EFaST: An Explainable Framework for Fair Sequential Task Assignment

Anna Dalla Vecchia[1][0000−0001−7026−5205], Sara Migliorini[1][0000−0003−3675−7243], Elisa Quintarelli[1][0000−0001−6092−6831], and Kostas Stefanidis[2][0000−0003−1317−8062]

[1] Department of Computer Science, University of Verona, Verona, Italy
{name.surname}@univr.it
[2] Data Science Research Centre, Tampere University, Tampere, Finland
konstantinos.stefanidis@tuni.fi

Abstract. In this demo, we present EFaST, an interactive framework for exploring and understanding fairness in multi-stakeholder sequential task assignment problems. EFaST allows users to iteratively define and modify stakeholder soft constraints, execute a fairness-aware optimization process, and observe how adjustments to these constraints influence fairness outcomes. Fairness is evaluated at two complementary levels: *local fairness* captures the satisfaction of individual stakeholders, while *global fairness* jointly measures overall satisfaction and balance across stakeholders. Users can inspect intermediate solutions, explore fairness trade-offs arising from conflicting constraints, and request natural-language explanations grounded in structured outputs. These explanations summarize constraint satisfaction, highlight critical violations, and provide guidance on potential constraint refinement, making fairness dynamics observable and controllable.

Keywords: Sequential tasks · Multi-stakeholder fairness · Explanations.

1 Introduction

Incorporating fairness as a core ethical principle enhances productivity, positivity, long-term relationships, job satisfaction, and goal achievement within organizations and, more generally, in various social environments [4]. Fair task distribution ensures equal opportunities for individuals to showcase their skills, while also preventing bias in recommendation systems and promoting inclusivity [3, 1]. However, fairness is a complex, context-dependent concept that varies across cultures and stakeholder perspectives [5]. This paper presents EFaST (Explainable FAir Sequential Task assignment), a framework for determining fair sequential task assignments among multiple stakeholders, a problem with competing interests and constraints that must be managed over time. The term *stakeholder* refers to any individual or group affected by or influencing task assignments. In the presence of multiple stakeholders, achieving fairness involves balancing the needs of all of them and, thus, is a process that involves multiple points of view.

We distinguish between local fairness, which measures satisfaction for a single stakeholder, and global fairness, which captures the overall balance across all stakeholders. Achieving optimal global fairness is a challenge, as a perfectly fair solution for all may not always exist.

Case study. We consider the problem of timetable creation in the university domain, where professors and students share lessons, have constraints, and they also may have preferences. In this case, the goal is to automatically generate a fair timetable, with fairness evaluated from the perspectives of both stakeholder groups. EFaST addresses this by solving two sub-problems separately to ensure local fairness for students and each professor, then combines the found best local solutions to produce a globally fair timetable.

Contribution. EFaST tackles the challenge of assigning shared tasks in domains with multiple stakeholders, aiming to ensure fairness by respecting each party's preferences and constraints. It relies on the FaST-AMOSA optimization algorithm [2], a multi-objective simulated annealing approach tailored for fair sequential task assignment. This demonstration focuses on making the fairness computation explainable and fully interactive in practice, allowing stakeholders to inspect, modify, and iteratively refine fairness configurations in real time. The key contributions provided are: (i) an interactive constraint specification layer that allows non-technical stakeholders to directly manipulate time slot preferences without requiring them to disclose motivations; (ii) real-time monitoring of the task assignment process, allowing users to observe how local and global fairness evolve across iterations; and (iii) the integration of a natural language explanation layer that complements numerical fairness metrics, helping users understand the numerical values and the unsatisfied constraints.

Demonstration. EFaST is a web-based, interactive platform that enables users to define, modify, and experiment with stakeholders' constraints and preferences in a flexible manner. Users can specify availability and preference levels directly on a weekly timetable, actively monitor the assignment process, and observe how different choices influence fairness outcomes. Once constraints are defined, the system executes the FaST-AMOSA optimization process, allowing users to explore algorithm iterations and visualize, in real time, both local and global fairness scores at each step. This interactive exploration, coupled with natural language explanations, makes fairness trade-offs explicit, helping users understand how individual constraints and preferences contribute to the overall solution. By enabling direct user control and immediate feedback, the demo supports informed decision-making and highlights the practical relevance of EFaST for analyzing and refining fairness in complex, multi-stakeholder settings.

