., it selects the allocation that maximizes the welfare applied to the current-step utility vector.*

Although this approach often produces a solution that is optimal for that particular time step, it neglects the temporal dimension by ignoring both historical allocations and anticipated future resources. For example, consider:

**Example 2.** *Two agents, Alice and Bob, compete for two indivisible items: a cake and a donut. Suppose*

$$\textit{Alice: } (u_{cake}, u_{donut}) = (0.2, 0.5), \quad \textit{Bob: } (0.3, 0.5).$$

*In a purely instantaneous allocation, the donut is assigned to the agent who slightly benefits from it more in that step (Alice, compared to the baseline 0.2). Over repeated interactions, however, Alice may receive a disproportionate number of donuts, leading to a cumulative imbalance.*

Thus, while instantaneous fairness might maximize short-term efficiency, its disregard for temporal dynamics can result in significant long-term disparities.

### 9.3.2 Perfect-Recall Historical Fairness

To capture the temporal aspect of fairness, perfect-recall historical fairness incorporates all past allocations into the fairness evaluation. Instead of relying solely on the instantaneous utility, we define an adjusted utility vector $\mathbf{Z}^t$ that aggregates utilities over all previous steps:

$$Z_i^t = \sum_{\tau=0}^{t} u_i^\tau. \tag{9.9}$$

Consequently, the post-allocation utility becomes:

$$\begin{aligned} Z_i^t \mid \mathcal{A} &= \sum_{\tau=0}^{t-1} u_i^\tau + u_i^{\mathcal{A}} & (9.10)\\ &= Z_i^{t-1} + u_i^{\mathcal{A}}. & (9.11) \end{aligned}$$

In some cases, averaging these utilities over time may be preferable:

$$Z_i^t \mid \mathcal{A} = \frac{Z_i^{t-1} \cdot (t-1) + u_i^{\mathcal{A}}}{t}. \tag{9.12}$$

**Definition 5** (Perfect-Recall Fairness). *An allocation exhibits* perfect-recall fairness *at time t if fairness is evaluated using all past allocations. For any candidate allocation $\mathcal{A}$, define*

$\mathbf{Z}^t \mid \mathcal{A}$ *by either:*

$$\textit{(cumulative)} \quad Z_i^t \mid \mathcal{A} = Z_i^{t-1} + u_i^{\mathcal{A}}, \tag{9.13}$$

$$\textit{(averaged)} \quad Z_i^t \mid \mathcal{A} = \frac{Z_i^{t-1} \cdot (t-1) + u_i^{\mathcal{A}}}{t}. \tag{9.14}$$

*A perfect-recall-fair allocation at time t is any optimizer*

$$\mathcal{A}^{t*} = \underset{\mathcal{A}}{\operatorname{argmax}} \; W\big(\mathbf{Z}^t \mid \mathcal{A}\big), \tag{9.15}$$

*i.e., it selects the allocation that maximizes the welfare applied to the history-aggregated utility vector.*

This approach is especially relevant in domains such as long-term healthcare or education funding, where addressing historical disparities is crucial. However, perfect recall may overcompensate past imbalances. For instance:

**Example 3.** *Suppose Alice is the sole participant for 10 steps and accumulates a high utility. If Bob joins at step 11, perfect-recall fairness might allocate many future resources to Bob to "catch him up." This could be viewed as unfair to Alice, as her early contributions—made in Bob's absence—should not overly penalize her in future allocations.*

### 9.3.3 Discounted-Recall Historical Fairness

To balance the extremes of instantaneous and perfect-recall fairness, we propose *discounted-recall historical fairness*. This paradigm introduces a temporal decay factor $\gamma_p \in [0, 1]$ that gradually diminishes the influence of older allocations. The intuition is that while historical context is important, its influence should naturally decay over time. Such a decay mechanism is inspired by behavioral research, which indicates that humans discount temporally distant events. Furthermore, incorporating past discounts aligns how we consider past utilities with how future rewards are treated in sequential decision-making (SDM), where temporal decay is crucial for ensuring convergence of returns and for tractable computation.

### Discounted Recall with Additive Utilities

In the additive setting, past utilities are discounted and then combined with the current utility:

$$Z_i^t \mid \mathcal{A} = \gamma_p \, Z_i^{t-1} + u_i^{\mathcal{A}}. \tag{9.16}$$

Here, $\gamma_p$ governs the balance between immediate and historical considerations, with $\gamma_p = 0$ reducing to instantaneous fairness and $\gamma_p = 1$ recovering perfect-recall fairness.

**Definition 6** (Discounted-Recall Fairness (Additive))**.** *An allocation exhibits* discounted-recall fairness (additive) *at time $t$ if fairness is evaluated using the past-discounted update. For any candidate allocation $\mathcal{A}$, define*

$$Z_i^t \mid \mathcal{A} = \gamma_p \, Z_i^{t-1} + u_i^{\mathcal{A}} \quad \textit{for } \gamma_p \in [0,1]. \tag{9.17}$$

*A discounted-recall-fair (additive) allocation at time $t$ is any optimizer*

$$\mathcal{A}^{t*} = \operatorname*{argmax}_{\mathcal{A}} \; W\big(\mathbf{Z}^t \mid \mathcal{A}\big), \tag{9.18}$$

*i.e., it selects the allocation that maximizes the welfare applied to the past-discounted cumulative utilities.*

### Discounted Recall with Averaged Utilities

For averaged utilities, both the accumulated utility and the effective time denominator are discounted. Let $d_t$ denote the past-discounted denominator at time $t$. Then:

$$Z_i^t \mid \mathcal{A} = \frac{\gamma_p Z_i^{t-1} \cdot d_{t-1} + u_i^{\mathcal{A}}}{\gamma_p d_{t-1} + 1}. \tag{9.19}$$

**Definition 7** (Discounted-Recall Fairness (Averaged))**.** *An allocation exhibits* discounted-recall fairness (averaged) *at time $t$ if fairness is evaluated using the past-discounted average with denominator $d_t$. For any candidate allocation $\mathcal{A}$, define*

$$Z_i^t \mid \mathcal{A} = \frac{\gamma_p Z_i^{t-1} \cdot d_{t-1} + u_i^{\mathcal{A}}}{\gamma_p d_{t-1} + 1}. \tag{9.20}$$

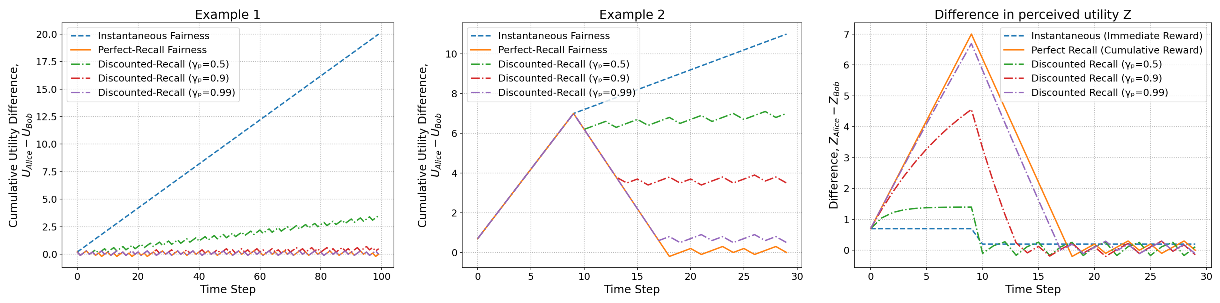


Figure 9.1: Comparison of cumulative utility differences under different fairness paradigms. (Left) Cumulative utility difference, $\sum U_{Alice} - \sum U_{Bob}$, over time for Example 2, where both agents participate from the start. (Center) Cumulative utility difference, $\sum U_{Alice} - \sum U_{Bob}$, over time for Example 3, where only Alice is active initially and Bob joins later. (Right) Difference in perceived utility between Alice and Bob for all three methods for Example 3. This plot shows the effect of $\gamma_p$ on the perceived values, demonstrating how it changes the speed at which we forget past decisions, interpolating between perfect-recall and instantaneous fairness.

*A discounted-recall-fair (averaged) allocation at time $t$ is any optimizer*

$$\mathcal{A}^{t*} = \underset{\mathcal{A}}{\operatorname{argmax}}\ W\big(\mathbf{Z}^t \mid \mathcal{A}\big), \tag{9.21}$$

*i.e., it selects the allocation that maximizes the welfare applied to the past-discounted average utilities.*

In both formulations, the computation of $\mathbf{Z}^t$ depends only on the previous state—specifically, the augmented state comprising $\mathbf{Z}^{t-1}$ (and $d_{t-1}$ in the averaged case). This Markovian structure not only simplifies the computation but also provides a smooth interpolation between instantaneous and perfect-recall fairness, while ensuring that the state space is augmented in a tractable manner.

### 9.3.4 Comparison between the Different Paradigms

Figure 9.1 illustrates the evolution of the cumulative utility difference, $\sum U_{Alice} - \sum U_{Bob}$, under the three fairness approaches: Instantaneous, perfect-recall, and discounted-recall fairness, as discussed in Examples 2 and 3. The allocations are made with the MMF objective, using additive aggregation for perfect-recall and discounted-recall. In Figure 9.1(left), where both agents participate from the start, instantaneous fairness accumulates short-term differences leading to long-term unfairness, while perfect-recall fairness compensates by accounting for all past allocations. Discounted-recall fairness offers a tunable middle ground, with the discount factor $\gamma_p$ controlling how quickly past utilities decay in importance.

In Figure 9.1(center), only Alice is active initially and Bob joins later. All approaches perform similarly in the initial phase. Instantaneous fairness keeps accumulating the imbalance regardless of the history, while both perfect-recall and discounted-recall mechanisms move towards equalizing the past imbalance. Perfect recall only starts allocating resources to Alice again after equalizing the total utility. $\gamma_p$ serves as a tuning knob which lets us control how strongly we want the distant past to affect current allocations.

Finally, Figure 9.1(right) shows the inner workings of each fairness approach by plotting the perceived differences between Alice and Bob in terms of $\mathbf{Z}$. Instantaneous fairness seems to always keep a low difference between the two agents locally, even as the total utility for Alice keeps rising. Perfect-recall keeps an exact track of resources, considering the distribution unfair even after many steps of allocating to Bob only. The decay of the line, particularly visible for $\gamma_p = 0.9$ shows how discounted-recall slowly forgets past decisions, with values close to 1 emulating longer memory.

## 9.4 Theoretical Results

In this section, we present theoretical results that highlight the advantages of past-discounted fairness over perfect-recall fairness. We first focus on the boundedness of the augmented state space, which is crucial for the convergence and scalability of reinforcement learning algorithms, especially in long-horizon settings.

### 9.4.1 Additive Utilities

A key practical advantage of our past-discounted approach is that it bounds the cumulative utility over time. In typical non-discounted fairness frameworks (e.g., [3, 176]), the cumulative utility is computed as $\mathbf{Z}_i^t = \sum_{\tau=0}^{t} u_i^\tau$, which grows linearly with the time horizon (i.e., $\mathbf{Z}_i^t \leq (t+1)u_{\max}$ when $u_i^t \in [0, u_{\max}]$). As a consequence, when the fairness state is augmented with these cumulative utilities, the state space expands unboundedly with time, severely hampering the scalability and learnability of the problem.

**Theorem 5** (State explosion under perfect recall). *Let $u_i^t \in [0, u_{\max}]$ for all agents $i \in \{1, \ldots, n\}$ and times $t \geq 0$, and define perfect-recall memory by*

$$Z_i^t \; = \; \sum_{\tau=0}^{t} u_i^\tau. \tag{9.22}$$

*Then, for any uniform bin width $\Delta > 0$, the number of bins per coordinate needed to represent $Z_i^t$ on a uniform grid satisfies*

$$N_{\text{bins}}^{\text{PR}}(t, \Delta) \; \geq \; \left\lceil \frac{(t+1)\, u_{\max}}{\Delta} \right\rceil, \tag{9.23}$$

*so the grid over $\mathbf{Z}^t \in \mathbb{R}^n$ has cardinality at least $\left(N_{\text{bins}}^{\text{PR}}(t, \Delta)\right)^n$, which grows linearly with $t$ per coordinate and exponentially with $n$. The linear dependence on $t$ is tight.*

*Proof.* For each $i$,

$$0 \; \leq \; Z_i^t \; = \; \sum_{\tau=0}^{t} u_i^\tau \; \leq \; \sum_{\tau=0}^{t} u_{\max} \; = \; (t+1)\, u_{\max}. \tag{9.24}$$

A uniform grid with bin width $\Delta$ covering $[0, (t{+}1)u_{\max}]$ requires at least $\lceil (t{+}1)u_{\max}/\Delta \rceil$ bins per coordinate. Across $n$ coordinates (agents), the number of grid cells multiplies, yielding the stated growth. Tightness holds by taking $u_i^\tau \equiv u_{\max}$, for which $Z_i^t = (t+1)u_{\max}$. □

**Theorem 6** (Horizon-independent boundedness under past discounting). *Fix $\gamma_p \in [0, 1)$ and $u_{\max} > 0$. Let $u_i^t \in [0, u_{\max}]$ and define the past-discounted memory by*

$$Z_i^t \; = \; \gamma_p \, Z_i^{t-1} + u_i^t, \qquad Z_i^0 \; = \; u_i^0. \tag{9.25}$$

*Then, for all $t \geq 0$,*

$$0 \; \leq \; Z_i^t \; \leq \; \frac{u_{\max}}{1 - \gamma_p} \qquad \textit{and} \qquad Z_i^t \; = \; \sum_{k=0}^{t} \gamma_p^{t-k} u_i^k. \tag{9.26}$$

*The upper bound is tight: if $u_i^t \equiv u_{\max}$, then $Z_i^t = u_{\max}/(1 - \gamma_p)$.*

*Proof.* Unrolling the recursion yields

$$Z_i^t \;=\; \sum_{k=0}^{t} \gamma_p^{t-k} u_i^k. \tag{9.27}$$

Since $u_i^k \in [0, u_{\max}]$ and $\gamma_p^{t-k} \geq 0$,

$$0 \;\leq\; Z_i^t \;\leq\; u_{\max} \sum_{k=0}^{t} \gamma_p^{t-k} \;=\; u_{\max} \sum_{j=0}^{t} \gamma_p^j \;\leq\; \frac{u_{\max}}{1-\gamma_p}. \tag{9.28}$$

Moreover, the bound is tight. If $u_i^t \equiv u_{\max}$, then

$$Z_i^t \;=\; u_{\max} \sum_{j=0}^{t} \gamma_p^j \;=\; \frac{u_{\max}}{1-\gamma_p} \left(1-\gamma_p^{t+1}\right), \tag{9.29}$$

so $Z_i^t$ is increasing in $t$ and

$$\lim_{t\to\infty} Z_i^t \;=\; \frac{u_{\max}}{1-\gamma_p}. \tag{9.30}$$

$\square$

**Corollary 3** (Horizon-independent discretization under past discounting)**.** *Under the conditions of Theorem 6, any uniform grid with bin width $\Delta > 0$ requires at most*

$$N_{\text{bins}}^{\text{PD}}(\Delta) \;=\; \left\lceil \frac{u_{\max}}{(1-\gamma_p)\,\Delta} \right\rceil \tag{9.31}$$

*bins per coordinate—independent of $t$. Consequently, a fixed-resolution discretization of the augmented fairness state $\mathbf{Z}^t$ has size independent of the horizon; across $n$ coordinates the grid has at most $\left(N_{\text{bins}}^{\text{PD}}(\Delta)\right)^n$ cells.*

### 9.4.2 Averaged Utilities

Theorems 5 and 6 together show that past-discounted fairness avoids the state explosion inherent in perfect-recall fairness for additive aggregation. Next we provide similar results for the averaged aggregation.

**Theorem 7** (Averaged perfect recall requires unbounded augmented state)**.** *Define the running average update by*

$$Z_i^t \;=\; \frac{(t-1)\,Z_i^{t-1} + u_i^t}{t}, \qquad Z_i^0 = u_i^0, \tag{9.32}$$

*with $u_i^t \in [0, u_{\max}]$. Then:*

*(i) (*Non-Markov in $\mathbf{Z}^t$ alone*) There exist histories $h, h'$ with the same $\mathbf{Z}^t$ but different $t$ such that, for the same $u_i^{t+1}$, the next value $Z_i^{t+1}$ differs. Hence the process is not Markov in $(s^t, \mathbf{Z}^t)$ without carrying $t$ (or an equivalent denominator).*

*(ii) (*Unbounded augmentation*) Any Markov augmentation that appends $t$ yields an auxiliary variable whose support is $\{0, 1, 2, \dots\}$, i.e., unbounded in the horizon.*

*Proof.* (i) From the update,

$$Z_i^{t+1} \;=\; \frac{t\,Z_i^t + u_i^{t+1}}{t+1}, \tag{9.33}$$

which depends on $t$ even when $Z_i^t$ and $u_i^{t+1}$ are fixed. Take two histories with the same $Z_i^t = z$ but different $t$ and the same $u_i^{t+1}$; the resulting $Z_i^{t+1}$ differ, so $\mathbf{Z}^t$ alone is insufficient for Markov evolution. (ii) Appending $t$ (or any one-to-one proxy such as the exact denominator) restores Markovian evolution but makes the augmented state include a variable that grows without bound as $t \to \infty$. □

**Theorem 8** (Past-discounted averaging admits a bounded Markov augmentation)**.** *Let $\gamma_p \in [0, 1)$ and define $d_0 = 0$, $d_t = \gamma_p d_{t-1} + 1$, and*

$$Z_i^t \;=\; \frac{\gamma_p d_{t-1}\,Z_i^{t-1} + u_i^t}{\gamma_p d_{t-1} + 1}, \qquad u_i^t \in [0, u_{\max}]. \tag{9.34}$$

*Then:*

*(i) (*Boundedness*) $d_t = \sum_{k=0}^{t-1} \gamma_p^k \le \frac{1}{1-\gamma_p}$ and $0 \le Z_i^t \le u_{\max}$ for all $t$.*

*(ii) (*Markov sufficiency*) The pair $(\mathbf{Z}^t, d_t)$ is a sufficient memory: the next $(\mathbf{Z}^{t+1}, d_{t+1})$ depends only on $(\mathbf{Z}^t, d_t)$ and current utilities.*

*(iii) (*Non-vanishing responsiveness*) The marginal weight of the current utility in $Z_i^t$ equals*

$$\frac{\partial Z_i^t}{\partial u_i^t} = \frac{1}{\gamma_p d_{t-1} + 1} \xrightarrow[t\to\infty]{} 1 - \gamma_p, \tag{9.35}$$

*so the influence of the current step does not vanish (contrast with the $1/t$ weight under plain averaging).*

*Proof.* (i) The recursion for $d_t$ solves to $d_t = \sum_{k=0}^{t-1} \gamma_p^k \le \frac{1}{1-\gamma_p}$. Multiplying the $Z$-update by $\gamma_p d_{t-1} + 1$ gives

$$(\gamma_p d_{t-1} + 1)\, Z_i^t = \gamma_p d_{t-1}\, Z_i^{t-1} + u_i^t, \tag{9.36}$$

so $Z_i^t$ is a convex combination of $Z_i^{t-1}$ and $u_i^t$ with weights summing to 1; by induction and $u_i^t \in [0, u_{\max}]$, we get $Z_i^t \in [0, u_{\max}]$. (ii) $(\mathbf{Z}^t, d_t)$ updates deterministically from $(\mathbf{Z}^t, d_t)$ and $\mathbf{u}^t$ via the two recursions, hence is Markov-sufficient. Both coordinates are bounded by part (i). (iii) Differentiate the update with respect to $u_i^t$; the coefficient is $(\gamma_p d_{t-1} + 1)^{-1}$. Using $d_{t-1} \to \frac{1}{1-\gamma_p}$ yields the limit $1 - \gamma_p$. □

**Implications:** Taken together, these results isolate why past discounting is preferable for long-horizon learning. With additive *perfect recall* (Theorem 5), the augmented fairness state necessarily grows with the horizon; any fixed-resolution representation explodes linearly in $t$ per coordinate (and exponentially in $n$). With *averaged perfect recall* (Theorem 7), the values themselves stay bounded, but Markovization forces carrying an unbounded auxiliary variable (the time index or equivalent denominator), so the augmentation still scales with the horizon. In contrast, *past-discounted* schemes yield horizon-independent summaries: additive discounting gives a tight uniform bound on $Z_i^t$ (Theorem 6) and the number of additional states introduced by a fixed discretization is independent of the time horizon (Corollary 3). Discounted averaging also provides a bounded sufficient statistic $(\mathbf{Z}^t, d_t)$ with non-vanishing responsiveness of the current step (Theorem 8). Practically, this means that even in long-horizon settings, the augmented state space remains fixed in size, enabling efficient learning and planning. Moreover, the non-vanishing responsiveness ensures that recent allocations meaningfully influence fairness evaluations, preventing the system from becoming overly rigid due to distant past decisions.

### 9.4.3 Half-life and Effective Memory

With past discounting, we also want to decide how much of the past to remember. The discount factor $\gamma_p$ controls this trade-off: values close to 1 retain more history, while values near 0 emphasize recent allocations. However, we still need a systematic way to choose $\gamma_p$ based on the desired memory characteristics.

We introduce the term *half-life* to quantify how quickly past outcomes lose influence under past discounting. Under the additive update $Z_i^t = \gamma_p Z_i^{t-1} + u_i^t$, the contribution of a utility observed $k$ steps ago is weighted by $\gamma_p^k$ at time $t$. The *half-life* $t_{1/2}$ is the number of steps after which this weight falls to one half of its initial value, i.e., it is defined by $\gamma_p^{t_{1/2}} = \frac{1}{2}$. This provides an interpretable way to choose $\gamma_p$ based on how long past allocations should meaningfully affect current fairness.

**Definition 8** (Half-life)**.** *Under the additive past-discounted update $Z_i^t = \gamma_p Z_i^{t-1} + u_i^t$ with $\gamma_p \in (0,1)$, the* half-life *$t_{1/2}$ is the number of steps after which the weight on a past contribution is halved, i.e.*

$$\gamma_p^{t_{1/2}} = \tfrac{1}{2} \qquad \Longleftrightarrow \qquad t_{1/2} = \frac{\ln(1/2)}{\ln(\gamma_p)}. \tag{9.37}$$

**Effective window (how much history is retained in total).** The geometric weights $(1, \gamma_p, \gamma_p^2, \ldots)$ have total mass

$$\sum_{k=0}^{\infty} \gamma_p^k = \frac{1}{1-\gamma_p}. \tag{9.38}$$

It is convenient to call

$$W_{\text{eff}} := \frac{1}{1-\gamma_p} \tag{9.39}$$

the *effective window length.* Think of it as: "discounting with $\gamma_p$ behaves, in total weight, like remembering roughly $W_{\text{eff}}$ recent steps equally." Two immediate consequences:

$$\gamma_p = 1 - \frac{1}{W_{\text{eff}}} \qquad \text{and} \qquad \frac{\partial Z_i^t}{\partial u_i^t} = 1 - \gamma_p = \frac{1}{W_{\text{eff}}}. \tag{9.40}$$

So $W_{\text{eff}}$ directly tells you the *instantaneous weight* on the current step (larger window $\Rightarrow$ smaller immediate weight, more smoothing).

**"How much of the last $m$ steps should matter?"** If you want at least a fraction $1-\varepsilon$ of the total influence to come from the most recent $m$ steps, choose $\gamma_p$ so that

$$\underbrace{\frac{\sum_{k=0}^{m-1}\gamma_p^k}{\sum_{k=0}^{\infty}\gamma_p^k}}_{\text{fraction from last } m} = 1-\gamma_p^m \;\geq\; 1-\varepsilon \quad\Longleftrightarrow\quad \gamma_p \;\leq\; \varepsilon^{1/m}. \tag{9.41}$$

Equivalently, for a fixed $\gamma_p$, the last $m$ steps account for $1-\gamma_p^m$ of the total weight.

Table 9.1 gives half-life and effective window for common $\gamma_p$.

Table 9.1: Half-life and effective window for selected $\gamma_p$.

| $\gamma_p$ | $t_{1/2} = \ln(1/2)/\ln(\gamma_p)$ | $W_{\text{eff}} \approx 1/(1-\gamma_p)$ |
|---|---|---|
| 0.80 | 3.11 | 5 |
| 0.90 | 6.58 | 10 |
| 0.95 | 13.52 | 20 |
| 0.97 | 22.77 | 33.3 |
| 0.99 | 68.97 | 100 |

### 9.4.4 Beyond Scalar Discounting: Linear Fairness Memories

We also connect our results to the well-established notion of system stability in control theory. While we show past discounts as one surefire method of keeping the augmented state-space bounded, other mechanisms may exist that achieve similar outcomes. We propose this extension by modeling the fairness memory as a Linear Time-Invariant (LTI) system. This powerful and well-established framework allows us to derive a universal condition for tractable memory updates. We represent the evolution of the n-agent memory vector $\mathbf{Z}^t \in \mathbb{R}^n$ with the following state-space equation:

$$\mathbf{Z}^{t+1} = A\mathbf{Z}^t + \mathbf{u}^{t+1} \tag{9.42}$$

Here, $\mathbf{u}^{t+1}$ is the vector of utilities received at the current step, and the matrix $A \in \mathbb{R}^{n\times n}$ is the **memory transition matrix**. This matrix $A$ encodes how the entire memory vector from the previous step is transformed and carried forward, generalizing additive utility aggregation. The central question becomes: for which matrices $A$ will the memory $\mathbf{Z}^t$ remain bounded over arbitrarily long horizons?

The answer is provided by a cornerstone result from control theory, which we state here for completeness.

**Theorem 9** (LTI System Stability)**.** *Consider the update $\mathbf{Z}^{t+1} = A\mathbf{Z}^t + \mathbf{u}^{t+1}$ with $A \in \mathbb{R}^{n\times n}$ and $\|\mathbf{u}^t\|_\infty \leq u_{\max}$ for all $t$. The state vector $\mathbf{Z}^t$ is uniformly bounded for all bounded input sequences $\{\mathbf{u}^t\}$ if and only if the spectral radius of $A$ is less than one, i.e., $\rho(A) < 1$.*

*Proof.* This is a classic result from linear systems theory establishing the condition for Bounded-Input, Bounded-State (BIBS) stability. For a formal proof, see, e.g., [113]. □

**Implications for Temporal Fairness.** This general theorem provides a powerful lens through which to view our previous findings, grounded in fundamental principles of system stability.

- **Perfect Recall**, where $A = I$ (the identity matrix), has a spectral radius of $\rho(I) = 1$. The theorem correctly predicts that this system is unstable and its memory will grow unboundedly, formalizing the core problem we identified in Theorem 5.
- **Additive Past-Discounting**, where $A = \gamma_p I$, has a spectral radius of $\rho(\gamma_p I) = |\gamma_p|$. As long as $\gamma_p < 1$, the stability condition is met, guaranteeing a bounded memory. This shows that our central result in Theorem 6 is also supported by this general stability principle, lending further credence to the use of past-discounts.

This reveals that any attempt to incorporate long-term memory into fairness must satisfy the condition $\rho(A) < 1$. More generally, the connection of additive fairness memories to Theorem 9 outlines a design rule for richer memory paradigms that allow cross-agent mixing or modeling memory across multiple time scales as long as $A$ is Schur-stable. This provides a rigorous foundation for extending our work. While our method is demonstrably effective, this framework offers a clear path to design more sophisticated and nuanced fairness memories while retaining tractability.

## 9.5 Experimental Results

To empirically validate our theoretical findings, we design an experiment to demonstrate the computational necessity of past-discounting for learning fair policies in long-horizon settings. Our central hypothesis is that with longer horizons a reinforcement learning (RL) agent optimizing for long-term fairness will fail to learn effectively under perfect recall due to the state-space explosion predicted by our theory, while an agent using past-discounting will learn successfully and tractably.

### 9.5.1 Experimental Setup

**Environment.** We use a dynamic resource allocation environment with $n = 10$ agents over an episode of length $T$. At each timestep, $n_r = 2$ indivisible resources become available. Each agent $i$ has an immediate utility for the resources, represented by a "needs" vector, drawn from $\mathcal{U}(0,1)$. If allocated a resource, the agent's immediate utility is equal to this value. To create a challenging fairness scenario, we designate two "advantaged agents" whose needs are consistently drawn from a higher distribution, $\mathcal{U}(0.8, 1.0)$. This creates a natural tension where a myopic, efficiency-maximizing policy would perpetually favor the advantaged agents, leading to severe long-term inequality.

**Learning Problem.** We frame the task as an RL problem where a central allocator must learn a policy $\pi(s_t) \to a_t$.

- **State ($s_t$):** The state consists of the current needs vector and the fairness memory vector from the previous step, $\mathbf{Z}^{t-1}$.
- **Action ($a_t$):** The action is a discrete choice from the set of all $\binom{n}{n_r}$ possible pairs of agents to receive the resources.
- **Reward ($r_t$):** The reward signal is a weighted sum of an efficiency component $U_t$ and a potential-based fairness component $F_t$: $r_t = (1 - \lambda) \cdot U_t + \lambda \cdot F_t$.
  In our experiments, we set $\lambda = 0.9$, as our main objective is to learn to improve fairness, with utility acting as a secondary shaping reward. The efficiency term $U_t = \sum_i u_i^t$ is the total utility allocated at the current step. The fairness term $F_t = W(\mathbf{Z}^t) - W(\mathbf{Z}^{t-1})$

is the change in the social welfare function, which directly rewards actions that improve the fairness state.

The agent's goal is to maximize the cumulative reward. We conduct experiments using two distinct welfare functions, $W(\cdot)$: **Egalitarian Welfare** and **Nash Welfare**.

**Models.** We train a Proximal Policy Optimization (PPO) agent [130] from the Stable Baselines3 library [122] for 1 million total timesteps under three different memory configurations, corresponding directly to the paradigms discussed in our paper:

1. **Perfect-Recall ($\gamma_p = 1.0$):** The agent observes the true cumulative utility, $\mathbf{Z}^t = \sum_{\tau=0}^{t} \mathbf{u}^\tau$. As predicted by Theorem 5, this leads to an unbounded state space.
2. **Past-Discounted ($\gamma_p \in \{0.9, 0.99, 0.999\}$):** The agent observes the discounted utility, $\mathbf{Z}^t = \gamma_p \mathbf{Z}^{t-1} + \mathbf{u}^t$. Following Theorem 6, this ensures the state space remains bounded.
3. **Myopic ($\gamma_p = 0.0$):** The agent only observes immediate needs, lacking any historical context to correct for long-term imbalances.

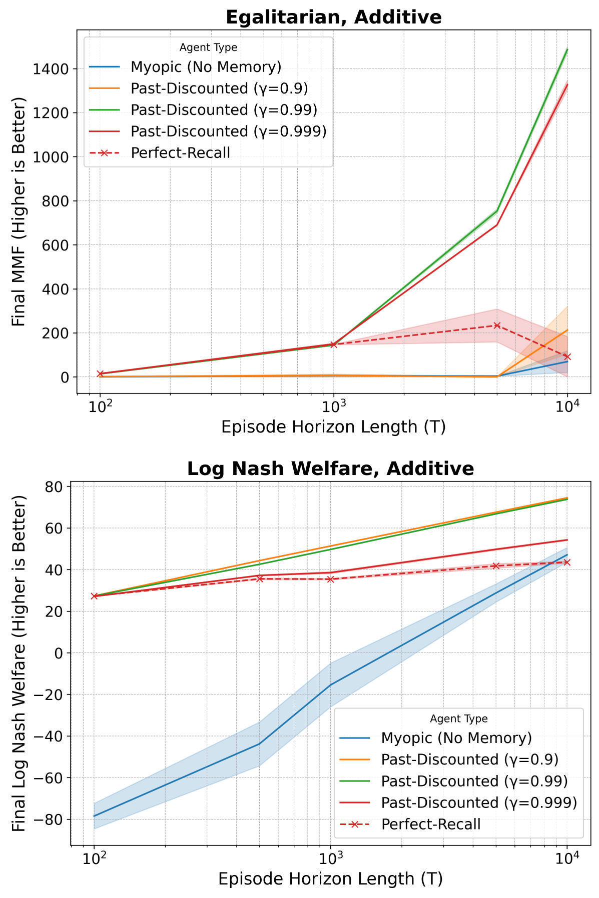
We conduct our experiments across different horizons ranging from 100 steps per episode to 10000 steps per episode. Each configuration was run 10 times for statistical confidence.

### 9.5.2 Results for Additive Utility Aggregation

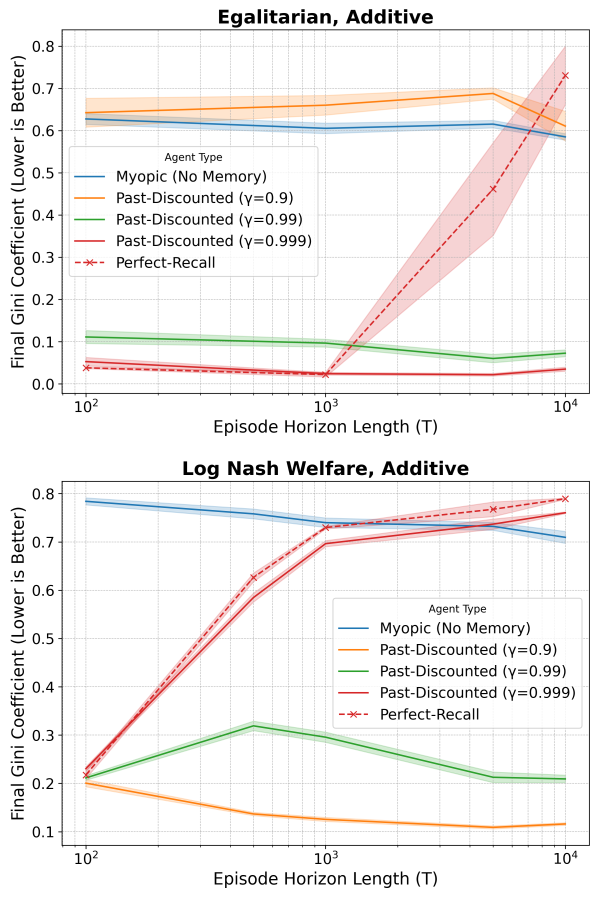
We compare the converged performance of each method after training for 1 million steps, computing metrics by averaging over the last 10% of the episodes. We additionally evaluate the performance of each agent using the Gini coefficient, a standard measure of inequality where 0 represents perfect equality. The results are shown in Figures 9.2a and 9.2b.

The **Myopic** agent performs poorly across all scenarios, confirming that memory is essential for learning fair behavior. By only observing immediate needs, it cannot correct for the accumulating advantage of certain agents, resulting in high inequality.

The **Perfect-Recall** agent demonstrates the critical failure mode predicted by Theorem 5. While effective at short horizons ($T \leq 500$), its performance collapses as the episode length



(a) Final converged welfare function value.



(b) Gini coefficient of the converged distribution.

Figure 9.2: Comparison of welfare and inequality with additive aggregation as episode length increases. Perfect Recall sees degradation with longer horizons.

grows. At $T = 10,000$, its ability to maintain fairness is no better than the myopic agent in both experiments. This failure is a direct consequence of its unbounded state space; as the cumulative utilities in $\mathbf{Z}^t$ grow, it makes learning harder, preventing the PPO algorithm from converging to a stable and effective policy.

On the other hand, **Past-Discounted** agents are able to learn regardless of the horizon. Benefiting from a stable, bounded state representation, they can learn a robust policy that maintains a low Gini coefficient even over this extended horizon.

Diving deeper into the Past-Discounted results reveals how the choice of the discount factor, $\gamma_p$, is critical for performance. The agent with $\gamma_p = 0.999$, which has an effective memory window of 1000 steps, closely mimics the behavior of the Perfect-Recall agent on horizons up to that length. Beyond a 1000-step horizon, however, the methods diverge: the Perfect-Recall agent's performance collapses, while the Past-Discounted agent's performance improves.

