# Human Preference-Aligned Construction Scheduling using Reinforcement Learning and Direct Preference Optimization

**Ajay Kumar Agrawal[1], Yang Zou[1,2], and Tao Yang[1]**

[1]Department of Civil and Environmental Engineering, University of Auckland, New Zealand
[2]School of Architecture, Building and Civil Engineering, Loughborough University, UK
aagr657@aucklanduni.ac.nz, y.zou2@lboro.ac.uk, tyan632@aucklanduni.ac.nz

**Abstract**

**Manual construction scheduling is prone to errors and suboptimality. To this end, there is growing attention to reinforcement learning (RL)-based approaches for automated construction scheduling due to their superior time efficiency and solution quality compared to metaheuristic algorithms. However, developing suitable reward functions for training RL agents to solve real-world problems is a significant challenge in practice. Moreover, current RL-based construction scheduling research has primarily focused on an algorithmic standpoint, lacking human aspects, such as incorporating human preferences into scheduling. These limitations hinder the real-world adoption of RL-based construction scheduling. This study presents a novel framework for automated reward function generation using human-AI collaboration. In the framework, the human provides a prompt consisting of project context and reward requirements to a large language model (LLM), which generates reward options. The reward options are used to train Proximal Policy Optimization-based scheduling agents. The trained agents predict schedules and compute objective values, which are presented to the human decision-maker. The human selects the preferred schedule, creating a preference dataset over the iterations. The dataset is used to fine-tune the LLM via Direct Preference Optimization, enabling it to generate human-preferred reward structures. Evaluation on a 20-activity construction project demonstrated the effectiveness of the proposed framework in generating schedules aligned with human preferences, as evidenced by higher reward margins and win rates than the base LLM.**

**Keywords –**

**Human-AI Collaboration, Reinforcement Learning, Construction Scheduling, Large Language Models, Direct Preference Optimization**

## 1 Introduction

Construction scheduling is often formulated as a Resource Constrained Project Scheduling Problem (RCPSP), which is Non-deterministic Polynomial time (NP)-hard in nature [1]. It involves scheduling a set of activities with fixed duration and resources under constraints, such as resource and precedence constraints. Manually solving such problems is time-intensive, suboptimal, and error-prone, which often leads to inadequate time and cost performance of the construction projects [1]. Extensive research has been conducted to address this problem through Artificial Intelligence (AI) and Optimization algorithms [2]. Traditionally, the heuristic and metaheuristic algorithms were widely utilized to this end [3]. However, owing to the high computational complexity and low adaptability of such methods, researchers have started exploring AI paradigms, such as reinforcement learning (RL) [4,5].

The industrial application of AI-based construction scheduling remains limited [2]. Construction projects are socio-technical in nature, requiring a suitable interplay between humans, processes, and technologies such as AI, to ensure that the technology-based systems are adequately adopted [6]. The existing AI-based construction scheduling research has primarily focused on an algorithmic standpoint, lacking the human aspects, such as human preferences, understanding, and interactions with AI [7,8]. This may result in issues such as a lack of trust and transparency, and thus, a lack of willingness to adopt AI-based systems [9,10].

At the same time, RL research has shown that its use for real-world problem-solving is challenging, limiting its practical applicability. Gabriel Dulac-Arnold et al. [11] identified nine such challenges, for instance, limited sample availability, high-dimensional action spaces, constraints, partially observable states, and inadequate reward functions. Ding and Dong [12] highlighted additional difficulties, including the Sim2Real gap,

generalizability, and the design of reward functions.

As several studies highlight [11,12], generating an effective reward function is challenging in real-world RL applications. A reward is a crucial component of the RL training cycle, enabling the agent to differentiate between good and bad actions based on the intended objectives and preferences in the problem [13]. A sparse reward can become ineffective in providing a good training signal, a mis-specified reward can result in sub-optimal agent behavior, and inadequate reward scaling can lead to training instability [12]. Moreover, the reward function might differ across problem types, scales, and objectives. This makes it difficult for construction schedulers, who often are not RL experts, to design effective reward functions [14]. As a result, construction schedulers can find it challenging to get RL agents to create schedules that align with their objectives and preferences.

