# LISTEN to Your Preferences: An LLM Framework for Multi-Objective Selection

**Adam Jovine***
ORIE, Cornell University
asj53@cornell.edu

**Tinghan Ye***
ISyE, Georgia Tech
joe.ye@gatech.edu

**David Shmoys**
ORIE, Cornell University
dbs10@cornell.edu

**Peter Frazier**
ORIE, Cornell University
pf98@cornell.edu

## Abstract

Multi-objective optimization often produces large sets of Pareto-optimal solutions, creating a bottleneck for human experts who must select the best option. This difficulty is compounded by the fact that expert preferences are often complex and hard to formalize. To address this, we introduce LISTEN, a framework that leverages a large language model (LLM) as a zero-shot preference oracle, guided only by an expert's high-level priorities in natural language. To operate within LLM constraints like context windows and inference costs, we propose two iterative algorithms: **LISTEN-U**, which uses the LLM to refine a parametric utility function, and **LISTEN-T**, a non-parametric method that performs tournament-style selections over small batches of solutions. We evaluate our framework on a real-world final exam scheduling problem. Preliminary results suggest this approach is promising, consistently identifying high-quality solutions and showing encouraging fidelity to the stated preferences. This work demonstrates a path toward steering complex multi-objective selection problem directly with natural language, bypassing the need for mathematical utility functions to express preferences.

## 1 Introduction

Multi-objective optimization (MOO) can generate hundreds of Pareto-optimal solutions (Knowles, 2006; Branke, 2008). When there are more than two metrics, it is difficult to visualize the tradeoffs implicit in these solutions. This creates a bottleneck for decision-makers. *How can an expert decision-maker efficiently sift through a vast set of viable candidates to find the one that best reflects their nuanced, often unstated, priorities?*

Traditional solutions are often inadequate. Manually comparing all options is time-consuming and prone to error. Algorithmic search methods that require the decision-maker to express pairwise preferences between solutions (Obayashi et al., 2007; Wang et al., 2022; Huber et al., 2025) or human-directed faceted search (Ozaki et al., 2024) can help, but still require significant human effort. The core difficulty is that *humans lack a time-efficient way to accurately articulate their preferences*.

Large language models (LLMs) offer a new paradigm for tackling this challenge. With their ability to interpret text, LLMs present an opportunity for zero-shot preference modeling, where a decision-maker's goals can be understood directly from a verbal description. This bypasses the need for rigid,

*Equal Contribution

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: ML×OR Workshop: Mathematical Foundations and Operational Integration of Machine Learning for Uncertainty-Aware Decision-Making.

numerical utility functions. While recent research has begun integrating LLMs into preference learning, they are typically used as components within larger systems—for instance, to guide questioning (Lawless et al., 2023; Austin et al., 2024), extract preferences from reviews (Bang and Song, 2025), or simulate user behavior (Okeukwu-Ogbonnaya et al., 2025; Zhang et al., 2025). However, it remains an open question whether an LLM can, on its own, effectively navigate the complex trade-offs inherent in a Pareto frontier using only high-level, natural language instructions.

To address this gap, we introduce **LISTEN**: **L**LM-based **I**terative **S**election with **T**rade-off **E**valuation from **N**atural-language, a framework that uses an LLM as a preference oracle for selection of a human expert's most preferred item from a list that is too long for the human to examine directly. In our framework, a human expert first describes their potentially complex priorities in natural language. An iterative search algorithm then uses this utterance in repeated LLM calls to compare subsets of items, summarizing the LLM's responses via a classical utility surrogate. Ultimately, the algorithm selects its estimate of the best item. Our approach must contend with limits on the LLM's context window and the LLM's ability to reason directly over large item lists, which prevent the LLM from directly selecting the best item in a single call. Our approach must also limit the number of calls to the LLM to save computational and financial cost.

Our approach combines principles from preference-based optimization (Chu and Ghahramani, 2005; Brochu et al., 2010) and active ranking (Jamieson and Nowak, 2011; Yue et al., 2012), situating these classical methods within the emerging field of LLMs for decision-making (Hao et al., 2023; Xi et al., 2025). While prior work uses LLMs to assist in planning or preference elicitation (Valmeekam et al., 2022; Zhang et al., 2023; Lawless et al., 2023, 2025), we specifically investigate if an LLM can serve as the primary, zero-shot oracle for selecting from a pre-computed Pareto frontier, guided only by high-level natural language goals.