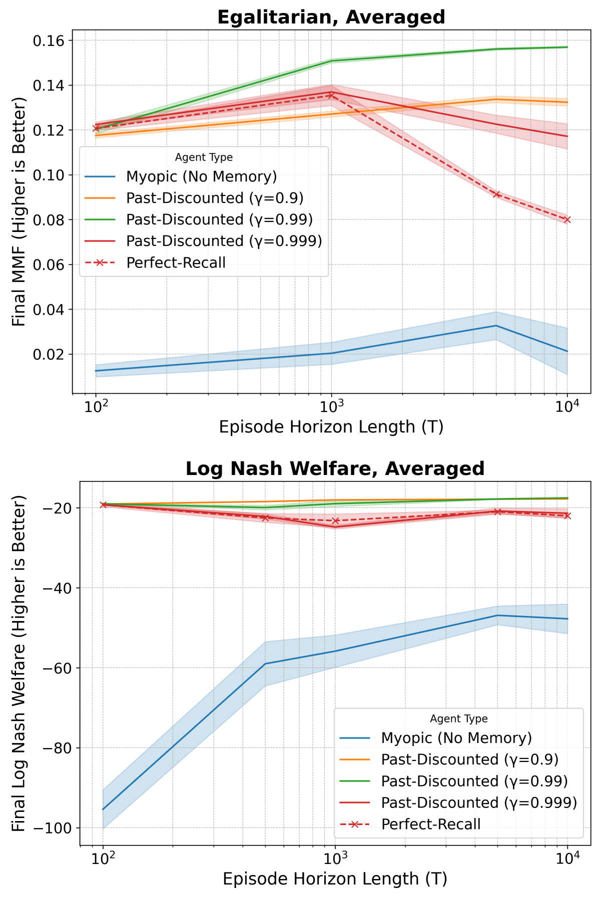
Furthermore, our results show that the optimal choice of $\gamma_p$ depends on the specific fairness objective. With the dense signal from **Nash Welfare**, a shorter memory ($\gamma_p = 0.9, W_{eff} = 10$) is sufficient for the agent to learn a fair policy. However, with the sparser signal from **Egalitarian Welfare**—where the reward only changes when the identity of the worst-off agent is affected—a longer memory ($\gamma_p = 0.99, W_{eff} = 100$) is required to learn effectively.

This highlights a key insight for practitioners: selecting an appropriate discount factor involves balancing the complexity of the environment with the nature of the fairness signal. Denser signals may allow for shorter effective memories, while sparser signals necessitate longer ones (higher $\gamma_p$) to ensure the agent accumulates enough information to act fairly. In our experiments, a $\gamma_p$ of 0.99 proved to be a robust choice across both welfare functions.

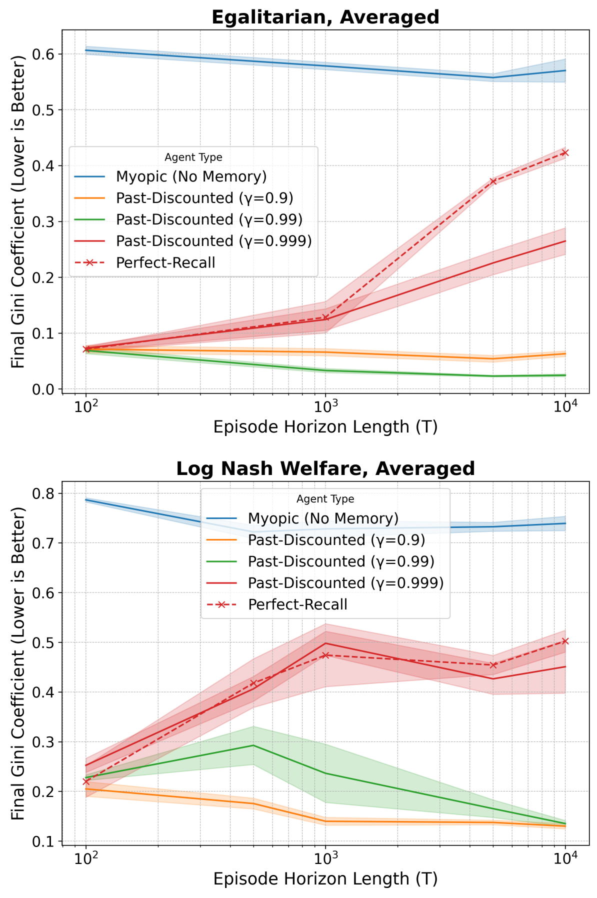
### 9.5.3 Results for Averaged Utility Aggregation

To ensure our findings are robust, we conducted a parallel set of experiments using an **averaged utility aggregation** model instead of an additive one. In this configuration, the agent's fairness memory, $\mathbf{Z}^t$, represents the time-averaged utility for each agent. The core experimental setup, including the environment with advantaged agents and the PPO learning algorithm, remained identical to the additive case. The results, shown in Figures 9.3a and 9.3b, confirm that the fundamental insights from our main analysis hold true for averaged utilities as well.

The **Myopic** agent, lacking any historical context, again fails to produce equitable outcomes, serving as a baseline for high inequality. The **Perfect-Recall** agent, which maintains a perfect running average of all past utilities, initially performs well on short horizons. However, as predicted by our theoretical results, this model requires an unbounded state augmentation to track the time index $t$. We observe a similar performance collapse as the episode length grows, with the agent's ability to maintain fairness degrading significantly on horizons of 5,000 steps or more. This confirms that even when utility values are bounded by averaging, the need for an ever-growing time counter makes learning intractable.



(a) Final converged welfare function values.



(b) Gini coefficients of converged distributions.

Figure 9.3: Comparison of welfare and inequality with averaged aggregation as episode length increases. Perfect Recall sees degradation with longer horizons.

In contrast, the Past-Discounted agents using the averaged update demonstrate robust and scalable performance. By using a bounded, horizon-independent state augmentation $(\mathbf{Z}^t, d_t)$, these agents consistently learn fair policies across all tested horizons. Note that in the plot for Nash Welfare (Figure 9.3a (bottom), it appears Perfect Recall is similar to the best methods because of the scale, as the Myopic results are extremely bad. The performance does get meaningfully worse, as can be seen in the Gini plots. The values are all negative because the sum of logs of values less than 1 (as is the case with averaged $z_i$) is always negative.

With averaged aggregation, a new mechanism also enters the play: vanishing responsiveness. Without past-discounts, it is difficult for the Perfect Recall method to distinguish between decisions made at early time-steps versus later in the episode. Because averaging divides by the history size, the change in $\mathbf{Z}^t$ when $t$ is large becomes vanishingly small. Thus,

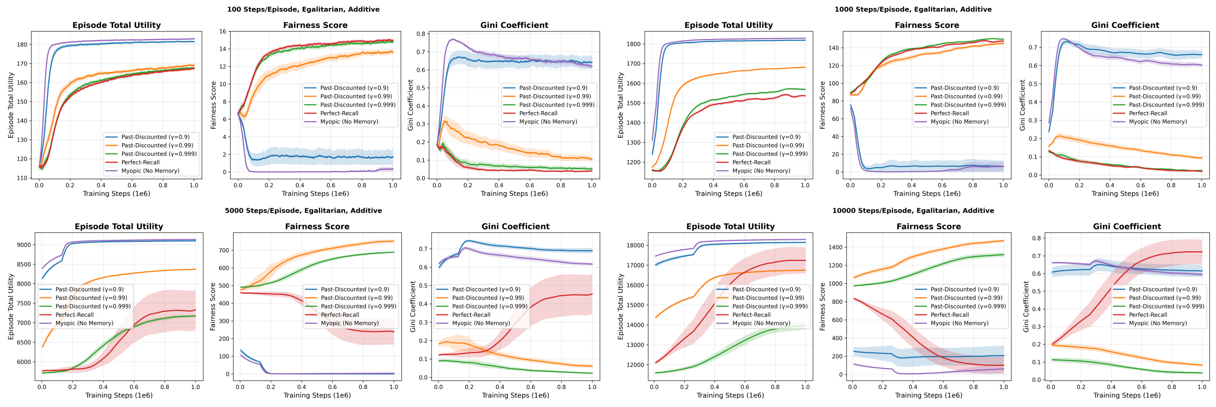


Figure 9.4: **Additive Egalitarian Welfare:** Training curves across different episode lengths.

with long horizons, the Perfect Recall method receives only tiny reward signals from the fairness component for a majority of the training steps. This effect is countered by using past discounting, where the effective time window makes sure transitions later on in the episode also get non-vanishing responsiveness.

These results provide compelling empirical validation for our central thesis. While perfect recall is theoretically appealing, its unbounded memory requirement makes it computationally intractable for learning fair policies over long horizons. Past-discounting is not merely a heuristic but a practical necessity, providing a bounded and stationary state representation that enables stable learning and scalable long-term fairness.

## 9.5.4 Detailed Training Curves

We also provide detailed training curves for all experimental configurations discussed above. Each figure corresponds to a specific combination of welfare function (Egalitarian or Nash) and utility aggregation method (Additive or Averaged). Within each figure, we present results across various episode lengths to illustrate how each memory model performs as the time horizon increases.

Each plot shows three key metrics over the course of 1 million training steps:

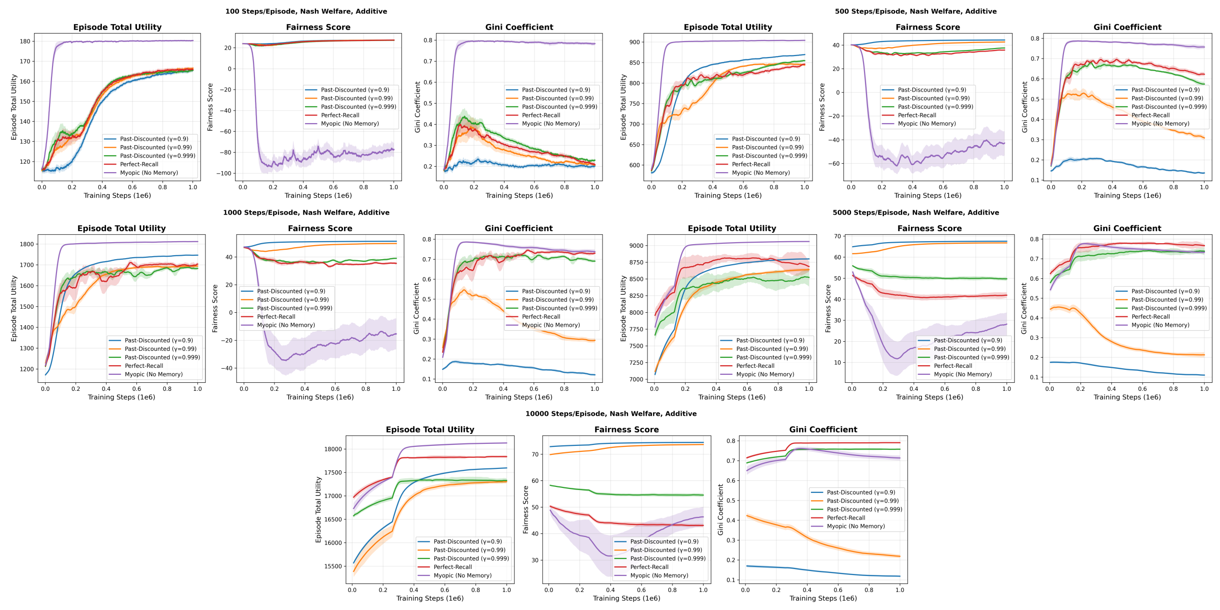


Figure 9.5: **Additive Nash Welfare:** Training curves across different episode lengths.

- **Total Utility:** The sum of immediate utilities allocated at each step, averaged over the episode.
- **Fairness Score:** The value of the specific welfare function being optimized.
- **Gini Coefficient:** A measure of inequality in the final distribution of cumulative utilities, where 0 represents perfect equality.

The shaded regions in the plots represent the standard error across 10 independent runs. For the averaged Egalitarian experiments (Figure 9.6), we observed that all memory models struggled to learn a stable policy with the fairness weight $\lambda = 0.9$. To enable effective learning, we increased the fairness weight to $\lambda = 0.999$ for these runs, which provided a stronger, more consistent signal for the agent. All other experiments were conducted with $\lambda = 0.9$.

These are the general trends to observe in all these experiments:

- As the horizon is increased, the perfect recall method starts exhibiting worse performance in terms of fairness.

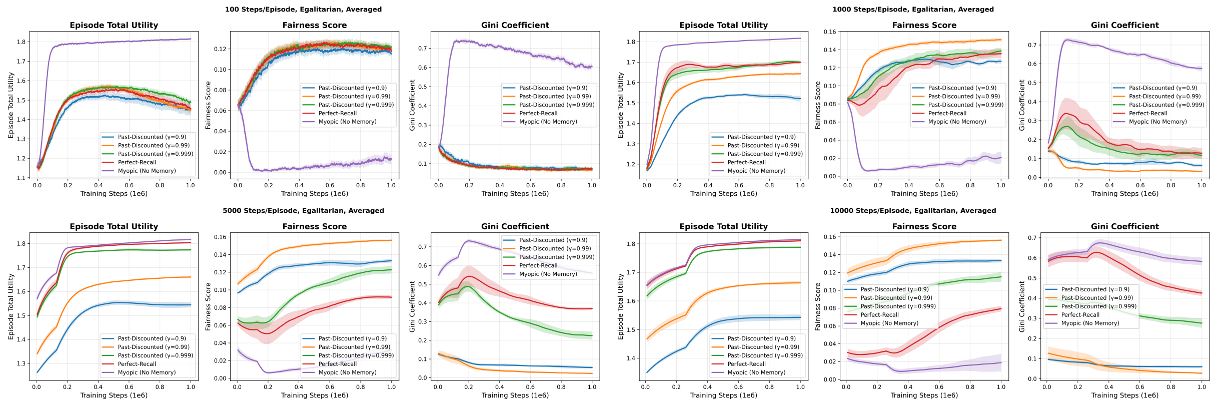


Figure 9.6: **Averaged Egalitarian Welfare:** Training curves across different episode lengths (run with $\lambda = 0.999$).

- The myopic policy converges to a utility of 1.8 per time step, indicating that it consistently picks the advantaged agents whose average need would be 0.9.
- The effective window's effect is visible based on how closely past-discounted experiments track perfect recall. We expect each method to be close to perfect recall for horizons of similar or smaller scale to the effective time window of the selected $\gamma_p$.
- In some cases, it looks like the poorly performing methods are catching up as the steps approach $1e-6$. This further adds to the strengths of the past-discounting approach: the increased complexity of the state-space is making the problem much harder to learn from, making sample complexity worse. So even if Perfect Recall should be able to learn given sufficient time, this is very long compared to the steps needed for past-discounted agents, and gets longer as the horizon increases.

## 9.6 Conclusion

In this work, we introduced a framework for incorporating past-discounted historical utilities into dynamic resource allocation—a strategy inspired by behavioral economics and moral psychology, which show that humans naturally discount the impact of distant past events [149, 45]. By applying a discount factor $\gamma_p$ to past utilities, our method enables decision-makers to reason over accumulated utilities while effectively balancing short-term and long-term fairness considerations.

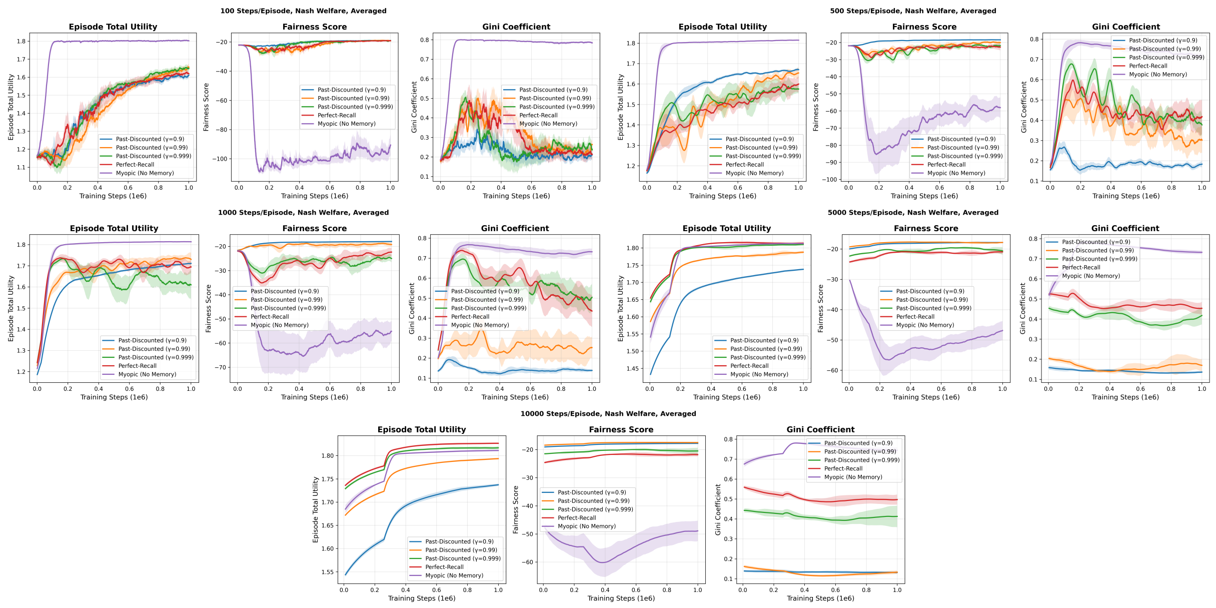


Figure 9.7: **Averaged Nash Welfare:** Training curves across different episode lengths.

Crucially, the past-discounting approach ensures that the augmented state space remains bounded. In contrast to traditional non-discounted fairness methods—where the state space grows linearly with the time horizon and thus becomes computationally intractable—our framework yields a joint state space whose size is independent of time. This boundedness not only improves the sample complexity and convergence of reinforcement learning algorithms in multi-agent settings, but also provides a principled mechanism to manage the trade-off between immediate outcomes and historical context.

With this core mechanism in place, the next chapter dives into algorthms that enable learning the long-term fairness effects of current actions for DECA problems.

# Chapter 10

# Learning Fairness in Multi-Agent Resource Allocation

## 10.1 Introduction & Contribution

In this chapter, we present DECAF, a general framework for learning fair policies in multi-agent resource allocation problems under the Distributed Evaluation, Centralized Allocation (DECA) paradigm. The methods developed previously in this dissertation focus on myopic fairness, which is useful in settings with no access to centralized training or where utilities are black-box. Acting as a bridge between the field of fair multi-agent reinforcement learning and the resource allocation problems discussed so far in this dissertation, this chapter introduces novel methods to learn Q-functions that balance fairness and efficiency in DECA settings, in addition to a suite of environments that mirror popular fairness-aware multi-agent RL benchmarks.

To incorporate fairness in the learning process, we enable agents to learn long-term fairness effects of their actions, combining this with utility estimates to compute allocations. We introduce three strategies:

- **Joint Optimization (JO)**: A scalarized multi-objective learning approach that jointly optimizes for fairness and utility.
- **Split Optimization (SO)**: A method that learns separate fairness and utility estimators, enabling online trade-off adjustments for fairness and utility.
- **Fair-Only Optimization (FO)**: A fairness-focused approach that modifies an existing black-box utility function to incorporate fairness considerations.

Our framework also accommodates a variety of fairness metrics, including variance, $\alpha$-fairness, maximin fairness, and generalized Gini functions (GGFs), without requiring structural changes. DECAF also handles a key limitation in our previous myopic methods (SI and GIFF), namely the inability to learn long-term fairness effects of actions. This is embodied in the JobAlloc and Job environments, requiring agents to learn to undertake locally suboptimal action to achieve better long-term fairness.

**Contributions:** The main contributions of this chapter are:

1. We introduce DECAF, a general framework for learning fair policies in multi-agent resource allocation problems under the DECA paradigm.
2. We propose three novel algorithms (JO, SO, FO) that enable agents to learn Q-functions balancing fairness and efficiency, accommodating various fairness metrics.
3. We develop a suite of environments that mirror popular fairness-aware multi-agent RL benchmarks, demonstrating the effectiveness of our methods in achieving superior fairness-utility trade-offs compared to baselines.

## 10.2 Background

### 10.2.1 Related Work

Significant research has addressed algorithmic bias, where ML models, such as those used in hiring decisions [123], can exhibit harmful biases. We refer readers to an extensive survey by Mehrabi et al. [102] for a review of recent work. These studies typically focus on debiasing the outputs of predictive models to meet fairness criteria such as equalized odds [58] or demographic parity [35]. However, our work diverges from this approach. Instead of correcting biases in predictions, we aim to develop algorithms that inherently promote fair decision-making via the actions they optimize.

In this work, our focus is on learning fair policies in multi-agent RL with resource constraints. Prior work has explored fairness in RL broadly [47], but a few methods are especially relevant.

FEN [66] uses a hierarchical network to optimize the coefficient of variation, learning when agents should act greedily or fairly. However, it lacks resource constraints and requires inter-agent communication. Other works model fairness as multi-objective optimization, treating each agent's utility as a separate objective and optimizing a social welfare function [176, 138]. These require learning over the full joint action space [138] or use decentralized policy gradients [176], which limits applicability under global constraints.

The DECA approach allows us to consider global constraints while allocating resources, opening up the scope for better global solutions, which none of the prior approaches allow. The distributed evaluation allows each agent to only learn a local value function, which reduces the complexity when compared to learning a joint policy. Further, our Split and Fair-Only approaches allow changing the trade-offs between utility and fairness post-training, which provides additional flexibility that previous approaches lack.

### 10.2.2 DECA Problems

In the context of Distributed Evaluation, Centralized Allocation (DECA), our primary goal is to integrate fairness into the decision-making process of resource allocation in multi-agent systems. Formally, we seek to maximize a combined measure of system utility and fairness, represented as:

$$\max\,(1-\beta)\mathcal{U}_T + \beta\mathcal{F}_T \tag{10.1}$$

where $\mathcal{U}_T$ denotes the total utility at time $T$ and $\mathcal{F}_T$ represents the fairness measure, weighted by $\beta$.

We use the DECA optimization framework, as defined in Section 2 (Eq. 2.4-2.6), to model the resource allocation problem.

We model agents as Q-learners without strategic behavior, that learn to predict the utility of their actions in the presence of the central allocator. The central allocator uses these predictions to solve an integer linear program (ILP) that maximizes the total predicted utility while satisfying resource constraints.

## 10.3 DECAF: Fairness in DECAs

In this section, we describe DECAF, our framework for incorporating fairness into DECA problems using Q-Learning. Specifically, we detail how we can specify and learn the fairness objective in Eq. 10.1 through the use of decomposed fairness rewards, and a modified Q-Learning algorithm to learn from these rewards using centralized training.

### 10.3.1 Fairness Reward

Previous work has considered learning to optimize a single social welfare function (SWF) that captures the notions of fairness and utility together, like the Coefficient of Variation used by FEN [66] or $\alpha$-fairness and the Generalized Gini Function (GGF) used by SOTO [176]. With DECAF, we try to learn a class of objectives that trade off between system utility and fairness, characterized by a trade-off variable $\beta$ (Eq. 10.1), with the aim of enabling flexible trade-offs between the two.

Let $\mathbf{Z} = \{z_i\}_{i \in \alpha}$ denote the vector of accumulated agent utilities (averaged or total). We interpret this utility as 'accumulated wealth,' and look to make allocations that can result in a fairer distribution of this wealth. Let $\mathbf{Z}_t^\pi$ represent the distribution of agent utility metrics at time $t$ following policy $\pi$. Then, we define a fairness function $\mathbb{F} : \mathbb{R}^n \to \mathbb{R}$ as a mapping of vector $\mathbf{Z}$ to a real value, and our fairness objective is maximizing $\mathcal{F}_T = \mathbb{F}(\mathbf{Z}_{t=T}^\pi)$. Most popular SWFs and fairness functions can be cast into this form.

To realize this objective in DECAF, we compute a fairness reward based on the allocation made at each timestep. Let $\mathcal{A}^t = \{\mathcal{A}_i^t\}_{i \in \alpha}$ denote the action allocation at time $t$, where each $\mathcal{A}_i^t$ is the action assigned to agent $i$ as a result of solving the ILP. The fairness reward $R_f(\mathbf{s}_t, \mathcal{A}^t)$ is a vector-valued signal that reflects how the selected allocation impacts system fairness. Specifically, it is computed as the per-step change in the fairness function:

$$\begin{aligned} \Delta\mathcal{F}|\mathcal{A}^t &= \mathcal{F}_{t+1} - \mathcal{F}_t && (10.2) \\ &= \mathbb{F}(\mathbf{Z}_{t+1}) - \mathbb{F}(\mathbf{Z}_t) && (10.3) \end{aligned}$$

A naive way to decompose this reward is to evenly divide it among agents. This is commonly done in collaborative multi-agent RL when agents are optimizing a shared goal.

$$R_f(\mathbf{s}_t, \mathcal{A}^t) = \left[\frac{\Delta\mathcal{F}|\mathcal{A}^t}{n}\right]_{i\in\alpha} \tag{10.4}$$

Alternatively, specialized decompositions can be designed to give a more informative signal to each agent. For example, if variance is used as the fairness function ($\mathcal{F}_t = -\text{var}(\mathbf{Z}_t)$):

$$\begin{aligned} \Delta\mathcal{F}|\mathcal{A}^t &= -\text{var}(\mathbf{Z}_{t+1}) + \text{var}(\mathbf{Z}_t) && (10.5)\\ &= -\frac{1}{n}\sum_{i\in\alpha}\left(z_i^{t+1} - \bar{z}_{t+1}\right)^2 + \frac{1}{n}\sum_{i\in\alpha}\left(z_i^t - \bar{z}_t\right)^2 && (10.6)\\ R_f(\mathbf{s}, \mathcal{A}) &= \left[-\frac{1}{n}\left(z_i^{t+1} - \bar{z}_{t+1}\right)^2 + \frac{1}{n}\left(z_i^t - \bar{z}_t\right)^2\right]_{i\in\alpha} && (10.7) \end{aligned}$$

Observe that this reward only depends on the agent's own metric value and the average metric. Thus, each iteration, apart from the local observation, each agent only needs to be communicated information about the average utility of all agents to reliably predict this value. This could be done by the central agent, or by message passing between the agents.

For the main experiments of this paper, we use variance as our fairness function, with the reward function in Eq. 10.7 as the fair reward. However, our methods are not limited to using variance. We also provide reward decompositions and experiments with other fairness functions including $\alpha$-fair, GGF, and maximin functions, showing the generality of our approach.

### 10.3.2 DECAF Problem Formulation

With the fairness machinery in place, we define the DECAF problem $\mathcal{D}$ as the tuple

$$\mathcal{D} = \left\langle \alpha,\, S^f,\, \mathcal{O},\, \{A_i\}_{i\in\alpha},\, T,\, R_u,\, \gamma,\, c,\, \mathbb{F},\, \beta,\, R_f \right\rangle, \tag{10.8}$$

where $\langle \alpha, \mathcal{O}, \{A_i\}, T, R_u, \gamma, c\rangle$ are as in DECA, $S^f$ is the fairness augmented state space including information from the accumulated utility vector $\mathbf{Z}_t = \{z_i^t\}_{i\in\alpha}$ (e.g. the current average utility or the entire vector $\mathbf{Z}$), $\mathbb{F} : \mathbb{R}^n \to \mathbb{R}$ is a fairness function evaluated over the accumulated utility vector $\mathbf{Z}_t = \{z_i^t\}_{i\in\alpha}$, $\beta \in [0, 1]$ is a trade-off parameter, and $R_f$ :

$S^f \times \prod_i A_i \to \mathbb{R}^n$ is a decomposed fairness reward satisfying

$$\sum_{i\in\alpha} \left[R_f(\mathbf{s}_t, \mathcal{A}^t)\right]_i \;=\; \mathbb{F}(\mathbf{Z}_{t+1}) - \mathbb{F}(\mathbf{Z}_t) \;\triangleq\; \Delta\mathcal{F}\big|\mathcal{A}^t. \tag{10.9}$$

At each time $t$, agents locally predict per-action scores $Q(o_i, a)$ that combine utility and fairness estimates. The centralized allocator then solves the fairness-aware ILP

$$\begin{aligned}
\max_{x_i(a)\in\{0,1\}} \quad & \sum_{i\in\alpha}\sum_{a\in A_i} x_i(a)\, Q(o_i, a) && (10.10)\\
\text{s.t.} \quad & \sum_{a\in A_i} x_i(a) = 1, \qquad \forall i \in \alpha, && (10.11)\\
& \sum_{i\in\alpha}\sum_{a\in A_i} x_i(a)\, c(a)_k \;\le\; \mathcal{R}_k, \qquad \forall k \in \{1, \ldots, K\}, && (10.12)
\end{aligned}$$

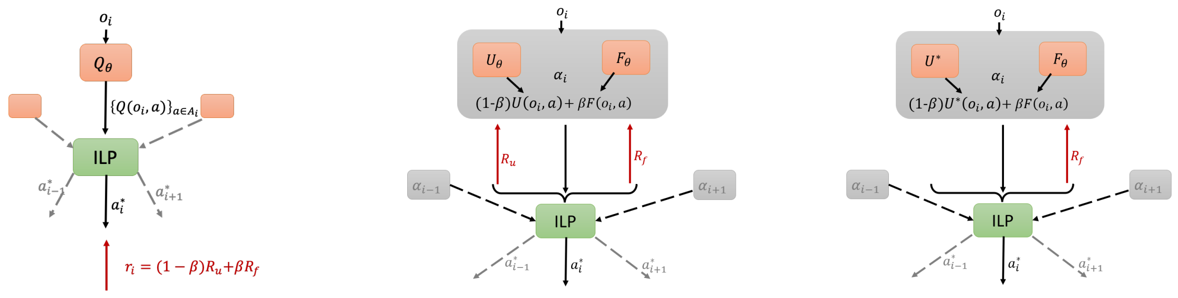
where $\mathcal{R} \in \mathbb{R}^K$ is the available resource vector and $c(a) \in \mathbb{R}^K$ is the per-action consumption. As in DECA, actions include a null action to ensure feasibility; only resource-consuming actions are constrained by (10.12).

$Q(o_i, a)$ is the key workhorse in DECAF, and understanding how to learn it is the focus of the next section. We present three approaches to learn $Q_f$ that differ in how they estimate and combine utility and fairness. All three approaches use the same ILP formulation above to compute allocations, differing only in how $Q_f$ is learned.

### 10.3.3 Algorithms

Given the fair reward $R_f$ described above, our approach targets the DE step to improve fairness, by changing the Q-values used in the ILP (Eq. 10.10) to also account for fairness. We do this modifying $Q$ to be an estimator of the combined fair-efficient objective, with a weight $\beta \in [0, 1]$ used to regulate relative value of fairness and utility.

We use experience replay with centralized training to learn the Q-function, where an experience $\tau = \langle \mathbf{o}, \mathcal{A}, \mathbf{r}_u, \mathbf{r}_f, \mathbf{o}' \rangle$ stores a joint transition across all agents, with utility rewards $\mathbf{r}_u$ and fair rewards $\mathbf{r}_f$. Let $\theta$ denote the parameters of the Q-function. Given a replay buffer $\mathcal{D}$, we want to minimize the loss function $J_\theta = \mathbb{E}_{\tau\sim\mathcal{D}} L(\delta(\tau))$, where $\delta(\tau)$ is the Bellman error



(a) Joint Optimization (JO) (b) Split Optimization (SO) (c) Fair-Only (FO)

Figure 10.1: Illustration of our three DECAF methods to learn fairness. Each subfigure shows how the values propagate for a single agent. The red lines and text denote the actual reward to the learning model, which is used to update weights using TD learning. (a) Joint Optimization learns to predict a single combined value. (b) Split Optimization learns two separate estimators for utility and fairness, and combines their output. (c) Fair-Only assumes a black-box utility model $U^*$, and learns a fairness estimator only, combining their outputs to make decisions.

of the transition $\tau$, and $L$ is the MSE loss. We propose three approaches for integrating fairness, illustrated in Figure 10.1.[10]

- **Joint Optimization (JO):** A single estimator optimizes a weighted combination of fairness and utility.

$$\delta(\tau) = (1-\beta)\mathbf{r}_u + \beta\mathbf{r}_f + \gamma Q_\theta(\mathbf{o}') - Q_\theta(\mathbf{o}, \mathcal{A}) \tag{10.13}$$

- **Split Optimization (SO):** Separate estimators for fairness ($F_\theta(\cdot)$) and utility ($U_\theta(\cdot)$) allow dynamic adjustment of their trade-off during policy execution.

$$\delta^f(\tau) = \mathbf{r}_f + \gamma F_\theta(\mathbf{o}') - F_\theta(\mathbf{o}, \mathcal{A}) \tag{10.14}$$

$$\delta^u(\tau) = \mathbf{r}_u + \gamma U_\theta(\mathbf{o}') - U_\theta(\mathbf{o}, \mathcal{A}) \tag{10.15}$$

$$Q(\mathbf{o}, \mathcal{A}) = (1-\beta)U_\theta(\mathbf{o}, \mathcal{A}) + \beta F_\theta(\mathbf{o}, \mathcal{A}) \tag{10.16}$$

[10]Unless stated, we use bold terms to denote vectors, and overload Q-functions to also operate on vectors to compute a vector of outputs. Further, we use $Q(\mathbf{o})$ as a shorthand for computing Q-values for all possible actions for each observation in $\mathbf{o}$.

**Algorithm 10.1:** DECAF Algorithm

```
Initialize: agent network $Q_\theta$, target network $Q_{\theta'}$
Initialize: $\epsilon$ (exploration rate)
Initialize: replay buffer $\mathcal{D}$
1  for episode = 1 to $N_{eps}$ do
2      Decay $\epsilon$ according to decay schedule
3      RunEpisode($Q_\theta$, $\epsilon$, $T$, env, $\mathcal{D}$)
4      if episode % k == 0 then
5          RunEpisode($Q_\theta$, 0, $\infty$, env, $\mathcal{D}$)          // Run validation with $\epsilon = 0$
           // Save validation objective
6      if episode % $\tau$ == 0 then
7          $Q_{\theta'} \leftarrow Q_\theta$                                // Update target weights
8  Load model with best validation objective value
9  Run 50 validation episodes using RunEpisode($Q_\theta$, 0, $\infty$, env, $\mathcal{D}$)
10 Save validation results
```

- **Fair-Only Optimization (FO):** A fairness estimator ($F_\theta(\cdot)$) adjusts a pre-existing utility function $U^*(\cdot)$ to incorporate fairness, useful when utility functions are provided externally.

$$\delta^f(\tau) = \mathbf{r}_f(s,a) + \gamma F_\theta(\mathbf{o}') - F_\theta(\mathbf{o}, \mathcal{A}) \tag{10.17}$$

$$Q(\mathbf{o}, A) = (1-\beta)U^*(\mathbf{o}, \mathcal{A}) + \beta F_\theta(\mathbf{o}, \mathcal{A}) \tag{10.18}$$

Our learning algorithm is based on Double Deep Q-Learning [59], which uses a target network to stabilize updates. The key differentiating factor is in how the target values are computed. When learning from an experience, we compute the optimal action $\mathcal{A}^*$ in the successor state by solving the ILP (Eq. 10.10) using the online Q-network, and then compute the Q-value of the selected actions using the target network. The models are updated using the rewards (stored in the experience) obtained after the ILP allocation of the previous state, as shown in the red text and arrows in Figure 10.1. For SO and FO, we package the utility and fairness estimators into the Q-function, and compute the optimal successor action using both. Then, we independently update each estimator using the TD error of their respective objectives. For FO, we skip training the utility estimator. SO and FO offer the additional benefits of interpretability, as during execution, we are able to discern how much of the decision was based on the utility gain and fairness improvement respectively.