2 EFaST Overview

EFaST is an interactive system built on top of the FaST algorithm [2] to support the exploration of fair sequential task assignment problems involving multiple stakeholders with heterogeneous preferences and constraints. Here, sequential means that assignments are performed across time slots (e.g., hours and days).

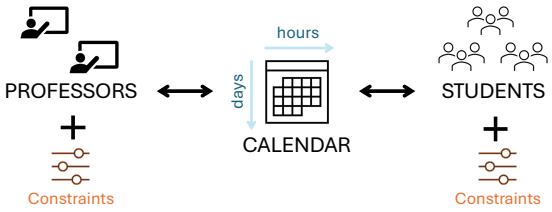

Fig. 1. Schematic representation of the considered sequential task assignment problem.

In this demo, we focus on the class scheduling problem in the university domain, where the stakeholders are represented by professors and students. As illustrated in Fig. 1, the shared resource is a calendar, in which classes are assigned to ordered time slots, yielding task sequences both within a day and across the week. Professors are modeled as individual stakeholders who can state their personal scheduling preferences. While students are represented at the cohort level, grouped by degree program and year of study, this reflects the standardized structure of university curricula.

As in [2], we distinguish between two types of constraints. *Hard constraints* are mandatory requirements that are intrinsic to the problem definition and must be satisfied by any valid solution (e.g., a professor cannot be assigned to two classes at the same time), whereas *soft constraints* encode a set of preferences and may be violated at the cost expressed by the stakeholder, e.g., a user may prefer a task in a specific time slot. Tasks (i.e., classes) are assigned over a discrete temporal domain defined by two orthogonal granularities (e.g., days and hours). Each professor specifies their preferences through a calendar where each time slot is associated with a value $n \in [-1, 0]$, where values closer to -1 indicate more undesirable slots. In addition to the calendar-based preferences, further soft constraints can be modeled through objective functions, for instance, to guarantee a lunch break. The final goal is to produce valid schedules that satisfy all hard constraints while minimizing the number of unsatisfied soft constraints.

To compute fair solutions, EFaST relies on the FaST-AMOSA algorithm, a fairness-aware extension of Multi-Objective Simulated Annealing (MOSA) [6]. Unlike traditional simulated annealing, which randomly selects perturbations at each iteration, this algorithm uses soft constraints to guide the process, accelerating convergence toward fair solutions. In line with the FaST formulation, fairness is evaluated at two complementary levels. **Local fairness** measures how well a valid solution $\mathcal{A}$ satisfies the soft constraints of a stakeholder k and is denoted as $f(\mathcal{A}, k)$. **Global fairness** assesses how evenly satisfaction is balanced across all stakeholders involved and is represented as $g(\mathcal{A}) = \langle g_a(\mathcal{A}), g_s(\mathcal{A}) \rangle$, where g_a is the average local fairness across stakeholders and g_s is its standard deviation. This formulation captures both overall satisfaction and the balance among stakeholders, preventing solutions that favor some stakeholders at the expense of others. Beyond fairness computation, this demo emphasizes fairness understanding. EFaST integrates an explanation layer that helps users interpret

fairness outcomes by highlighting how constraints are violated, which conflicts are most critical, and why certain still persist even in globally fair solutions. This enables users to inspect the underlying reasons for the final fairness score.

3 EFaST Architecture and User Interface

EFaST is implemented as a web-based interactive platform designed to support the exploration and understanding of fairness in sequential task assignment. The architecture follows a client–server model in which users interact with a web interface to define constraints and preferences, trigger the optimization process, and inspect both local and global fairness outcomes. The backend executes the FaST-AMOSA optimization algorithm and generates intermediate and final results in real time. The user interface is designed to make fairness trade-offs explicit and inspectable. Stakeholders interact with a weekly timetable view where each time slot is annotated with a preference value n, following the formalization introduced earlier. In particular, slots can be marked as available ($n = 0$), mildly undesirable ($n = -0.5$), or strongly undesirable ($n = -1$). Constraints are validated at insertion time according to administrator-defined limits to ensure feasibility. During execution, the interface visualizes the evolution of the solution by showing: (i) the student fairness across different cohorts, (ii) professor fairness, and (iii) the global fairness indicators.