This paper's primary contribution is the introduction and exploration of a framework, LISTEN, that uses an LLM as a preference oracle for multi-objective decision-making. Our preliminary evaluation on a real-world final exam scheduling dataset suggests this approach is promising, identifying higher-quality solutions than baselines and showing encouraging fidelity to the stated preferences. We present these initial findings as part of our ongoing work to understand the potential and limitations of using LLMs for multi-objective decision-making. While we focus here on selection from a finite pre-computed list, we view this work as a starting point for prompt-guided MOO in mixed integer programming, continuous optimization, and other optimization domains.

## 2 Problem Description

We are given a collection of items $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$. Each item $s_i$ is a $d$-dimensional vector giving the values for each of $d$ metrics. We are focused on settings where $N > 1000$ is too large for the user to directly examine all items. Such collections are often produced in MOO when a MOO algorithm identifies a large collection of items on the Pareto frontier. We are also given a natural language utterance describing a human decision-maker's preferences, the name of each attribute, and a pre-trained LLM. An example utterance and items are given in the appendix. In our experiments, items are final exam schedules considered by a university registrar. Our goal is to use a limited number of calls to the LLM to select the item that the human decision maker most prefers.

## 3 Methodology

To solve the selection problem defined in Section 2, we introduce **LISTEN** (**L**LM-based **I**terative **S**election with **T**rade-off **E**valuation from **N**atural-language). LISTEN is an iterative framework that employs an LLM as a zero-shot preference oracle to identify a decision-maker's most preferred item from a large candidate set. The framework begins with a natural language utterance describing the user's priorities and then progressively refines the set of candidates to find the single best option, as illustrated in Figure 3 (see Appendix). This iterative design is crucial for navigating large solution spaces while respecting the LLM's inherent context window and query budget limitations.

Our first algorithm, LISTEN-Utility (**LISTEN-U**), is a parametric approach. It assumes a linear utility function, $u(s) = \mathbf{w}^\top s^{\text{num}}$, defined exclusively over the **numerical** attributes of an item (denoted $s^{\text{num}}$). The non-numerical attributes provide context for the LLM's reasoning but are not part of this

scoring function. The algorithm iteratively refines the weight vector $\mathbf{w}$ to identify the best item, as detailed in Algorithm 1 (see Appendix).

Our second algorithm, LISTEN-Tournament (**LISTEN-T**), is a non-parametric method that emulates a tournament to efficiently search the solution space. It is designed for a budget of $T \geq 3$ calls to the LLM, as detailed in Algorithm 2 (see Appendix).

## 4 Experiments

To evaluate the LISTEN framework, we apply it to the real-world problem of final exam scheduling.

**The Final Exam Scheduling Problem** We conduct our experiments using a MOO benchmark introduced in Ye et al. (2024). This benchmark focuses on final exam scheduling using mixed integer programming (MIP). There are 11 metrics describing exam conflicts and when students complete their exam. They are listed in Appendix C. For each metric, smaller is better. Using ParEGO (Knowles, 2006) and optimizing over MIP hyperparameters, we generate a diverse set $\mathcal{S}$ of 5,000 unique candidate solutions on the Pareto frontier over these metrics.

**Experimental Setup** Across all experiments, we use `Llama-3.3-70B-Versatile` (Meta AI, 2024) as our llm preference oracle. To ensure the robustness of our results, all algorithmic runs are replicated 50 times with different random seeds for generation while keeping the prompt consistent for each replication.

The ground truth for evaluation is derived from human expert rankings. For each dataset, as it is impractical to rank the entire set of $N$ items, an expert manually ranked a subset of the most relevant items ($m \ll N$). This ranked list serves as the ground truth for measuring how well an algorithm's selected item aligns with nuanced human preferences.