Recent research on RL with Human Feedback (RLHF) [15] and Direct Preference Optimization (DPO) [16], have shown the potential to enable RL agents to align with human preferences. They avoid the need for humans to define an explicit reward function, as the reward is learned from human feedback [15]. They are widely used to prevent large language models (LLMs) from generating harmful content [15]. However, their potential to enable humans and RL agents to develop construction schedules collaboratively has not yet been realized.

Moving forward in this direction, this study proposes a novel human-AI collaboration framework for reward function generation in RL-based construction scheduling using LLMs and DPO [16]. The framework utilizes LLMs to generate reward structures based on the project context, which are converted into Python reward functions and are used to train the Proximal Policy Optimization (PPO) algorithm [17]-based scheduling agents. The agents predict schedules, which are shown to the user, who selects the preferred one. Such preference datasets are then used to fine-tune the LLM using DPO. This cycle is repeated multiple times, enabling the LLM to learn to develop reward functions and, consequently, construction schedules based on human preferences.

## 2 Literature review

### 2.1 Reinforcement Learning-based construction scheduling

Kedir et al. [18] proposed a method to integrate RL with Agent-based modeling to automate construction activity sequencing for minimizing construction cost. Soman et al. [19] used RL in conjunction with linked data to generate constraint-free lookahead schedules. To improve the training efficiency, Yao et al. [5] sampled valid actions while training a deep Q-learning (DQN)-based scheduling agent [5]. Seilabi et al. [20] presented a bi-level optimization method for urban road scheduling, minimizing construction cost, vehicle emissions, and travel times. Agrawal et al. [4,21] proposed a PPO-based method for multi-objective optimization of precast assembly schedules under weather uncertainty.

Previous studies mainly focused on minimizing construction time [4,5,19], cost [20], and constraint violations [4,19]. However, the reward structures were considerably different. For instance, to minimize construction time, Kedir et al. [18] provided a negative time value as a reward, while Yao et al. [5] provided a positive reward for starting and finishing activities, a negative reward for each timestep, a large positive reward for completing all the components, and a positive reward for executing activities in parallel. Similarly, for cost minimization, Soman et al. [19] provided a negative reward for the total duration for which a resource was assigned, while Agrawal et al. [4] assigned a negative reward for the cost incurred at each timestep, as well as a negative reward for the total cost at the end of the episode.

Different studies used different reward functions, even for similar objectives. Moreover, the scales of different reward functions varied, underscoring the need for expertise in designing appropriate reward functions to achieve the desired preferences across different problems. However, automatically aligning rewards with the problem context and human preferences has received little attention, hindering RL-based construction scheduling agents from adapting to problem type and scale and to human scheduling preferences.

## 3 Proposed Framework

The proposed framework is demonstrated in Figure 1. It consists of four components, i.e., LLM-based reward function generation, PPO-based scheduling agent training, human preference collection, and DPO-based LLM fine-tuning. The first component generates options for the reward function based on the prompt provided by the human. The second component trains scheduling agents using the reward options and predicts the schedules using the trained agents. The third component collects the human preferences over the generated schedule options. The fourth component fine-tunes the reward-generation LLM using the collected human preference data. Overall, the AI is responsible for generating reward structures and schedules, while humans provide input information and preferences to guide the AI in generating preferred schedules. The details of the individual components are provided below.