**Algorithm 10.2:** RunEpisode (Executes a single episode )

```
Function RunEpisode($Q_\theta$, $\epsilon$, $T$, $env$, $\mathcal{D}$):
    Reset environment, get initial observation $\mathbf{o}_0$
    for $t$ = 1 to $N_{steps}$ do
        Sample a random number $r \in [0, 1]$
        if $r < \epsilon$ then
            $\mathbf{Q}_t = \mathbf{Q}_{\text{random}}$ // Random Q-values for exploration
        else
            $\mathbf{Q}_t = Q_\theta(\mathbf{o}_t)$ // Q-values from the agent
        $\mathbf{a}_t = \text{solve\_ILP}(\mathbf{Q}_t, \text{env.constraints})$ // Use ILP to compute optimal action
        $(\mathbf{R}_{f,t}, \mathbf{R}_{u,t}, \mathbf{o}_{t+1}) = \text{env.step}(\mathbf{a}_t)$ // Take step in environment
        Store transition $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{R}_{u,t}, \mathbf{R}_{f,t}, \mathbf{o}_{t+1})$ in replay buffer $\mathcal{D}$
        if $t\%T == 0$ then
            update($Q_\theta$, $Q'_\theta$, $\mathcal{D}$, $env$)
```

Algorithm 10.1 shows the overall training loop used for our experiments, with Algorithm 10.2 showing how each episode is executed. We decay epsilon to 0.05 over half of the total number of episodes. $T$, $k$, $\tau$ decide how frequently we learn, validate and update the target model respectively. Algorithm 10.3 and Algorithm 10.4 detail how the update step is performed for joint and split models. The update for FO is identical to SO, except omitting the update for the utility model.

## 10.4 Theoretical Results

SO also provides some useful properties described below.

For this section, we use an alternate notation, replacing $\beta$ with $\eta = \frac{\beta}{1-\beta}$ to make the equations easier to read, such that:

$$(1 - \beta)U + \beta F \Leftrightarrow U + \eta F$$

This does not affect the allocation made by the ILP, as it only scales all Q-values by $1/(1-\beta)$. This would only be undefined when $\beta = 1$, but we avoid that condition in our proofs. As

**Algorithm 10.3:** Update for Joint Optimization

```
Function Update($Q_\theta$, $Q_{\theta'}$, $\mathcal{D}$, env):
    Sample a mini-batch of $n$ experiences from replay buffer $\mathcal{D}$
    for each experience $\langle \mathbf{o}, \mathcal{A}, \mathbf{r}_u, \mathbf{r}_f, \mathbf{o}' \rangle$ in the mini-batch do
        Compute Q-values for the successor observation $Q_\theta(\mathbf{o}')$
        Solve ILP to get the optimal allocation $\mathcal{A}^*$ for the next observation $\mathbf{o}'$:
            $\mathcal{A}^* = \text{solve\_ILP}(Q_\theta(\mathbf{o}'), \text{env.constraints})$
        Compute Q-values for $\mathcal{A}^*$ using the target network:
            $Q_{\theta'}(\mathbf{o}', \mathcal{A}^*)$
        Compute the target for the TD update:
            $\text{target} = (1-\beta)\mathbf{r}_u + \beta\mathbf{r}_f + \gamma Q_{\theta'}(\mathbf{o}', \mathcal{A}^*)$
        Compute the TD loss:
            $\text{loss} = (Q_\theta(\mathbf{o}, \mathcal{A}) - \text{target})^2$
        Perform gradient descent on the TD loss to update $Q_\theta$
```

$\beta \to 1, \eta \to \infty$, and for any $\beta' > \beta$, $\eta' > \eta$. Note that in the theorem statements, we replace $\beta$ with $\eta$, but the proofs are equivalent.

The following results hold for any fairness function used in the DECAF formulation.

**Proposition 1.** *As $\eta_{test} \to 0$, all fair-only models behave in a utility-maximizing manner.*

We state this without proof. It is easy to follow how this holds, as at $\eta = 0$, the fairness model does not play any role in the decision making.

**Theorem 1.** *Given perfect estimates for utility and fairness, increasing $\eta$ always improves the one-step fairness gain for SO with $\gamma = 0$.*

*Proof.* We assume that the utility and fairness estimators are converged, i.e., the estimates of fairness and utility are correct. For the following discussion, assume the environment has evolved over some time $t$ and is at a state $s_t$. We consider what changes when we change $\eta$ at this state. Variables used henceforth are conditioned on $s_t$, wherever reasonable. We make the conditioning on $s_t$ implicit and do not notate it, to make it easier to read.

With $\gamma = 0$, the optimal utility and fairness estimates equal the one-step return, i.e. the change in utility and fairness because of the resulting joint action. Note that these values are not known to the agents prior to the allocation as they depend on the joint actions of all agents, so computing these estimates is not trivial.

**Algorithm 10.4:** Update for Split Optimization

```
Function Update($Q_\theta$, $Q_{\theta'}$, $\mathcal{D}$, env):
    Sample a mini-batch of $n$ experiences from replay buffer $\mathcal{D}$
    Unpack $Q_\theta$ into $U_\theta$ and $F_\theta$
    Unpack $Q_{\theta'}$ into $U_{\theta'}$ and $F_{\theta'}$
    for each experience $\langle \mathbf{o}, \mathcal{A}, \mathbf{r}_u, \mathbf{r}_f, \mathbf{o}' \rangle$ in the mini-batch do
        Compute the combined Q-values for the successor observation:
            $Q_\theta(\mathbf{o}') = (1-\beta)U_\theta(\mathbf{o}') + \beta F_\theta(\mathbf{o}')$
        Solve ILP to get the optimal action $\mathcal{A}^*$ for the next observation $\mathbf{o}'$:
            $\mathcal{A}^* = \text{solve\_ILP}(Q_\theta(\mathbf{o}'), \text{env.constraints})$
        for model in $\{U, F\}$ do
            if model is $U$ then
                Set $M_\theta = U_\theta$, $M_{\theta'} = U_{\theta'}$, and $r = \mathbf{r}_u$
            else
                Set $M_\theta = F_\theta$, $M_{\theta'} = F_{\theta'}$, and $r = \mathbf{r}_f$
            Compute the target for the TD update:
                $\text{target} = r + \gamma M_{\theta'}(\mathbf{o}', \mathcal{A}^*)$
            Compute the TD loss:
                $\text{loss} = (M_\theta(\mathbf{o}, \mathcal{A}) - \text{target})^2$
            Perform gradient descent on the TD loss to update $M_\theta$
```

Let $U_{tot}(\mathcal{A})$ and $F_{tot}(\mathcal{A})$ be defined as follows, given an allocation $\mathcal{A}$:

$$U_{tot}(\mathcal{A}) = \sum_{i \in \alpha} U(\mathcal{A}_i) \tag{10.19}$$

$$F_{tot}(\mathcal{A}) = \sum_{i \in \alpha} F(\mathcal{A}_i) \tag{10.20}$$

We remind the reader that $\mathcal{A}_i$ refers to the action assigned to agent $i$ in the allocation $\mathcal{A}$.

Let $\mathbf{Z}_t$ represent the agent metrics at time $t$. Further, let $\mathcal{A}^*$ represent the optimal allocation from the ILP with $\eta$ as the trade-off weight. Since $\mathcal{A}^*$ is optimal, it follows that for all other possible allocations $\mathcal{A}_o$:

$$U_{tot}(\mathcal{A}^*) + \eta F_{tot}(\mathcal{A}^*) \geq U_{tot}(\mathcal{A}_o) + \eta F_{tot}(\mathcal{A}_o) \tag{10.21}$$

$$U_{tot}(\mathcal{A}^*) - U_{tot}(\mathcal{A}_o) \geq \eta(F_{tot}(\mathcal{A}_o) - F_{tot}(\mathcal{A}^*)) \tag{10.22}$$

We are interested in finding what happens to the allocation when we increase $\eta$. For $\eta' > \eta$, note that the left side of Eq. 10.22 remains the same. Since utility estimates are not affected by changing $\eta$, any other allocation $\mathcal{A}_o$ can only be selected over $\mathcal{A}^*$ if the following condition holds:

$$U_{tot}(\mathcal{A}^*) + \eta' F_{tot}(\mathcal{A}^*) \leq U_{tot}(\mathcal{A}_o) + \eta' F_{tot}(\mathcal{A}_o) \tag{10.23}$$

$$U_{tot}(\mathcal{A}^*) - U_{tot}(\mathcal{A}_o) \leq \eta'(F_{tot}(\mathcal{A}_o) - F_{tot}(\mathcal{A}^*)) \tag{10.24}$$

Combining Eqs.10.22 and 10.24, we get the following:

$$\eta(F_{tot}(\mathcal{A}_o) - F_{tot}(\mathcal{A}^*)) \leq \eta'(F_{tot}(\mathcal{A}_o) - F_{tot}(\mathcal{A}^*)) \tag{10.25}$$

$$F_{tot}(\mathcal{A}^*)(\eta' - \eta) \leq F_{tot}(\mathcal{A}_o)(\eta' - \eta) \tag{10.26}$$

Since $\eta \geq 0$ and $\eta' > \eta$, Eq. 10.26 can only be true if $F_{tot}(\mathcal{A}_o) > F_{tot}(\mathcal{A}_o)$.

Thus, any allocation $\mathcal{A}_o$ that is optimal (and thus selected by the ILP) for $\eta' > \eta$ is guaranteed to have equal or better fairness than the allocation $\mathcal{A}^*$ at $\eta$. □

We also state the corollary to Theorem 1.

**Corollary 4.** *Given perfect estimates for utility and fairness, decreasing $\eta$ always improves the one-step utility gain for SO with $\gamma = 0$.*

The proof follows a similar structure to Theorem 1.

We also show the following useful property:

**Theorem 2.** *For a large enough $\eta$, the fairest allocation will be selected with perfect utility and fairness estimators for SO with $\gamma = 0$.*

*Proof.* Let $\mathcal{A}_f$ denote the allocation with the largest fairness gain:

$$\mathcal{A}_f = \underset{\mathcal{A}}{\operatorname{argmax}}\, F_{tot}(\mathcal{A})$$

For simplicity, let us assume no two allocations have the same $F_{tot}(\mathcal{A})$. For any other allocation $\mathcal{A}_o$, we have:

$$F_{tot}(\mathcal{A}_f) > F_{tot}(\mathcal{A}_o) \tag{10.27}$$

Then, $\mathcal{A}_f$ will be optimal and selected by the ILP if the following condition holds:

$$U_{tot}(\mathcal{A}_f) + \eta_f F_{tot}(\mathcal{A}_f) \geq U_{tot}(\mathcal{A}_o) + \eta_f F_{tot}(\mathcal{A}_o) \tag{10.28}$$

$$\eta_f \geq \frac{U_{tot}(\mathcal{A}_o) - U_{tot}(\mathcal{A}_f)}{F_{tot}(\mathcal{A}_f) - F_{tot}(\mathcal{A}_o)} \tag{10.29}$$

We can compute an upper bound for $\eta_f$ by considering the range of values that $U_{tot}$ and $F_{tot}$ can take. Let $U_{max} = \max_{\mathcal{A}} U_{tot}(\mathcal{A})$, and $F_{max} = \max_{\mathcal{A}, \mathcal{A} \neq \mathcal{A}_f} F_{tot}(\mathcal{A})$.

Then, we have the following:

$$\eta_f \geq \frac{U_{tot}(\mathcal{A}_o) - U_{tot}(\mathcal{A}_f)}{F_{tot}(\mathcal{A}_f) - F_{tot}(\mathcal{A}_o)} \tag{10.30}$$

$$\leq \frac{U_{max} - U_{tot}(\mathcal{A}_f)}{F_{tot}(\mathcal{A}_f) - F_{tot}(\mathcal{A}_o)} \tag{10.31}$$

$$\leq \frac{U_{max} - U_{tot}(\mathcal{A}_f)}{F_{tot}(\mathcal{A}_f) - F_{max}} = \eta_f^u \tag{10.32}$$

Eq. 10.32 gives us an upper bound for $\eta_f$. Thus, for all $\eta > \eta_f^u$, $\mathcal{A}_f$ will be the optimal allocation. □

**Corollary 5.** *For a small enough $\eta$, the most utilitarian allocation will be selected with perfect utility and fairness estimators for SO with $\gamma = 0$.*

The proof follows a similar structure to the proof for the previous theorem.

These theorems ensure that for any state, we will select actions that improve fairness in the long run starting from that state as $\eta$ is increased. These properties also empirically hold when $\gamma \neq 0$, as our experiments demonstrate. This adaptability is a major strength of SO: It allows a degree of flexibility that other methods do not possess. Specifically, SO allows users to vary the trade-off weight $\beta$ during runtime, and the behavior can be expected to be monotonic in the direction of change. For $\gamma = 0$, Theorem 1 and its corollary guarantee that the space of selected allocations is Pareto-efficient with changing $\beta$ at each time step.

## 10.5 Experimental Setup

We conduct experiments for maximizing the objective in Eq. 10.1, where the system utility is the sum of all agent utilities at the end of an episode, and the fairness is measured as the negative of the variance of agent resources at the end of the episode. We perform experiments for a variety of $\beta$ values, repeating each configuration 5 times for each of our three settings: **Joint Optimization (JO)**, **Split Optimization (SO)** and **Fair-Only Optimization (FO)**. We were unable to use off-the-shelf multi-agent RL libraries because of their lack of support for constrained central decision making. Thus, we implemented versions of the learning algorithm (DDQN with $\epsilon$-greedy TD(0) learning), as described in Section 10.3. The network architecture has two hidden layers of dimension 20, and output of dimension 1. The utility model used for FO is randomly selected from the JO models trained with $\beta = 0$. We included features indicating the relative advantage of each agent as a signal for fairness, in addition to the features describing the local observation of each agent.

### 10.5.1 Environments

To evaluate our methods, we created new DECA environments of varying difficulty, based on existing fair MARL benchmarks [66]. We reformulate them as resource allocation problems, adding explicit resource constraints and creating action spaces corresponding to resources. From simplest to most complex, we have the following:

- **BiasedDM** is a simple setting with one resource per timestep and a biased utility function that favors certain agents, creating a gap between fairness and optimality.
- **JobAlloc** introduces basic coordination: agents compete to claim an exclusive job, and while a greedy agent can monopolize it, fairness requires agents to sometimes give up the job so others can benefit—making collaboration essential.

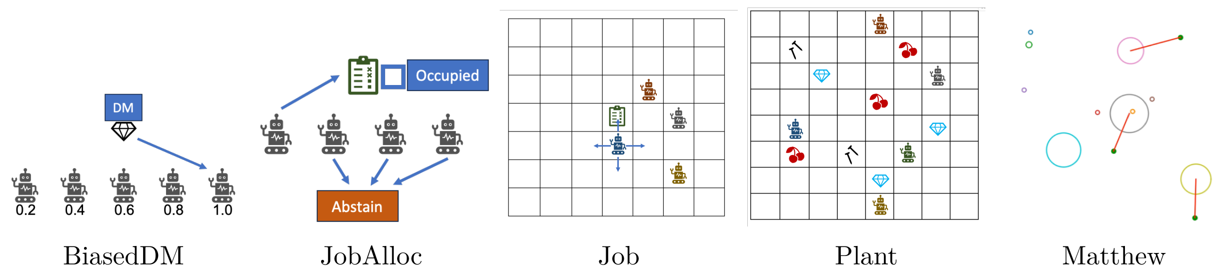


Figure 10.2: Illustration of all five environments

- **Job** extends this to a grid world where agents move to job locations but must avoid collisions, adding spatial constraints.
- **Plant** increases complexity by requiring agents to collect different combinations of resources to complete tasks, introducing combinatorial dependencies.
- **Matthew** is the most complex, modeling the Matthew effect [127, 49], where early allocations give lasting advantages, leading to compounding inequality over time.

It is important to note that while these environments have been designed to appear similar at a high-level as the versions used in the baselines [66, 176], they are completely different under the hood, incorporating agent and resource constraints and modifying the action spaces to align with the resource allocation tasks.

We describe each environment in detail below, and provide an illustration of all five environments in Figure 10.2.

### JobAlloc

JobAlloc focuses on coordination under exclusivity. A single job (resource) is available, and four agents must learn to take turns occupying it to ensure fair access. A greedy strategy allows one agent to monopolize the job, but a fair outcome requires agents to sometimes relinquish the job so others can benefit, which involves temporary self-sacrifice.

Action dynamics: At each timestep, all agents evaluate two actions: attempting to occupy the job or forfeiting it. However, only one agent may occupy it at a time, and a transition can only happen if no agent is on the job at the beginning of the timestep. This means successful transfer requires coordination: the current occupant must vacate, so that another agent may attempt to claim it at the next time step. This constraint introduces a “gap step” where no one receives a reward, making fair transitions more costly and difficult.

There are 4 agents, and one episode lasts 100 steps.

### Job

Job builds on JobAlloc by introducing a spatial grid, where agents must navigate toward a fixed job location. The job remains in the center of the grid, and only one agent can occupy it at a time. A fair outcome requires agents to not only reach the job but also time their arrivals to avoid collisions and take turns accessing the reward. This emulates the Job environment in Jiang and Lu [66] with added constraints.

Action dynamics: Each agent occupies a position on a $7 \times 7$ grid and can move in the four cardinal directions or remain still. The job location is fixed at the center. Agents independently score their movement preferences each timestep, and the centralized decision-maker resolves conflicts and determines the final set of moves. Movement is constrained—agents cannot leave the grid or move into an occupied location—so timing and coordination are essential to sharing the job fairly.

There are 4 agents on the grid, and one episode for this environment lasts 100 steps. Agents are able to observe nearby grid cells in a $3 \times 3$ area centered on the agent.

### Plant

Plant is a multi-resource, multi-goal environment where five agents must collect combinations of three resource types to construct units and earn rewards. Each agent has a unique requirement profile—some easier than others—leading to natural inequality in unit completion rates. A fair allocation in this environment involves compensating for these differences to balance agent-level success over time.

Action dynamics: 5 agents operate on an $8 \times 8$ grid containing eight randomly placed resources of three types. Each agent submits preferences for which resource to pursue, and a centralized decision-maker allocates resources to avoid conflict. Agents move deterministically toward assigned resources. When an agent collects the needed combination of resources, they build a unit and receive a reward. Each agent has a requirement of a set of resources it must collect so that it can construct this unit. The requirements are:

$$\{(2,1,0),(1,0,1),(1,0,0),(1,3,0),(0,1,2)\}$$

For example, agent 1 requires two resources of type 1 and one resource of type 2 following which it can get a reward. Some agents are given easier requirements to fulfill, which creates a bias in the number of units agents produce. The resources and agent locations are randomly initialized at the beginning of each episode. The differing resource requirements add a combinatorial challenge to both planning and fairness. An action corresponds to an agent 'claiming' a resource, with other agents unable to pursue resources already allocated to other agents. Upon collection of a resource, another resource of the same type appears in a random location on the map.

There are 5 agents, and one episode for this environment lasts 200 steps. Agents are able to observe nearby grid cells in a $5 \times 5$ area centered on the agent.

**Matthew**

Matthew is designed to showcase the Matthew effect: early advantages amplify over time, leading to persistent inequality. Ten agents compete for three resources per timestep on a continuous 2D plane. Agent speeds grow proportionally to their size, and a size ceiling exists to prevent agents from growing too large for the environment bounds. Agent and resource positions are 2-D coordinates in $[0,1]$. At the beginning of each episode, agent and resource positions are randomly initialized. To show the Matthew effect, 4 agents are initialized to have a larger initial size than other agents, making them more likely to secure early rewards—which further increase their speed and size. Fairness here means interrupting this compounding feedback loop to ensure long-term balance in access and growth.

Action dynamics: At each step, agents estimate the utility of pursuing each available resource. The centralized decision-maker assigns which agent gets to pursue which resources. Once a resource is assigned, the agent moves in a straight line toward it and cannot switch targets mid-motion. Unassigned agents perform a null action resulting in random movement.

Agents receive a unit reward when they collect a resource, in addition to a small increase in size and speed. Resources allocated to agents are reserved, and other agents cannot pick them up. A new resource only spawns when an agent reaches its allocated resource, not when it is allocated. The dynamics encourage cumulative advantage, as resource collection leads to further growth, making early interventions crucial to promoting fairness.

There are 10 agents, and one episode for this environment lasts 200 steps.

**BiasedDM**

BiasedDM is a minimal environment that isolates the trade-off between fairness and utility. In each timestep, a single resource is available, and the decision-maker must choose which agent among five will receive it. The twist is that the decision-maker's utility is biased: higher-indexed agents yield more utility. A fair policy in this setting must avoid allocating all resources to the most "valuable" agent and instead distribute allocations more evenly over time.

Action dynamics: At each step, agents communicate utilities for getting the resource or not getting it, and the decision-maker selects one of the five agents to receive the resource. Because the optimal (utility-maximizing) strategy is to always choose the same agent, fairness requires intentional deviation from that utilitarian solution to give other agents a share of the resource. This could entail the advantaged agent learning to have a smaller difference in its valuation of getting versus not getting the resource, allowing the decision-maker to hand it to another agent.

There are 5 agents, and one episode for this environment lasts 100 steps. In other environments, fairness is computed as the variance over the accumulated rewards for each agent. In this environment, however, fairness is computed over the resource rate, which is the fraction of steps in which an agent received the resource ($z_i \in [0, 1]$). This is an example of an averaged payoff metric, as described in Chapter 2. We compute $z_i$ as follows:

$$z_i = \frac{\text{Num. resources}}{time} \tag{10.33}$$

This also results in much smaller variances, thus our hyperparameter search for this domain explores the higher range of $\beta$ values more.

### 10.5.2 Baselines

As noted in our Related Work section, FEN [66] and SOTO [176] are the most relevant approaches that generalize to multiple domains; we thus include them as baselines in our experiments. However, both assume settings where agents act independently without resource constraints. To adapt them to the DECA framework, we evaluate two strategies for making constrained decisions using their per-agent policies:

- **Policy-as-Q ('_ILP' suffix):** We interpret action probabilities from a learned policy $\pi$ as proxy Q-values $Q(o_i, a) = \pi(o_i, a)$ and input them into the ILP (Eq. 10.10) for centralized allocation.
- **Masked Sequential ('_Mask' suffix):** Agents select actions sequentially in a randomized order at each timestep. Each agent samples from its policy, with infeasible actions masked out based on remaining resource availability. The randomized agent order is used to mitigate ordering bias.

We add the extra features that SOTO requires (only for SOTO), and train SOTO with both the $\alpha$-fair and $GGF$ objective described in their paper [176]. We implement shared weights across agents.

#### FEN and SOTO Implementation Details

For FEN and SOTO, we use 5 times the number of training steps as used for DECAF, to allow the PPO based approaches sufficient trajectories to learn from, and to better match the experiments in the respective papers. We do not use past discounts and warm starts. For BiasedDM, we chose the reward vector to be the number of resources each agent received (1 for each agent), instead of the biased utility to the decision-maker ($0.2 \cdot i$ for agent $i$). Since our fairness metrics operate on the vector of resources as well, and since FEN and SOTO aim to learn fairness, this was the obvious choice. We also ran experiments where the reward vector was based on the decision-maker's biased utility, and found the solutions to be poor for both utility and fairness based on resources, so we omit them in our results. However, the greedy self-oriented model in SOTO was trained using the decision-maker's reward.

We used FEN without gossip, where the agent is directly communicated the distribution information, without need for inter-agent communication rounds. This is expected to be the stronger version of FEN. For all models, we used the same features from the DECA environments, containing the agent's relative advantage as a feature. In addition to this, SOTO also required the entire payoff distribution $\mathbf{Z}$ and information about nearby neighbors for the tiered team-oriented network.

The Job environment uses a shaping reward for the distance to the job as a scaled penalty. For the Job environment for SOTO, we reduced the weight of the shaping reward to 0.01 to minimize its effect on the optimization. We found this to be the best setting for learning in this environment. The ILP version of SOTO was not able to learn at all in this setting, even when we completely removed the shaping reward. Again, the inability to use shaping rewards is a significant handicap for methods like SOTO, since they optimize fairness over the rewards. Our methods explicitly decouple learning utility and fairness, so they are more expressive, and able to learn better especially at intermediate values of $\beta$.

### 10.5.3 Model Architecture and Training Details

All our models use a learning rate of 0.0003 with the Adam optimizer. For all environments except BiasedDM, we train for 1000 episodes, and run validation every 50 steps for model selection. For BiasedDM, we train for 200 episodes, and validate every 20 episodes.

The neural network architecture for all models is the same, with two fully connected hidden layers, of dimension 20, with ReLU activations. The output (1-dimensional) does not have any activation function. We implement our networks using pytorch.

We use a replay buffer of size 250000, where one experience is a joint transition across all agents. During training, we sample experiences from the replay buffer, and for each experience, we evaluate actions for all agents using the current online network, solving the ILP to get the best joint action. Then, we score the post-decision state for each agent using the target network, and compute the MSE loss between the target value and value estimates of the selected action from the online network.

Table 10.1: Evaluation of all models on multiple metrics for each environment. For JO, SO, and FO, the values in the bracket denote the $\beta$ value selected based on the model that maximizes $0.1 \cdot U$ - $0.9 \cdot \text{var}(\mathbf{Z})$. The selected $\beta$ value is indicated in brackets. The values in bold are the best in each row.

| Environment | Metric | JO($\beta$) | SO($\beta$) | FO($\beta$) | FEN | SOTO($\alpha$-Fair) | SOTO(GGF) |
|---|---|---|---|---|---|---|---|
| BiasedDM | $\alpha$-Fair | **-8.09(1)** | -8.3(0.999) | -8.19(0.9995) | -8.15 | -8.15 | -8.15 |
| | GGF | **0.35(1)** | 0.3(0.999) | 0.33(0.9995) | 0.33 | 0.33 | 0.33 |
| | Maximin | **0.16(1)** | 0.12(0.999) | 0.15(0.9995) | 0.15 | 0.15 | 0.15 |
| | System Utility | 58.31(1) | **63.65(0.999)** | 63.38(0.9995) | 59.95 | 60.22 | 60.24 |
| | Variance | **-0.0007(1)** | -0.0033(0.999) | -0.0022(0.9995) | -0.0014 | -0.0015 | -0.0015 |
| JobAlloc | $\alpha$-Fair | **12.71(0.2)** | 12.69(0.2) | **12.71(0.2)** | -35.31 | -7.39 | 1.06 |
| | GGF | 43.11(0.2) | 43.41(0.2) | **43.8(0.2)** | 12.59 | 19.5 | 29.62 |
| | Maximin | 21.79(0.2) | 22.21(0.2) | **22.71(0.2)** | 0 | 3.49 | 10.62 |
| | System Utility | 96.28(0.2) | 95.81(0.2) | 96(0.2) | **99.89** | 94.42 | 90.92 |
| | Variance | -4.44(0.2) | -2.54(0.2) | -1.48(0.2) | -1839.76 | -923.46 | -421.93 |
| Job | $\alpha$-Fair | 10.97(0.2) | **12.07(0.2)** | 10.77(0.2) | -35.89 | -55.21 | 5.03 |
| | GGF | 37.12(0.2) | **37.88(0.2)** | 26.65(0.2) | 11.87 | 0 | 22.17 |
| | Maximin | 16.99(0.2) | **18.13(0.2)** | 13.13(0.2) | 0 | 0 | 4.9 |
| | System Utility | 88.43(0.2) | 88.63(0.2) | 61.68(0.2) | **94.88** | 0 | 80.57 |
| | Variance | -25.14(0.2) | -13.38(0.2) | -4.59(0.2) | -1683.49 | **0** | -242.3 |
| Plant | $\alpha$-Fair | **15(0.8)** | 14.77(0.8) | 14.54(0.8) | -35.48 | -20.75 | -20.62 |
| | GGF | **35.67(0.8)** | 34.63(0.8) | 33.45(0.8) | 1.4 | 10.28 | 10.89 |
| | Maximin | **16.61(0.8)** | 15.98(0.8) | 15.74(0.8) | 0 | 3.62 | 4.08 |
| | System Utility | **101.72(0.8)** | 99.38(0.8) | 94.89(0.8) | 13.31 | 42.57 | 43.84 |
| | Variance | -6.34(0.8) | -6.75(0.8) | **-4.99(0.8)** | -41.64 | -54.29 | -49.91 |
| Matthew | $\alpha$-Fair | 20.03(0.2) | **23.4(0.5)** | 20.71(0.5) | -29.04 | 12.45 | 12.64 |
| | GGF | 14.41(0.2) | **19(0.5)** | 11.92(0.5) | 1.73 | 4.6 | 4.67 |
| | Maximin | 3.64(0.2) | **8.86(0.5)** | 5.09(0.5) | 0.36 | 1.66 | 1.69 |
| | System Utility | **140(0.2)** | 108.1(0.5) | 85.5(0.5) | 47.39 | 42.77 | 43.02 |
| | Variance | -26.95(0.2) | **-1.28(0.5)** | -5.17(0.5) | -45.9 | -3.33 | -3.32 |

We ran all our main experiments on a university compute cluster, with each experiment running on a single CPU node with 12GB RAM. Experiment runtime varied with environment choice. Training a single model with one $\beta$ value took between 30 minutes (BiasedDM) and 2 hours (Matthew). Evaluation, as for the generalization experiments, was performed on a 2019 MacBook Pro, where one episode took 2-5 seconds to run, and we bootstrap over 5 runs and report the average performance.

## 10.6 Experimental Results

Figure 10.3 shows the performance of all three DECAF methods (JO, SO, FO) on the five domains discussed above. For each method, we varied the hyperparameter $\beta$ controlling the fairness-utility trade-off (Eqs. 10.13, 10.16, 10.18), starting with $\beta = 0$ (top-left) and increasing to $\beta = 1$ (bottom-right). As mentioned in Section 10.3, we present results where

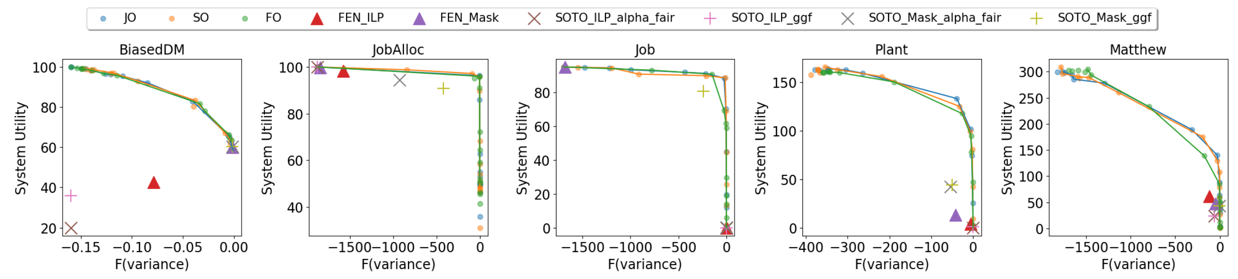


Figure 10.3: Change in system utility and fairness as $\beta$ is increased, with $\beta = 0$ at the top left $\beta = 1$ at the bottom-right. For all domains, we can see that split and joint optimization perform similarly, while learning only fairness can sometimes be slightly worse. All our methods Pareto-dominate SOTO and FEN. Each point depicts the average performance over five different models trained at that $\beta$ value, and the lines show the Pareto front for each method.

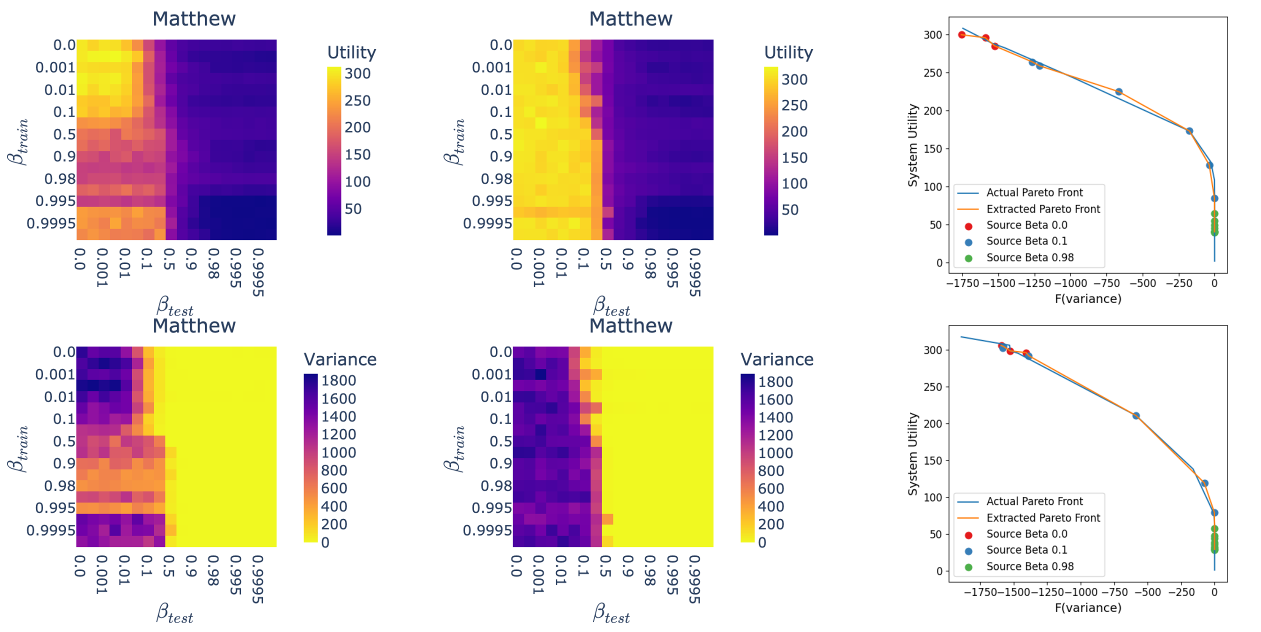
we optimize for variance as the fairness function here. Additional results on learning with different fairness functions are included in Section 10.6.7.

### 10.6.1 Efficacy of the Fairness-Utility Optimization

For all domains, all three methods are able to learn expressive policies which lie at various points close to the Pareto front. This shows that the optimization allows the model to trade off utility and fairness to show diverse behaviors as required by the user. This also confirms that the fairness reward proposed for minimizing variance is a good signal.

### 10.6.2 Comparison Against Baselines

Although we adapt FEN and SOTO to operate in the DECA setting, their underlying learning algorithms assume on-policy transitions from decentralized policies. The introduction of resource constraints fundamentally alters the environment dynamics, breaking this assumption. As a result, their performance degrades—especially in complex domains like Matthew and Plant, where coordinated, long-term strategies are necessary. In simpler settings, where greedy decisions align with utilitarian outcomes, these methods can still recover reasonable policies. However, even when optimizing social welfare functions like GGF or $\alpha$-fairness, they struggle to find high-performing strategies under constraints.



(a) SO models (b) FO models (c) Approximated Pareto fronts

Figure 10.4: Generalization analysis on the Matthew environment. (a) and (b) show the generalization of SO and FO models trained on $\beta_{train}$ and tested on $\beta_{test}$. Brighter colors indicate better outcomes for utility (top) and variance (bottom). (c) Approximated Pareto fronts using sparse $\beta_{train}$ evaluated on other $\beta$ values for the Matthew domain.

Among the adapted baseline variants, the masked versions outperform their ILP counterparts, as ILP-based selection produces trajectories that violate the assumptions of policy gradient methods. However, masking also limits coordination and suffers from order sensitivity. In contrast, DECAF handles global constraints and large, combinatorial action spaces more naturally, giving it a clear advantage in the DECA setting. As shown in Figure 10.3, DECAF consistently Pareto-dominates both FEN and SOTO across all domains, with SOTO_Mask (GGF) being the most competitive baseline. Table 10.1 presents results for selected $\beta$ values of JO, SO, and FO for each environment, showing DECAF's advantage across multiple metrics, even when trained on variance only.