We compare the performance of our LISTEN algorithms against the following two baseline methods:

1. Uniform Random Selection (`baseline/random`). This non-informative baseline selects an item uniformly at random from the full candidate set $\mathcal{S}$. It is used to establish a lower bound on performance and quantify the improvement gained by any guided selection strategy.
2. Normalized Average Score (`baseline/z-score-avg`). This method considers only the numerical attributes of the items. For each numerical attribute, it first standardizes the values across the entire set $\mathcal{S}$ to have zero mean and unit variance (i.e., calculates the z-score). The scores for attributes that are to be minimized (e.g., price) are then negated. The algorithm selects the item with the highest average standardized score across all numerical attributes. This baseline represents a simple, non-learned linear utility function that weights all normalized metrics equally and ignores all categorical or textual information.

We evaluate algorithm performance using the rank of the selected item within the human-generated ground-truth ranking. For a given dataset with $N$ total items, let the ground-truth list contain $m$ items ranked by a human expert $(1, 2, \ldots, m)$. For an item $s^\star$ selected by an algorithm, its rank is determined as follows: If $s^\star$ is within this top-$m$ list, its rank is its position in that list. If $s^\star$ is not in the list, it is considered unranked. All $N - m$ unranked items are assigned a shared average rank of $(m+1+N)/2$, representing the mean of the available ranks from $m+1$ to $N$. To ensure a consistent scale across datasets of different sizes, we normalize this rank by dividing by the total number of items, $N$. The final metric is a value in $(0, 1]$, where a lower score indicates better performance.

**Results** Figure 1 highlights the effectiveness of both proposed algorithms, outperforming both baselines. LISTEN-U starts with performance comparable to the random baseline but progressively refines its utility function to find dramatically better solutions, ultimately achieving the best performance of any method.

To isolate the value of the user's natural language preference utterance, we conducted an ablation study (Figure 2) comparing performance with a full, preference-guided prompt against a *base prompt* containing only the persona and metric definitions. It's interesting to note that including the preference utterance never degraded performance. This suggests that providing more specific, user-aligned context to the LLM is beneficial or, at worst, neutral.

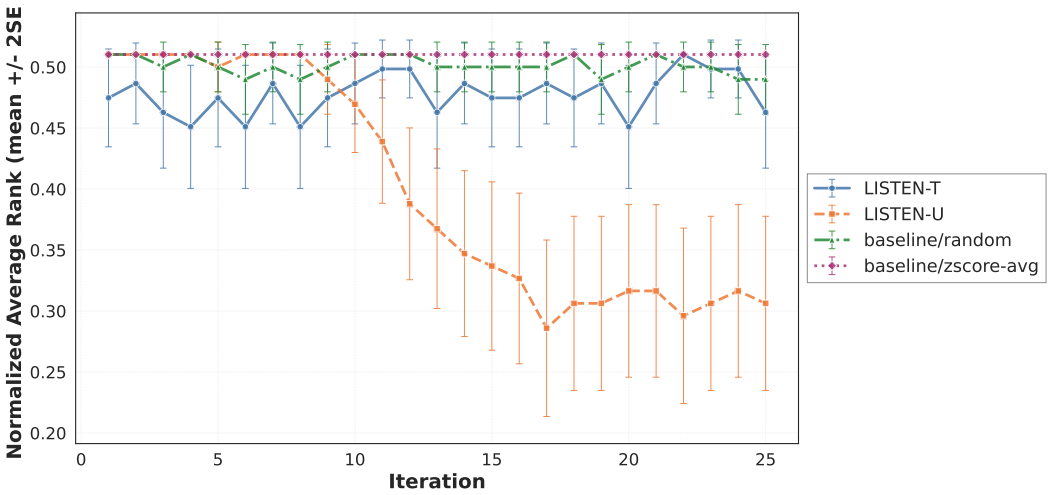


Figure 1: Performance of LISTEN algorithms and baselines, showing the Normalized Average Rank (lower is better) over 25 iterations.