The core of the web app is the explainability module, which supports the interpretation of fairness outcomes. For any solution, users can request a natural language explanation describing which constraints are violated, which stakeholders are most affected, and why specific unfair values persist. Explanations are generated through an external LLM service accessed via API. The demonstration supports interactive selection among different models, allowing the user to balance explanation quality, latency, and computational cost depending on the deployment context. LLMs have demonstrated the ability to translate structured representations into coherent natural language descriptions [7]. In EFaST, these models process fairness scores, violated constraints, and the current assignment solely to translate formal structured output into textual explanations. The prompt enforces a fixed three-part structure (fairness summary, structural conflicts, and preference adjustment) and does not introduce new numerical values or modify results. This approach enhances interpretability for non-expert users, which can be challenging in multi-stakeholder scenarios where explanations need to address multiple types of constraints and conflicting preferences.

EFaST supports incremental execution of the optimization algorithm. As the algorithm progresses, intermediate fairness results are displayed in the interface, allowing users to observe how local and global fairness evolve over time through a plot. Administrators can monitor convergence, inspect stakeholder conflicts, and request explanations at different stages of the optimization. This interactive loop enables users to explore how modifying constraints or preferences affects fairness trade-offs, transforming fairness optimization from a black-box process into an analyzable, controllable decision-support workflow.

4 Demonstration

In this section, we demonstrate how users can interact with the EFaST web application in a real-world university timetabling scenario. As illustrated in previous sections, we have two groups of stakeholders: students and professors. The latter are those that can express their own preferences in a calendar, while students are modeled as cohorts associated with degree programs. To showcase the potential of our proposal, we apply the EFaST system to data from the Department of Computer Science at the University of Verona, comprising three bachelor's degree programs in Computer Science, Bioinformatics, and Biotechnology. Participants can interact with the web application to (i) change constraints, (ii) inspect optimization results, and (iii) explore their explanations in real time. Overall, the demonstration is designed to show how fairness trade-offs emerge under different preference configurations and how stakeholders can interpret them. The following subsections illustrate the main interaction points offered by EFaST during the demonstration[3].

Constraint Specification. The demonstration starts with the specification of stakeholder constraints, in which each professor can specify their individual availability on a weekly calendar. For each time slot, the professor can select one of three options (Fig. 2): *Available* (blue), *Better not* (yellow), representing a soft constraint with $n = -0.5$, and *Impossible* (red), representing a stronger soft constraint ($n = -1$). To avoid unrealistic configurations in which only a few time slots remain effectively usable, the system allows the administrator to impose a limit on the number of constraints per professor. For instance, in Fig. 2, *prof8* expresses a preference to avoid classes scheduled between 1:30 and 3:30 p.m. on any day, while explicitly marking it as strongly unavailable on Wednesday.

Fair Solution Generation. Once preferences for each professor are collected by assessing the administrator dashboard, it is possible to inspect the submitted configurations. From the interface, the administrator can review and download detailed information for each professor and, if needed, adjust the professors' preferences thresholds. The optimization process is then triggered by selecting the maximum number of iterations for the FaST-AMOSA algorithm, which is set to 100 by default. Before the first iterations of the algorithm, EFaST estimates and displays the expected end time. As soon as the first solution is generated, the interface visualizes the current fairness status. Fig. 3 shows an example of the monitoring interface displayed during the optimization process. The upper table provides a summary of the current fairness status, including the Global Fairness value $\langle g_a(A), g_s(A)\rangle = \langle -0.66, 0.31\rangle$, as well as the average and standard deviation of local fairness for professors and student cohorts. It also reports the number of professors whose preferences are not fully satisfied. In the shown configuration, all professor preferences are satisfied; consequently, the indicator corresponding to the percentage of satisfied constraints reaches 100%. The second table details student cohort fairness by degree program and