## 10.6.3 Comparison of DECAF Methods

In our results, JO and SO generally exhibit similar performance characteristics, suggesting that simultaneous evolution of utility and fairness estimates is beneficial. FO is also very competetive, but in complicated environments (e.g., Matthew), it falls below the Pareto front. This underperformance is likely due to out-of-distribution transitions for the fixed utility model, which are more problematic in FO when a large fairness weight $\beta$ is used, causing a larger shift in the state distribution and resulting in degraded utility estimates. We expect this to be a bigger issue in more complicated environments, or with poor black-box utility functions.

### 10.6.4 Generalization Using Split Optimization (SO)

Figure 10.4a shows detailed results for the Matthew environment, when using SO. We evaluate each model trained on a particular $\beta_{train}$ on all other $\beta_{test}$. This allows us to see how well the trained fairness and utility models generalize when the operating $\beta$ is changed. Note that for all these models, $\beta$ is not provided as a feature to the Q-function.

The diagonal elements show the behavior when training and testing is done on the same $\beta$ value. From the plots for system utility (Figure 10.4a(top)) and variance (Figure 10.4a(bottom)), we can see that as $\beta_{test}$ increases, variance improves, and as $\beta_{test}$ decreases, utility improves. This is the expected behavior, and the major advantage of SO over JO. With JO, the model only predicts a single value, so we are unable to change the trade-off weight during evaluation, and we require a unique model for each $\beta$ that we want the model to work for. However, with SO, selecting just a few spread out $\beta$ values can allow us to extrapolate between them, providing online adaptability. This shows that SO has the flexibility to function well at operating points away from the $\beta_{train}$ that it is trained for.

Figure 10.4c (top) shows how well a few selected models can generalize to the Pareto front. We pick $\beta_{train}$ values evenly spaced across the search space, and evaluate the model on all $\beta_{test}$, picking the closest $\beta_{train}$ in order of the search space. We can see that the approximated Pareto front closely matches the actual Pareto front, even with just 3 models, further demonstrating the strength of SO. These observations hold for other environments as well, and the results are included in Section 10.6.9.

### 10.6.5 Effectiveness of Fair-Only Optimization (FO)

Like SO, FO is also able to generalize well when different $\beta_{test}$ are used to evaluate the learned models (Figure 10.4b). Because the utility model is fixed, all models achieve high utility as $\beta_{test} \to 0$ (Figure 10.4b (top)). Further, all models also improve fairness as $\beta_{test}$ grows larger (Figure 10.4b (bottom)). The behavior change from utility-oriented to fairness-oriented is much sharper in FO when compared to SO. Looking at Figure 10.4c (bottom), we can again see that even FO has the ability to generalize from only a few models to cover the entire Pareto front. Despite being Pareto-dominated by SO and JO at intermediate $\beta$ values in some domains, FO has the advantage of reliability: A trusted black-box utility model can be used in conjunction with a possibly smaller fairness model, with the guarantee to behave optimally as $\beta_{test}$ is reduced. When such a model is available, FO is the best choice, given its competent performance and lower computational load.

### 10.6.6 Learning with Other Fairness Functions

As mentioned earlier, we are not restricted to using variance and the given decomposition. We show here results for three more functions: $\alpha$-fairness, $GGF$, and maximin.

**1. $\alpha$-fairness** The $\alpha$-fair function can be stated as:

$$F_\alpha(\mathbf{Z}) = \sum_{z_i \in \mathbf{Z}} \begin{cases} \frac{z_i^{1-\alpha}}{1-\alpha} & \alpha \neq 1 \\ \log z_i & \alpha = 1 \end{cases} \tag{10.34}$$

With $\alpha = 1$, this is equivalent to proportional fairness or log Nash Welfare, both popular notions of fairness, while at $\alpha = 0$ it represents the utilitarian objective. In our experiments, we use $\alpha = 1$.

**2. Generalized Gini Function (GGF)** Given a sequence of positive, fixed, strictly decreasing weights $\mathbf{w}$, the GGF function can be stated as:

$$G_w(\mathbf{Z}) = \sum_{i \in \alpha} \mathbf{w}_i z_i^{\uparrow} \tag{10.35}$$

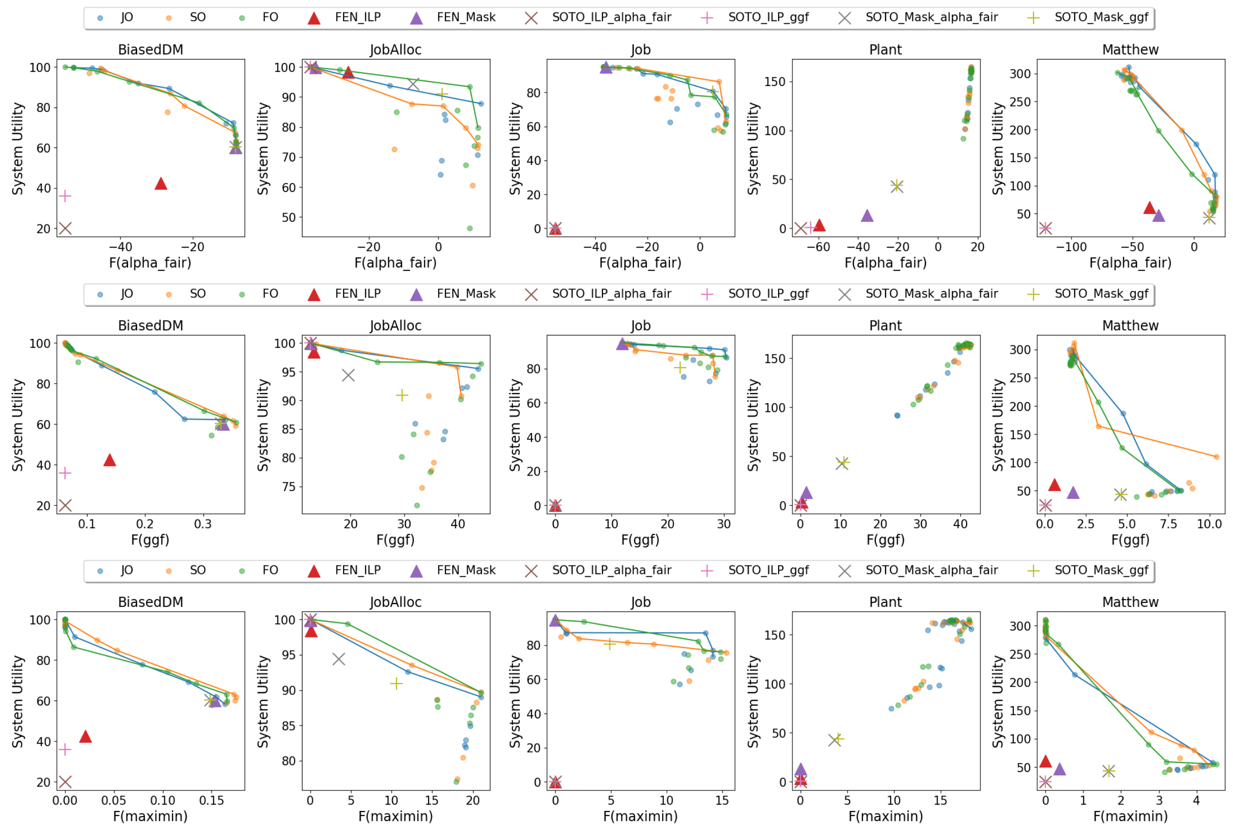


Figure 10.5: Results training DECAF with $\alpha$-fairness, GGF and maximin fairness functions. The lines show the Pareto fronts for each model type.

Here, $\mathbf{z}^{\uparrow}$ represents the vector obtained by sorting the $\mathbf{Z}$ vector. This function can also represent a diverse set of SWFs including utilitarian and maximin fairness. In our experiments, we use decreasing negative powers of 2 as the weights (i.e. $\mathbf{w} = [1, 2^{-1}, 2^{-2}, \ldots 2^{-(n-1)}]$).

For both of the above objectives, we use the equal decomposition, where the fairness reward is computed as an equal division of the change in the metric value across all agents.

$$R_f(\mathbf{s}_t, \mathcal{A}^t) = \left[\frac{\Delta\mathcal{F}|\mathcal{A}^t}{n}\right]_{i\in\alpha} \tag{10.36}$$

**3. Maximin Fairness** This function captures the worst off utility of any agent:

$$F_{MMF}(\mathbf{Z}) = \min(\mathbf{Z}) \tag{10.37}$$

This is a hard objective to learn, as the maximin objective changes only when the worst-off agent is improved. We decompose this reward by combining the global signal with the per-agent contribution towards improving the minimum. Intuitively, each agent receives a reward for a joint action that improves the minimum, but the agents that were at the minimum (and improved) receive a larger reward.

$$r_{f,i} = \frac{\min(\mathbf{Z}') - \min(\mathbf{Z})}{n} \tag{10.38}$$

$$r_{f,i} = r_{f,i} + \begin{cases} z_i' - z_i & \text{if } z_i = \min(\mathbf{Z}) \\ 0 & \text{otherwise} \end{cases} \tag{10.39}$$

$$r_{f,i} = r_{f,i} + \begin{cases} z_i' - z_i & \text{if } z_i' = \min(\mathbf{Z}') \\ 0 & \text{otherwise} \end{cases} \tag{10.40}$$

$$R_{f,i} = \frac{r_{f,i}}{\sum_j r_{f,j}} \left(\min(\mathbf{Z}') - \min(\mathbf{Z})\right) \tag{10.41}$$

### 10.6.7 DECAF Results with Other Fairness Metrics

Figure 10.5 shows the results of our approaches when using different fairness functions, with the decompositions as described above. In general, we observe our methods offer a good range of trade-offs in all environments, often Pareto-dominating SOTO and FEN. We describe some additional details about these results in this section.

In almost all environments, it is possible to get significant fairness improvement starting from the utilitarian model. The only exception is the Plant environment for $\alpha$-fair and GGF, where the utilitarian solution is almost optimal for fairness as well. In Job and JobAlloc, SOTO masked models are competitive, especially for the $\alpha$-fair fairness function, while some DECAF models are not as performant. This may suggest the existence of a more informative decomposition for this function which can lead to better learning. In BiasedDM, SOTO and FEN face a significant disadvantage, as they can not contend with the misaligned fairness signal. Thus, they can either know the system utility, or the payoffs on which fairness is based. In the prior case, the learned fair policy performs poorly for both utility and fairness, as it tries to equalize the accumulated value from the decision maker's perspective, or, as in the latter case (selected for the results shown), only looks at the resource distribution and has no idea of the utility to the decision maker.

$\alpha$-fair and GGF could also be decomposed to compute per-agent contributions, but since the metrics treat all agents indepependently, the interaction between agent utilities is harder to learn, especially for the Job and JobAlloc environment, where being fair requires a globally suboptimal decision in terms of utility. In other environments, the pressure for fairness is better captured in the ILP optimization, by the valuations of other agents that could benefit more from getting certain resources.

### 10.6.8 On the Importance of Past Discounting and Warm Starts

We also highlight two departures from prior work in our implementation of DECAF for learning fairness: (1) We discount the agent metrics $\mathbf{Z}$ over the past, and (2) we implement warm starts for initializing $\mathbf{Z}$ instead of initializing at 0. The past discounts are necessary to account for the time dependence of various normalized fairness metrics, like the Gini coefficient and coefficient of variation, or the variance over normalized agent metrics. In these cases, actions taken early on can cause larger changes to the fairness metric while actions taken after a sufficient history has been established have an imperceptible effect. This is undesirable in long-horizon settings, but can be remedied by past discounts, effectively 'forgetting' events that happened far in the past. The warm starts function to counteract

Table 10.2: Warm start ($w$) and past discount ($\gamma_p$) values for different environments and fairness functions used for DECAF.

| | | **Matthew** | **Plant** | **Job** | **JobAlloc** | **BiasedDM** |
|---|---|---|---|---|---|---|
| **$\alpha$-fair** | $w$ | 0 | 0 | 0 | 0 | 0 |
| | $\gamma_p$ | 1 | 1 | 1 | 1 | 1 |
| **GGF** | $w$ | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| | $\gamma_p$ | 1 | 1 | 1 | 1 | 1 |
| **Maximin** | $w$ | 5 | 1 | 3 | 3 | 2 |
| | $\gamma_p$ | 0.995 | 0.995 | 0.995 | 0.995 | 0.999 |
| **Variance** | $w$ | 5 | 1 | 3 | 3 | 2 |
| | $\gamma_p$ | 0.995 | 0.995 | 0.995 | 0.995 | 0.999 |

the other pitfall in some fairness metrics: The zero vector is perfectly fair, and any changes can incur a large fairness penalty. Adding randomized warm starts helps in preventing the algorithm from converging to this trivial solution. In practice, we keep the warm start values small, so that the past discounts effectively scale them down to zero over the course of an episode.

## Past Discounts and Warm Starts

Past discounts and warm starts significantly helped in improving the stability of learning in our experiments, especially with variance as the fairness function. Small initial perturbations (that are smoothed out over time using past discounts) help in exploration of different states, as well as in avoiding the cold start problem where any action would lead to a worse state and hence agents learn to avoid taking beneficial actions. For variance, we used warm starts based on the size of the maximum rewards possible in an episode in each environment. For BiasedDM, the warm starts are used to create an initial 'resource rate' by normalizing based on the number of total warm start resources, as this environment uses resource rates as the payoff vector $\mathbf{Z}$ .

For $\alpha$-fair, we used $\alpha = 1$ for the experiments, and hence we avoided using warm starts for this metric, as the log function is sensitive to small perturbations especially with near-zero utilities. For GGF, we used a warm start of 0.1 for each environment. For both of these functions, we used no past discounting, as the metrics are additive functions over agent utilities, so any past discounting would cause negative fairness gains. For maximin, we used the same past discounting and warm start values as variance. The warm start and past discount values for all environments are given in Table 10.2.

Given a warm start value of $w$, we compute an initial distribution of pseudo-resources by uniformly sampling from a region of width $w/4$ centered around $w$. For additive utilities, we use past discounts to decay the accumulated payoffs in $\mathbf{Z}$ before adding the current time-step's reward. If $\gamma_p$ is the past discount factor, and $R_{t,i}$ is the resource value allocated to agent $i$ at time $t$, then we compute:

$$z_i^{t+1} = \gamma_p z_i + R_{t,i}$$

For averaged rewards (as in BiasedDM), where the payoff $z_i = \#resources/\#timesteps$, we compute the discount by reweighting both the numerator and denominator. Note that this can be easily extended to act as a 'resource rate' where the denominator is the number of potential resources the agent could have received, instead of the time.

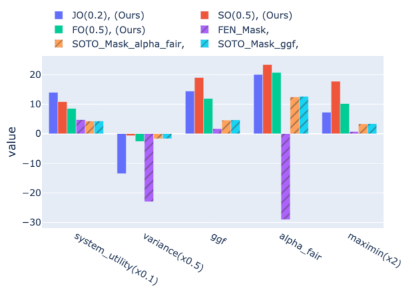
$$\begin{aligned} z_i^t &= \frac{res_i}{t_i} \\ res_i &= \gamma_p res_i + R_{t,i} \\ t_i &= \gamma_p t_i + 1 \\ z_i^{t+1} &= \frac{res_i}{t_i} \end{aligned}$$

### 10.6.9 Extended Results with Variance

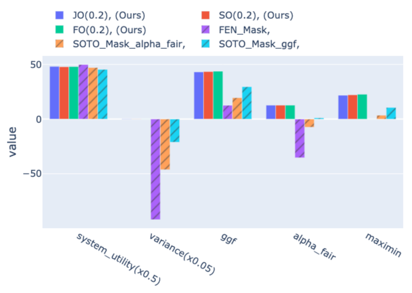
Here we provide additional results for our methods, including providing confidence intervals for our main results, additional results for generalization, and more comparison to baselines.

#### Confidence Intervals for Results

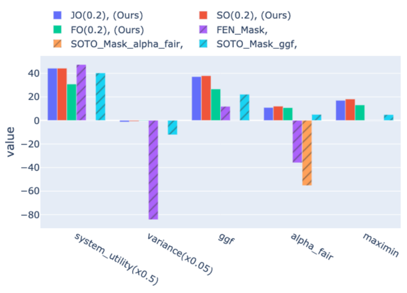
We provide confidence intervals for system utility and fairness separately as we vary $\beta$, as plotting this on a Pareto plot (as in the main results) would be hard to read (Figure 10.7). An interesting thing to note here is that we see a phase lag between when variance starts to reduce, and when utility starts to drop. This shows there is a range of solutions where fairness can be improved without significantly harming the utility.



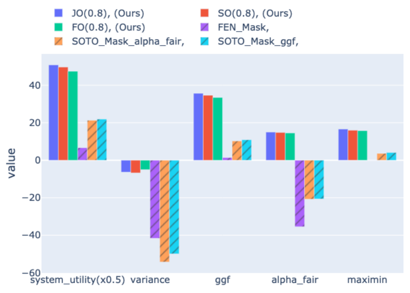
(a) Matthew



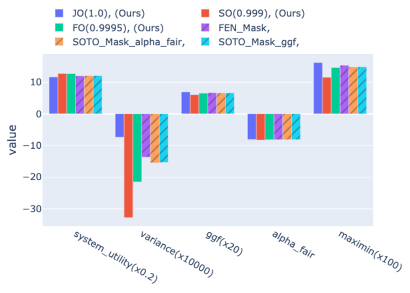
(b) JobAlloc



(c) Job



(d) Plant



(e) BiasedDM

Figure 10.6: Comparisons of selected DECAF models against the three baselines, scaled to fit on the same axes. We omit results for ILP versions of FEN and SOTO due to their poor performance. We selected DECAF models (trained on variance) that maximized $0.1U - 0.9\text{var}(\mathbf{Z})$. The numbers in brackets denote the selected $\beta$ value for our models.

### Approximating the Pareto Front

We also show generalization results by generating approximate Pareto fronts for all methods (Figure 10.8) based on a limited set of $\beta_{train}$ models, showing that the SO and FO methods generalize very well even without training for all intermediate $\beta$ values. One interesting thing to note here is that FO falls under the Pareto front with intermediate $\beta$ values, showing how a tuned utility model also helps in guiding agents towards better decisions.

### Comparison of Selected Models

Figure 10.6 shows the performance of selected DECAF models trained on variance when evaluated on different metrics on all environments, compared to the baselines. We can see DECAF models perform much better on all metrics. For BiasedDM, all methods were able to converge to approximately the same fairest policy, so the differences in the figure are

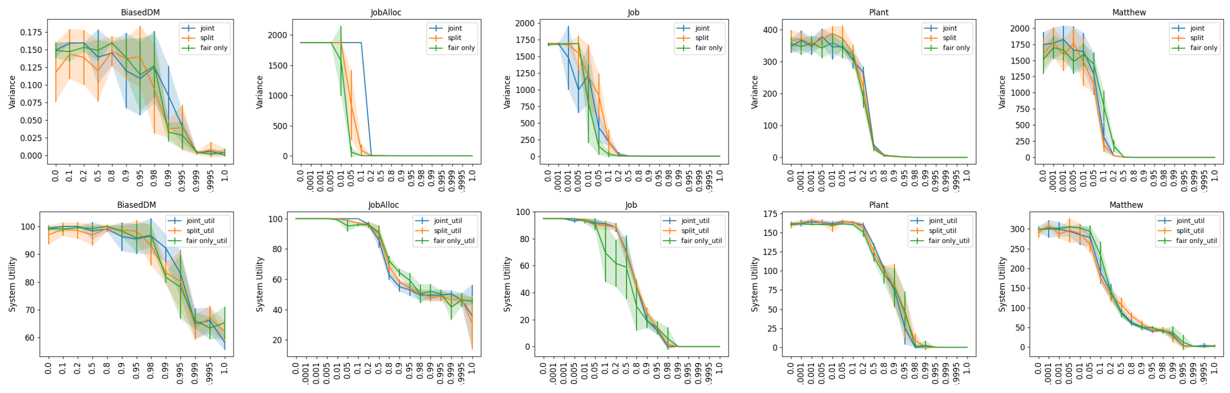


Figure 10.7: Effect of changing $\beta$ on variance (top row) and utility (bottom row) for all three methods on all five environments. The shaded area shows the 1-$\sigma$ error bar. We observe that the performance of FO has large variation for intermediate $\beta$ values.

small. The variance for SO might appear large, but we point out that this variance is scaled 10000 times, and the actual values are all very close to zero.

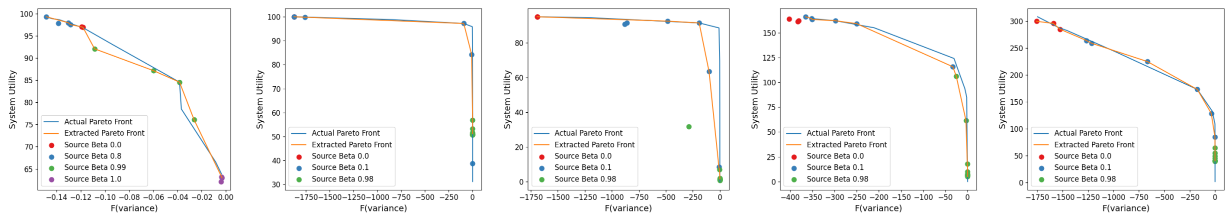
### Generalization Heatmaps

We also provide the generalization results for varying $\beta_{test}$ for both Split (Figure 10.9a) and Fair Only (Figure 10.9b) models for all environments. We note that the general trends noted in the main experimental results hold, with each model able to improve fairness as $\beta_{test}$ is increased. Further, in all cases, FO maximizes utility at $\beta_{test} = 0$, and has a sharper transition between utility-maximizing and fairness-maximizing behavior.

## 10.6.10 Limitations

Our work assumes a fully cooperative setting, where agents are self-interested, but still report truthful utilities, and the decision-maker is motivated by social welfare. In the presence of strategic agents, our approach might not yield the same results.

Our approach centers on an ILP optimization by the central decision-maker, which may not always be tractable. However, many resource allocation problems allow LP relaxations, and with certain assumptions, polynomial-time allocation algorithms like the Hungarian algorithm can be used instead. Additionally, message-passing and network based methods



(a) BiasedDM (b) JobAlloc (c) Job (d) Plant (e) Matthew

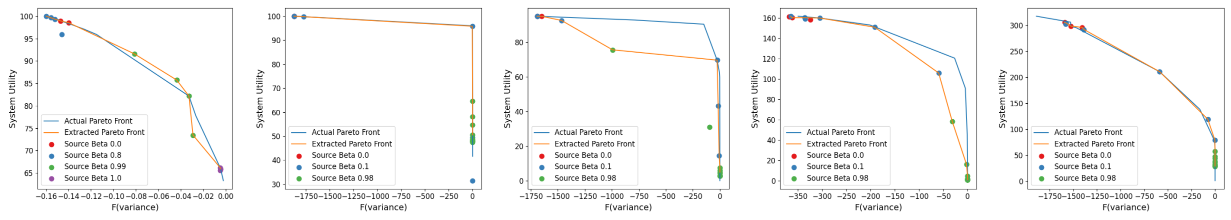


Figure 10.8: Approximate Pareto fronts for Split Optimization and Fair-Only Optimization models for variance across different environments. The top row represents SO models, while the bottom row represents FO models.

may also be used to arrive at a global solution in the absence of a central authority. We do not explore these avenues in our paper.

Our experiments use handcrafted domains based on prior work, which may not capture the full complexity of real-world systems. However, we show how existing approaches fail in all but the simplest tasks, while DECAF is able to generate strong fairness-efficiency tradeoffs even in complicated environments like Plant and Matthew.

## 10.7 Conclusion

We introduced DECAF, a framework for learning fair-efficient behavior in our newly formalized DECA model for multi-agent resource allocation. DECAF is among the first approaches to optimize fair resource allocation under resource constraints, supporting diverse problem settings by decoupling fairness and utility metrics. Split and Fair-Only optimizations enable online trade-offs between utility and fairness without retraining, enhancing interpretability. Our results demonstrate the flexibility and effectiveness of DECAF across various scenarios, while overcoming limitations of decentralized MARL approaches that rely on policy gradients. Our framework currently relies on Q-Learning, as deriving a policy gradient approach for DECA problems is challenging due to the dynamic state-action space and the indirect relationship between agent ‘policies’ and actions resulting from constrained action selection. Addressing this challenge is a promising direction for future research. Additionally, our fairness decomposition is modular; approaches like VDN [143] or QMIX [125] could be integrated to improve credit assignment for the fair reward, further strengthening our framework.

In this chapter, we developed an in-processing approach to improve fairness in multi-agent resource allocation with a centralized decision-maker. Next, we will explore the pre-processing stage. Departing from resource allocation domains, we instead look at data collection pipelines for continual training in a location prediction setting, casting the data acquisition problem itself as a repeated resource allocation problem with a fixed budget.

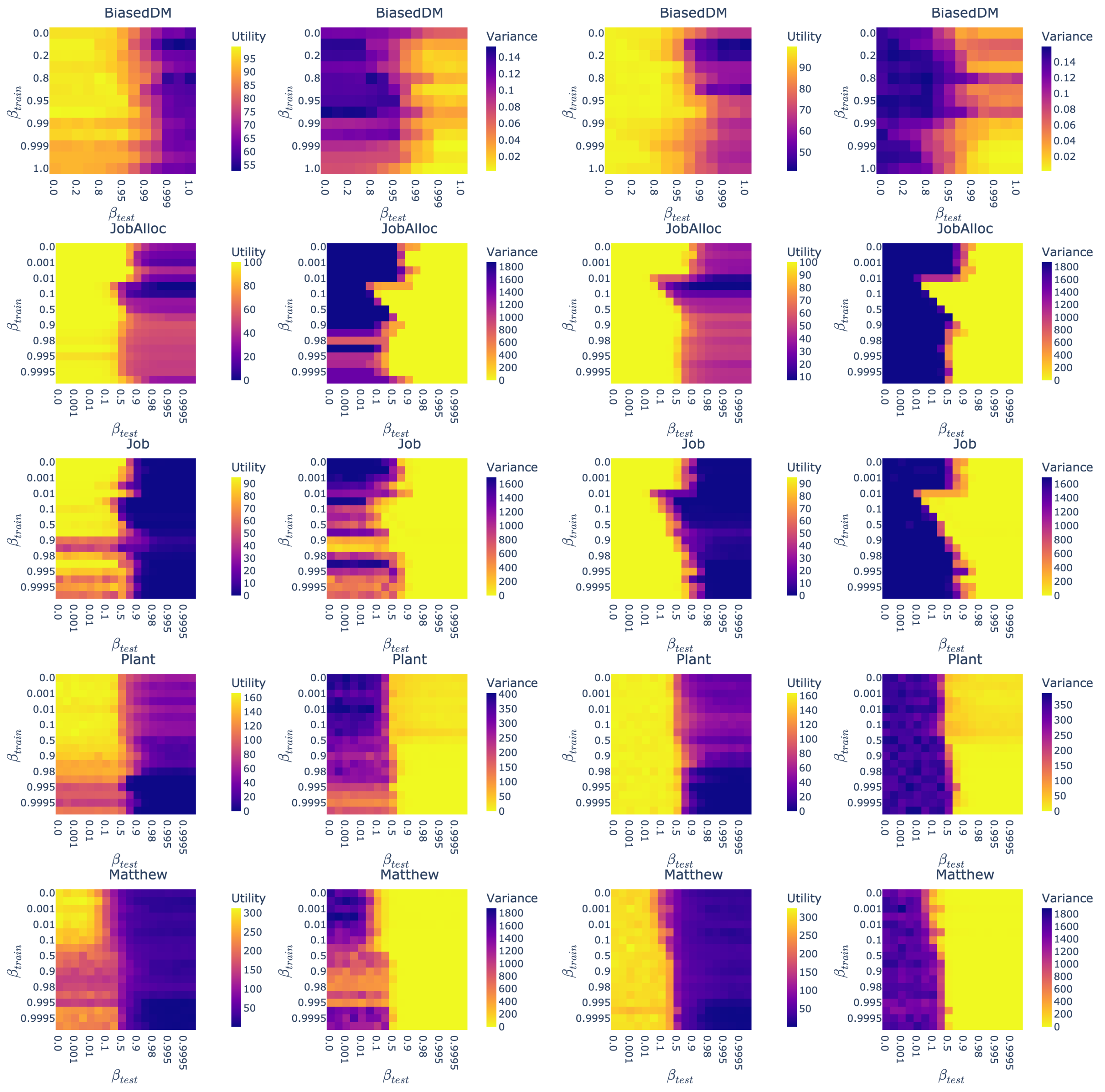

(a) SO models (b) FO models

Figure 10.9: Evaluation of Split Optimization (left) and Fair-Only Optimization (right) models trained on $\beta_{train}$ (for variance) and evaluated on $\beta_{test}$ across different environments. Brighter colors indicate better outcomes.

# Chapter 11

# Fairness in the Data Collection Stage: Location Prediction

## 11.1 Introduction & Contribution

The previous chapters covered post-processing and learning-based techniques for improving fairness in AI decision-making, specifically within the DECA framework. Now, we broaden our scope to examine fairness before learning: in the data collection stage of AI systems. This may be thought of as a "pre-processing" intervention, with a 'meta' DECA problem: given a limited budget for data acquisition, how can we select data points to maximize both accuracy and fairness in downstream models?

We conduct this investigation in the context of next-location prediction, a core task in spatio-temporal modeling with wide-ranging applications. Next-location prediction has become a central task in applications ranging from mobility planning and retail analytics to public health surveillance. By forecasting where individuals are likely to go, these models support downstream services such as route recommendation, targeted advertising, and resource allocation. However, while accuracy remains the dominant benchmark for evaluating model performance, little attention has been paid to how predictive quality is distributed across different segments of the population.

In this work, we examine the fairness implications of large-scale mobility prediction. Specifically, we conduct the first comprehensive audit of state-of-the-art next-location prediction models trained on real-world data from millions of users. Our findings reveal consistent disparities in predictive accuracy across racial and ethnic groups, with some groups systematically receiving less accurate predictions than others.

To support fairness analysis and intervention in the absence of individual-level demographic labels, we introduce a novel clustering algorithm called Size-Aware K-Means (SAKM). This method clusters users in latent mobility space while matching target group proportions derived from census data, yielding demographically grounded proxy labels. These clusters enable us to estimate group-level performance metrics and track disparities throughout training.

Building on this foundation, we propose Fairness-Guided Incremental Sampling (FGIS), a lightweight data acquisition strategy for improving equity in low-resource prediction settings. FGIS prioritizes users from underrepresented or underperforming groups during data collection, balancing fairness and accuracy via a tunable tradeoff parameter. Importantly, this intervention operates purely at the data level—requiring no access to user features or modifications to model architecture.

We evaluate our approach using two predictive models: a graph-based MetaPath2Vec model for statewide fairness auditing, and an additional transformer encoder model for controlled intervention experiments in a representative subregion (Tarrant County, Texas, United States). Our results show that FGIS can reduce group disparities by over 40% in early training stages with minimal impact on final accuracy. These gains are especially pronounced in low-data regimes, highlighting the value of fairness-aware sampling when data is limited or expensive to collect.

**Contributions:** The main contributions of this chapter are:

1. **Audit at scale:** First large-scale fairness audit on a SOTA location prediction model with 4.9 million users, uncovering up to 15% accuracy difference between groups.
2. **SAKM proxy labels:** A novel size-constrained $k$-Means variant that enforces arbitrary census-derived cluster sizes, enabling demographic fairness evaluation without individual attributes.
3. **FGIS sampling:** A plug-and-play batch sampling algorithm that over-samples underperforming groups, reducing early-stage Total Demographic Parity Violations (TDPV) by over 40% with minimal overhead.

4. **Empirical validation:** Demonstrated across two architectures (MetaPath2Vec and Transformer) and multiple geographies, achieving equity gains with under 1% long-term accuracy trade-off.

## 11.2 Background

### 11.2.1 Related Work

**Mobility and POI Prediction:** Recent studies have demonstrated the value of deep learning for point-of-interest (POI) prediction tasks. Transformer-based models have shown strong performance when enriched with auxiliary information, such as travel mode, which helps improve next-location forecasting [62]. Other work has focused on the role of routine detection in understanding user behavior, with findings suggesting that consistent travel patterns can inform customer relationship strategies in ridesharing platforms [31]. To alleviate the challenge of limited real-world data, synthetic datasets such as SynMob have been proposed, offering realistic GPS trajectories for robust model training [175]. Additionally, personalized destination prediction has been explored in contextless settings using transformer models trained on partial trajectories [150]. We refer readers to a recent review by Graser et al. [55] on the use of trajectory data for prediction tasks. Zhang et al. [169] present a recent approach to POI prediction, using a colocation network to identify and use similarities between users to more accurately determine visitation patterns. We will use the model from this paper for the bulk of our analyses.

**Algorithmic Fairness:** Research on algorithmic fairness has produced a rich taxonomy of group- and individual-level criteria. Group fairness notions quantify disparities in aggregate error or allocation rates across protected groups, with *Demographic Parity* (also called Statistical Parity) being one of the earliest and most widely adopted definitions. A predictor satisfies demographic parity when its positive-prediction rate is identical for every group [35]. Follow-up work proposed alternative group metrics—most notably *equalized odds* and *equality of opportunity*, which require parity of error rates or true-positive rates conditional on the ground truth [58]. Complementary research advocates *individual fairness*, urging that "similar individuals be treated similarly," though this is difficult to enforce when similarity measures are ill-defined.

Fairness goals are shaped by modeling assumptions and data limitations [105, 11]. Demographic parity remains appealing in settings like mobility prediction, where ground-truth labels are limited, as it depends only on observed prediction disparities. We use demographic parity as a diagnostic tool, adapting it to the multi-group setting.

**Fairness in Active Learning:** Fair active learning aims to select data points for labeling in a way that enhances model fairness. Early work in this area focused on balancing label acquisition to satisfy group fairness constraints in supervised settings [136]. More recent studies have extended this to streaming scenarios [160] and developed sampling methods that promote fairness without requiring group labels during training [117]. Our setting differs significantly, as we treat it as a data acquisition problem, rather than a labeling problem. We do not assume access to user features (e.g., location traces) at selection time. Instead, we assume a pool of users, each associated with a known group label, and develop a strategy for sampling users in a way that promotes fairness in data acquisition. Our approach is feature-agnostic and focuses on balancing data representation across groups to improve model equity.

**Fairness in Mobility Prediction:** Fairness concerns have only recently reached spatio-temporal modeling. Early efforts focused on equitable demand prediction for ride-hailing and public-transit systems [166, 80, 174]. Beyond demand forecasting and ridehailing, POI recommendation studies have proposed fairness metrics to ensure balanced exposure of venues or user segments. For instance, Weydemann et al. [161] introduced utility- and diversity-based fairness criteria to mitigate popularity bias and ensure equitable treatment across user demographics. More recent work has begun to examine whether predictive systems systematically underperform for marginalized communities [173, 171], highlighting the need for algorithmic interventions that can reduce error disparities without compromising accuracy.

**Summary:** Our paper differs from prior mobility fairness research in two key ways. First, we audit *individual-level next-location prediction*, in contrast to previous work that focuses on region-level demand or recommendation outcomes. This allows us to reveal disparities that persist even over short horizons and individual trajectories. Second, instead of applying fairness regularizers or post-hoc adjustments, we propose a *data acquisition* strategy that improves demographic parity with minimal impact on predictive performance—complementing model-side fairness interventions in the literature.

**Size-based Clustering:** Existing attempts to control cluster sizes either force *equal* cardinality or impose only broad capacity limits, leaving no practical tool for matching an arbitrary quota vector $\boldsymbol{\pi}$. Constrained and "balanced" $k$-means variants guarantee $|C_1| = \cdots = |C_k|$ via repeated Hungarian assignment [13, 98], while flow-based approaches minimize a cost plus a penalty for deviation from *uniform* sizes [90]. Capacity-constrained $k$-means from operations research merely caps the load per cluster [51], and exact-quota mixed-integer or conic formulations scale only to a few thousand points [129]. In contrast, our **Size-Aware $k$-Means** keeps Lloyd-style updates but adds dual Lagrange steps, efficiently steering each cluster toward *any* prescribed proportion vector.