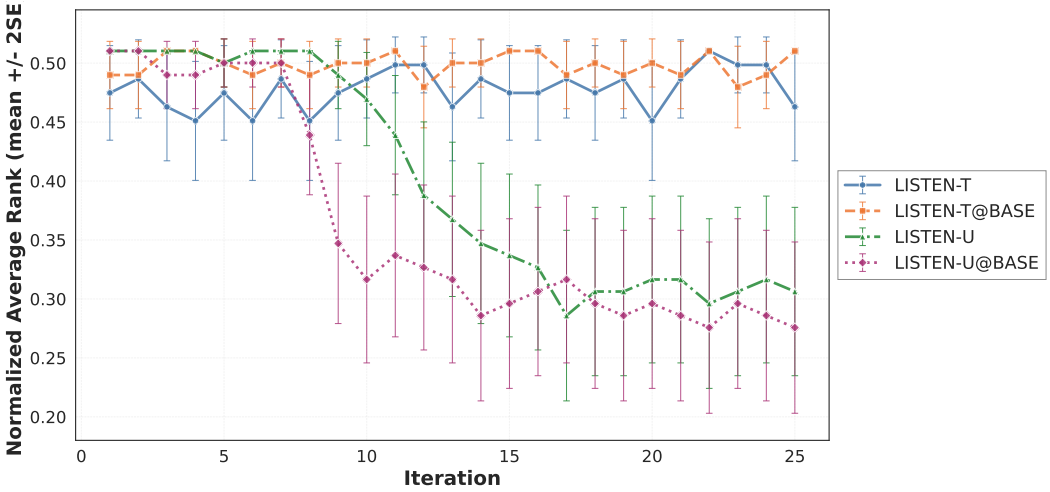


Figure 2: Ablation study evaluating the impact of preference utterance. Performance using the preference-guided prompt is compared to a base prompt containing only the persona and metric definitions.

## 5 Discussion & Future Work

While promising, the LISTEN framework is not without its limitations. A core challenge is evaluation: future research in this area would be accelerated by a high-quality benchmark dataset spanning multiple domains that can better assess the performance of strategies in the LISTEN framework.

We are excited about opportunities to expand the LISTEN framework. One future direction is leveraging multi-turn conversations with users to better identify desirable solutions. Conversation is an important technique when one human (e.g., a salesperson) selects from a list to satisfy the needs of another human (e.g., a customer). Another direction is going beyond selection from lists to methods that can optimize over large search spaces. We also expect that developing a better understanding of the core statistical properties of different types of LLM queries will support improvement of future algorithms in the LISTEN framework. Finally, we view this work on prompt-guided selection from a finite list as a jumping off point for optimization over large spaces using ideas from mixed integer programming, continuous optimization, and other optimization frameworks.

## References


Austin, D. E., Korikov, A., Toroghi, A., and Sanner, S. (2024). Bayesian optimization with llm-based acquisition functions for natural language preference elicitation. *arXiv preprint arXiv:2405.00981*.

Bang, S. and Song, H. (2025). Llm-based user profile management for recommender system. *arXiv preprint arXiv:2502.14541*.

Branke, J. (2008). *Multiobjective optimization: Interactive and evolutionary approaches*, volume 5252. Springer Science & Business Media.

Brochu, E., Cora, V. M., and De Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*.

Chu, W. and Ghahramani, Z. (2005). Preference learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pages 137–144.

Hao, S., Gu, Y., Ma, H., Hong, J. J., Wang, Z., Wang, D. Z., and Hu, Z. (2023). Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Huber, F., Gonzalez, S. R., and Astudillo, R. (2025). Bayesian preference elicitation for decision support in multiobjective optimization. *arXiv preprint arXiv:2507.16999*.

Jamieson, K. G. and Nowak, R. (2011). Active ranking using pairwise comparisons. *Advances in neural information processing systems*, 24.

Knowles, J. D. (2006). Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66.

Lawless, C., Li, Y., Wikum, A., Udell, M., and Vitercik, E. (2025). Llms for cold-start cutting plane separator configuration. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 51–69. Springer.

Lawless, C., Schoeffer, J., Le, L., Rowan, K., Sen, S., Hill, C. S., Suh, J., and Sarrafzadeh, B. (2023). "I Want It That Way": Enabling Interactive Decision Support Using Large Language Models and Constraint Programming. *arXiv preprint arXiv:2312.06908*. Submitted December 12, 2023; revised October 1, 2024.

Meta AI (2024). The llama 3 herd of models.

Obayashi, S., Jeong, S., Chiba, K., and Morino, H. (2007). Multi-objective design exploration and its application to regional-jet wing design. *Transactions of the Japan Society for Aeronautical and Space Sciences*, 50(167):1–8.