### 3.1 Reward function generation using Large Language Models

LLMs are trained on large amounts of text data,

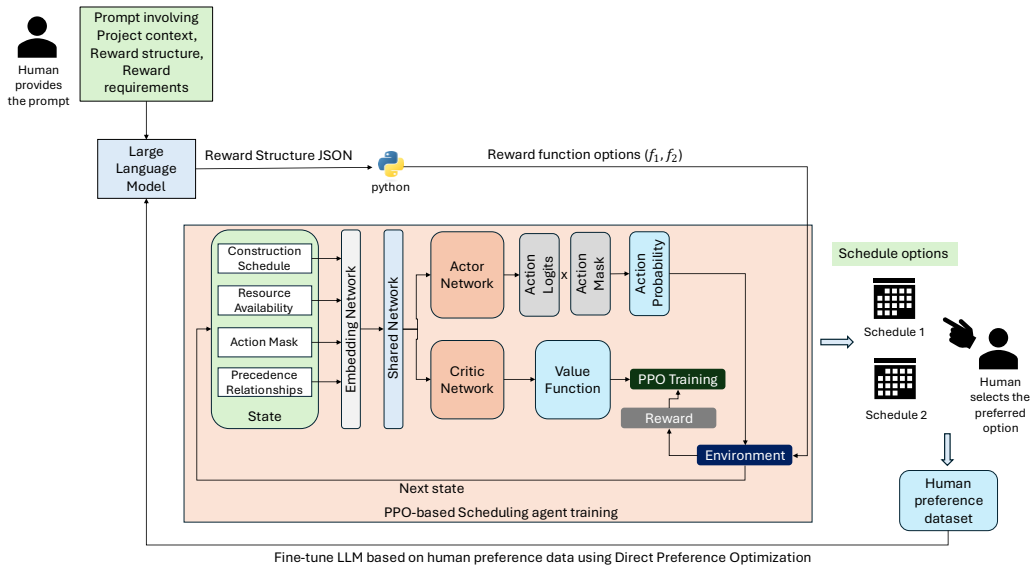


Figure 1: Proposed framework for reward generation using human-AI collaboration

primarily using transformer architectures [24], enabling them to perform several natural language tasks like humans. Using these capabilities, the proposed framework generates variations of the reward function based on the human-provided prompt. The human provides the LLM with a prompt containing the information needed to create the reward functions. However, instead of generating the reward function code, we use an LLM to create reward structures in a predefined JSON format, as LLMs can introduce syntax errors during code generation. JSON provides a structured representation that can be easily processed in programming languages to instantiate a reward function. The prompt consists of the following information:

a) Project context: Including the contextual information about the project, such as project type, requirements, and optimization objectives.
b) Reward components: During the training process, the RL environment sends the information to the reward function, which computes the reward and sends it to the agent. The LLM can generate a variety of reward components. However, if the environment doesn't provide the reward function with information about those components, it will lead to runtime errors. Therefore, we provide the LLM with pre-specified reward components. The LLM must create the reward structures based solely on those components. The reward components can be comprehensively identified from the literature, and the LLM can explore different combinations of them.
c) Output format: To ensure that the LLM generates the reward structure that can be accurately converted into a Python reward function, the format of the reward structure is provided.
d) Output constraints: The constraints are placed on the type of mathematical transformations that can be performed on the reward components (e.g., linear, quadratic, and logarithmic). Additional information is provided to guide the generation of the reward structure, such as the use of positive and negative values to support and penalize a reward component, respectively. Instructions are added to ensure that only JSON is returned, avoiding any unnecessary text.

## 3.2 PPO-based scheduling agent training

For each generated reward function, a separate PPO-based scheduling agent is trained. PPO was chosen for its superior training efficiency and solution quality compared to value-based RL algorithms, such as DQN [4]. While the state information and scheduling agent training architecture can be adjusted according to the problem, this study focused on the proof-of-concept of the overall framework. Therefore, the agent training procedure was mostly consistent with the authors' previous study [4]. The state information included the construction schedule generated up to that timestep, represented as a graph. The graph nodes represent activities, with the features including activity progress status, resource requirements, and duration. The resource availability and action masks at that timestep, as well as the activity precedence relationship, are also included in the state information. To create the embeddings, graph neural networks (GNNs) are used for the construction schedule, and fully connected layers are used for other parts of the state information. The embeddings are then

passed through the actor and critic network layers. The reward is provided by the environment based on the selected action and the reward function. The <state, action, reward, next state> transitions are collected and used for training the network parameters using the clipped loss function of PPO [17]. The trained agents are then used to generate the schedule and objective values.

### 3.3 Human preference collection

The predicted schedules and corresponding objective values are presented to the human in a tabular form. The human selects the preferred schedule by entering its serial number; the remaining schedule options are considered rejected. This interaction provides a data point comprising (prompt, preferred response, and rejected responses), where the prompt corresponds to the input prompt for reward generation, and the preferred and rejected responses correspond to the reward structures for the selected and rejected schedules of the human. The same process is repeated across multiple iterations, yielding a dataset of human preferences that is used to fine-tune the LLM with DPO.