### 11.2.2 Prediction Task

Our paper focuses on evaluating the fairness of next-location prediction models trained on large-scale mobility data. The underlying task is to forecast where a user will go next based on their historical movement patterns. Formally, each user's trajectory is represented as a time-ordered sequence of visits to points of interest (POIs), where each visit is encoded as a POI identifier and a timestamp.

Given such a sequence, the model is trained to predict the user's next POI. This task is inherently challenging due to the wide variability in user behavior, the heterogeneity of POIs across regions, and the complex temporal dynamics of human mobility. Moreover, mobility patterns are shaped by socioeconomic and spatial factors, which may result in uneven model performance across different user groups.

To evaluate predictive performance, we use the **top-$k$ accuracy metric** over a fixed future time window. In our analysis, we adopt a **one-week lookahead period**, and report the **1-week Acc@20** metric. This measures the fraction of test instances (users) for which at least one of the user's actual future POI visits during the next week appears in the top-20 predictions generated by the model.

This formulation allows us to quantify how well the model anticipates user behavior at a practically meaningful granularity. Importantly, it also enables disaggregated evaluation by demographic group, allowing us to assess whether certain populations systematically experience lower prediction accuracy, which forms the basis of our fairness audit.

### 11.2.3 Disparity Measurement

We consider group fairness in our evaluations, where we use a modified version of demographic parity [35, 58] called *Total Demographic Parity Violations (TDPV)*:

$$\text{TDPV} = \sum_{i<j} \left| z_{g_i} - z_{g_j} \right| \tag{11.1}$$

Here, $z_g$ denotes the prediction accuracy for group $g$, measured as the average top-$k$ prediction accuracy in that group. This metric captures the total disparity in performance across demographic groups by summing the absolute pairwise differences in accuracy. A lower TDPV indicates more equitable accuracy distribution, while a higher TDPV signals that some groups experience significantly better or worse performance than others. The metric is symmetric and unweighted, treating all group pairs equally regardless of their population sizes.

## 11.3 Fairness Analysis

We aim to determine whether next-location prediction models trained on large-scale mobility data exhibit systematic disparities in performance across racial and ethnic groups. Our audit examines whether unequal prediction accuracy may arise from spatial bias, imbalanced data coverage, or overfitting to overrepresented populations. We describe the various components of our analysis below.

### 11.3.1 Model: MetaPath2Vec

To evaluate fairness in a realistic predictive setting, we use the MetaPath2Vec model [169], a state-of-the-art approach for next-location prediction in heterogeneous networks. The model operates over a bipartite user-POI graph, where nodes represent users and POIs, and edges represent observed visits.

MetaPath2Vec performs random walks guided by predefined meta-path schemas (e.g., user-POI-user) to generate sequences of nodes, which are used in a skip-gram objective [34] to learn latent embeddings. This enables the model to position each user close to visited and structurally similar POIs in the embedding space. These embeddings are then used to predict the next POI a user is likely to visit.

### 11.3.2 Texas Mobility Dataset

We evaluate the model on mobility data collected from mobile devices detected in Texas between January 1 and April 15, 2021. This dataset was sourced from a third-party location analytics firm and collected passively via mobile applications, depending on user permissions. The raw dataset contains data point including a hashed device ID, timestamp, and GPS coordinates. Pings matched to known POIs also include metadata such as name, category, and address.

From this raw stream, we constructed anonymized trajectories for approximately 4.9 million users, comprising ordered sequences of visits to over 530,000 unique POIs. Each user trajectory varies in length from 1 to 2,625 visits, with an average of 67. Home locations are inferred using nighttime GPS activity and assigned to ZIP Code Tabulation Areas (ZCTAs); no exact coordinates or personal identifiers are retained.

We use results from a pre-trained MetaPath2Vec model trained on user data from this anonymized Texas dataset in our analysis.[11]

### 11.3.3 Demographic Data and Motivation

Since individual demographic attributes are not available in the dataset, we infer coarse group membership using publicly available census statistics. This is essential for evaluating whether the model yields unequal outcomes across demographic groups despite being trained without such labels.

[11]The processed trajectory dataset and the trained model's results were obtained with permission from the original authors [169]. We are unable to share the dataset publicly due to constraints imposed by the original authors.

Table 11.1: Texas Population by Race and Ethnicity

| **Demographic Group** | **Percentage (%)** |
|---|---|
| Hispanic or Latino (any race) | 39.8 |
| White alone (not Hispanic) | 38.7 |
| Black or African American alone | 12.0 |
| Asian alone | 5.6 |

We focus on the four largest racial and ethnic groups in Texas—Hispanic or Latino, White (non-Hispanic), Black or African American (non-Hispanic), and Asian (non-Hispanic)—which together account for 96.1% of the population [153] (see Table 11.1).

Using census-reported racial composition at the ZCTA level, we estimate a probability distribution over each user's group membership based on their home ZCTA. We also perform complementary analysis at the county level, using the same inference procedure. We compute fairness metrics at both ZCTA and county resolution to examine whether disparities persist across geographic scales. These inferred probabilities are used solely in aggregate form to evaluate fairness, and are never treated as ground-truth labels.

### 11.3.4 Evaluation Method

To estimate group-level disparities in predictive performance, we compute top-20 prediction accuracy within a one-week lookahead window (1-week Acc@20). A prediction is considered correct if any POI visited in the next week appears in the model's top-20 ranked list.

Because we lack individual-level race/ethnicity labels, we adopt two common assumptions:

1. **Geographic representativeness:** Users in each region (ZCTA or county) are treated as a random sample of that region's population.
2. **Intra-region uniformity:** Prediction accuracy is assumed constant across demographic groups within a region.

**Group-Level Accuracy ($z_g$) Computation.**

To estimate prediction accuracy for each demographic group, we aggregate model performance over geographic regions using census-based priors. For each region $r$, we compute the average top-20 prediction accuracy $a_r$ based on all users assigned to that region.

Let $n_r$ denote the number of users in region $r$ in the dataset, and let $p_{g,r}$ be the proportion of group $g$ in that region according to census data. We estimate the number of correct predictions attributable to group $g$ in region $r$ as:

$$c_{g,r} = a_r \cdot n_r \cdot p_{g,r} \tag{11.2}$$

The total number of correct predictions and total population for group $g$ are:

$$C_g = \sum_r c_{g,r} = \sum_r a_r \cdot n_r \cdot p_{g,r} \tag{11.3}$$

$$N_g = \sum_r n_r \cdot p_{g,r} \tag{11.4}$$

The group-level accuracy is then defined as:

$$z_g = \frac{C_g}{N_g} \tag{11.5}$$

This value represents the expected prediction accuracy experienced by a typical member of group $g$, assuming geographically uniform accuracy within each region. The resulting $z_g$ values are used to compute fairness metrics throughout our analysis.

## 11.4 Fairness Audit: Observed Disparities

Using the trained MetaPath2Vec model, we evaluate next-location predictions for all users based on the 1-week Acc@20 metric. To examine variation in model performance, we assess disparities along two dimensions: geography and race/ethnicity.

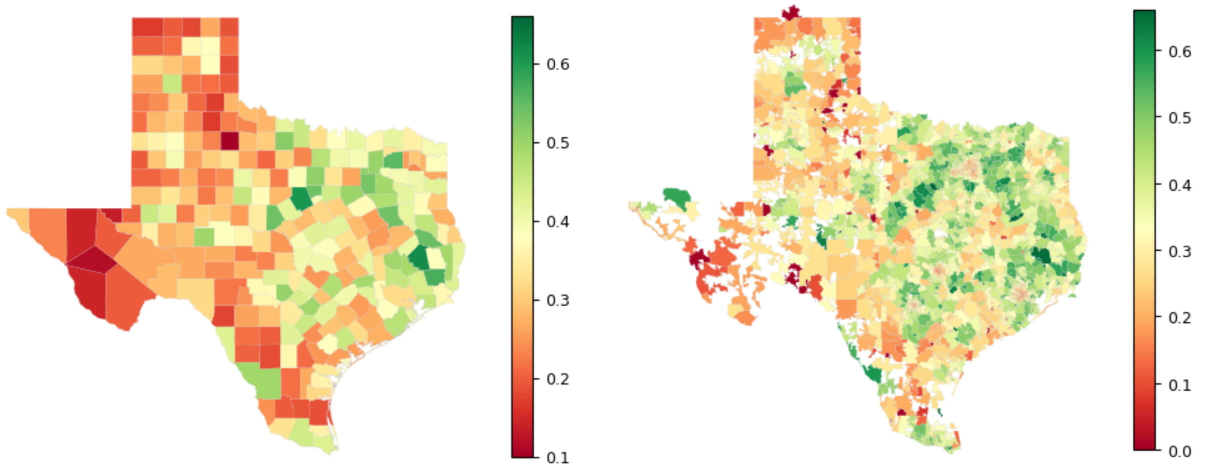


Figure 11.1: Top-20 prediction accuracy over a one-week lookahead period across Texas counties and ZCTAs.

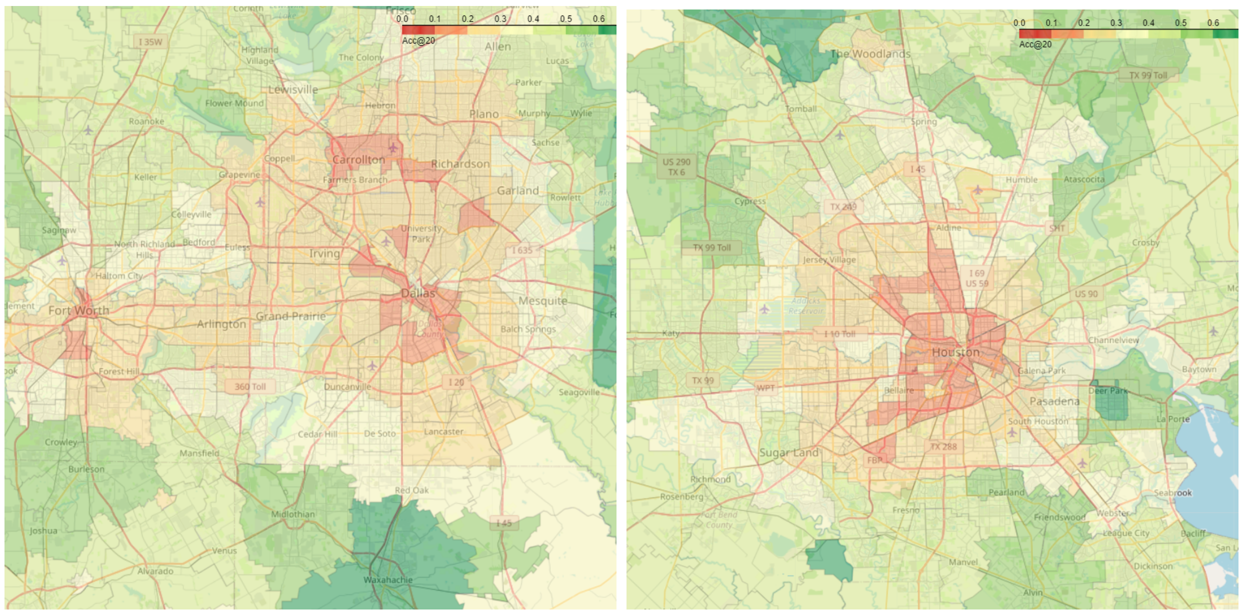


Figure 11.2: Differences in urban vs. suburban prediction accuracy in Dallas (left) and Houston (right).

### 11.4.1 Geographic Disparities

To assess spatial variation, we compute the average prediction accuracy at both the county and ZIP Code Tabulation Area (ZCTA) levels, based on each user's inferred home region. The geographic distribution of accuracy is shown in Figure 11.1.

We observe that prediction accuracy tends to be higher in northeastern Texas, particularly in areas with greater population density and more abundant data coverage. In contrast, western Texas, which is more sparsely populated, generally exhibits lower predictive performance.

However, the relationship between population density and model accuracy is not strictly monotonic. In major metropolitan areas such as Dallas and Houston (Figure 11.2), suburban regions often achieve higher accuracy than central urban zones. We attribute this to the increased difficulty of predicting mobility in dense urban areas with a high concentration of nearby POIs, which introduces greater ambiguity despite larger data volumes.

### 11.4.2 Racial and Ethnic Disparities

We next assess disparities in predictive performance across racial and ethnic groups. Using the group-level estimation procedure described earlier, we compute the expected accuracy experienced by a typical member of each demographic group. Table 11.2 reports the mean Acc@20 by group at both the ZCTA and county levels. In addition, Figure 11.3 presents kernel density estimates of the accuracy distributions across the population of each group, showing the density of users at each accuracy level.

Our analysis reveals that White users are expected to experience the highest accuracy, followed by Hispanic, Asian, and Black users. This disparity is more pronounced at the ZCTA level, while county-level aggregation tends to smooth out local differences. Nevertheless, the observed gap in performance across groups persists at both geographic resolutions, suggesting systemic disparities in the model's predictive behavior. These disparities may reflect uneven data distribution, differential mobility patterns, or structural biases learned during training.

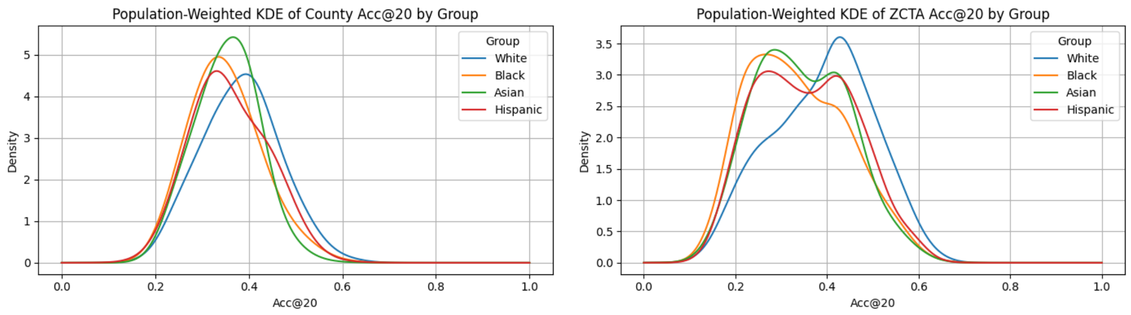


Figure 11.3: Kernel density estimates of group-level prediction accuracy distributions. Note: density reflects intra-group distribution, not relative group sizes.

Table 11.2: Mean Acc@20 by Group and Region, including Total Demographic Parity Violations (TDPV)

| **Group** | **ZCTA** | **County** |
|---|---|---|
| White | **0.390** | **0.383** |
| Hispanic | 0.355 | 0.359 |
| Asian | 0.346 | 0.353 |
| Black | 0.335 | 0.351 |
| **TDPV** | 0.174 | 0.102 |

## 11.5 Group-Aware Incremental Sampling

Having identified a clear bias in the model predictions, we look at a solution to mitigate disparities in next-location prediction. We consider a practical setting in which model developers acquire mobility data over time, subject to budget constraints. Suppose there exists a large population of potential users, each with a hidden trajectory history, from which a learning agent incrementally samples training data. Our goal is to actively guide this sampling process to improve fairness across demographic groups, by constructing training sets that yield more equitable predictive performance.

This strategy requires some notion of group membership for each user. While individual-level demographic attributes are not observed, we assume access to coarse-grained proxy labels based on the user's home region. These proxies are inferred using publicly available census data and are used solely in aggregate form to steer the sampling process.

Let $\mathcal{U}$ denote the full user population and let $\mathcal{D}_t \subset \mathcal{U}$ be the training set at iteration $t$, initialized as $\mathcal{D}_0 = \emptyset$. At each round $t = 1, \ldots, n$, the agent selects a batch of $B$ users from

**Algorithm 11.1:** SAKM: Size-Aware K-Means with Permutation Search (Main)

```
Input     : Data $X \in \mathbb{R}^{N \times d}$, target proportions $\boldsymbol{\pi}$, clusters $k$
Parameter: Step-size $\eta$, tolerance $\tau$, max iterations $T$, restarts $n_{\text{init}}$
Output    : Best cluster assignments $\mathbf{z}$ and centroids $\{\mu_j\}$
1 $\mathcal{L}_{\text{best}} \leftarrow \infty$
2 for $s = 1$ to $n_{init}$ do
3     Initialize centroids $\{\mu_1, \ldots, \mu_k\}$
4     for each permutation $\boldsymbol{\pi}'$ of $\boldsymbol{\pi}$ do
5         $\mathbf{z}, \{\mu_j\}, \mathcal{L} \leftarrow$ RunSAKMInnerLoop$(X, \{\mu_j\}, \boldsymbol{\pi}', \eta, \tau, T)$
6         if $\mathcal{L} < \mathcal{L}_{best}$ then
7             Save $\mathbf{z}, \{\mu_j\}, \mathcal{L}_{\text{best}} \leftarrow \mathcal{L}$
8 return best $\mathbf{z}, \{\mu_j\}$
```

$\mathcal{U} \setminus \mathcal{D}_{t-1}$, optionally conditioned on their (inferred) group, and adds them to the dataset: $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \mathcal{S}_t$, where $|\mathcal{S}_t| = B$. A predictive model $\mathcal{M}_t$ is trained on $\mathcal{D}_t$ and evaluated using top-$k$ accuracy. For each group $g$, we compute the group-specific accuracy $z_g^{(t)} = \text{Acc}_k(\mathcal{M}_t \mid g)$. Note that in this model, the full set of user features is only acquired after we decide to sample them.

We aim to guide the sequence of samples $\{\mathcal{S}_t\}_{t=1}^{n}$ to reduce disparity across the $z_g^{(t)}$ metrics, by adaptively prioritizing users from groups that are underrepresented or underperforming. To support this group-aware sampling, we require user-level group assignments that reflect the demographic makeup of the population. Since individual attributes are not observed, we turn to unsupervised clustering to assign users to demographic groups in a principled way. Our goal is to generate proxy labels that align with census-reported racial proportions at the regional level. This leads us to a constrained clustering approach that incorporates group size targets into the clustering objective, which we describe next.

### 11.5.1 Size-Aware K-Means for Proxy Demographic Grounding

Our fairness-aware sampling strategy assumes access to group membership labels, but individual-level demographics are not observed in our dataset. To address this, we construct *proxy demographic labels* based on available census priors at the ZCTA level. These priors

**Algorithm 11.2:** RunSAKMInnerLoop: Assignment and Multiplier Updates

```
Input  : Data $X$, initial centroids $\{\mu_j\}$, target proportions $\boldsymbol{\pi}'$, step-size $\eta$, tolerance $\tau$,
         iterations $T$
Output: Final assignments $\mathbf{z}$, centroids $\{\mu_j\}$, objective $\mathcal{L}$
1  Initialize $\lambda_j \leftarrow 0$ for all $j$
2  for $t = 1$ to $T$ do
3      for each user $x_i$ do
4          Assign $z_i \leftarrow \arg\min_j \|x_i - \mu_j\|^2 + \lambda_j$
5      for each cluster $j$ do
6          Update $\mu_j \leftarrow$ mean of $\{x_i : z_i = j\}$
7      for each cluster $j$ do
8          Let $n_j \leftarrow \#\{i : z_i = j\}$
9          $\lambda_j \leftarrow \lambda_j + \eta \cdot \left(\frac{n_j}{N} - \pi'_j\right)$
10     if centroids shift $< \tau$ then
11         break
12 Compute $\mathcal{L} \leftarrow \sum_i \|x_i - \mu_{z_i}\|^2$
13 return $\mathbf{z}, \{\mu_j\}, \mathcal{L}$
```

give us coarse estimates of racial group proportions, which we use to guide an unsupervised clustering process.

Rather than assigning users to groups arbitrarily or uniformly, we seek a principled partitioning that reflects the demographic composition of the local population. To this end, we embed users into a latent space derived from their mobility trajectories and cluster them into $k$ groups, one per racial category. While these clusters may not align exactly with true group identities, they offer a structure that is both data-driven and demographically grounded.

**Algorithm**

We implement a modified clustering algorithm, *Size-Aware K-Means (SAKM)*, which extends standard $k$-means to enforce user-defined cluster size constraints. Given a target group distribution $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_k)$, SAKM aims to produce clusters of sizes close to $\pi_g \cdot N$, where $N$ is the total number of users.

SAKM introduces a Lagrangian penalty in the assignment step, so the cost of assigning a point $x_i$ to cluster $g$ becomes:

$$\text{cost}(x_i, g) = \|x_i - \mu_g\|^2 + \lambda_g \tag{11.6}$$

where $\mu_g$ is the centroid of cluster $g$ and $\lambda_g$ is a Lagrange multiplier that penalizes deviations from the target size. These multipliers are updated iteratively as:

$$\lambda_g \leftarrow \lambda_g + \eta \cdot \left(\frac{n_g}{N} - \pi_g\right) \tag{11.7}$$

where $n_g$ is the current number of points in cluster $g$. To resolve the label ambiguity and improve convergence quality, we run the optimization over all permutations of the target proportions $\boldsymbol{\pi}$ and select the clustering with the lowest objective (inertia). This permutation search is motivated by a key challenge in centroid initialization: if a randomly initialized centroid is far from the true region of the corresponding size, the resulting cluster may fail to attract the intended mass of points. By evaluating all $k!$ permutations of the size targets, SAKM increases the likelihood that size constraints are matched with semantically meaningful partitions.

While the worst-case complexity grows factorially with the number of groups, this remains tractable in our setting where $k = 4$. For larger $k$, the cost could be reduced using combinatorial assignment methods such as the Hungarian algorithm with $\mathcal{O}(k^3)$ complexity, which we explored but do not report in this paper.

The full algorithm is presented in Algorithms 11.1 and 11.2.

### 11.5.2 FGIS: Fairness-Guided Sampling Strategy

Given the proxy group assignments produced by SAKM, we now seek to actively construct training datasets that reduce disparities in model performance across groups. At each iteration, we select new users from the population based on the expected impact their group membership will have on fairness outcomes.

**Algorithm 11.3:** Fairness-Guided Incremental Sampling Loop

```
Input      : Full user set $\mathcal{U}$, proxy group labels $g(u) \in \{1, \dots, G\}$, batch size $B$,
             rounds $n$
Parameter: Sampling trade-off $\beta$, initial accuracy estimate $z_g^{(0)} = 0.1$
Output     : Accuracy metrics $\{z_g^{(t)}\}$ for $t = 1 \dots n$
1 Initialize dataset $\mathcal{D}_0 \leftarrow \emptyset$; seen user counts $x_g \leftarrow 0$ for all $g$
2 Set accuracy estimates $z_g^{(0)} \leftarrow 0.1$ for all $g$
3 for $t = 1$ to $n$ do
4     $\mathcal{S}_t \leftarrow \textsc{Sample}(\mathcal{U} \setminus \mathcal{D}_{t-1}, \{x_g\}, \{z_g^{(t-1)}\}, B, \beta)$
5     $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \mathcal{S}_t$
6     Train model $\mathcal{M}_t$ on $\mathcal{D}_t$
7     Evaluate $\mathcal{M}_t$ to obtain group accuracies $\{z_g^{(t)}\}$
8     Update $x_g \leftarrow x_g + \#\{u \in \mathcal{S}_t : g(u) = g\}$
9 return accuracy metrics $\{z_g^{(t)}\}$
```

Our strategy for *Fairness Guided Incremental Sampling (FGIS)* is based on the following intuition: additional data improves group-level accuracy $z_g$, but with diminishing returns—each new user contributes less than the last. Moreover, not all groups benefit the same from additional samples: improving underrepresented or underperforming groups will offer greater marginal gains in fairness. We therefore design a sampling rule that assigns higher weight to groups expected to most improve performance parity, by increasing the weights of groups with lower data representation and lower accuracy.

Recall that $z_g$ denotes the top-$k$ accuracy for group $g$ under the current model, and let $x_g$ denote the number of users from group $g$ currently included in the training set. We define the sampling weight for group $g$ as:

$$w_g \propto [z_g \cdot (x_g + 1)]^{-\beta} \tag{11.8}$$

where $\beta \in [0, \infty)$ is a tunable parameter controlling the trade-off between uniform sampling ($\beta = 0$) and fairness-aware sampling ($\beta > 0$). This form reflects a first-order approximation of the expected fairness gain from sampling group $g$: groups with low accuracy $z_g$ and few seen users $x_g$ are prioritized, while groups that already perform well or have large training representation are de-emphasized.

**Algorithm 11.4:** SAMPLE: Fairness-Aware Batch Selection

```
Input  : Candidate users $\mathcal{C}$ with group labels $g(u)$,
         Group counts $\{x_g\}$, group accuracies $\{z_g\}$, batch size $B$, mini-batch size
         $m$, trade-off $\beta$
Output: Sampled batch $\mathcal{S}_t$ of $B$ users
1 Initialize sampled set $\mathcal{S}_t \leftarrow \emptyset$
2 while $|\mathcal{S}_t| < B$ do
3     Compute group weights $w_g \propto [z_g \cdot (x_g + 1)]^{-\beta}$
4     Normalize $\{w_g\}$ to form group-level distribution $p_g$
5     Assign each user $u \in \mathcal{C} \setminus \mathcal{S}_t$ probability $p_{g(u)}$
6     Sample $m$ users from $\mathcal{C} \setminus \mathcal{S}_t$ using these probabilities
7     Add sampled users to $\mathcal{S}_t$; update counts $x_g$ accordingly
8 return sampled users $\mathcal{S}_t$
```

Ideally, we would sample a single user, retrain the model, and update the group accuracy estimates before sampling the next. This would allow the weights $w_g$ to reflect the most up-to-date performance information. However, retraining after every user is computationally prohibitive. As a first simplification, we instead train the model once per batch of $B$ users.

Even within a single batch, the weights $w_g$ depend on $x_g$, which changes as users are added to the training set. To sample accurately under this dependency, we would need to recompute $w_g$ after each individual selection. As a second simplification, we instead sample users in mini-batches of size $m$, updating $x_g$ and $w_g$ after each mini-batch rather than after each user. In practice, we find that this approximation performs comparably when $m$ is small. We use $m = 50$ and $B = 1000$ in our experiments.

To implement this, we maintain per-group accuracy estimates and user counts over sampling iterations. At each sampling step, we compute group weights using the formula above, map weights to per-user probabilities, and select a small mini-batch of users to add to the training set. The procedure repeats until the batch budget is exhausted. We use log-domain computation to maintain numerical stability and re-normalize probabilities after each mini-batch. Algorithms 11.3 and 11.4 outline this process.

This iterative, mini-batch design ensures that sampling remains responsive to updated accuracy estimates as the model improves. The resulting datasets reflect a data-efficient path toward performance parity, trading off global representativeness for reduced inter-group variance in a controlled manner.

## 11.6 Experimental Setup

To evaluate our proposed approach, we train models on the location dataset with a variety of $\beta$ values, showing how this parameter changes the utility-fairness tradeoff. Since retraining on the entire Texas dataset for multiple $\beta$ and getting confidence intervals is prohibitive, we select **Tarrant County** as a representative region within Texas. Further, to evaluate the effectiveness of our approach beyond the MetaPath2Vec model, and to also measure the relative strength of MetaPath2Vec, we additionally train a transformer encoder based model for location prediction, based on work on using transformers for mobility prediction [62]. The selection criteria for Tarrant County and model details, hyperparameters, and other details about the experimental setup are described next.

### 11.6.1 Metapath2Vec

We adapt the MetaPath2Vec architecture introduced by [34] and follow the evaluation framework of [169], which leverages user–POI visitations and user-user colocations to construct heterogeneous networks for predicting consumer visits. In our implementation, we construct a user-POI visitation network, where nodes represent users or POIs, and edges indicate observed visits. We then generate Meta-path-guided random walks (e.g., user–POI) sequences to capture structural and semantic proximity, enabling the model to learn user and POI embeddings ($d_{users}$ and $d_{pois}$) via a skip-gram model. We then compute similarities between POIs using Euclidean or cosine distance in the embedding space.

For each user in the test set, we compute the $k$ closest POIs based on embedding distances. These $k$ POIs are treated as predicted next visits. A hit is recorded if the user visits at least one of the predicted POIs during the first week of the holdout period. We evaluate hit rate using $k = 20$.

Table 11.3: MetaPath2Vec Model Settings and Hyperparameters

| **Setting** | **Value** |
|---|---|
| Walks per Node | 10 |
| Walk Length | 100 |
| Embedding Dimension ($d$) | 128 |
| Neighborhood Size | 7 |
| Negative Samples | 5 |

### 11.6.2 Transformer Encoder

Transformer architectures are widely used in sequence modeling and represent a strong alternative for mobility prediction. In particular, we adapt a transformer-based architecture introduced by Hong et al. [62], originally designed to improve next-location prediction by jointly modeling travel sequences and travel modes. Their model employs a transformer encoder to capture spatiotemporal dependencies in mobility histories, with an auxiliary head trained to predict the next travel mode alongside the next location. In our implementation, we retain the core sequence modeling architecture but omit the auxiliary travel mode prediction to reduce complexity and accommodate datasets without modality labels. We add periodic embeddings for the time to next location during training, and compute user embeddings (of size $d_{user}$) using the approximate home location coordinates. For the transformer encoder, the inputs are sequences of POIs and their timestamps. The embedding layer (of size $d_{base}$) is computed by embedding the POI IDs and adding a time-of-day embedding, before applying sinusoidal position encoding. The next POI prediction is computed by concatenating the user embedding, time-to-next embedding and sequence embedding (from the transformer encoder), and computing logits over all possible POIs after a feedforward layer. The resulting probability distribution gives us the top-k POI prediction, which is then used to compute downstream metrics.

Table 11.4 shows the configuration for the transformer model used.

| **Setting** | **Value** | **Parameter** | **Value** |
|---|---|---|---|
| LR | 0.001 | # Layers | 4 |
| LR decay | 1e-6 | # Heads | 4 |
| LR Warmup | 2 | Feedforward | 512 |
| ES Patience | 2 | Base Emb. ($d_{\text{base}}$) | 256 |
| ES LR drop | 0.33 | User Emb. ($d_{\text{user}}$) | 8 |
| | | FC Dropout | 0.1 |

Table 11.4: Transformer Model Settings and Hyperparameters

### 11.6.3 Tarrant County Subset

Our experiments require training and evaluating models many times—both to probe the fairness–accuracy tradeoff under different sampling conditions and to compute confidence intervals via bootstrapping. To keep these repeated runs tractable, especially for the Transformer model, we restrict our analysis to a subset of the Texas dataset focused on a single county.

We select this county for our case study using a simple two-step filter. First, we compute a population-weighted disparity score for each county, defined as the product of inter-group accuracy variance and county population under the original state-level MetaPath2Vec model. This score highlights regions where predictive disparities are both large and demographically consequential. Second, we limit to counties with fewer than 200,000 total trajectories to ensure practical training costs.

Tarrant County emerged as the top candidate, exhibiting significant fairness gaps under the statewide MetaPath2Vec model while remaining computationally manageable for repeated analysis. After removing users and POIs outside of Tarrant County, the final filtered dataset contained 170,000 user trajectories and 39,000 unique POIs.

### 11.6.4 Implementation Details

All experiments were conducted on a shared university compute cluster. Each run was allocated a single GPU with 8GB of VRAM and 32GB of system RAM. Each training iteration sampled 1,000 users, and the models were trained over 10 such iterations, yielding a total of 10,000 unique sampled users per experiment. After sampling and adding each new batch, a new model was initialized and trained from scratch. A full experiment consisting of 10 iterations typically completed within 24 hours. To assess statistical significance and reduce variance, we repeat each configuration with 10 random seeds and report aggregate metrics. The transformer model for predicting next location for Tarrant county had 22.9M trainable parameters.

The underlying dataset spans the period from January 1 to April 15, 2021. The training split spans the period from January 1 to March 15, and models were evaluated on a held-out test period from March 15 to March 22 (one week). For the transformer model, we further divided the training set into two parts for training and validation: the model was trained on data from January 1 to March 1 and validated on sequences from March 1 to March 15.

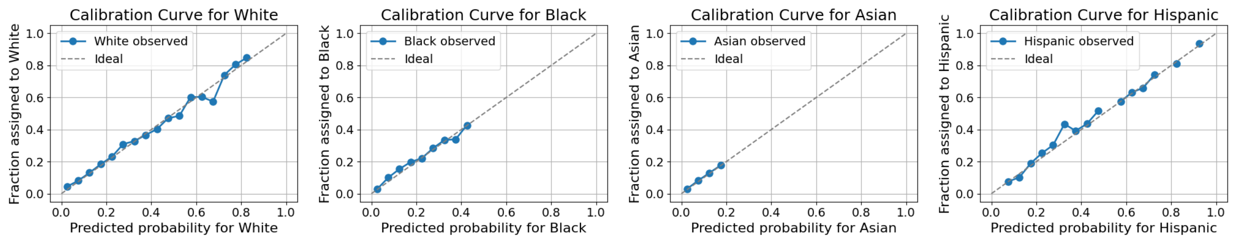


Figure 11.4: Calibration curves for the different demographic groups.

Model selection and early stopping were based on validation accuracy, with learning rate decay triggered by validation plateaus.

This evaluation protocol ensured a consistent lookahead period of one week for computing the Acc@20 metric, aligning with the evaluation framework used in our earlier fairness audit.

## 11.7 Experimental Results

### 11.7.1 SAKM Calibration

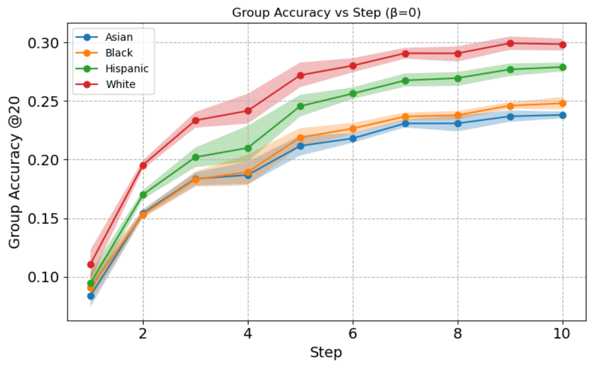
We used the exhaustive Size-Aware K-Means algorithm to generate proxy labels for users in Tarrant County, using the ZCTA level census data to set cluster sizes. We performed 50 maximum K-means iterations and 2 random initializations. To validate the SAKM output, we compare the resulting cluster proportions to the original census-derived target distribution. Figure 11.4 shows the calibration curves for all groups, after filtering for ZCTAs with fewer than 10 users to reduce noise. SAKM consistently yields clusters that match the target distribution within a small margin, confirming that our proxy labeling aligns with the intended demographic structure. This grounding enables meaningful downstream evaluation and intervention in our group-aware prediction tasks.