Okeukwu-Ogbonnaya, A., Amatapu, R., Bergtold, J., and Amariucai, G. (2025). Llm-based community surveys for operational decision making in interconnected utility infrastructures. *arXiv preprint arXiv:2507.13577*.

Ozaki, R., Ishikawa, K., Kanzaki, Y., Suzuki, S., Takeno, S., Takeuchi, I., and Karasuyama, M. (2024). Multi-objective bayesian optimization with active preference learning. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence*, pages 14490–14498, Vancouver, Canada. AAAI Press.

Valmeekam, K., Olmo, A., Sreedharan, S., and Kambhampati, S. (2022). Large language models still can't plan (a benchmark for llms on planning and reasoning about change). In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Wang, X., Jin, Y., Schmitt, S., and Olhofer, M. (2022). Recent advances in bayesian optimization. *arXiv preprint arXiv:2206.03301*.

Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., Zhang, M., Wang, J., Jin, S., Zhou, E., et al. (2025). The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Ye, T., Jovine, A., van Osselaer, W., Zhu, Q., and Shmoys, D. B. (2024). Cornell university uses integer programming to optimize final exam scheduling. *arXiv preprint arXiv:2409.04959*.

Yue, Y., Broder, J., Kleinberg, R., and Joachims, T. (2012). The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556.

Zhang, H., Zhu, Q., and Dou, Z. (2025). Enhancing reranking for recommendation with llms through user preference retrieval. *Proceedings of the 31st International Conference on Computational Linguistics*, pages 658–671.

Zhang, M. R., Desai, N., Bae, J., Lorraine, J., and Ba, J. (2023). Using large language models for hyperparameter optimization. *arXiv preprint arXiv:2312.04528*.

## A Schematic Overview

Figure 3 shows the schematic of the LISTEN framework. A human decision maker provides preferences in natural language. An iterative algorithm, either LISTEN-U or LISTEN-T, then uses an LLM as a preference oracle to progressively filter a large set of candidate items (e.g., the Pareto frontier) and identify the single most preferred solution.

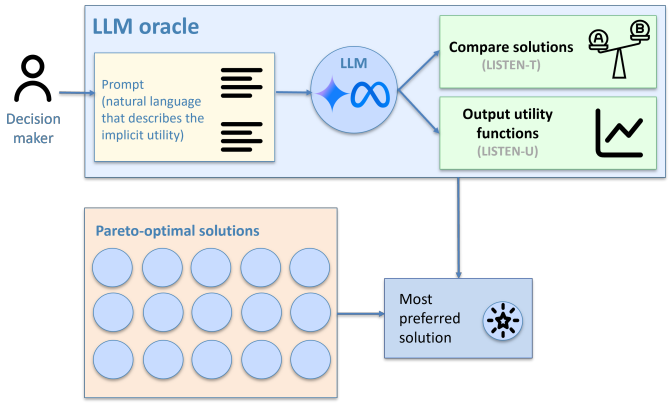


Figure 3: A schematic overview of the LISTEN framework.

## B Algorithm Pseudocode

This section provides the pseudocode of our proposed algorithms.

**Algorithm 1** LISTEN-U: Iterative Utility Refinement

**Require:** Solution set $\mathcal{S}$, max iterations $T$, LLM oracle $\mathcal{L}$, preference utterance $U$

1: **Initialization:**
2: Construct initial prompt $p_1$ with $U$ and definitions of all attributes
3: Elicit initial weight vector $\mathbf{w}_1$ for numerical attributes from $\mathcal{L}(p_1)$
4: For each $s_i \in \mathcal{S}$, let $s_i^{\text{num}}$ be the vector of its numerical attributes
5: Let $\tilde{s}_i^{\text{num}}$ be the version of $s_i^{\text{num}}$ with attributes normalized to $[0, 1]$
6: $s^\star \leftarrow \arg\max_{s_i \in \mathcal{S}} \mathbf{w}_1^\top \tilde{s}_i^{\text{num}}$
7: **Iterative Refinement:**
8: **for** $t \leftarrow 2$ to $T$ **do**
9: Construct refinement prompt $p_t$ using $U$, $\mathbf{w}_{t-1}$, and all attributes (numerical and non-numerical) of the *unnormalized* solution $s^\star$
10: Elicit refined weight vector $\mathbf{w}_t$ for the numerical attributes from $\mathcal{L}(p_t)$
11: $s^\star \leftarrow \arg\max_{s_i \in \mathcal{S}} \mathbf{w}_t^\top \tilde{s}_i^{\text{num}}$
12: **end for**
13: **return** final weight vector $\mathbf{w}_T$ and solution $s^\star$