### 3.4 DPO-based LLM fine-tuning

To align an LLM with human preferences, it can be fine-tuned using preference data. In a seminal work, Ouyang et al. [22] utilized RLHF to fine-tune GPT-3. Their approach started with supervised fine-tuning, followed by training a reward model on a human-preference dataset. Finally, the language model is further fine-tuned using PPO and the learned reward model. In contrast, Bai et al. [15] directly trained a human preference model based on the preference dataset, which was used for fine-tuning the LLM using PPO. However, these approaches suffer from high computational costs and instability [16]. Addressing this, Rafailov et al. [16] proposed DPO, which directly fine-tunes the LLM based on the preference dataset, avoiding the need for training a separate reward or preference model, as well as the use of RL for fine-tuning. Unlike RLHF, DPO learns from the preference dataset via a loss function that increases the likelihood of preferred responses and decreases the likelihood of rejected responses. DPO can produce comparable or better quality solutions relative to RL-based LLM fine-tuning on human preferences, however, in a more efficient manner [16]. Therefore, DPO is selected in the proposed framework.

The dataset collected in Section 3.3 is used to fine-tune the LLM utilizing DPO. Instead of fine-tuning the entire base LLM, which would require significant compute, Low Rank Adaptation (LoRA) [23] was adopted, a widely used method for efficient fine-tuning of LLMs. LoRA keeps the base model parameters fixed and introduces trainable low-rank decomposition parameters in the transformer layers. Instead of training the base model parameters, a small set of trainable parameters is trained [23].

The updated LLM parameters are used to generate the reward structures and preference dataset again. This cycle of dataset collection and DPO-based fine-tuning is repeated for a fixed number of iterations.

## 4 Case study

### 4.1 Project details

The proposed framework is evaluated on a case study project (taken from Moselhi and El-Rayes [24]) to demonstrate its ability to generate human-aligned schedules. The project involved constructing a bridge in four sections. Each activity required one crew. The details of the project activities are shown in Table 1. Table 2 presents the daily cost for each crew and the maximum number of available crews. The optimization objectives were set to project time and cost, given their prevalence in construction scheduling [4,19].

### 4.2 Training specifications

The Owen2.5-3B-Instruct [25] model was selected as the LLM due to its manageable size and competitive performance across a wide range of tasks [26]. The PPO network architecture was kept largely the same as in [4], except the learning rate was set to 0.00025, the entropy coefficient to 0.01, and the number of training episodes to 350. Due to the unavailability of human-elicited preferences, this study adopted a set of lexicographic rules that represented the decision-maker's preferences. The preferences (Table 3) were based on a previous study [27], which found that reducing the project timeline is most preferred by construction managers, followed by maintaining crew and task unit continuity. Based on this, a test scenario was created, as demonstrated in Table 3. The objective was to show that, given a specific set of preferences, the proposed framework can automatically generate reward functions and, consequently, schedules increasingly aligned with those preferences as training progresses. In Table 3, $t_I$, $c_I$, and $w_I$ indicate the schedule time, crew continuity, and work continuity for the 1st-generated schedule option in each iteration. $c_I$ refers to the sum of the time difference between the predecessor and successor activities where a crew is idle, and $w_I$ the sum of the time difference between the predecessor and successor activities in a work section [28]. To evaluate the performance of the proposed framework relative to human-generated reward functions, separate PPO agents were trained with the reward function for time and cost objectives based on Agrawal et al. [4].

The first author acted as the human preference

selector. In each feedback step, the two schedules are displayed in a table, showing the start and end dates of the activities, along with the corresponding values of time, cost, crew, and task continuity. The user is prompted to enter the serial number of the preferred schedule (i.e., 1 or 2). This dataset is automatically transformed into a structured format, as discussed in Section 3.3. The specified reward format is shown in Figure 2. It divides the reward into a step and a terminal term, corresponding to the reward provided at a timestep and at the episode end, respectively. The "channel" indicates whether the reward component corresponds to time or cost. The components are indicated by "source", as described in Table 4. The "shape" and "weight" define the allowed mathematical transformation and coefficient of the component. The DPO training was conducted in five iterations, with 30 feedback samples per iteration.