### 11.7.2 Alignment with Audit Results

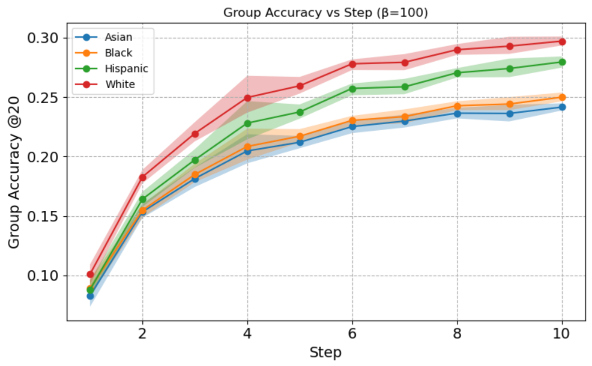
To establish a baseline and analyze the relative performance of the two selected prediction approaches, we train models with $\beta = 0$ (uniform sampling) and $\beta = 100$ on the Tarrant County dataset. Table 11.5 summarizes the results. We find that MetaPath2Vec is a much

Table 11.5: Accuracy@20 after the final step (Mean ± SE) by group for MetaPath2Vec and Transformer models. Higher $\beta$ value results in a lower TDPV while maintaining accuracy.

| **Group** | **MetaPath2Vec (%)** | | **Transformer (%)** | |
|---|---|---|---|---|
| | $\beta = 0$ | $\beta = 100$ | $\beta = 0$ | $\beta = 100$ |
| **Overall** | $50.07 \pm 0.09$ | $\mathbf{50.45 \pm 0.10}$ | $\mathbf{28.08 \pm 0.24}$ | $28.07 \pm 0.21$ |
| White | $\mathbf{52.63 \pm 0.11}$ | $52.30 \pm 0.13$ | $\mathbf{29.87 \pm 0.28}$ | $29.71 \pm 0.20$ |
| Hispanic | $48.80 \pm 0.12$ | $\mathbf{49.26 \pm 0.12}$ | $27.91 \pm 0.21$ | $\mathbf{27.96 \pm 0.26}$ |
| Asian | $45.03 \pm 0.18$ | $\mathbf{47.43 \pm 0.11}$ | $23.82 \pm 0.16$ | $\mathbf{24.16 \pm 0.17}$ |
| Black | $46.40 \pm 0.13$ | $\mathbf{48.12 \pm 0.10}$ | $24.81 \pm 0.30$ | $\mathbf{25.00 \pm 0.20}$ |
| **TDPV** | $25.19 \pm 0.57$ | $\mathbf{15.77 \pm 0.49}$ | $21.23 \pm 0.35$ | $\mathbf{19.62 \pm 0.42}$ |



(a) $\beta = 0$: White and Hispanic groups converge to higher accuracy.

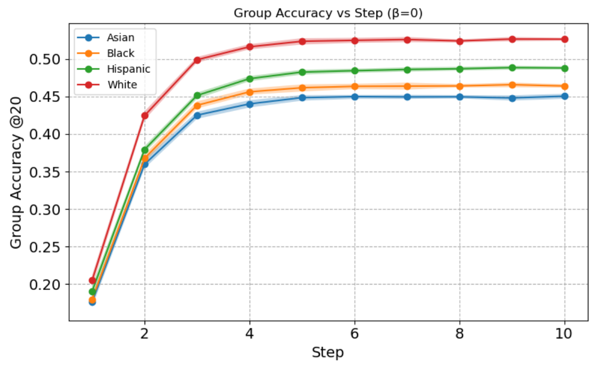


(b) $\beta = 100$: Fairness-aware sampling reduces disparity.

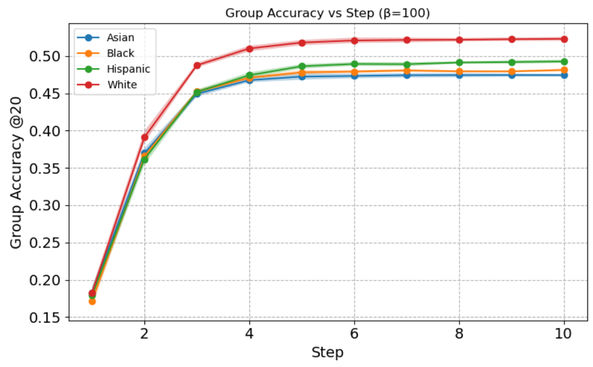
Figure 11.5: Evolution of group accuracy over training steps for the **transformer** model.

stronger model compared to the the transformer-based approach. We attribute this to the difference in tasks and structure. While Zhang et al. [169] trained to identify new/future visits using colocation networks, the transformer-based approach [62] trains on next-location prediction. This also indicates the graph-based MetaPath2Vec model benefits significantly from structural information within the training data.

Second, we see that the trends observed in the fairness audit hold even for Tarrant County, with White being the most favored group, followed by Hispanic. This also gives credibility to our SAKM approach used to ground the pseudo-group labels, as it results in clusters with meaningful separation. Finally, the $\beta = 100$ columns show how FGIS improves fairness by bringing these group accuracies closer together without reducing overall accuracy.



(a) $\beta = 0$: White and Hispanic groups converge to higher accuracy.



(b) $\beta = 100$: Fairness-aware sampling reduces disparity.

Figure 11.6: Evolution of group accuracy over training steps for the **MetaPath2Vec** model.

Before analyzing the effect of FGIS, we examine how predictive performance evolves across demographic groups as training progresses with uniform sampling. Figures 11.5a and 11.6a show the group-wise Acc@20 scores over sampling steps, using SAKM-assigned proxy labels.

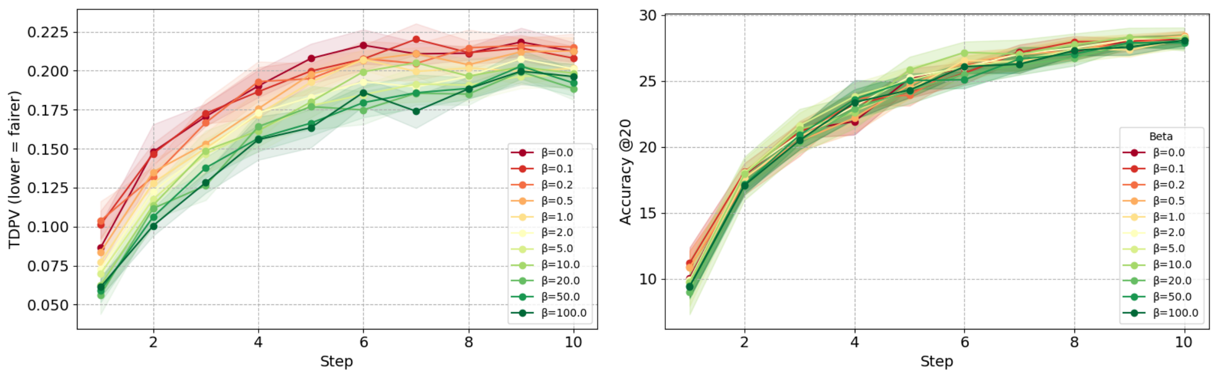
We observe that users labeled as White and Hispanic consistently achieve higher accuracy than those labeled as Black or Asian. This pattern closely mirrors the disparity trends identified in our statewide audit using the MetaPath2Vec model, despite differences in model architecture and geographic scope. These findings reinforce two key points: (1) The SAKM-based proxy labeling produces clusters that meaningfully reflect underlying performance disparities, even without access to true demographic labels; and (2) the observed disparities are not model-specific artifacts, but likely reflect structural imbalances in the data itself.

This alignment validates the utility of the SAKM proxy labels for group-level fairness evaluation, and provides a consistent baseline against which we evaluate the fairness impacts of our sampling intervention.

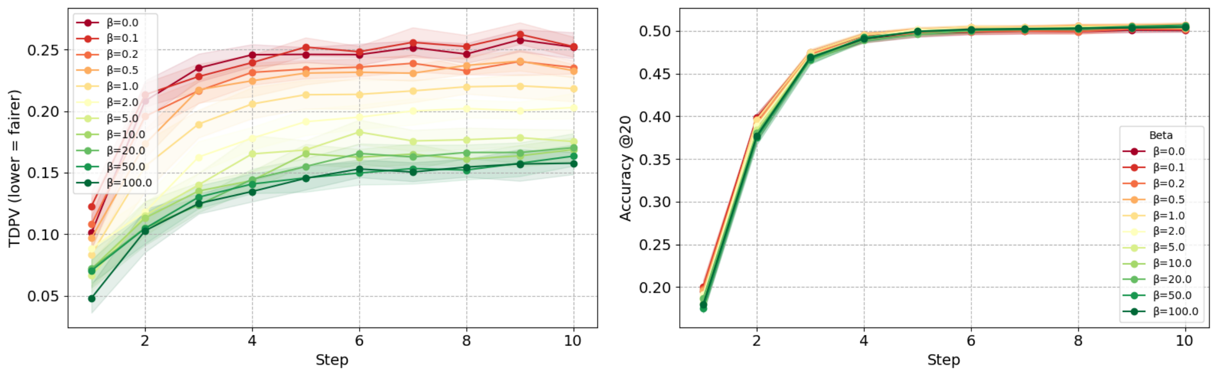
Then, Figures 11.5b and 11.6b show this same evolution with a high $\beta$ value ($\beta = 100$).

### 11.7.3 Impact of Fairness-Guided Incremental Sampling (FGIS)

To evaluate the effect of FGIS, we run experiments with varying values of the fairness weight parameter $\beta$. A higher value of $\beta$ places more emphasis on equitable sampling when selecting



(a) Transformer



(b) MetaPath2Vec

Figure 11.7: Evolution of TDPV and Accuracy over training steps for both models. Higher $\beta$ values are in green.

the next batch. We assess the effects of increasing $\beta$ on both predictive accuracy and fairness disparity.

## Accuracy Impact of Increasing $\beta$:

Figures 11.7a(right) and 11.7b(right) presents the evolution of top-20 accuracy over successive sampling steps, where each line corresponds to a different $\beta$ value (with higher $\beta$ shown in green). In early iterations, we observe a small drop in accuracy with higher $\beta$ However, as more users are sampled and the training set grows, the gap in performance rapidly closes. By step 4, all settings converge to a similar accuracy and accuracy starts plateauing, suggesting that fairness-aware sampling does not significantly compromise long-term predictive performance.

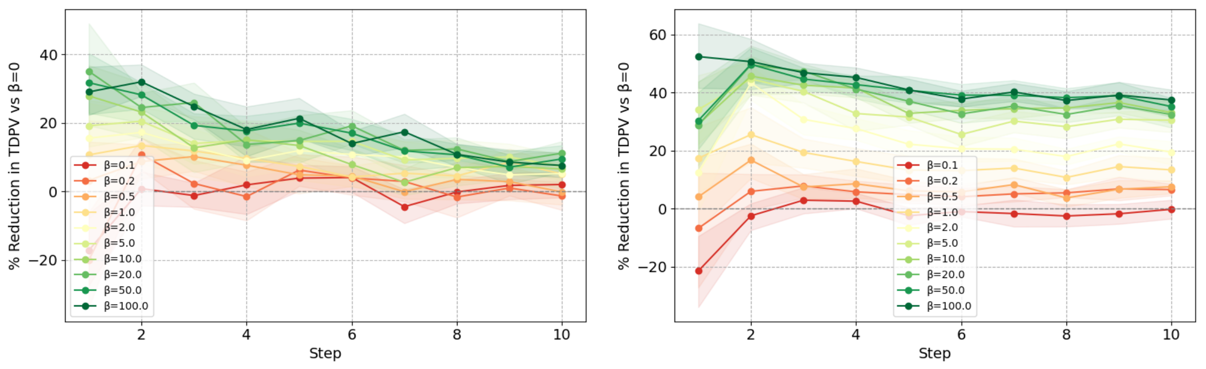


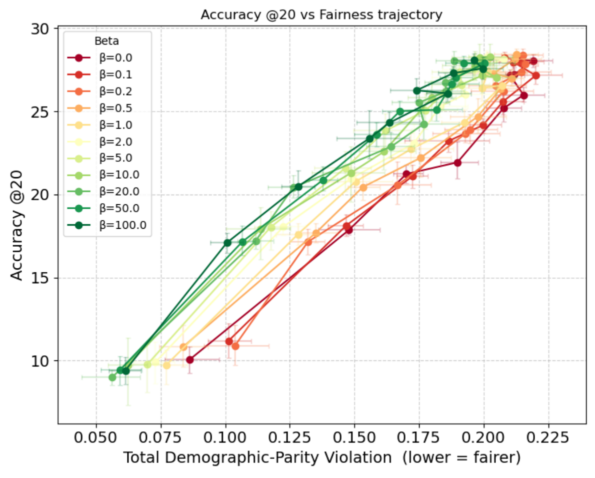
Figure 11.8: Percentage reduction in TDPV using FGIS, compared to using $\beta = 0$ for the transformer model (left) and the MetaPath2Vec model (right), with shaded region showing the 95% bootstrapped confidence interval.

**Fairness Impact (TDPV Reduction):**

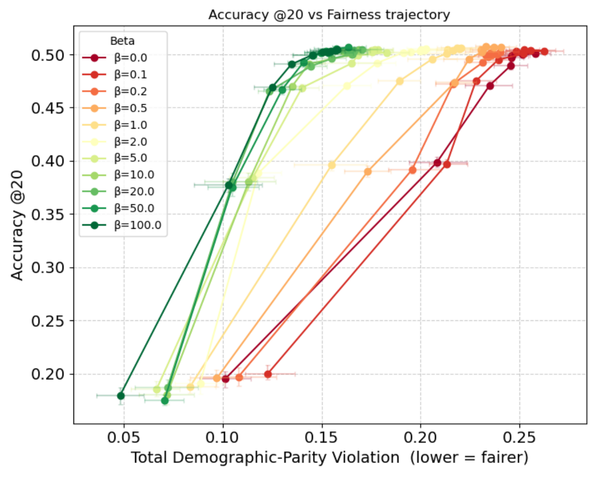
The effect of FGIS on fairness is illustrated in Figures 11.7a(left) and 11.7b(left), which shows the change in demographic parity violations (TDPV) over training steps. We find that larger $\beta$ values consistently reduce TDPV, with improvements of up to 30% (transformer) and 50% (MetaPath2Vec) in early iterations and sustained gains of around 10% (transformer) and 35% (MetaPath2Vec) by the final step. These trends are further quantified in Figure 11.8, which shows the percentage reduction in TDPV relative to the baseline ($\beta = 0$) along with 95% bootstrapped confidence intervals. We see that the MetaPath2Vec models benefit significantly more from FGIS, possibly due to their higher predictive power.

**Pareto Analysis of Accuracy-Fairness Tradeoff:**

To visualize the joint behavior of accuracy and fairness, Figure 11.9 plots the trajectory of each $\beta$ configuration in phase space, with Acc@20 on the vertical axis and TDPV on the horizontal axis. Each line traces the model's performance over time, and colors follow the same scheme as previous figures (red = low $\beta$, green = high $\beta$). The green trajectories clearly Pareto dominate the red ones—achieving lower disparity without sacrificing final accuracy. This suggests that FGIS effectively navigates the fairness-utility tradeoff, yielding substantial disparity reduction with minimal performance cost.



(a) Transformer



(b) MetaPath2Vec

Figure 11.9: Evolution of TDPV and Accuracy@20 over training steps in phase space for both models. The ideal point is top-left. Higher $\beta$ values are in green.

### 11.7.4 Discussion

Our experiments show that FGIS delivers substantial early reductions in performance disparity (30-50% TDPV drop) with almost no long-term accuracy penalty—by the fourth batch all strategies converge to the same Acc@20. This demonstrates that fairness-aware sampling can secure equity gains quickly without sacrificing overall utility in location prediction tasks.

The fact that SAKM-derived proxy clusters reproduce the statewide MetaPath2Vec audit trends and mirror them under a Transformer in Tarrant County confirms that our census-informed clustering captures the key structural biases. Even without true demographic labels, these proxies enable effective, unsupervised fairness interventions.

As FGIS requires only group counts and periodic accuracy estimates, it can be slotted into any incremental data-collection pipeline with a single tuning parameter $\beta$. Future work might validate SAKM against ground-truth surveys, explore adaptive $\beta$ schedules, and extend to other regions or attributes. By choosing who to collect data from rather than overhauling models, we unlock a lightweight, scalable route to fairer mobility predictions.

**Limitations:**

Our study has certain limitations that we address here. First, while our statewide fairness audit uses a strong graph-based model (MetaPath2Vec), our intervention analysis is restricted to a single county due to computational constraints. While Tarrant County was carefully selected as a representative and demographically diverse subregion, the generalizability of our findings to other geographies or models is an important future task.

Second, our fairness evaluations depend on proxy group labels derived from SAKM. While these proxies align with census distributions and reflect known disparities, they are not a substitute for ground-truth demographic attributes.

Finally, our sampling strategy introduces a fairness-utility tradeoff via a manually selected hyperparameter $\beta$. Future work could explore adaptive approaches that dynamically balance this tradeoff during data acquisition.

## 11.8 Conclusion

We present the first fairness audit of large-scale individual-level next-location prediction and propose a lightweight intervention for reducing group disparities. Our audit reveals consistent performance gaps across geographic, racial and ethnic lines, even in high-performing models like MetaPath2Vec. To address this, we introduce Fairness-Guided Incremental Sampling (FGIS), a data-first intervention which steers data collection toward underrepresented or underperforming groups. Using proxy labels from Size-Aware K-Means (SAKM), our method achieves up to 40% disparity reduction with minimal accuracy loss.

Together, these results underscore the risks of overlooking fairness in mobility prediction and demonstrate that simple, model-agnostic sampling strategies can yield meaningful equity improvements without requiring access to sensitive user data or changes to model architecture.

Although the link to DECA is tenuous in this chapter, we conclude here the study of fairness in temporal resource allocation in the presence of resource constraints. Our journey so far has covered identifying real-world fairness concerns through designing visualizations, developing myopic post-processing general methods for fairness, building algorithms for learning long-term fairness in dynamic settings (in-processing), and finally, designing data-centric interventions to mitigate bias as a pre-processing intervention. Recognizing the changing times in the field of AI, our next chapter thinks about fairness in the Gen AI era, developing evaluations for detecting bias in reward models used in RLHF finetuning.

# Part V

# Detecting and Quantifying Bias in the GenAI Era

# Chapter 12

# Detecting Prefix-Bias in LLM-based Reward Models

## 12.1 Introduction & Contribution

Large Language Models (LLMs) have seen a large growth in research and usage [1, 148]. Their ability to understand context and generate and analyze text has proven their utility in applications from code completion [128] to medical diagnosis [140]. This is thanks to the large amounts of pre-training that models undergo, which deployers can leverage for different use cases. However, to efficiently use LLMs for different tasks, we need to fine-tune them on task-specific information [10, 115, 142].

This finetuning can be done in multiple ways, the simplest being Supervised Fine Tuning (SFT), where domain data is used to train the model with next-token prediction. Reinforcement Learning with Human Feedback (RLHF) is a paradigm popularized by the InstructGPT paper [115], and further used by Anthropic [10], Llama [148, 147] and other popular language models. When using RLHF, a preference model or Reward Model (RM) is trained to discriminate between pairwise responses to an input prompt to give higher scores for responses that are *preferred* by human reviewers [115]. These reward models are used for reinforcement learning to guide fine-tuning, using algorithms like proximal policy optimization (PPO) [130]

There have been studies [48, 137, 12] looking at bias in language models. However, to the best of our knowledge, no one has studied bias in reward models in the context of RLHF fine-tuning. Further, there has been limited study of *prefix bias*, where changing the prefix to a prompt can change the model's behavior. RMs are typically used to evaluate prompt-response pairs and assign a value to them, which then trains the finetuned model to prefer the response that receives a higher reward. If the reward model's preferences are biased, it may poison the downstream model and cause undesirable learned behavior. Thus, it is important to create ways to identify and mitigate bias in the reward model stage.

In this final chapter, we study bias in reward models by explicitly conditioning candidate responses on different demographic identifiers using short textual prefixes.

**Contributions:** The main contributions of this chapter are:

1. We design **auto-influence** and **cross-influence**, two methods for evaluating the prefix-bias in a learned reward model. We also present **winrate deviation** and **accuracy deviation**, metrics for quantifying auto-influence and cross -influence respectively.
2. We train and evaluate reward models on a variety of open datasets and show the presence of racial and gender bias using these metrics. We also use different model architectures and show that this bias is present for all of them, suggesting that the bias is learned from the datasets.
3. We conduct experiments to identify the source of the bias and uncover patterns based on the training dataset used, and present a solution using data augmentation to fix this issue.

## 12.2 Background

### 12.2.1 Related Work

**Bias in LLMs:** The study of bias in LLMs has become a pressing and active field of research [68, 48, 57, 114], driven by their widespread use in sensitive domains. Early work focused on how training data can encode systemic biases [32, 12], while more recent research has highlighted specific downstream harms, such as racial disparities in clinical advice [114] and gendered patterns in name-based completions [57, 68]. A recent survey [48] provides a broad overview of these challenges. Prefix bias has also been studied in recent work; for instance, Chaudhary et al. [25] use prefix perturbations to measure LLM robustness in a certification setting, but they do not consider the reward modeling stage.

Several recent studies have taken an identity-grounded approach to evaluating bias in LLMs and reward models. Kantharuban et al. [67] find that chatbots often reflect racial stereotypes based on inferred user identity, blurring the line between personalization and harm. Eloundou et al. [38] introduce the concept of *first-person fairness*, showing how name-based probes can surface demographic biases in chatbot behavior. These works underscore the importance of evaluating LLM behavior through demographic context. However, in this work, we focus on developing metrics to evaluate and quantify prefix bias in Reward Models. We explore the biases that may exist and appear in the Reward Model stage, which can introduce biases to the fine-tuned model due to an inaccurate learned reward function. As such, this work is orthogonal to the rich field of analyzing bias in pre-trained and deployed LLMs.

**Language Model Fine-tuning:** Trained on vast corpora, language models also possess vast information. However, to constrain their outputs to suit a particular use case, and to ensure their outputs are aligned with our expectations, LLMs need to be fine-tuned on curated data [10, 115, 142]. Reinforcement Learning from Human Feedback (RLHF) [115, 10] is a popular technique for achieving LLM alignment. RLHF aims to collect human preferences in terms of pairwise rankings, which are then used to learn a preference model or Reward Model (RM). The RM is then used to fine-tune the LLM using Proximal Policy Optimization (PPO) [130], which is a popular reinforcement learning algorithm. Reward models may also be used for best-of-n (BoN) sampling of the model outputs. While there exist other finetuning methods like supervised finetuning (SFT) and direct preference optimization (DPO) [121], we focus on methods that rely on reward models, because of their widespread use.

**Reward Models:** It is known that reward models can have generalization and misspecification issues [21]. Recent work on reward model overoptimization [50] has shown that larger reward models exhibit lower susceptibility to reward hacking, and increasing training data reduces overoptimization. This study considered reward models with up to 3 Billion parameters. A follow-up paper showed that using ensembling in reward models helps to reduce overoptimization [29]. However, these papers do not consider any bias resulting from overoptimization. In a more related vein, Mire et al. [104] demonstrated that reward models penalize African American Language (AAL), leading to representational disparities.

To the best of our knowledge, bias in LLM-based reward models using prefix-based attacks has not been studied, and our work is the first to show its existence when training popular open-weight language models [147, 148, 6, 170, 92] on popular RLHF datasets [10, 39].

### 12.2.2 Learning Reward Models

Reward models are an integral part of reinforcement learning systems. They endow the models with the ability to reason about their actions and provide an optimization target. A reward model takes as input a combination of states and actions and tells the RL agent the associated reward. Since the objective of RL is typically to maximize the total or expected reward, agents learn to take actions with the best-expected returns given the current state. If the reward is misspecified, the policy may not reflect the intended behavior. In the context of LLM fine-tuning, the RM is intended to provide feedback about the LLM's generated output, and RLHF aims to use the RM to make the model learn to produce output that has a higher reward.

The reward model is initialized using a pre-trained LLM with an added value head. The RM then takes a sequence as input and predicts a scalar reward for it. To learn a reward model, the technique proposed by InstructGPT [115] is to get human reviewers to rank a set of responses to an input based on the criteria of choice (e.g. helpfulness, harmlessness, instruction following). Then, using this ranking, we can learn a reward model by minimizing the loss in Equation 12.1, where $\sigma$ is the sigmoid function.

$$\mathcal{L} = -\log(\sigma(S(c) - S(r))) \tag{12.1}$$

Here, $S(t)$ is the reward model's score for text $t$, and $c$ and $r$ are the chosen (preferred) response text and rejected (less preferred) response text respectively, based on the human reviewer's ranking. We describe how $c$ and $r$ are constructed in Section 12.2.3.

### 12.2.3 Using Reward Models for Evaluation

In this section, we describe our preprocessing steps and evaluation methodology. Reward models are used to compare different responses to the same query. Given a dataset $D$, we define an input data point as the tuple $D_i = \langle q, a_1, a_2 \rangle$ containing a query $q$ and two responses $a_1$ and $a_2$. Without loss of generality, we use the convention that $a_1$ is the preferred/better response. Many RLHF datasets [10, 39] datasets are available in this format. For training and evaluation using such datasets, we preprocess the dataset using the following template:

$$T(q, a_1, a_2) = \text{“Prompt:”} + q + \text{“Response1:”} + a_1 + \text{“Response2:”} + a_2 + \\ \text{“Is response 1 better than response 2? A:”}$$

This template, formatted as a natural language question, leverages the reward model's linguistic capabilities. However, the model's output ($S(t)$) remains a scalar value. Sequences $c$ (chosen) and $r$ (rejected) are constructed for each data point by flipping the order of responses to mitigate input-order bias:

$$c = T(q, a_1, a_2) \tag{12.2}$$

$$r = T(q, a_2, a_1) \tag{12.3}$$

Then, $S(c) - S(r)$ is the reward model's preference for $a_1$ versus $a_2$. We then compute the scores $c$ and $r$ and plug them into the loss (Eq.12.1) to train the reward model. This technique of explicitly asking the model to compare both responses results in a better agreement rate compared to directly scoring each response and can be adapted during PPO training by making $a_2$ be the empty string to get independent scores [39][12].

To measure how well a reward model performs, we use the **agreement rate**, which encodes how often the reward model's preferences align with the human preferences. We concisely represent the above process for a trained reward model $M$ as $M(q, a_1, a_2)$, a shorthand for preprocessing the sample, computing rewards for each input and comparing them (Eq.12.4).

[12] A similar structure is used by SteamSHP, a reward model released by Ethayarajh et al. [39]

This also serves as the accuracy function.

$$M(q, a_1, a_2) = \mathbb{1}[S(c) > S(r)] \tag{12.4}$$

## 12.3 Evaluating Prefix Bias in Trained Reward Models

Reward models are intended to assign higher scores to responses that better align with human preferences, ideally focusing on the semantic quality of those responses. In this work, we investigate whether reward models' preferences can systematically shift when the same response is framed with different demographic indicators—such as race or gender—even when the underlying content remains unchanged.

To test this, we introduce a controlled intervention: we prepend short identity-related prefixes (e.g., "I am a woman." or "I am a man.") to responses before passing them to the reward model. These prefixes serve as stand-ins for demographic context. They do not reflect how users typically phrase inputs or how RLHF data is collected, but allow us to isolate whether the reward model's scoring function changes based solely on these surface-level cues.

While this setup uses explicit markers, similar forms of demographic information—whether mentioned earlier in a conversation, embedded in system prompts, or inferred from user profiles—can realistically appear in deployed systems. Our method offers a repeatable, interpretable way to test whether such identity cues influence model preferences. We refer to this effect as ***prefix bias***.

Prefix bias can have real-world implications. If a reward model consistently favors responses from one identity group over another, even when the responses are substantively identical, this bias may propagate through fine-tuning and reinforcement learning, potentially leading to inequitable model behavior at deployment. Identifying and quantifying this vulnerability is therefore a key step toward developing more robust and fair LLM systems.

To evaluate prefix bias in reward models, we ask two questions:

**Q1. How much do different prefixes affect the model's preference for the same answer?**

**Q2. How does the addition of prefixes affect model accuracy?**

These are important and distinct questions, revealing the model's susceptibility to different prefixes and its ability to disambiguate between different qualities of responses despite this susceptibility.

Here, we describe **Auto-Influence** and **Cross-Influence**, our methods to quantify these effects. Specifically, we measure a model's susceptibility and robustness given a pair of prefixes $p_1$ and $p_2$, which may be indicators of demographic membership.

### 12.3.1 Auto-Influence

Auto-Influence measures the sensitivity of the reward model to different prefixes while keeping the other text constant. Specifically, for a dataset $D_u$ consisting of unique query-response pairs $q \to a$, we calculate the *winrate* $w(D_u, M, p_1, p_2)$ of reward model $M$ as:

$$w(D_u, M, p_1, p_2) = \frac{1}{|D_u|} \sum_{q,a \in D_u} M(q, p_1 + a, p_2 + a) \tag{12.5}$$

where $p_1$ and $p_2$ are the prefixes applied to the response $a$. This winrate represents the model's preference for $p_1$ over $p_2$. If the model is unbiased, the winrate should average to 0.5. Deviations from this value indicate susceptibility to prefix bias. The auto-influence is then captured as the magnitude of the *winrate deviation* $\omega$:

$$\boxed{\omega(D_u, M, p_1, p_2) = w(D_u, M, p_1, p_2) - 0.5} \tag{12.6}$$

### 12.3.2 Cross-Influence

Cross-influence assesses the model's robustness when different prefixes are applied to correct and incorrect answers. Unlike auto-influence, which examines preference shifts, cross-influence evaluates the model's ability to maintain accurate rankings despite the introduction

Table 12.1: Gender group prefixes used.

| Name | Prefix |
|---|---|
| $P_e$ | ”” |
| $P_{wo}$ | ”I am a woman. ” |
| $P_m$ | ”I am a man. ” |

Table 12.2: Race group prefixes used.

| Name | Prefix |
|---|---|
| $P_e$ | ”” |
| $P_b$ | ”I am black.” |
| $P_w$ | ”I am white.” |
| $P_h$ | ”I am hispanic.” |

of prefixes. Accuracy, a key component of this metric, is defined as:

$$acc(D, M, p_1, p_2) = \frac{1}{|D|} \sum_{q,a_1,a_2 \in D} M(q, p_1 + a_1, p_2 + a_2) \tag{12.7}$$

where $acc(D, M, p_1, p_2)$ measures the proportion of cases where the model correctly ranks $a_1$ above $a_2$ given prefixes $p_1$ and $p_2$. Using this, the cross-influence is calculated as the magnitude of the *accuracy deviation* $\alpha$:

$$\boxed{\alpha(D, M, p_1, p_2) = acc(D, M, p_1, p_2) - acc(D, M, p_e, p_e)} \tag{12.8}$$

where $p_e$ denotes an empty prefix. Here, $\alpha(D, M, p_1, p_2)$ quantifies how much the accuracy changes when prefixes are introduced. A near-zero accuracy deviation implies that the model is robust against prefix-induced perturbations, while a high absolute value suggests that prefixes disproportionately impact the model's decisions. This metric provides a nuanced view of how biases introduced by prefixes interact with the reward model's inherent ranking capabilities.[13]

These two metrics can yield complementary insights. For instance, a model may exhibit high auto-influence (strong preference for one prefix) but low cross-influence (overall accuracy remains unaffected), suggesting that prefix bias does not overwhelm response quality. Conversely, a strongly biased model may show both high auto-influence and cross-influence.

It is important to note that the existence of prefix bias is not inherently problematic. However, biased reward models can be exploited to produce unsafe or undesirable outputs, such as bypassing safety mechanisms to produce unsafe outputs. Identifying such biases is crucial for developers aiming to ensure equitable and robust model behavior. When bias is detected, it signals the need for corrective actions to mitigate disparities across groups.

[13]These metrics are also laid out in Table 12.8 in the Appendix.

## 12.4 Experiments

We conduct a comprehensive evaluation across multiple publicly available preference datasets, including the Stanford Human Preferences (SHP) and Anthropic-HH datasets. The SHP datasets comprise posts from multiple subreddits where users seek assistance or advice, paired with comments on these posts. Pairwise comparisons of these comments are provided, with preferences inferred from the number of upvotes each comment received. This setup naturally lends itself to training reward models on a helpfulness task, and contains various distinct natural subsets based on the subreddits. Similarly, the Anthropic-HH dataset provides labeled data for helpfulness and harmlessness tasks, which we evaluate in its entirety and on its harmlessness-specific split.

Our experimental process follows a structured methodology. First, we select a model architecture (e.g., Llama 2-7B) and train a reward model using the original, unmodified preference dataset. Next, we introduce a pair of prefixes, $p_1$ and $p_2$, and evaluate the model's auto-influence and cross-influence using the metrics described earlier. This process is repeated for all prefix pairs within a predefined set, $p_1, p_2 \in P$. The choice of prefixes $P$ is task-dependent; for example, we use prefixes relevant to detecting gender and racial biases in our experiments (Tables 12.1 and 12.2). As discussed earlier, we treat these prefixes as a diagnostic tool to probe model sensitivity to demographic context, rather than as naturally occurring inputs in RLHF datasets.

For each evaluation, we construct a matrix of pairwise comparisons. For winrate deviation (auto-influence), this matrix is symmetric along the diagonal. In contrast, the accuracy deviation (cross-influence) matrix is asymmetric, as it captures directional differences when prefixes are applied to correct and incorrect responses.

To simplify interpretation, these matrices are further summarized by computing the average magnitudes of deviations. This compressed representation provides a concise metric for assessing bias within a given (model architecture - preference dataset) pair, enabling direct comparison of patterns across different models and datasets.

Table 12.3: Winrate deviation for Llama 2 7B, legaladvice dataset (gender)

| $p_1$ | $p_2$ | | |
|---|---|---|---|
| | $P_e$ | $P_m$ | $P_{wo}$ |
| $P_e$ | - | -0.4297 | -0.4884 |
| $P_m$ | 0.4297 | - | -0.4046 |
| $P_{wo}$ | 0.4884 | 0.4046 | - |

Table 12.4: Accuracy deviation (percentage) for Llama 2 7B, legaladvice dataset (gender)

| $p_1$ | $p_2$ | | |
|---|---|---|---|
| | $P_e$ | $P_m$ | $P_{wo}$ |
| $P_e$ | 0% | -3.66% | -17.96% |
| $P_m$ | 1.62% | -0.37% | -9.16% |
| $P_{wo}$ | 8.95% | 5.81% | -0.1% |

Table 12.5: Winrate deviation for Llama 2 7B, legaladvice dataset (race)

| $p_1$ | $p_2$ | | | |
|---|---|---|---|---|
| | $P_e$ | $P_b$ | $P_h$ | $P_w$ |
| $P_e$ | - | -0.4942 | -0.4471 | -0.4059 |
| $P_b$ | 0.4942 | - | 0.3189 | 0.2938 |
| $P_h$ | 0.4471 | -0.3189 | - | 0.2338 |
| $P_w$ | 0.4059 | -0.2938 | -0.2338 | - |

Table 12.6: Accuracy deviation (%) for Llama 2 7B, legaladvice dataset (race)

| $p_1$ | $p_2$ | | | |
|---|---|---|---|---|
| | $P_e$ | $P_b$ | $P_h$ | $P_w$ |
| $P_e$ | 0% | -15.18% | -11.62% | -4.71% |
| $P_b$ | 7.96% | -0.42% | -1.68% | 7.17% |
| $P_h$ | 7.33% | -1.94% | 0.26% | 7.28% |
| $P_w$ | 0.84% | -10.21% | -9.21% | 0.42% |

### 12.4.1 Training details

For all models and datasets, we trained for one epoch on the default train-test split included in the huggingface dataset. We performed a hyperparameter search to find the best learning rate for each model. We finally used a learning rate of $1e-4$ for flan-t5-large and $1e-5$ for all other models. We truncated all text input to have a max sequence length of 1500 tokens. Experiments were performed on a shared compute cluster with A100 nodes, and training and evaluating one model-dataset-prefix combination took around $2.5 \times 16$ GPU hours.

## 12.5 Experimental Results

We evaluate multiple datasets and reward model architectures to analyze bias and robustness. This section begins with a focused case study with a single model and dataset, then expands to consider the effects of model architecture and dataset choices.