[3] The source code and data have been made available at https://github.com/4nnina/EFaST

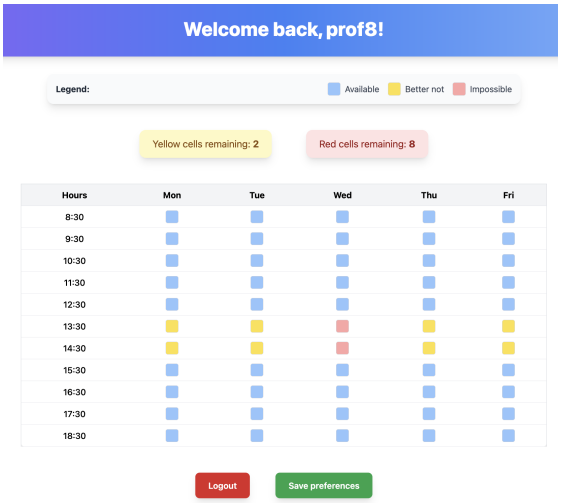

Fig. 2. User preferences webpage illustrating how professors express constraints for weekly classes.

academic year. Each value can be inspected interactively to visualize the corresponding timetable, highlighting in the assignment the professor constraints that are violated in that specific schedule. At the bottom of the page, a plot shows the evolution of fairness scores across iterations, allowing users to observe how global and local fairness improve over time.

This monitoring becomes particularly informative in configurations where stakeholder preferences conflict structurally. For example, if multiple professors mark Wednesday as strongly undesirable while student cohorts require compact weekly schedules, satisfying all professor constraints forces lectures to be redistributed across the remaining days, leading to fragmented timetables for students and a decrease in their local fairness score. In such cases, users can observe how prioritizing professor satisfaction increases the global fairness average $g_a(A)$, while simultaneously increasing its standard deviation $g_s(A)$, revealing a growing imbalance across stakeholders. While the interface makes these trade-offs quantitatively visible, the underlying reasons for persistent dissatisfaction are not immediately evident from numerical indicators alone. To clarify the causes behind these fairness dynamics, the violet pop-up on the right side of the interface provides direct access to the fairness explanation of the current assignment, supporting the interpretation of the reported scores and trade-offs.

Fairness Explanation. In realistic scenarios, a perfectly fair solution satisfying all soft constraints is often unachievable, as schedules may violate some preferences due to structural conflicts among stakeholders. To support fairness, EFaST allows users to request an explanation of the current assignment at any stage of the optimization process. The system produces a structured explanation that summarizes the overall fairness score and highlights the constraint viola-

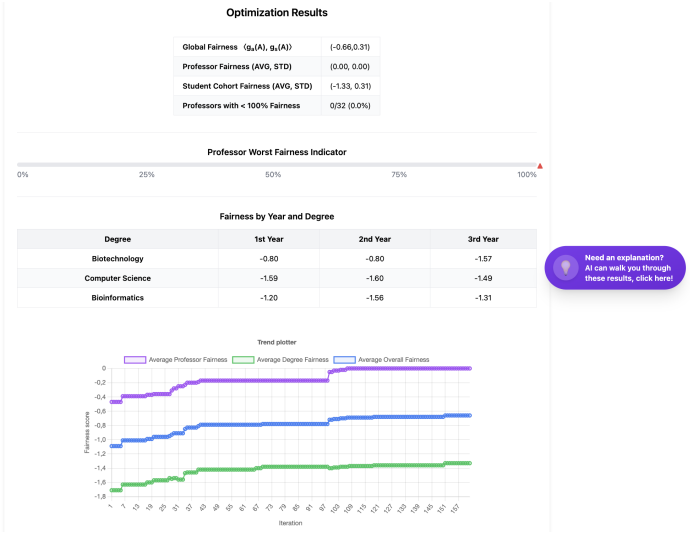

Fig. 3. Monitoring interface showing fairness indicators during the optimization process.