Table 1: Project details

| ID | Activity | Section | Duration (Days) | Predecessor |
|---|---|---|---|---|
| A | Excavation | 1 | 13 | - |
| B | Foundation | 1 | 12 | A |
| C | Columns | 1 | 18 | A, B |
| D | Beams | 1 | 9 | A-C |
| E | Slabs | 1 | 16 | A-D |
| F | Excavation | 2 | 16 | - |
| G | Foundation | 2 | 12 | F |
| H | Columns | 2 | 15 | F, G |
| I | Beams | 2 | 9 | F-H |
| J | Slabs | 2 | 16 | F-I |
| K | Excavation | 3 | 11 | - |
| L | Foundation | 3 | 11 | K |
| M | Columns | 3 | 23 | K, L |
| N | Beams | 3 | 10 | K-M |
| O | Slabs | 3 | 13 | K-N |
| P | Excavation | 4 | 17 | - |
| Q | Foundation | 4 | 10 | P |
| R | Columns | 4 | 17 | P, Q |
| S | Beams | 4 | 8 | P-R |
| T | Slabs | 4 | 17 | P-S |

```
OUTPUT FORMAT (JSON):
    {{
    "step_terms": [
        {{"channel": "time|cost", "source": "variable_name",
        "shape": "linear|quadratic|sqrt|log", "weight": number}}
    ],
    "terminal_terms": [
        {{"channel": "time|cost", "source": "variable_name",
        "shape": "linear|quadratic|sqrt|log", "weight": number}}
    ],
    "apply_terminal_only_if_complete": true|false
    }}
```

Figure 2: Reward structure given in the prompt

Table 2: Crew daily costs (USD/day) and maximum crew availability

| Crew type | Cost, Crew No. | Crew type | Cost, Crew No. |
|---|---|---|---|
| Excavation | 340, 1 | Beams | 3931, 1 |
| Foundation | 3804, 2 | Slabs | 2230, 2 |
| Columns | 1875, 1 | | |

Table 3: Preferences for selecting schedule options

| S.No. | Preference scenario |
|---|---|
| 1 | If $t_I$-$t_{II}$ > 20, select II, and vice versa |
| 2 | Elif $c_I$-$c_{II}$ > 50, select II, and vice versa |
| 3 | Elif $w_I$-$w_{II}$ > 50, select II, and vice versa |
| 4 | Otherwise, select the schedule with the lower cost |

Table 4: Reward components

| Source | Meaning |
|---|---|
| $\Delta C$ | Cost incurred in the current timestep |
| $C$ | Total cost till the current timestep |
| $T$ | Current timestep of the agent |
| $N_s$ | Number of started tasks in the timestep |
| $\Delta T$ | Time increment in the current timestep |
| $T_s$ | Current schedule time |
| $N_F$ | Number of finished tasks in the timestep |
| $N$ | Total number of tasks |

### 4.3 Results

Each iteration took, on average, 138.38 minutes for agent training and preference collection, and 66.03 seconds for DPO training. Figure 3 demonstrates the evolution of $R_m$ for the four training iterations. The figures demonstrate a broadly increasing trend with the number of epochs. The overall $R_m$ values attained in iteration 5 were almost two orders of magnitude higher than those for iteration 1.

Table 5 shows the results for $R_w$ across 30 evaluation runs. As is evident, the final fine-tuned model generated reward structures that led to preferred schedules in 16 cases, resulting in a 59.3% win rate. On the other hand, the base model led to an $R_w$ of 40.7%, clearly lagging behind the fine-tuned model. In three of 30 scenarios, both models led to the same schedule objectives.

An average across ten training iterations with different random seeds demonstrated the compatibility of the schedules generated by the proposed framework with a human-generated reward function. Table 6 shows that the schedules generated by the proposed framework had comparable time, but lower values for crew continuity and task continuity.

Table 5: Win Rate results

| Quantity | Value |
|---|---|
| Number of comparisons | 30 |
| DPO Wins $(A)$ | 16 |
| Base model Wins $(B)$ | 11 |
| Ties | 3 |
| DPO Win Rate $[(A/(A+B)) * 100]$ | 59.3% |
| Base model Win Rate $[(B/(A+B)) * 100]$ | 40.7% |

Table 6: Comparison with human-generated reward (Average of ten training iterations)

| Quantity | Proposed framework | Baseline reward function |
|---|---|---|
| Schedule cost | 302773 | 314702 |
| Schedule time | 151.1 | 152.7 |
| Crew continuity | 114.8 | 121.7 |
| Task continuity | 52.9 | 55.1 |

## 5 Discussion

A higher $R_m$ indicates that the model can better distinguish between preferred and rejected responses [29]. Therefore, an increasing trend of $R_m$ in Figure 3 shows that, as the number of training iterations increases, the model assigns a higher implicit reward to the reward structures associated with the preferred schedules than to those associated with the rejected schedules. The model parameters after one DPO iteration assigned a higher $R_m$ to the same preference pairs in the next iterations, indicating it is moving closer to the provided preferences. The $R_w$ results further strengthen this, as the fine-tuned model clearly outperformed the base model. This shows that the model is able to learn to generate reward structures that lead to schedules that are better aligned with human preferences [16].