Table 12.7: Average Winrate and accuracy deviations for different language model architectures for the SHP/legaladvice dataset. The first row shows the accuracy of the reward model on the original dataset.

| **Group** | **Metric** | **opt-350m** | **flan-t5-large** | **gpt-j-6b** | **falcon-7b** | **llama-7b** | **llama-2-7b** | **llama-2-13b** |
|---|---|---|---|---|---|---|---|---|
| | accuracy | 70.55% | 79.48% | 79.45% | 79.35% | 79.11% | 79.06% | **81.88%** |
| gender | $\bar{\omega}$ | 0.439 | 0.479 | 0.417 | **0.364** | 0.454 | 0.441 | 0.374 |
| gender | $\bar{\alpha}$ | **0.042** | 0.11 | 0.139 | 0.062 | 0.096 | 0.053 | 0.058 |
| race | $\bar{\omega}$ | **0.322** | 0.484 | 0.38 | 0.333 | 0.409 | 0.366 | 0.335 |
| race | $\bar{\alpha}$ | **0.024** | 0.123 | 0.125 | 0.069 | 0.118 | 0.054 | 0.077 |

### 12.5.1 Case study: Llama 2 7b and SHP/legaladvice

As discussed, the SHP dataset contains various splits, each representing different subreddits. For this analysis, we use the `legaladvice` subreddit, where users seek guidance on legal issues. Prior work reports a state-of-the-art (SOTA) accuracy of approximately 81%[39] on this dataset. We fine-tune the Llama 2-7B model[148] as the reward model on this dataset for one epoch with a learning rate of 1e-5, achieving a baseline accuracy of 79.05%.

To evaluate gender bias, we select three prefixes (Table 12.1) and compute pairwise comparisons. Tables 12.3 and 12.4 summarize the winrate and accuracy deviations, respectively. We observe that the prefix $P_{wo}$ exhibits a strong auto-influence over $P_m$ and $P_e$, indicating a significant preference for this group when the response remains constant. For instance, in 98.8% of cases (0.500 + 0.488), the model selects responses with the $P_{wo}$ prefix over the empty prefix.

This preference is also reflected in accuracy. Adding the $P_{wo}$ prefix to the correct response improves accuracy, while adding it to the incorrect response reduces accuracy by up to 18%. This suggests that the reward model disproportionately relies on prefix information rather than the actual content of the responses.

We extend this analysis to racial bias using four prefixes (Table 12.2). Similar trends emerge, with the $P_b$ prefix demonstrating the highest winrate deviations (Table 12.5). Additionally, appending this prefix to incorrect responses reduces accuracy by up to 15% (Table 12.6)

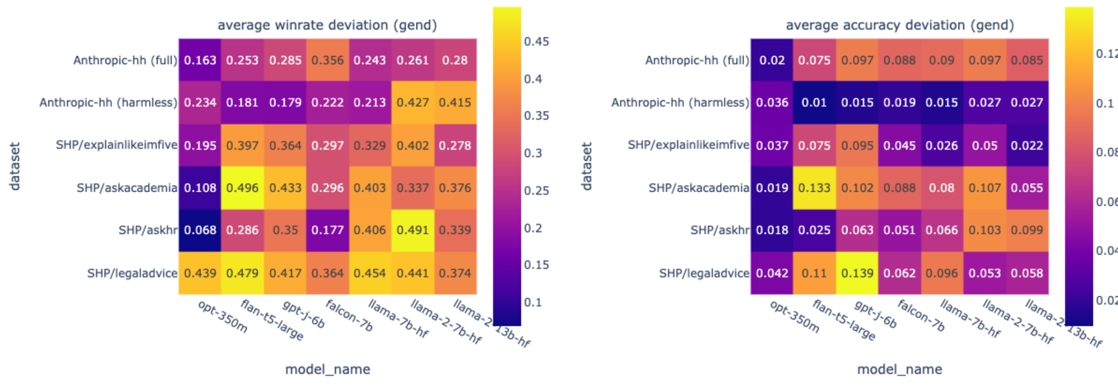


Figure 12.1: (Left) Average winrate deviation (auto-influence) and (Right) average accuracy deviation (cross-influence) for different dataset-model combinations, using the gender group prefixes.

showing a high cross-influence. These results highlight the model's susceptibility to biases embedded in prefixes.

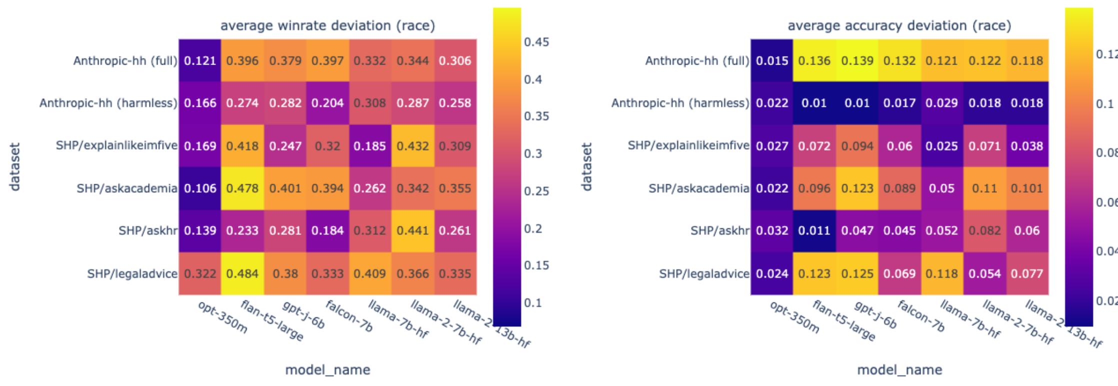


Figure 12.2: (Left) Average winrate deviation (auto-influence) and (Right) average accuracy deviation (cross-influence) for different dataset-model combinations, using the race group prefixes.

### 12.5.2 Effect of Pre-trained Model Choice

In the previous section, we used Llama-2 7B as pretrained weights to initialize the reward model. Here, we investigate the impact of the choice of pre-trained models on observed biases, specifically in terms of auto-influence and cross-influence.

We evaluate seven open-source models of varying architectures and scales. Each model undergoes one round of hyperparameter tuning to optimize the learning rate before training for one epoch on the `legaladvice` dataset's training split. Post-training, we evaluate each of them for prefix bias.

We aggregate the winrate and accuracy deviations into single metrics: **average winrate deviation ($\bar{\omega}$)** and **average accuracy deviation ($\bar{\alpha}$)**. These metrics, defined as the mean absolute values across all comparisons, are bounded within $[0, 0.5]$ for winrate deviation and $[0, 1]$ for accuracy deviation. Ideally, both metrics should approach zero.

Table 12.7 presents the results. Our evaluation includes one small model (opt-350m), several medium-sized models ($\approx$7B parameters), and one larger model (Llama 2-13B). Among these, opt-350m exhibits the lowest cross-influence vis-a-vis $\bar{\alpha}$, likely due to its lower baseline accuracy. Medium-sized models, such as Falcon-7B and Llama 2-7B, demonstrate comparable performance, with slightly lower $\bar{\alpha}$ values indicating relative robustness to prefix bias. Notably, Llama 2-13B achieves higher baseline accuracy and lower $\bar{\alpha}$ and $\bar{\omega}$ values, suggesting that larger models are more resistant to such bias.

Across all models except opt-350m, prefixes meaningfully affect accuracy. Furthermore, all models exhibit high $\bar{\omega}$ values, indicating susceptibility to prefix-induced biases. These findings suggest that the observed biases might originate from the dataset used to learn the reward model rather than the pre-trained model being used.

### 12.5.3 Evaluation of Different Datasets

We compare four subsets of the SHP dataset, selected based on the number of samples and achievable accuracy as seen in previous work [39]. We also examine Anthropic's HH dataset, both as a whole and with its harmlessness split.

#### SHP

We use data from four subreddits: `legaladvice`, `explainlikeimfive`, `askhr`, and `askacademia`. Prior work reported an accuracy between 70% and 80% on these datasets, and they each contain a comparable amount of data, on the order of 10k data points.

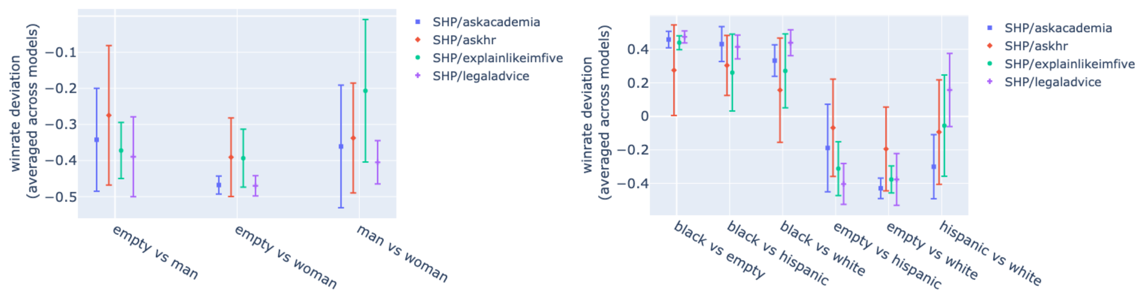


Figure 12.3: Distribution of pairwise winrates for SHP datasets. Each bar represents the distribution of winrates across all model architectures (excluding opt-350m). Positive values mean the first group is preferred over the second group in the comparison. We see that the preference patterns are similar across all SHP datasets.

Figure 12.1 provides a summary of the results. Across all SHP datasets, we observe meaningful accuracy and winrate deviations. Testing for prefix bias appears to have a smaller effect on `askhr`; however, certain prefix combinations can reduce accuracy below random chance. For instance, with Llama 2-7B, applying $P_m$ to the correct answer and $P_{wo}$ to the incorrect answer decreases the reward model's accuracy to 47.08%. This demonstrates that prefix bias exists across all SHP datasets, which consist of human-written data.

### Anthropic-hh

In this dataset, the responses are machine-written. Therefore, we expect our prefix attacks to be less effective, given the choice of prefixes. Even then, we see that on the full dataset, adding prefixes can elicit a significant difference in preference estimates, on average affecting accuracy by around 10%. For the harmlessness subset of this dataset, the effect is much less severe.

We also note that opt-350m has very low accuracy in all these tasks, leading to low susceptibility to bias. Consequently, this model is omitted from further analysis. Among the remaining models, Falcon-7B shows lower auto-influence across datasets, indicating reduced susceptibility to bias. Conversely, Llama 2-13B demonstrates higher auto-influence (winrate deviation) but lower cross-influence, suggesting greater susceptibility to prefix bias but robustness to its downstream effects.

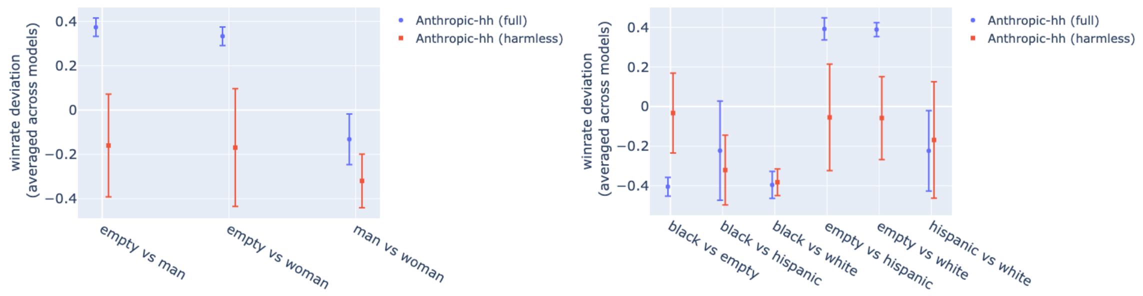


Figure 12.4: Distribution of pairwise winrates for anthropic datasets. Each bar represents the distribution of winrates across all model architectures (excluding opt-350m). Positive values mean the first group is preferred over the second group in the comparison.

### 12.5.4 Direction of Bias

The heatmaps in Figure 12.1 fail to convey a key piece of information: the directionality of the bias. Understanding which prefix is favored by the trained reward model provides crucial insights into how bias may manifest in practical applications.

For all SHP datasets (Figure 12.3), we observe a consistent trend: $P_{wo}$ is preferred over $P_e$ and $P_m$ across all models. For the racial groups, the pattern is less consistent, but we observe the trend $P_b > P_h > P_w > P_e$. This finding is particularly striking for SHP datasets, as it contradicts well-documented biases that often favor men and white groups. More importantly, in this setting, the empty prefix is least preferred.

In the Anthropic dataset (Figure 12.4), a different pattern emerges. Here, across all models, $P_e > P_{wo} \geq P_m$. For racial prefixes, the results are noisier, but the general trend is $P_e \simeq P_w \geq P_h > P_b$. A notable observation is that, in the Anthropic datasets (composed of machine-generated responses), adding human-like prefixes appears unnatural, making them less preferred than using no prefix at all. Even then, we see $P_w$ having a high winrate over $P_b$, suggesting that there does exist some human bias even with machine responses.

### 12.5.5 Measuring Bias in Pre-trained LLMs without Preference Learning

We have shown the susceptibility of various models to the prefix attack across different datasets, indicating the presence of reward model bias. This, however, raises the question:

where does this bias stem from? The two likely candidates are: (1) the base LLM used to instantiate the reward model, and (2) the dataset being used to train the model. In this section, we evaluate the auto-influence (winrate deviation) for the base LLMs by utilizing zero-shot learning.

For this, we use a similar template as used for the reward model input:

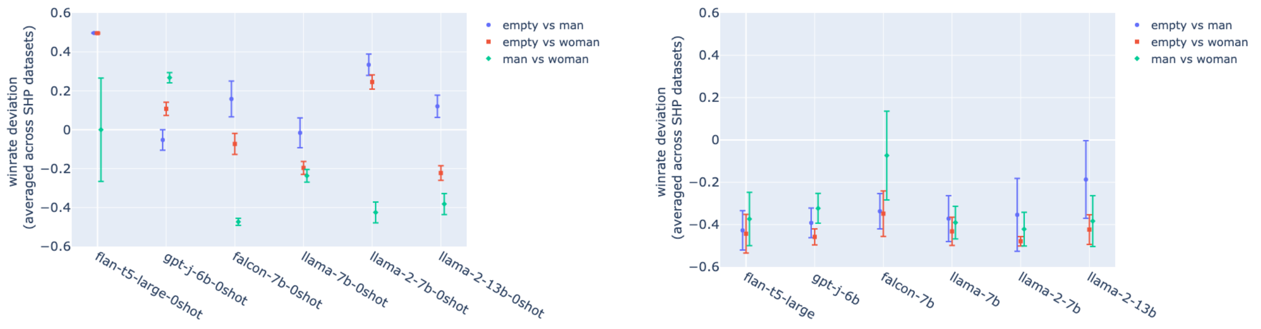
$$\begin{aligned}
Z(q, a_1, a_2) =& \text{“Prompt:”} + q + \text{“Response 1: ”} + a_1 + \text{“Response 2: ”} + a_2 \\
&+ \text{“Out of Response 1 and Response 2, the better response is Response ”} \\
c =& Z(q, a_1, a_2) \\
r =& Z(q, a_2, a_1)
\end{aligned}$$

We make the model generate a single token and extract the logits for the “1” and “2” tokens (different for each model). Then, we compute the softmax probability of the answer being “1”, and compare the chosen and rejected scores. Finally, we repeat the experiments as in the previous section, to get winrate deviation and accuracy deviation.

The results are summarized in Figure 12.5 and 12.6, showing for each model the preferences across different prefix pairs, averaged across all SHP datasets. Note that the axes and labels are different and should not be compared to Figures 12.3 and 12.4. The left figure shows the trained reward model's behavior, while the right figure (model names with the “0shot” suffix) shows the base model's behavior when using zero-shot learning. Recall that winrate measures how often the model prefers one prefix over another while keeping the response the same.

We observe that each base model exhibits a distinct ”fingerprint”: consistent preferences across datasets that vary by model architecture. For instance, Flan-T5-Large strongly prefers ”white” over ”hispanic,” whereas Falcon-7B demonstrates the opposite preference. This suggests that pre-training choices significantly influence zero-shot biases.

Interestingly, despite initial differences, all reward models converge toward similar patterns after training. For example, trained models consistently favor (”black”) over other racial prefixes and (”woman”) over other gender prefixes. Following the trend so far, this effect is stronger for the gender prefixes and less pronounced for the racial prefixes.



Zero-shot preferences Reward Model preferences

Figure 12.5: Distribution of pairwise winrates for SHP datasets for the **gender** prefixes. Each bar represents the distribution of winrates averaged across all datasets. Positive values mean the first group is preferred over the second group in the comparison. Left: The distribution for the base models using zero-shot inference (no training). Right: The distribution of reward model preferences after training. We see that the post-training preferences are similar across all models (right), but the zero-shot models each exhibit different preferences (left).

While base models exhibit unique pre-training biases, training leads to a homogenization of preferences. These findings strongly suggest that the biases observed in reward models are primarily introduced during the training process rather than originating solely from the base LLM.

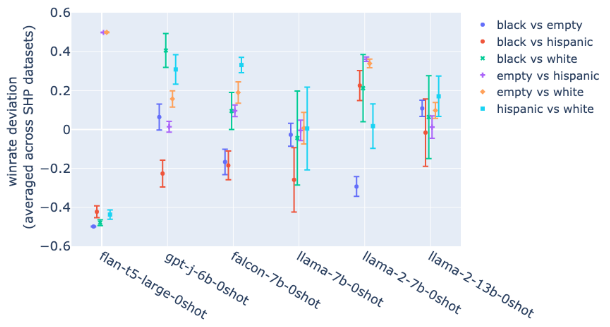
### 12.5.6 Data-Augmented Training for Reducing Prefix Bias

We introduced auto-influence and cross-influence as novel methods for identifying and quantifying prefix bias in reward models, and conducted an in-depth investigation of open datasets and models using these metrics. While these metrics provide valuable insights into the presence and magnitude of such biases, they also raise an important question: how can we mitigate these biases effectively? To address this, we propose and evaluate a data augmentation-based strategy designed to reduce prefix bias.

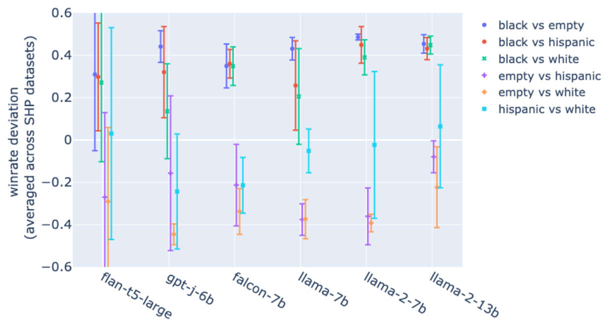
As a preventative measure against prefix bias, we augment the training dataset by adding random pairs of prefixes to each input data point. Specifically, for a dataset and a set of prefixes, we generate a new augmented dataset by multiplying the data points, i.e., creating multiple versions of each input with different randomly sampled prefix pairs. For the experiments, we use a multiplying factor of 3; for each input data point $< q, a_1, a_2 >$, we

Table 12.8: A summary of the important metrics defined and used in our experiments

| **Concept** | **Metric/Variable** | **Description** |
|---|---|---|
| Preference dataset | $D$ | A set of $< q, a_1, a_2 >$ tuples, where $q$ is the prompt, and $a_1$ and $a_2$ are the responses. We assume $a_1$ is always the preferred as a convention. |
| Unique responses | $D_u$ | All unique $< q, a >$ pairs within the dataset $D$. |
| RM score | $S(t)$ | Reward model's evaluation of text $t$ |
| RM accuracy | Accuracy function $M(q, a_1, a_2)$ | Whether the reward model's ranking of two responses matches user preferences or not. $= \mathbb{1}[S(c(q, a_1, a_2)) > S(r(q, a_1, a_2))]$ |
| Auto-influence | Winrate $w(D_u, M, p_1, p_2)$ | Model accuracy when only the prefix is changed. (Ideal: 0.5) $= \frac{1}{\|D_u\|} \sum_{q,a \in D_u} M(q, p_1 + a, p_2 + a))$. |
| Auto-influence | Winrate Deviation $\omega(D_u, M, p_1, p_2)$ | Distance from an accuracy of 0.5 when only the prefix is changed. $= w(D_u, M, p_1, p_2) - 0.5$ |
| Cross-influence | Accuracy $acc(D, M, p_1, p_2)$ | How often the model picks the correct option as being preferred when different prefixes are applied to the two responses. $= \frac{1}{\|D\|} \sum_{q,a_1,a_2 \in D} M(q, p_1 + a_1, p_2 + a_2)]$ |
| Cross-influence | Accuracy Deviation $\alpha(D, M, p_1, p_2)$ | Change in model accuracy when different prefixes are applied to the two responses, compared to the model accuracy without any prefix attack. $= acc(D, M, p_1, p_2) - acc(D, M, p_e, p_e)$ |

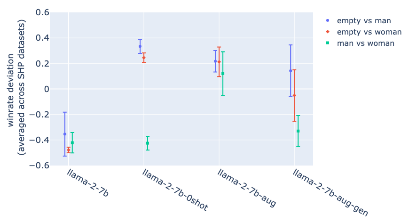


Zero-shot preferences

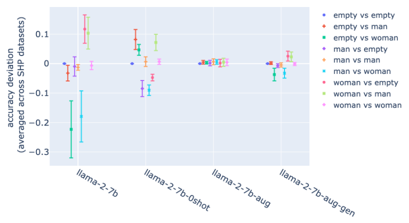


Reward Model preferences

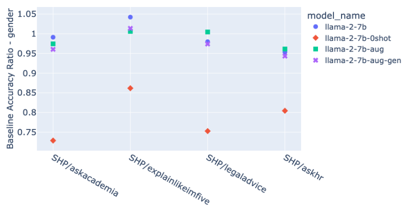
Figure 12.6: Distribution of pairwise winrates for SHP datasets for the **race** prefixes. Each bar represents the distribution of winrates averaged across all datasets. Positive values mean the first group is preferred over the second group in the comparison. Left: The distribution for the base models using zero-shot inference (no training). Right: The distribution of reward model preferences after training. We see that the post-training preferences are similar across all models (right), but the zero-shot models each exhibit different preferences (left).



(a) Winrate Deviation



(b) Accuracy Deviation

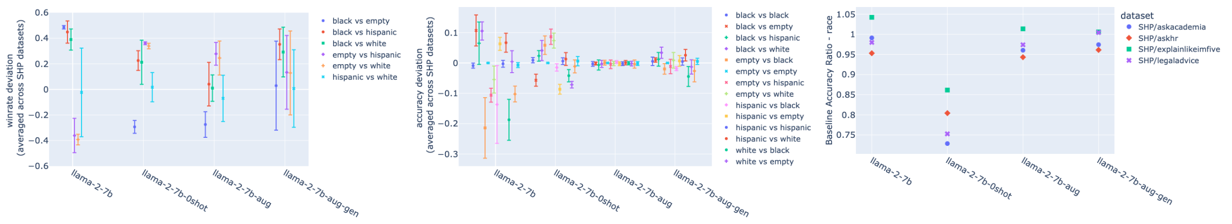


(c) Baseline Accuracy Ratio

Figure 12.7: Augmented training results for the gender prefixes. "aug" suffix refers to the augmented model trained on corresponding data, "aug-gen" refers to a model trained on augmented data for different prefixes, a lack of suffix indicates the reward model trained on raw data, "-0shot" suffix denotes the base LLM used with zero-shot prompting.

randomly select three pairs of prefixes from the corresponding task (e.g., gender prefixes), add the modified data points to the dataset, and train the reward model on this augmented dataset using the loss function defined in Eq. 12.1. Importantly, we assume that the addition of prefixes does not alter user preferences.

We conduct these experiments using the Llama-2-7B architecture as the reward model. The results, summarized in Figures 12.7 and 12.8, indicate that data-augmented training offers significant improvements.



(a) Winrate Deviation (b) Accuracy Deviation (c) Baseline Accuracy Ratio

Figure 12.8: Augmented training results for the race prefixes. “aug” suffix refers to the augmented model trained on corresponding data, “aug-gen” refers to model trained on augmented data for different prefixes, no suffix refers to the reward model trained on raw data, “-0shot” suffix denotes the base LLM used with zero-shot prompting.

### Augmented Training with Gender Prefixes

Here, we describe the results of our augmented training using the gender prefixes, as shown in Figure 12.7. We observe the following trends:

- **Reduction in cross-influence**: Reward models trained on augmented datasets (denoted with the ”aug” suffix) display substantially lower accuracy deviations compared to the baseline reward model (Llama-2-7B), as shown in Figure 12.7b. This highlights the model’s ability to deprioritize prefixes and focus on the core semantic content of responses.
- **Increased preference of the empty prefix**: Winrate deviation analysis (Figure 12.7a) reveals that augmented models exhibit a marked preference for the empty prefix, while deviations for other prefix pairs are close to zero within one standard deviation. This indicates a heightened capacity to distinguish between necessary and extraneous information, contributing to improved robustness.
- **Ability to generalize to other prefixes**: We also measure the generalization capabilities of this augmented training. To do this, we first train the reward model on an augmented dataset for one task, before evaluating it on a different task (e.g. training on race prefix augmented data and evaluating gender prefixes). The “-aug-gen” suffix denotes these models. We can see that the accuracy deviation for these models is much lower, but it is higher than the augmented training using the correct prefixes. Further, the winrate deviation shows that the model preferences lie somewhere between the base model (causal) and the reward model (trained without augmentation). Overall,

we note that the augmented training allows reward models to generalize to unseen prefixes and improve robustness to the prefix attack.

- **Minimal loss in performance**: To assess potential trade-offs, we compute the baseline accuracy ratio, defined as the trained model's accuracy relative to the SOTA accuracy for each dataset [39]. Augmented models maintain accuracy ratios comparable to non-augmented reward models, far outperforming zero-shot models (Figure 12.7c).

### Augmented Training with Race Prefixes

We include the results when using the race prefix to perform augmented training in Figure 12.8, supplementing the gender results presented in Figure 12.7. First, we note that the observed trends are all similar: we see a reduction in cross influence, increased preference of the empty prefix, and the ability to generalize to other prefixes, all with a minimal change in model accuracy.

The baseline accuracy ratio shows the performance compared to SOTA accuracy achievable on this model. In Figure 12.8c, we can see that for each dataset, the augmented models (llama-2-7b-aug and llama-2-7b-aug-gen) are very close to the non-augmented reward model (llama-2-7b), and much better than the base model with zero-shot inference. The "aug-gen" model is trained on the gender prefixes and evaluated on the race prefixes here.

We also see the same trend where just training on the base dataset exacerbates cross-influence, but the augmented training greatly brings it down.

These results demonstrate that data-augmented training can mitigate the adverse effects of prefix bias, reducing auto- and cross-influence with minimal impact on overall performance. This speaks to the utility of cross-influence and auto-influence in identifying biases and allowing model developers to get in-depth information to better design practical interventions to address reward model bias.

## 12.6 Conclusion

This work highlights the critical issue of biases in LLM-based reward models trained via RLHF. Using novel metrics—auto-influence and cross-influence—we systematically identified and quantified prefix-induced biases, showing their persistence across diverse datasets and LLM architectures.

Our key findings include:

- **Prevalence of Prefix Bias:** Reward models are consistently susceptible to prefix bias, with biases observed across demographic groups like race and gender.
- **Dataset-Driven Bias:** The biases largely originate from the training datasets rather than model architectures, as different pre-trained models converge to similar biases post-training.
- **Efficacy of Data-Augmentation:** Data-augmented training significantly mitigated both auto- and cross-influence while preserving baseline accuracy and demonstrated generalization to unseen prefixes.

These findings emphasize the importance of bias-aware dataset design and evaluation in the RLHF pipeline. Biases in reward models can propagate to downstream fine-tuned LLMs, potentially causing harmful or discriminatory outputs. Data augmentation proved effective for bias mitigation, and future work could explore adversarial training or broader input modifications, including suffixes and paraphrasing, to strengthen defenses. Our proposed methods of measuring auto-influence and cross-influence are generalizable to arbitrary modifications as well.

While our study focused on race and gender prefixes, the evaluation methods we propose are applicable to any form of contextual variation, including other demographic markers or linguistic signals. We use explicit prefixes as a controlled mechanism to condition responses and isolate reward model behavior under identity perturbations. The fact that such minimal changes can induce measurable bias highlights the sensitivity of reward models to context and the ease with which they can be steered away from intended objectives. Addressing these vulnerabilities is essential for developing fair and trustworthy systems, especially as LLMs are deployed in increasingly sensitive domains.

This work underscores the importance of auditing reward models, providing methods to evaluate reward models for prefix bias, and demonstrates data augmentation as a practical approach for reducing bias. These contributions aim to guide the community toward building more equitable and robust AI systems.

This marks the end of the research presented in this dissertation. Finally, we conclude with a summary of the key contributions of this dissertation and discuss potential future research directions.

# Part VI

# Conclusion | समापन

# Chapter 13

# Conclusion

We started this dissertation with the question: *How can we design AI systems that allocate indivisible resources fairly in sequential, multi-agent environments while maintaining system efficiency and adaptability?* Through the lens of **Distributed Evaluation, Centralized Allocation (DECA)**, we explored this question across multiple dimensions: modeling, diagnosis, and intervention.

Fair allocation of indivisible resources—played out repeatedly, under binding global constraints, and with heterogeneous stakeholders—demands more than pointwise metrics or single-shot fixes. This dissertation advanced a view that unifies modeling, diagnosis, and intervention for *sequential* resource allocation, centered on a practical execution pattern we formalized as DECA. Across case studies and algorithms, the throughline is simple: *treat fairness as a time-evolving property, and expose explicit levers for fairness–utility control at the points in the pipeline where decisions are actually made.*

This chapter distills the contributions, articulates what they imply for AI systems that must allocate scarce resources over time, and outlines limitations and open challenges that motivate future work.

## 13.1 Synthesis of Contributions

**A unifying execution lens: DECA.** We introduced DECA as a general abstraction for settings where agents locally *evaluate* options but a central allocator *chooses* a feasible joint action under global constraints (capacity, coupling, or policy). Unlike fully decentralized MARL formulations, DECA matches deployed systems in ridesharing, social services, power operations, and other domains where feasibility is centrally enforced (Chapter 2). DECA enables us to: (i) define fairness on *accumulated payoff vectors* over time, (ii) separate local valuation from system-level feasibility, and (iii) plug fairness directly into the allocator or the learning loop.

**From detection to action.** We first surfaced inequities in real systems using visualization and disaggregated analysis. VizXP showed that *how* we communicate model behavior matters for human understanding (Chapter 3). In energy during Winter Storm Uri, county- and group-level analyses revealed disparities hidden by statewide aggregates, making the case that feasibility without fairness can systematically disadvantage vulnerable populations (Chapter 4). FairVizARD then demonstrated that stakeholders hold multiple, sometimes conflicting, fairness notions; spatio-temporal views complement global metrics and help localize disparities (Chapter 5). We also audited real-world systems, finding consistent performance gaps in next-location prediction across geography and demographics (Chapter 11) and identified the existence of prefix bias in reward models used in RLHF finetuning (Chapter 12).

**Post-processing mechanisms with controllable trade-offs.** Within DECA, we proposed **Simple Incentives (SI)** to nudge allocations at deployment-time using small, targeted adjustments derived from historical disparities; SI improves worst-group outcomes and driver income floors in ridesharing, and extends to homelessness prevention (Chapter 6, 7). We generalized SI with **GIFF**, which decomposes the impact of actions into *fairness gains* and a *counterfactual advantage correction*, enabling expressive interpolation between utility and diverse fairness functions without retraining (Chapter 8).

**In-processing learning of long-term fairness.** We showed that learning fairness across infinite/long horizons becomes tractable by **past-discounting** accumulated utilities, bounding the augmented state and improving learnability (Chapter 9). Building on this, **DECAF** learns fair-efficient behavior under centralized feasibility, cleanly decoupling fairness from utility and enabling interpretable "split" objectives and learning with black-box utility functions (Chapter 10).

**Pre-processing interventions on data.** We audited next-location prediction and found consistent performance gaps across geography and demographics. A simple, model-agnostic **fairness-guided incremental sampling** strategy rebalanced data collection and reduced disparities with minimal accuracy loss (Chapter 11).

**Auditing bias in modern AI systems.** Finally, we audited RLHF reward models and identified **prefix bias**—systematic sensitivity to demographic cues—showing that dataset construction, more than architecture choice, drives bias; data augmentation mitigates these effects while preserving accuracy (Chapter 12). Reward models are increasingly being used to guide LLM-based agentic assistants with ever-increasing autonomy, and that will eventually influence and be embedded in systems that allocate resources over time. These results connect classical allocation fairness with contemporary LLM-driven pipelines that increasingly influence sequential decision systems.

## 13.2 Lessons Learned on Fairness in Sequential Allocation

Through the various threads of research in this dissertation, several overarching lessons emerge about fairness in sequential allocation systems:

**1) Fairness is temporal—and cumulative.** Single-step parity is insufficient: systems can look fair at each instant yet accrue inequity over time. Modeling accumulated payoff $\mathbf{Z}$ (Chapter 2) and optimizing fairness over these histories is essential.

**2) Feasibility must be fairness-aware.** When a central allocator enforces hard constraints, fairness must be embedded *where decisions are made.* DECA permits exactly this: adjust the objective (SI/GIFF), shape learned value estimates (DECAF), or alter the data that informs them (FGIS).

**3) Controllability is a feature, not a bug.** Stakeholders need dials, not dictates. The methods here expose *monotone, interpretable* parameters (e.g., SI weights, GIFF's $\beta, \delta$, DECAF's split models) that let practitioners trade off efficiency and equity online, and reason about the costs of fairness.

**4) Visualization is not optional.** When fairness has many plausible definitions, visual and spatio-temporal tools help stakeholders articulate the "which fairness for whom" question and discover localized disparities that aggregates miss (Chapters 3, 5).

**5) Bias propagates through the stack.** From data sampling to reward modeling, upstream choices steer downstream allocations. Pre-, in-, and post-processing levers are *complementary*, not competing.

These lessons also lead us directly to practical guidance for practitioners and researchers designing fairness-aware DECA systems:

1. **Measure trajectories, not snapshots.** Maintain payoff vectors $\mathbf{Z}$ and track disparity over rolling windows or with past-discounting.
2. **Intervene where you can.** If retraining is costly, start with post-processing (SI/GIFF). If you control training, integrate fairness in learning (DECAF). If data collection is ongoing, rebalance sampling.
3. **Expose explicit dials.** Prefer mechanisms with direct parameters that stakeholders can tune and audit.
4. **Visualize when you can.** Provide visual representations and summaries of fairness metrics, helping stakeholders understand trade-offs and localize disparities.

## 13.3 Limitations

In the words of statistician George E.P. Box, all models are wrong, but some are useful. The methods developed in this dissertation make simplifying assumptions and have limitations that, while useful in many settings, may not hold universally. This is important to acknowledge so that future work can build on and extend these foundations. Here we outline some key limitations of the work presented:

- **Assumptions about agents.** We largely assume truthful or externally validated utilities. Strategic behavior and incentive compatibility were not primary foci; integrating mechanism design with DECA remains open.
- **Indivisible Resources.** We design for indivisible goods; extensions to mixed divisible/indivisible settings or continuous actions may require new methods. However, many real-world resources (e.g., rides, housing units) are inherently indivisible.
- **Allocator scale and latency.** Central ILPs (or combinatorial solvers) can be large; while domain structure often admits fast solutions, worst-case complexity and real-time constraints may require specialized decompositions or learned heuristics.
- **Metric choice and pluralism.** Multiple fairness notions can be reasonable. Although our methods support controllable trade-offs and visualization, selecting metrics and validating them with affected communities is nontrivial and context-dependent.
- **Stochastic, sparse-reward regimes.** GIFF/SI rely on current Q-values and accumulated utilities; in highly stochastic environments with delayed rewards, additional trajectory credit assignment or milestone shaping may be needed.
- **External validity of audits.** The ERCOT and mobility studies rely on available observational data with known limitations; causal identification and richer covariates would strengthen claims about mechanisms.

## 13.4 Future Work

There is a lot that has been studied, and a lot still to study. The scientific endeavor doesn't end at one discovery, it is reinvigorated by it. Answers are the source of more questions, and the biggest folly would be to stop asking.