tion. Fig. 4 shows an example of such an explanation. First, the current fairness is analyzed by qualitatively identifying which constraints remain unsatisfied and which stakeholders are most affected. Second, it articulates the structural conflicts underlying these violations, clarifying why certain constraints cannot be jointly satisfied. Finally, the explanation provides illustrative directions for improving fairness, including which preference adjustments could reduce imbalance, supporting the administrator in exploring alternative, potentially fairer configurations. The explanation relies on an LLM that translates structured outputs produced by the optimization process into natural language. This model can be iteratively configured by the user, and it does not perform any independent computations or optimizations, nor does it influence the algorithms' decision-making process. Any illustrative suggestions included in the explanation are interpretative and require an explicit human validation before any constraint edits.

5 Conclusion

This demo presented EFaST, an interactive framework for exploring fairness in sequential task assignment problems. EFaST enables users to monitor local and global fairness in real time, analyze trade-offs among stakeholders, and inspect why specific constraints remain unsatisfied. By combining optimization with natural language explanations grounded in structured outputs, the system transforms fairness-aware optimization from a black-box process into an inspectable and human-controllable decision-support workflow. Overall, EFaST illustrates how fairness-aware optimization can become transparent, interactive, and practically actionable in complex multi-stakeholder environments.

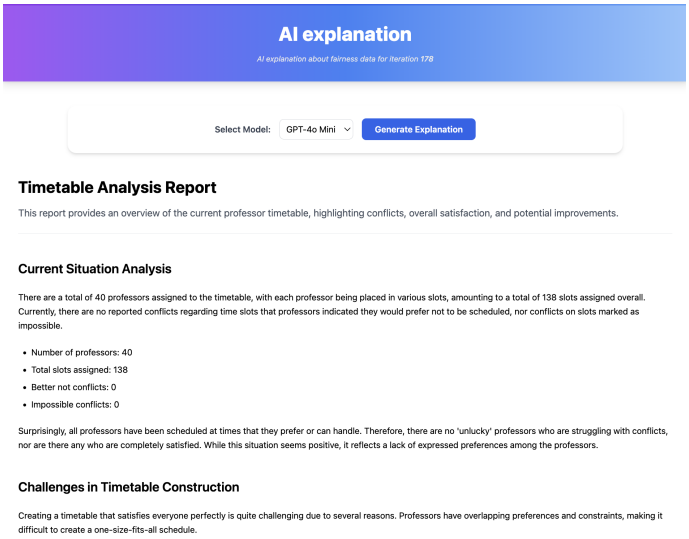

Fig. 4. Example of EFaST fairness explanation.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Borges, R., Sahlgren, O., Koivunen, S., Stefanidis, K., Olsson, T., Laitinen, A.: Multi-objective fairness in team assembly. In: New Trends in Database and Information Systems - ADBIS 2023 Short Papers, Doctoral Consortium and Workshops Proceedings. CCIS, vol. 1850, pp. 106–116. Springer (2023)
2. Dalla Vecchia, A., Migliorini, S., Quintarelli, E., Stefanidis, K.: Multi-sided fairness in sequential task assignment. Information Systems p. 102700 (2026)
3. Migliorini, S., Quintarelli, E., Gambini, M., Belussi, A., Carra, D.: Sequence recommendations for groups: A dynamic approach to balance preferences. Information Systems **108**, 102023 (2022)
4. Pitoura, E., Stefanidis, K., Koutrika, G.: Fairness in rankings and recommendations: an overview. VLDB J. **31**(3), 431–458 (2022)
5. Shafiloo, R., Stratigi, M., Peltonen, J., Olsson, T., Stefanidis, K.: The many facets of fairness in recommender systems: Consumers, providers and items. Inf. Syst. **137**, 102643 (2026)
6. Suman, B.: Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem. Computers & Chemical Engineering **28**(9), 1849–1871 (2004)
7. Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J.Y., Wen, J.R.: A survey of large language models (2025), https://arxiv.org/abs/2303.18223