Although an increasing trend in $R_m$ is visible, the values also show instability. This is a widely reported limitation in DPO literature. DPO relies on a static base policy for updates, which may become increasingly less informative over time [30]. Recent studies have suggested the use of dynamic training settings in DPO, such as adaptive reward margins [30], which can help stabilize training by controlling the extent of updates from good and bad samples. Moreover, the DPO performance relies highly on data quality [31]. If the schedule pairs are not sufficiently different, they might not provide a strong enough learning signal [31].

The results in Table 6 demonstrate the ability of the proposed framework to incorporate human preferences relative to the baseline reward function. The proposed framework generated schedules with comparable duration, but better crew and task continuity. This can be attributed to the design of the baseline reward function, which primarily emphasized minimizing time and cost.

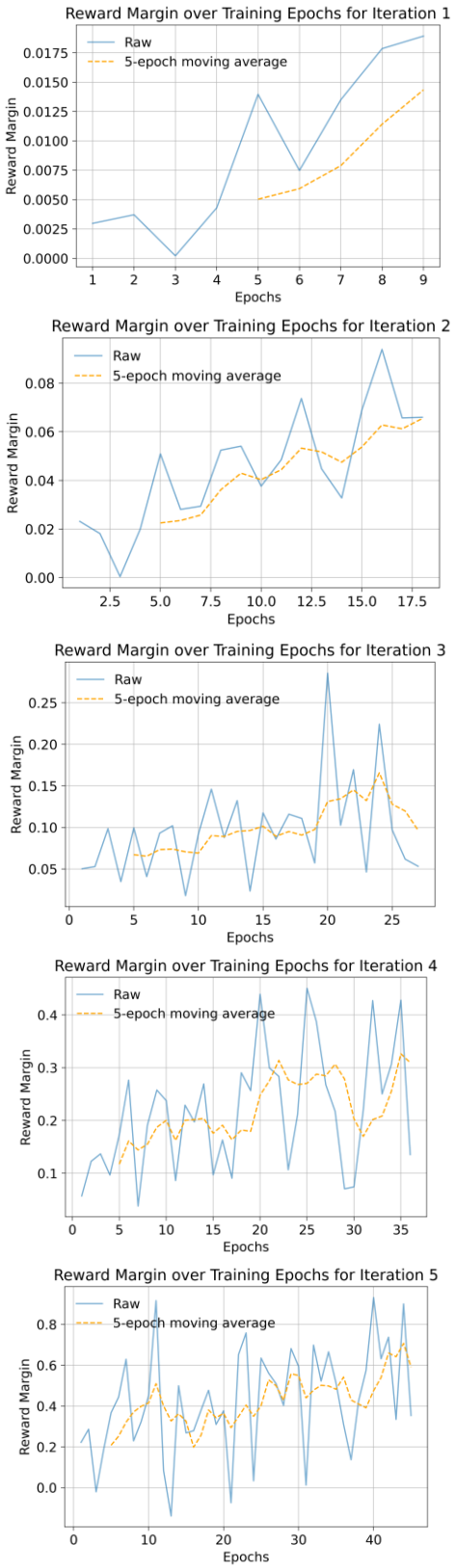


Figure 3: Reward margin trend over the training iterations

In contrast, the reward function generated by the proposed framework captures the crew and task continuity preferences. Therefore, the resulting schedules compensated for the time while improving the task and crew continuity. It is important to emphasize that the aim of this study is not to establish the superiority of reward functions designed by RL experts. Rather, it is to enable construction schedulers, who are often non-RL experts, to effectively generate schedules using RL without requiring reward engineering expertise.

Overall, the study's findings demonstrate the potential of the proposed framework to automatically generate human preference-aligned construction schedules in collaboration with humans. The study contributes to the existing literature on RL-based construction scheduling and provides a framework for humans to work with LLMs and RL agents, and guide them to generate reward functions and schedules tailored to their objectives and preferences. Through this, it also addresses the problem of complex reward function generation, a widely reported problem in the practical applications of RL [11].