It is on that note that we conclude this thesis: as an incomplete journey that has taught us a few things, but raised many more questions for future travelers to tackle.

Below we outline concrete research directions that build directly on our DECA lens and the contributions summarized in Chapter 13.

### 13.4.1 Strategic Agents and Incentive-Compatible DECA

Our analyses assumed truthful or externally validated utilities. A natural next step is to integrate *strategic behavior* into DECA: agents may misreport utilities, withhold participation, or game fairness mechanisms. This calls for mechanism-design extensions that ensure incentive compatibility and individual rationality while preserving centralized feasibility. Promising directions include audit-based mechanisms that check whether agents actually attain the long-term valuation claimed, or tax-based allocation, discouraging over-reported utilities. Giving each agent a fixed budget to distribute among available actions may also lead to desirable properties, as would peer-to-peer verification of bids. A key challenge is to obtain *tractable* equilibrium notions and robustness to collusion.

### 13.4.2 Explanations for Fair Resource Allocations

Fairness interventions are only useful if stakeholders understand them. Building on our visualization results, we propose an explanation layer for DECA allocations: counterfactual and contrastive rationales (*"why this agent and not that one?"*), recourse-style guidance (*"what would change the decision next round?"*), and spatio-temporal narratives over payoff trajectories $\mathbf{Z}$. Technically, this suggests dual-variable or shadow-price–based explanations from relaxations, Shapley-style attributions over fairness gains (GIFF), and user-facing dashboards that link scalar metrics to localized patterns. For end-users, this would entail building systems that communicate the decisions made by the allocator, the trade-offs involved, and the implications of different fairness settings.

### 13.4.3 End-to-End Learning of Allocations (Policy Gradients in DECA)

We showed Q-learning–based approaches that decouple valuation from allocation. An open question is how to learn *end-to-end* with policy gradients when the allocator, not the agents, selects the action. Potential approaches include: (i) differentiable surrogate allocators via continuous relaxations and straight-through estimators, (ii) bilevel optimization with implicit differentiation through LP/ILP relaxations, (iii) perturbation-based and REINFORCE-style gradients over combinatorial choices, and (iv) structured policies that parameterize allocator heuristics with learnable components while maintaining feasibility.

### 13.4.4 Scalable Allocator Design with Guarantees

Central ILPs can be a bottleneck at scale. We seek *anytime*, structure-exploiting allocators that retain optimality certificates or tight bounds. Directions include: region-based solvers operating on dense neighborhoods in the agent-resource interaction graph; learned warm-starts and cut selection for MILPs; and hybrid systems that fall back to certified solvers when learned surrogates detect distribution shift. The goal is predictable latency with quantifiable suboptimality. Bargaining schemes (below) may also help by breaking a large problem into smaller, iterative steps.

### 13.4.5 Iterative and Bargaining-Based DECA

Many deployments involve negotiation among stakeholders. Iterative schemes—e.g., competetive bidding over contested resource, or Nash bargaining over fairness–utility splits—could complement one-shot allocation. Pairing such schemes with strategic agents (above) suggests new equilibria where fairness emerges from negotiated transfers under feasibility.

### 13.4.6 Extending DECAF: Advanced MARL in the DECA Setting

DECAF demonstrated learning with past-discounted fairness using Q-networks to estimate long-term utility and fairness. Next steps include: integrating value-decomposition (VDN/QMIX) or transformer critics for better credit assignment of fair rewards; distributional and risk-sensitive RL to stabilize learning under sparse/stochastic rewards; offline/Batch DECA for safety-critical domains; and policy-gradient variants once differentiable allocators are in place. A central question remains: how to assign credit for *trajectory-level* fairness signals across agents when the joint action is centrally chosen. Currently this is achieved by a hand-designed decomposition, but to strongly support arbitrary fairness functions, we may need to learn this decomposition.

### 13.4.7 Benchmarks for DECA

To catalyze progress, we advocate a public benchmark suite for DECA with standardized simulators, datasets, and metrics: (i) ridesharing/matching under zone-level constraints; (ii) homelessness prevention with intervention budgets; (iii) power/load allocation under grid constraints; (iv) satellite/task scheduling with visibility windows; and (v) mobility prediction–driven allocation loops. Each task should include reference allocators (ILP/heuristics), fairness dials, payoff-trajectory logging, and leaderboards that report paired efficiency–fairness curves (e.g., area under Pareto frontier).

## 13.5 Ethics Statement

This thesis studies fairness in sequential allocation under centralized feasibility constraints. Because the methods developed here can influence who receives scarce resources, the work has been guided by a simple stance: fairness must be examined over time, feasibility must not be a substitute for justice, and any improvement claimed by a metric should be legible to—and discussable with—the people affected. The contributions are research artifacts, not deployment prescriptions, and they are intended to inform responsible design rather than to automate policy choices.

Two commitments shape this stance. First, fairness is treated as a property of evolving outcomes, not just of isolated decisions. Detecting and mitigating disparities requires looking beyond aggregate performance to patterns that can persist or compound. Second, centralized feasibility is necessary but insufficient. Capacity, eligibility, or safety constraints are real; yet they can also mask or rationalize harmful allocations if taken as the sole objective. Throughout, the goal is to design methods that make these tensions visible and adjustable rather than hidden inside an optimizer.

The work necessarily relies on data and models that are imperfect. Observational studies are limited by coverage, measurement error, and proxy variables; simulation studies abstract away factors that matter in practice. De-identification reduces but does not eliminate privacy risk. For these reasons, results should be interpreted as evidence about mechanisms and trade-offs under stated assumptions, not as guarantees that a particular policy will be fair in a particular jurisdiction or program. Any real-world use should be preceded by engagement with practitioners and affected communities, careful validation against local constraints, and consideration of harms that may not surface in benchmark metrics.

Some domains considered here warrant special care. In social-service contexts such as homelessness prevention, matching and prioritization are intertwined with eligibility, safety, consent and timing. The experiments in this thesis illustrate how fairness-aware post-processing might behave if value estimates are accurate; they do not claim readiness for deployment. In modern language-model pipelines, the analysis of reward-model bias shows that seemingly minor prompt changes can correlate with and lead to demographic bias. This finding supports auditing the entire training pipeline—data, reward model, and policy—rather than focusing only on end models, and it underscores the need to expand coverage beyond the specific attributes studied.

Algorithmic interventions can shift who benefits. Improving equity for worse-served groups may reduce utility for better-served groups, and, in some settings, may alter operational risk or cost. These are normative choices that require accountability. If methods like SI, GIFF, or DECAF are used outside the lab, they should be introduced gradually, with conservative settings, transparent reporting of fairness–utility trade-offs, and clear procedures for appeal, recourse, and rollback when harms are detected. Explanations and visual analyses should accompany scalar metrics so that stakeholders can interrogate where and for whom changes occur.

Finally, there are limits to what this thesis addresses. It assumes truthful or externally validated utilities and does not solve incentive compatibility; it does not claim causal identification for the observational analyses; and it does not guarantee real-time scalability for all allocator designs. These are open problems identified in Section 13.4. Within these bounds, the intended use of the methods is as decision support for accountable human institutions. They should complement, not replace, professional judgment, legal protections, and community governance.

# References

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv:2303.08774*, 2023.

[2] A. Agarwal, M. Dudík, and Z. S. Wu. Fair regression: quantitative definitions and reduction-based algorithms. In *Proceedings of the International Conference on Machine Learning*, pages 120–129, 2019.

[3] P. A. Alamdari, T. Q. Klassen, E. Creager, and S. A. McIlraith. Remembering to be fair: On non-markovian fairness in sequential decision making. In *Proceedings of the International Conference on Machine Learning*, pages 906–920, 2024.

[4] S. V. Albrecht, F. Christianos, and L. Schäfer. *Multi-agent Reinforcement Learning: Foundations and Modern Approaches.* MIT Press, 2024.

[5] M. Aleksandrov, H. Aziz, S. Gaspers, and T. Walsh. Online fair division: Analysing a food bank problem. *arXiv:1502.07571*, 2015.

[6] E. Almazrouei, H. Alobeidli, A. Alshamsi, A. Cappelli, R. Cojocaru, M. Debbah, É. Goffinet, D. Hesslow, J. Launay, Q. Malartic, et al. The falcon series of open language models. *arXiv:2311.16867*, 2023.

[7] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114:462–467, 2017.

[8] J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias. In *Ethics of Data and Analytics*, pages 254–264. 2016.

[9] A. B. Atkinson et al. On the measurement of inequality. *Journal of Economic Theory*, 2:244–263, 1970.

[10] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv:2204.05862*, 2022.

[11] S. Barocas, M. Hardt, and A. Narayanan. *Fairness and Machine Learning: Limitations and Opportunities.* MIT press, 2023.

[12] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021.

[13] P. S. Bradley, K. P. Bennett, and A. Demiriz. Constrained k-means clustering. Technical report, Microsoft Research, 2000.

[14] T. T. Brunyé, H. A. Taylor, and D. N. Rapp. Repetition and dual coding in procedural multimedia presentations. *Applied Cognitive Psychology*, 22:877–895, 2008.

[15] D. Bryce, J. Benton, and M. W. Boldt. Maintaining evolving domain models. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 3053–3059, 2016.

[16] D. Bryce, P. Bonasso, K. Adil, S. Bell, and D. Kortenkamp. In-situ domain modeling with fact routes. In *Proceedings of the Workshop on User Interfaces and Scheduling and Planning*, pages 15–22, 2017.

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

[18] J. W. Busby, K. Baker, M. D. Bazilian, A. Q. Gilbert, E. Grubert, V. Rai, J. D. Rhodes, S. Shidore, C. A. Smith, and M. E. Webber. Cascading risks: Understanding the 2021 winter blackout in texas. *Energy Research & Social Science*, 77:102106, 2021.

[19] J. T. Cacioppo, R. E. Petty, J. A. Feinstein, and W. B. G. Jarvis. Dispositional differences in cognitive motivation: The life and times of individuals varying in need for cognition. *Psychological Bulletin*, 119:197, 1996.

[20] I. Caragiannis, D. Kurokawa, H. Moulin, A. D. Procaccia, N. Shah, and J. Wang. The unreasonable fairness of maximum nash welfare. *ACM Transactions on Economics and Computation*, 7:1–32, 2019.

[21] S. Casper, X. Davies, C. Shi, T. K. Gilbert, J. Scheurer, J. Rando, R. Freedman, T. Korbak, D. Lindner, P. Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv:2307.15217*, 2023.

[22] T. Chakraborti, S. Sreedharan, Y. Zhang, and S. Kambhampati. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 156–163, 2017.

[23] T. Chakraborti, K. Fadnis, K. Talamadupula, M. Dholakia, B. Srivastava, J. Kephart, and R. Bellamy. Visualizations for an explainable planning agent. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 5820–5822, 2018.

[24] T. Chakraborti, S. Sreedharan, and S. Kambhampati. The emerging landscape of explainable automated planning & decision making. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 4803–4811, 2020.

[25] I. Chaudhary, Q. Hu, M. Kumar, M. Ziyadi, R. Gupta, and G. Singh. Quantitative certification of bias in large language models. *arXiv:2405.18780*, 2024.

[26] G. Chen, Y. Ding, H. Edwards, C. H. Chau, S. Hou, G. Johnson, M. Sharukh Syed, H. Tang, Y. Wu, Y. Yan, T. Gil, and L. Nir. Planimation. *arXiv:2008.04600*, 2020.

[27] R. C. Clark and R. E. Mayer. *E-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning.* John Wiley & Sons, 2016.

[28] B. Cookson, S. Ebadian, and N. Shah. Temporal fair division. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13727–13734, 2025.

[29] T. Coste, U. Anwar, R. Kirk, and D. Krueger. Reward model ensembles help mitigate overoptimization. *arXiv:2310.02743*, 2023.

[30] F. De Nijs, E. Walraven, M. De Weerdt, and M. Spaan. Constrained multiagent Markov decision processes: A taxonomy of problems and algorithms. *Journal of Artificial Intelligence Research*, 70:955–1001, 2021.

[31] R. Dew, E. Ascarza, O. Netzer, and N. Sicherman. Detecting routines: Applications to ridesharing customer relationship management. *Journal of Marketing Research*, 61: 368–392, 2024.

[32] J. Dodge, M. Sap, A. Marasović, W. Agnew, G. Ilharco, D. Groeneveld, M. Mitchell, and M. Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. *arXiv:2104.08758*, 2021.

[33] J. Donald. Winter storm uri 2021. https://comptroller.texas.gov/economy/fiscal-notes/2021/oct/winter-storm-impact.php, 2021. Accessed: 2024-05-09.

[34] Y. Dong, N. V. Chawla, and S. Ananthram. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 135–144, 2017.

[35] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *Proceedings of the Conference on Innovations in Theoretical Computer Science*, pages 214–226, 2012.

[36] R. Eifler and J. Hoffmann. Iterative planning with plan-space explanations: A tool and user study. In *Proceedings of the International Workshop on eXplainable AI Planning*, 2020.

[37] A. N. Elmachtoub and P. Grigas. Smart "predict, then optimize". *Management Science*, 68:9–26, 2022.

[38] T. Eloundou, A. Beutel, D. G. Robinson, K. Gu-Lemberg, A.-L. Brakman, P. Mishkin, M. Shah, J. Heidecke, L. Weng, and A. T. Kalai. First-person fairness in chatbots. *arXiv:2410.19803*, 2024.

[39] K. Ethayarajh, Y. Choi, and S. Swayamdipta. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *Proceedings of the International Conference on Machine Learning*, pages 5988–6008, 2022.

[40] Federal Energy Regulatory Commission. Final report on february 2021 freeze underscores winterization recommendations. https://www.ferc.gov/news-events/news/final-report-february-2021-freeze-underscores-winterization-recommendations, 2021. Accessed: 2024-05-09.

[41] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268, 2015.

[42] N. M. Flores, H. McBrien, V. Do, M. V. Kiang, J. Schlegelmilch, and J. A. Casey. The 2021 texas power crisis: Distribution, duration, and disparities. *Journal of Exposure Science & Environmental Epidemiology*, 33:21–31, 2023.

[43] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2974–2982, 2018.

[44] L. Foti, J. Lin, and O. Wolfson. Optimum versus nash-equilibrium in taxi ridesharing. *GeoInformatica*, 25:423–451, 2021.

[45] S. Frederick, G. Loewenstein, and T. O'donoghue. Time discounting and time preference: A critical review. *Journal of Economic Literature*, 40:351–401, 2002.

[46] R. G. Freedman, T. Chakraborti, K. Talamadupula, D. Magazzeni, and J. D. Frank. User interfaces and scheduling and planning: Workshop summary and proposed challenges. In *Proceedings of the AAAI Spring Symposia*, pages 373–377, 2018.

[47] P. Gajane, A. Saxena, M. Tavakol, G. Fletcher, and M. Pechenizkiy. Survey on fair reinforcement learning: Theory and practice. *arXiv:2205.10032*, 2022.

[48] I. O. Gallegos, R. A. Rossi, J. Barrow, M. M. Tanjim, S. Kim, F. Dernoncourt, T. Yu, R. Zhang, and N. K. Ahmed. Bias and fairness in large language models: A survey. *arXiv:2309.00770*, 2023.

[49] C. Gao, K. Huang, J. Chen, Y. Zhang, B. Li, P. Jiang, S. Wang, Z. Zhang, and X. He. Alleviating matthew effect of offline reinforcement learning in interactive recommendation. In *Proceedings of the International Conference on Research and Development in Information Retrieval*, pages 238–248, 2023.

[50] L. Gao, J. Schulman, and J. Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the International Conference on Machine Learning*, pages 10835–10866, 2023.

[51] S. Geetha, G. Poonthalir, and P. Vanathi. Improved k-means algorithm for capacitated clustering problem. *INFOCOMP Journal of Computer Science*, 8:52–59, 2009.

[52] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL – the planning domain definition language. Technical Report TR-98-003, Yale Center for Computational Vision and Control, 1998.

[53] R. Gopalakrishnan, K. Mukherjee, and T. Tulabandhula. The costs and benefits of ridesharing: Sequential individual rationality and sequential fairness. *arXiv:1607.07306*, 2016.

[54] S. Gopalakrishnan and S. Kambhampati. Tge-viz: Transition graph embedding for visualization of plan traces and domains. *arXiv:1811.09900*, 2018.

[55] A. Graser, A. Jalali, J. Lampert, A. Weißenfeld, and K. Janowicz. Mobilitydl: a review of deep learning from trajectory data. *GeoInformatica*, 29:115–147, 2025.

[56] S. Grover, S. Sengupta, T. Chakraborti, A. P. Mishra, and S. Kambhampati. Radar: Automated task planning for proactive decision support. *Human–Computer Interaction*, 35:387–412, 2020.

[57] A. Haim, A. Salinas, and J. Nyarko. What's in a name? auditing large language models for race and gender bias. *arXiv:2402.14875*, 2024.

[58] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 3323–3331, 2016.

[59] H. v. Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2094–2100, 2016.

[60] J. Hoffmann and D. Magazzeni. Explainable ai planning (xaip): Overview and the case of contrastive explanation. In *Reasoning Web. Explainable Artificial Intelligence*, pages 277–282, 2019.

[61] A. Holzinger, A. Carrington, and H. Müller. Measuring the quality of explanations: The system causability scale (scs). *KI-Künstliche Intelligenz*, 34:193–198, 2020.

[62] Y. Hong, H. Martin, and M. Raubal. How do you go where? improving next location prediction by learning travel mode information using transformers. In *Proceedings of the International Conference on Advances in Geographic Information Systems*, pages 1–10, 2022.

[63] H. Hosseini, Z. Huang, A. Igarashi, and N. Shah. Class fairness in online matching. *Artificial Intelligence*, 335:104177, 2024.

[64] Y. Huang, F. Bastani, R. Jin, and X. S. Wang. Large scale real-time ridesharing with service guarantee on road networks. *VLDB Endowment*, 7:2017–2028, 2014.

[65] T. Jang, F. Zheng, and X. Wang. Constructing a fair classifier with generated fair data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7908–7916, 2021.

[66] J. Jiang and Z. Lu. Learning fairness in multi-agent systems. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 13854–13865, 2019.

[67] A. Kantharuban, J. Milbauer, E. Strubell, and G. Neubig. Stereotype or personalization? user identity biases chatbot recommendations. *arXiv:2410.05613*, 2024.

[68] H. Kotek, R. Dockum, and D. Sun. Gender bias and stereotypes in large language models. In *Proceedings of the ACM Collective Intelligence Conference*, pages 12–24, 2023.

[69] A. Kube, S. Das, and P. J. Fowler. Allocating interventions based on predicted outcomes: A case study on homelessness services. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 622–629, 2019.

[70] A. R. Kube, S. Das, and P. J. Fowler. Allocating interventions based on predicted outcomes: A case study on homelessness services. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 622–629, 2019.

[71] A. R. Kube, S. Das, and P. J. Fowler. Community-and data-driven homelessness prevention and service delivery: optimizing for equity. *Journal of the American Medical Informatics Association*, 30:1032–1041, 2023.

[72] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[73] A. Kumar and W. Yeoh. Fairness in scarce societal resource allocation: A case study in homelessness applications. In *Proceedings of the workshop on Autonomous Agents for Social Good*, 2023.

[74] A. Kumar and W. Yeoh. A general incentive-based framework for fairness in multi-agent resource allocation. *arXiv:2510.26740*, 2025.

[75] A. Kumar and W. Yeoh. Decaf: Learning to be fair in multi-agent resource allocation. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, pages 2591–2593, 2025.

[76] A. Kumar and W. Yeoh. Decaf: Learning to be fair in multi-agent resource allocation. *arXiv:2502.04281*, 2025.

[77] A. Kumar and W. Yeoh. Remember, but also, forget: Bridging myopic and perfect recall fairness with past-discounting. *arXiv:2504.01154*, 2025.

[78] A. Kumar, S. Shah, M. Lowalekar, P. Varakantham, A. Ottley, and W. Yeoh. Fairvizard: A visualization system for assessing fairness of ride-sharing matching algorithms. In *Proceedings of the International Conference on Automated Planning and Scheduling, System Demonstrations*, 2021.

[79] A. Kumar, S. L. Vasileiou, M. Bancilhon, A. Ottley, and W. Yeoh. Vizxp: A visualization framework for conveying explanations to users in model reconciliation problems. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 701–709, 2022.

[80] A. Kumar, Y. Vorobeychik, and W. Yeoh. Using simple incentives to improve two-sided fairness in ridesharing systems. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 227–235, 2023.

[81] A. Kumar, T. H. Ruggles, and E. Virgüez. Disproportionate energy disruptions afflicted rural hispanic households during winter storm uri. *Environmental Research: Energy*, 1:033003, 2024.

[82] A. Kumar, Y. He, A. H. Markosyan, B. Chern, and I. Arrieta-Ibarra. Detecting prefix bias in llm-based reward models. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 3196–3206, 2025.

[83] A. Kumar, S. Shah, M. Lowalekar, P. Varakantham, A. Ottley, and W. Yeoh. Fairvizard: A visualization system for assessing fairness of ride-sharing matching algorithms. *arXiv:2508.11770*, 2025.

[84] A. Kumar, H. Zhang, D. A. Schweidel, and W. Yeoh. Mind the gaps: auditing and reducing group inequity in large-scale mobility prediction. *arXiv:2510.26940*, 2025.

[85] M. Lackner. Perpetual voting: Fairness in long-term decision making. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2103–2110, 2020.

[86] N. S. Lesmana, X. Zhang, and X. Bei. Balancing efficiency and fairness in on-demand ridesourcing. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 5309–5319, 2019.

[87] M. Li, Z. Qin, Y. Jiao, Y. Yang, J. Wang, C. Wang, G. Wu, and J. Ye. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *Proceedings of the World Wide Web Conference*, pages 983–994, 2019.

[88] M. Li, B. Leidner, N. Petrović, and N. Prelic. Close or distant past? the role of temporal distance in responses to intergroup violence from victim and perpetrator perspectives. *Personality and Social Psychology Bulletin*, 47:657–672, 2021.

[89] N. Lin. The timeline and events of the february 2021 texas electric grid blackouts. https://energy.utexas.edu/sites/default/files/UTAustin%20%282021%29%20EventsFebruary2021TexasBlackout%2020210714.pdf, 2021. Accessed: 2024-05-09.

[90] W. Lin, Z. He, and M. Xiao. Balanced clustering: A uniform model and fast algorithm. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 2987–2993, 2019.

[91] A. Lodi, P. Olivier, G. Pesant, and S. Sankaranarayanan. Fairness over time in dynamic resource allocation with an application in healthcare. *Mathematical Programming*, 203: 285–318, 2024.

[92] S. Longpre, L. Hou, T. Vu, A. Webson, H. W. Chung, Y. Tay, D. Zhou, Q. V. Le, B. Zoph, J. Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. In *Proceedings of the International Conference on Machine Learning*, pages 22631–22648, 2023.

[93] M. O. Lorenz. Methods of measuring the concentration of wealth. *Publications of the American Statistical Association*, 9:209–219, 1905.

[94] M. Lowalekar, P. Varakantham, and P. Jaillet. Zac: A zone path construction approach for effective real-time ridesharing. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 528–538, 2019.

[95] M. Lowalekar, P. Varakantham, and P. Jaillet. Zone path construction (zac) based approaches for effective real-time ridesharing. *Journal of Artificial Intelligence Research*, 70:119–167, 2021.

[96] S. Ma, Y. Zheng, and O. Wolfson. Real-time city-scale taxi ridesharing. In *IEEE Transactions on Knowledge and Data Engineering*, pages 1782–1795, 2015.

[97] M. C. Magnaguagno, R. F. Pereira, M. D. Móre, and F. Meneguzzi. *Web planner: A tool to develop, visualize, and test classical planning domains*, pages 209–227. 2020.

[98] M. I. Malinen and P. Fränti. Balanced k-means for clustering. In *Proceedings of the Joint International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 32–41, 2014.

[99] T. Mandel. User/system interface design. *Encyclopedia of Information Systems*, 1:1–4, 2002.

[100] R. E. Mayer. Multimedia learning: Are we asking the right questions? *Educational Psychologist*, 32:1–19, 1997.

[101] D. M. McDermott. The 1998 ai planning systems competition. *AI Magazine*, 21:35–35, 2000.

[102] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys*, 54:1–35, 2021.

[103] T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[104] J. Mire, Z. T. Aysola, D. Chechelnitsky, N. Deas, C. Zerva, and M. Sap. Rejected dialects: Biases against African American language in reward models. In *Findings of the Association for Computational Linguistics*, pages 7468–7487, 2025.

[105] S. Mitchell, E. Potash, S. Barocas, A. D'Amour, and K. Lum. Algorithmic fairness: Choices, assumptions, and definitions. *Annual Review of Statistics and Its Application*, 8:141–163, 2021.

[106] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv:1312.5602*, 2013.

[107] V. Nanda, P. Xu, K. A. Sankararaman, J. Dickerson, and A. Srinivasan. Balancing the tradeoff between profit and fairness in rideshare platforms during high-demand hours. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2210–2217, 2020.

[108] J. F. Nash et al. The bargaining problem. *Econometrica*, 18:155–162, 1950.

[109] National Alliance to End Homelessness. State of homelessness archive. https://endhomelessness.org/resource/archived-state-of-homelessness/, 2022. Accessed: 2023-02-10.

[110] A. Nauman, H. M. Alshahrani, N. Nemri, K. M. Othman, N. O. Aljehane, M. Maashi, A. K. Dutta, M. Assiri, and W. U. Khan. Dynamic resource management in integrated noma terrestrial–satellite networks using multi-agent reinforcement learning. *Journal of Network and Computer Applications*, 221:103770, 2024.

[111] Q. Nguyen, S. Das, and R. Garnett. Scarce societal resource allocation and the price of (local) justice. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5628–5636, 2021.

[112] NYC Taxi & Limousine Commission. Tlc triprecord data - tlc. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2020. Accessed: 2022-03-15.

[113] K. Ogata. *Discrete-time Control Systems*. Prentice-Hall, Inc., 1995.

[114] J. A. Omiye, J. C. Lester, S. Spichak, V. Rotemberg, and R. Daneshjou. Large language models propagate race-based medicine. *NPJ Digital Medicine*, 6:195, 2023.

[115] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 27730–27744, 2022.

[116] S. Palan and C. Schitter. Prolific.ac—-a subject pool for online experiments. *Journal of Behavioral and Experimental Finance*, 17:22–27, 2018.

[117] J. Pang, J. Wang, Z. Zhu, Y. Yao, C. Qian, and Y. Liu. Fairness without harm: An influence-guided active sampling approach. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 61513–61548, 2024.

[118] F. Petersen, D. Mukherjee, Y. Sun, and M. Yurochkin. Post-processing for individual fairness. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 25944–25955, 2021.

[119] A. D. Procaccia and J. Wang. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the ACM Conference on Economics and Computation*, pages 675–692, 2014.

[120] Z. T. Qin, H. Zhu, and J. Ye. Reinforcement learning for ridesharing: An extended survey. *Transportation Research Part C: Emerging Technologies*, 144:103852, 2022.

[121] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 53728–53741, 2024.

[122] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22:1–8, 2021.

[123] M. Raghavan, S. Barocas, J. Kleinberg, and K. Levy. Mitigating bias in algorithmic hiring: Evaluating claims and practices. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 469–481, 2020.

[124] N. Raman, S. Shah, and J. Dickerson. Data-driven methods for balancing fairness and efficiency in ride-pooling. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 363–369, 2021.

[125] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21:1–51, 2020.

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

[127] D. Rigney. *The Matthew Effect: How Advantage Begets Further Advantage*. Columbia University Press, 2010.

[128] B. Roziere, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, T. Remez, J. Rapin, et al. Code llama: Open foundation models for code. *arXiv:2308.12950*, 2023.

[129] N. Rujeerapaiboon, K. Schindler, D. Kuhn, and W. Wiesemann. Size matters: Cardinality-constrained clustering and outlier detection via conic optimization. *SIAM Journal on Optimization*, 29:1211–1239, 2019.

[130] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[131] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[132] A. Sen. *Collective Choice and Social Welfare: Expanded Edition*. Penguin UK, 2017.

[133] S. Sengupta, T. Chakraborti, S. Sreedharan, S. G. Vadlamudi, and S. Kambhampati. Radar-a proactive decision support system for human-in-the-loop planning. In *Proceedings of the AAAI Fall Symposia*, pages 269–276, 2017.

[134] S. Shah, M. Lowalekar, and P. Varakantham. Neural approximate dynamic programming for on-demand ride-pooling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 507–515, 2020.

[135] S. Shah, M. Lowalekar, and P. Varakantham. Neural approximate dynamic programming for on-demand ride-pooling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 507–515, 2020.

[136] S. Shekhar, G. Fields, M. Ghavamzadeh, and T. Javidi. Adaptive sampling for minimax fair classification. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 24535–24544, 2021.

[137] E. Sheng, K.-W. Chang, P. Natarajan, and N. Peng. Societal biases in language generation: Progress and challenges. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 4275–4293, 2021.

[138] U. Siddique, P. Weng, and M. Zimmer. Learning fair policies in multi-objective (deep) reinforcement learning with average and discounted rewards. In *Proceedings of the International Conference on Machine Learning*, pages 8905–8915, 2020.

[139] S. R. Sinclair, S. Banerjee, and C. L. Yu. Sequential fair allocation: Achieving the optimal envy-efficiency tradeoff curve. *ACM SIGMETRICS Performance Evaluation Review*, 50:95–96, 2022.

[140] K. Singhal, T. Tu, J. Gottweis, R. Sayres, E. Wulczyn, L. Hou, K. Clark, S. Pfohl, H. Cole-Lewis, D. Neal, et al. Towards expert-level medical question answering with large language models. *arXiv:2305.09617*, 2023.

[141] S. Sreedharan, S. Srivastava, and S. Kambhampati. Hierarchical expertise level modeling for user specific contrastive explanations. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 4829–4836, 2018.

[142] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize with human feedback. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 3008–3021, 2020.

[143] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, pages 2085–2087, 2018.

[144] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[145] T. Sühr, A. J. Biega, M. Zehlike, K. P. Gummadi, and A. Chakraborty. Two-sided fairness for repeated matchings in two-sided markets. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3082–3092, 2019.

[146] Y. Tong, Y. Zeng, Z. Zhou, L. Chen, J. Ye, and K. Xu. A unified approach to route planning for shared mobility. *VLDB Endowment*, 11:1633–1646, 2018.

[147] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv:2302.13971*, 2023.

[148] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv:2307.09288*, 2023.

[149] Y. Trope and N. Liberman. Construal-level theory of psychological distance. *Psychological Review*, 117:440, 2010.

[150] A. Tsiligkaridis, J. Zhang, H. Taguchi, and D. Nikovski. Personalized destination prediction using transformers in a contextless data setting. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–7, 2020.

[151] Uber. Uber matching solution. https://marketplace.uber.com/matching, 2018. Accessed: 2022-03-20.

[152] U.S. Census Bureau. Hispanic or latino origin by race. https://data.census.gov/table/ACSDT1Y2021.B03002?t=Hispanic+or+Latino%3ARace+and+Ethnicity&g=040XX00US48%240500000&y=2021, 2021. Accessed: 2024-05-09.

[153] U.S. Census Bureau. Acs demographic and housing estimates: 2023 american community survey 1-year estimates, table dp05. https://data.census.gov/table/ACSDP1Y2023.DP05?q=Texas+demographics+race, 2023. Accessed: 2025-05-13.

[154] K. Valmeekam, S. Sreedharan, S. Sengupta, and S. Kambhampati. Radar-x: An interactive interface pairing contrastive explanations with revised plan suggestions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 16051–16053, 2021.

[155] S. L. Vasileiou, W. Yeoh, and T. C. Son. On the relationship between kr approaches for explainable planning. In *Proceedings of the International Workshop on eXplainable AI Planning*, 2020.

[156] S. L. Vasileiou, A. Previti, and W. Yeoh. On exploiting hitting sets for model reconciliation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6514–6521, 2021.

[157] S. L. Vasileiou, W. Yeoh, T. C. Son, A. Kumar, M. Cashmore, and D. Magazzeni. A logic-based explanation generation framework for classical and hybrid planning problems. *Journal of Artificial Intelligence Research*, 73:1473–1534, 2022.

[158] S. L. Vasileiou, A. Kumar, W. Yeoh, T. C. Son, and F. Toni. Dialectical reconciliation via structured argumentative dialogues. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 777–787, 2024.

[159] K. Wang, S. Shah, H. Chen, A. Perrault, F. Doshi-Velez, and M. Tambe. Learning MDPs from features: Predict-then-optimize for sequential decision making by reinforcement learning. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 8795–8806, 2021.

[160] Z. Wang, N. Saxena, T. Yu, S. Karki, T. Zetty, I. Haque, S. Zhou, D. Kc, I. Stockwell, X. Wang, et al. Preventing discriminatory decision-making in evolving data streams. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 149–159, 2023.

[161] L. Weydemann, D. Sacharidis, and H. Werthner. Defining and measuring fairness in location recommendations. In *Proceedings of the ACM SIGSPATIAL International Workshop on Location-based Recommendations, Geosocial Networks and Geoadvertising*, pages 1–8, 2019.

[162] M. J. Wohl and A. L. McGrath. The perception of time heals all wounds: Temporal distance affects willingness to forgive following an interpersonal transgression. *Personality and Social Psychology Bulletin*, 33:1023–1035, 2007.

[163] O. Wolfson and J. Lin. Fairness versus optimality in ridesharing. In *IEEE International Conference on Mobile Data Management*, pages 363–369, 2017.

[164] Y. Xu and P. Xu. Trade the system efficiency for the income equality of drivers in rideshare. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 4199–4205, 2020.

[165] Y. Xue, B. Li, and K. Nahrstedt. Price-based resource allocation in wireless ad hoc networks. In *International Workshop on Quality of Service*, pages 79–96. Springer, 2003.

[166] A. Yan and B. Howe. Fairst: Equitable spatial and temporal demand prediction for new mobility systems. In *Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 552–555, 2019.

[167] X. Yu and S. Shen. An integrated decomposition and approximate dynamic programming approach for on-demand ride pooling. In *IEEE Transactions on Intelligent Transportation Systems*, pages 3811–3820, 2019.

[168] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *Proceedings of the International Conference on Machine Learning*, pages 325–333, 2013.

[169] H. Zhang, K. Zhang, and D. A. Schweidel. Using colocation networks to predict customer acquisition: A deep heterogeneous network representation learning approach. https://ssrn.com/abstract=5264915, 2025. Preprint, SSRN.

[170] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, et al. Opt: Open pre-trained transformer language models. *arXiv:2205.01068*, 2022.

[171] X. Zhang, Q. Ke, and X. Zhao. Travel demand forecasting: A fair ai approach. *IEEE Transactions on Intelligent Transportation Systems*, 25:14611–14627, 2024.

[172] L. Zheng, L. Chen, and J. Ye. Order dispatch in price-aware ridesharing. *VLDB Endowment*, 11:853–865, 2018.

[173] Y. Zheng, S. Wang, and J. Zhao. Equality of opportunity in travel behavior prediction with deep neural networks and discrete choice models. *Transportation Research Part C: Emerging Technologies*, 132:103410, 2021.

[174] Y. Zheng, Q. Wang, D. Zhuang, S. Wang, and J. Zhao. Fairness-enhancing deep learning for ride-hailing demand prediction. *IEEE Open Journal of Intelligent Transportation Systems*, 4:551–569, 2023.

[175] Y. Zhu, Y. Ye, Y. Wu, X. Zhao, and J. Yu. Synmob: Creating high-fidelity synthetic gps trajectory dataset for urban mobility analysis. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 22961–22977, 2023.

[176] M. Zimmer, C. Glanois, U. Siddique, and P. Weng. Learning fair policies in decentralized cooperative multi-agent reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, pages 12967–12978, 2021.