Algorithm 1 operates in two phases. In the **initialization phase** (lines 2-6), the LLM is prompted to define an initial weight vector, $\mathbf{w}_1$, for the numerical attributes based on the user's utterance. To score the items, the numerical attributes of every solution in $\mathcal{S}$ are first normalized to a common $[0, 1]$ scale. The algorithm then computes the utility of each item by taking the dot product of the weights and the normalized numerical attribute vector, selecting the highest-scoring item as the initial best solution, $s^\star$.

The **iterative refinement phase** (lines 8-12) then begins. In each iteration, the prompt for the LLM is constructed using the current weight vector $\mathbf{w}_{t-1}$ alongside the complete, **unnormalized** description of the current best solution, $s^\star$, including both its numerical and non-numerical attributes. Presenting the true values and full context is crucial for the LLM to reason about real-world trade-offs (e.g., "the price is too high *for this particular brand*"). The LLM is asked to critique this solution and propose a refined weight vector, $\mathbf{w}_t$. After the LLM returns the updated weights, the algorithm once again

scores the entire solution set using the consistently **normalized** numerical attributes to select the new best solution.

**Algorithm 2** LISTEN-T: Tournament-Based Selection

**Require:** Solution set $\mathcal{S}$, batch size $B$, max iterations $T \geq 3$, LLM oracle $\mathcal{L}$, utterance $U$

1: **Preliminary Rounds:**
2: $m \leftarrow T - 1$ ▷ Number of preliminary rounds
3: $\mathcal{K} \leftarrow \emptyset$ ▷ Initialize set of batch champions
4: **for** $j \leftarrow 1$ to $m$ **do**
5: $\quad \mathcal{C}_j \leftarrow \text{Sample}(\mathcal{S}, B)$ ▷ Sample a batch of size B
6: $\quad c_j \leftarrow \text{LLM-Choose}(\mathcal{C}_j, U)$ ▷ LLM selects batch champion
7: $\quad \mathcal{K} \leftarrow \mathcal{K} \cup \{c_j\}$
8: **end for**
9: **Final Playoff:**
10: $s^\star \leftarrow \text{LLM-Choose}(\mathcal{K}, U)$ ▷ Select winner from champions
11: **return** solution $s^\star$

Algorithm 2 proceeds in two stages using a total of $T$ calls to the LLM. First, in the **preliminary rounds** (lines 2-6), the algorithm conducts $m = T - 1$ rounds. In each round, it samples a batch of $B$ items uniformly at random from the full solution set $\mathcal{S}$ and prompts the LLM to select the single most preferred item from that batch. This winning item, or "batch champion," is added to a set of champions, $\mathcal{K}$.

Second, in the **final playoff** (line 8), the LLM is presented with the set of all $m$ champions collected from the preliminary rounds and is asked to select the single overall winner. By requiring $T \geq 3$, this structure ensures the final playoff is a meaningful comparison between at least two distinct batch champions, allowing the LLM to make a final, decisive trade-off.

## C Metrics Used in Final Exam Scheduling

Table 1 documents the objectives used in the final exam scheduling problem.