## 6 Conclusion and outlook

This study presents a human-AI collaboration framework for reward function generation in RL-based construction scheduling. An LLM base model is used to generate reward options based on a human-provided prompt, including the project context, expected reward structure, and reward constraints. The reward options, generated in JSON format, are converted into Python code and plugged into a PPO-based scheduling agent. The scheduling agent trains on the reward options and generates schedules accordingly. The schedules are displayed to the user, who provides feedback by selecting the preferred one. Such a preference dataset is iteratively collected, and the LLM is fine-tuned on it using DPO. For fine-tuning, LoRA is utilized. Over iterations, the LLM agent learns to generate reward structures that lead to the development of schedules synchronized with human preferences. The proposed framework was evaluated through a case study project. The results demonstrated that over the course of training, the LLM agent learns to better differentiate between preferred and rejected reward structures. Moreover, the fine-tuned LLM achieved a higher win rate than the base LLM, demonstrating superior performance in terms of human preference alignment.

This study is a preliminary step towards true human-AI collaboration in construction scheduling. Technically, future research can tune DPO hyperparameters to improve stability and explore different LLM fine-tuning strategies to identify the best approaches for reward generation and fine-tuning. While the proposed framework only involves humans in the preference selection stage, future research should work towards more comprehensive human-AI collaboration by addressing aspects such as explainability, transparency, communication, and adaptability, among others [7]. Finally, integrating the reward-generation pipeline with the task-generation pipelines should be explored to enable human-preferred, optimized schedule generation based solely on project-specific prompts.

From the perspective of evaluation and generalizability, future work should focus on directly involving human stakeholders in validation and on evaluating the framework's ability to adapt schedules based on stakeholders' individual knowledge and preferences. Additionally, the framework should be tested across a diverse range of projects with varying types and scales to assess its generalizability.

The human preferences in decision-making may stem from their tacit knowledge [32]. In the proposed framework, over the course of training iterations, the agent seeks to understand these preferences and to create reward structures that are better aligned with them. As a result, the agent indirectly learns from the human's tacit knowledge of decision-making. Future studies can further explore the potential of the proposed framework to capture and reuse such knowledge.

## 7 References


[1] M. Kannimuthu, B. Raphael, E. Palaneeswaran, et al., Optimizing time, cost, and quality in multi-mode resource-constrained project scheduling. *Built Environment Project and Asset Management*, 9 (1):44–63, 2019.

[2] F. Amer, H.Y. Koh, M. Golparvar-Fard, Automated Methods and Systems for Construction Planning and Scheduling: Critical Review of Three Decades of Research. *Journal of Construction Engineering and Management*, 147 (7), 2021.

[3] N. Essam, L. Khodeir, F. Fathy, Approaches for BIM-based multi-objective optimization in construction scheduling. *Ain Shams Engineering Journal*, 14 (6):102114, 2023.

[4] A.K. Agrawal, Y. Zou, H. Jin, et al., Automated assembly sequence planning and scheduling in precast building projects using reinforcement learning, In *42nd International Symposium on Automation and Robotics in Construction (ISARC 2025)*, pages 1355–1362, Montreal, Canada.

[5] Y. Yao, V.W.Y. Tam, J. Wang, et al., Automated construction scheduling using deep reinforcement learning with valid action sampling. *Automation in Construction*, 166:105622, 2024.

[6] C. Liu, V.A. González, I. Pavez, et al., Exploring the Socio-Technical Nature of Lean-Based Production Planning and Control Using Immersive

Virtual Reality, In *Lean Construction 4.0*, Routledge.
[7] H. Nabizadeh Rafsanjani, A.H. Nabizadeh, Towards human-centered artificial intelligence (AI) in architecture, engineering, and construction (AEC) industry. *Computers in Human Behavior Reports*, 11:100319, 2023.
[8] R.K. Soman, K. Farghaly, G. Mills, et al., Digital twin construction with a focus on human twin interfaces. *Automation in Construction*, 170:105924, 2025.
[9] F. Aman, S.R. Abdul Hamid, A. Mat Isa, et al., A Systematic Review of Trends in Human–AI Collaboration Research. *The International Journal of Technology, Knowledge, and Society*, 21 (1):189–217, 2025.
[10] N. Montealegre-López, Exploring the role of trust in AI-driven decision-making: a systematic literature review. *Management Review Quarterly*, 2025.
[11] G. Dulac-Arnold, D. Mankowitz, T. Hester, Challenges of Real-World Reinforcement Learning. *arXiv:1904.12901*, 2019, Preprint.
[12] Z. Ding, H. Dong, Challenges of Reinforcement Learning, In *Deep Reinforcement Learning: Fundamentals, Research and Applications*, pages 249–272, Springer, Singapore, . https://doi.org/10.1007/978-981-15-4095-0_7, (Accessed: 01/22/2026).
[13] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, Second, The MIT Press, Cambridge, Massachusetts.
[14] P. Christiano, J. Leike, T.B. Brown, et al., Deep reinforcement learning from human preferences. *arXiv:1706.03741*, 2023, Preprint.
[15] Y. Bai, A. Jones, K. Ndousse, et al., Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. *arXiv.2204.05862*, 2022, Preprint.
[16] R. Rafailov, A. Sharma, E. Mitchell, et al., Direct Preference Optimization: Your Language Model is Secretly a Reward Model. *arXiv.2305.18290*, 2024, Preprint.
[17] J. Schulman, F. Wolski, P. Dhariwal, et al., Proximal Policy Optimization Algorithms. *arXiv:1707.06347*, 2017, Preprint.
[18] N.S. Kedir, S. Somi, A.R. Fayek, et al., Hybridization of reinforcement learning and agent-based modeling to optimize construction planning and scheduling. *Automation in Construction*, 142:104498, 2022.
[19] R.K. Soman, M. Molina-Solana, Automating look-ahead schedule generation for construction using linked-data based constraint checking and reinforcement learning. *Automation in Construction*, 134:104069, 2022.
[20] S.E. Seilabi, M. Saneii, M. Pourgholamali, et al., Reinforcement learning-based approach for urban road project scheduling considering alternative closure types. *Computer-Aided Civil and Infrastructure Engineering*, 40 (6):721–740, 2025.
[21] A.K. Agrawal, Y. Zou, M. Abdelmegid, et al., Automated generation of assembly schedules for precast building projects under uncertainty using reinforcement learning and Monte Carlo sampling. *Advanced Engineering Informatics*, 73:104510, 2026.
[22] L. Ouyang, J. Wu, X. Jiang, et al., Training language models to follow instructions with human feedback. *arXiv.2203.02155*, 2022, Preprint.
[23] E.J. Hu, Y. Shen, P. Wallis, et al., LoRA: Low-Rank Adaptation of Large Language Models. *arXiv:2106.09685*, 2021, Preprint.
[24] O. Moselhi, K. El-Rayes, Scheduling of Repetitive Projects with Cost Optimization. *Journal of Construction Engineering and Management*, 119 (4):681–697, 1993.
[25] Qwen/Qwen2.5-3B-Instruct · Hugging Face. On-line.: https://huggingface.co/Qwen/Qwen2.5-3B-Instruct, (Accessed: 01/27/2026).
[26] Qwen, A. Yang, B. Yang, et al., Qwen2.5 Technical Report. *arXiv:2412.15115*, 2025, Preprint.
[27] M. Tomczak, P. Jaśkowski, Preferences of Construction Managers Regarding the Quality and Optimization Criteria of Project Schedules. *Sustainability*, 13 (2), 2021.
[28] K. Hyari, K. El-Rayes, Optimal Planning and Scheduling for Repetitive Construction Projects. *Journal of Management in Engineering*, 22 (1):11–19, 2006.
[29] Y. Yan, X. Lou, J. Li, et al., Reward-Robust RLHF in LLMs. *arXiv:2409.15360*, 2024, Preprint.
[30] J. Wu, X. Wang, Z. Yang, et al., AlphaDPO: Adaptive Reward Margin for Direct Preference Optimization. *arXiv.2410.10148*, 2025, Preprint.
[31] P. Bernardelle, G. Demartini, Optimizing LLMs with Direct Preferences: A Data Efficiency Perspective, In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 236–240, Association for Computing Machinery, New York, NY, USA.
[32] R.K. Soman, K. Farghaly, G. Mills, et al., Digital twin construction with a focus on human twin interfaces. *Automation in Construction*, 170:105924, 2025.