Table 1: Final Exam Scheduling Conflict Metrics

| Metric | Description |
|---|---|
| Conflicts | Instances of a student having two or more exams in the same time slot. |
| Quints | Instances of a student having five exams in consecutive time slots. |
| Quads | Instances of a student having four exams in consecutive time slots. |
| Four in Five Slots | Instances of a student having four exams within five consecutive time slots. |
| Triple in 24h (no gaps) | Instances of a student having three back-to-back exams in a 24-hour period. |
| Triple in Same Day (no gaps) | Instances of a student having three back-to-back exams on the same day. |
| Triples | Triple in 24h (no gaps) + Triple in Same Day (no gaps). |
| Three in Four Slots | Instances of a student having three exams within four consecutive time slots. |
| Evening/Morning B2Bs | Instances of a student having an exam in the last slot of one day and the first slot of the next day. |
| Other B2Bs | All other instances of a student having exams in adjacent time slots. |
| Back-to-backs | Evening/Morning B2Bs + Other B2Bs. |
| Two in Three Slots | Instances of a student having two exams within three consecutive time slots. |
| Average Time of Last Exam | Average time slot in which students take their last exams. |

## D Prompts

At the core of our framework is the use of an LLM to simulate the decision-maker's preferences in a zero-shot manner. We do not fine-tune the LLM; instead, all queries follow a structured prompt that encapsulates the decision context. Each prompt sent to the LLM consists of five key components:

1. **Persona Context**: Assigns a role to the LLM (e.g., "You are a University registrar").
2. **Metric Definitions**: Defines the attributes of the items (e.g., "Conflicts: students with two exams at the same time").
3. **User Priorities**: The natural language utterance $U$ describing the user's goals (e.g., "prioritize minimizing back-to-back exams").
4. **Solutions**: The candidate item(s) to be evaluated in the current step.
5. **Format Instructions**: Specifies the desired output format (e.g., a JSON object).

This consistent structure provides the LLM with all the necessary context to act as the preference oracle for a given algorithmic step, whether it is refining a utility function or selecting a champion from a batch.

We give the first three parts of the prompts for exam scheduling below. The fourth part (solutions) pulls the solution from the previous iteration, if any. The fifth part of the instructions, which covers output format, has different structures for LISTEN-U and LISTEN-T. For LISTEN-U, the task is to output the weights for numerical features in JSON format. For LISTEN-T, the task is to identify the single best solution from the options provided and return it using the exact format: `FINAL A (B, C ...)`.

As shown in Figure 4, the first line, "You are an expert university registrar..." provides the persona context. The next section details the metrics. Following the metric definitions in a bulleted list format, the user priorities are highlighted. In this case, the priority is to "minimize student stress and administrative burden by optimizing a specific set of metrics...".

Exam Scheduling Prompt

```
You are an expert university registrar choosing the better final-exam schedule.
Use these definitions (lower is better unless stated otherwise):
    - conflicts: students with overlapping exams (must be minimized)
    - quints: 5 exams in a row
    - quads: 4 in a row
    - four in five slots: 4 exams within 5 slots
    - triple in 24h (no gaps): 3 exams within 24 hours (strictly consecutive
      blocks)
    - triple in same day (no gaps): 3 exams within the same calendar day
      consecutively
    - three in four slots: 3 exams within 4 slots
    - evening/morning b2b: back-to-back from an evening into next-morning slot
    - other b2b: any other back-to-back pair
    - two in three slots: 2 exams within 3 slots
    - avg max: is the average slot of the last exam for each student. Students
      tend to want to leave as early as possible.

Policy guidance: To generate an optimal final examination schedule that
minimizes student stress and administrative burden by optimizing a specific
set of metrics according to a tiered priority system.
      *Context:* There are three exam slots available per day.
      Solutions must have a conflicts value of 0 or 1.

      *Tier 1:* Minimize Mandatory Reschedules
      The highest priority is to minimize the metrics associated with intense
      exam clusters that require a mandatory reschedule. Please minimize the
      following metrics in the order listed:
          quints (five exams in 24 hours)
          quads (four exams in 24 hours)
          four in five slots
          triple in 24h (no gaps)
          triple in same day (no gaps)

      *Tier 2:* Reduce Student Exam Fatigue
      Once Tier 1 objectives have been met, the primary tie-breaker is to
      minimize the total number of back-to-back exams. This objective is
      achieved by minimizing the sum of the following two metrics:
          evening/morning b2b
          other b2b

      *Tier 3:* Enhance Overall Schedule Quality
      As a final set of tie-breakers, prioritize schedules that improve the
      general student experience by optimizing these metrics in order of
      preference:
          Lower the avg_max (the average slot of a student's final exam).
          Minimize two in three slots.
          Minimize three in four slots.
```

Figure 4: An example prompt for the exam scheduling dataset.