http://dx.doi.org/10.1016/j.eswa.2023.122092

A heuristic rule adaptive selection approach for multi-work package project scheduling problem

Yaning Zhang[1,2], China, Email: yaningzhang@mail.nwpu.edu.cn

Xiao Li[3], China, Email: shell.x.li@hku.hk

Yue Teng[1], China, Email: yue.teng@polyu.edu.hk

Geoffrey Qiping Shen[1], China, Email: geoffrey.shen@polyu.edu.hk

Sijun Bai[2], China, mail: baisj@nwpu.edu.cn

[1] Department of Building and Real Estate, The Hong Kong Polytechnic University, Hong Kong SAR.
[2] School of Management, Northwestern Polytechnical University, Xi'an, China.
[3] Department of Civil Engineering, The University of Hong Kong, Hong Kong SAR. Email: xl1991@hku.hk, Tel.: (852) 5692 1399; Corresponding author

Abstract

Effectively scheduling a project is crucial for its success, especially after generating work packages from the work breakdown structure during the planning phase. Nevertheless, solving project scheduling problems with multiple work packages is challenging due to the inefficient utilization of work package information in existing scheduling approaches. To address this issue, this paper proposes the Heuristic Rule Adaptive Selection (HAS) approach for the Multi-Work Package Project Scheduling Problem (MWPSP). This approach involves work package information and employs reinforcement learning (RL) for intelligent decision-making in scheduling. First, the MWPSP with the optimization objective of minimizing the Portfolio Delay (PDEL) and the Average Percent Delay (APD) is defined, and a scheduling environment is established that integrates information from both work packages and tasks. Second, a Double Deep Q-network (DDQN) is employed to train agents for adaptively selecting heuristic rules of tasks and work packages. The performance of the HAS approach is then evaluated using a case project and the newly created MWPSP dataset. The experimental results demonstrate that the HAS approach exhibits superior solution quality and computational efficiency in optimizing PDEL and APD compared to heuristics approaches, e.g., single-priority rule-based heuristics and genetic algorithms. This achievement sets the stage for the development of next-generation adaptive scheduling for construction projects.

Keywords: Multi-work package project scheduling problem; Reinforcement learning; Adaptive scheduling; Deep Q-network

1. Introduction

Efficient project scheduling plays a critical role in ensuring the successful completion

of construction projects within specified timeframes and resource constraints (Hua et al., 2022). Timely completion is essential as it helps meet stakeholders' expectations, optimizes resource allocation, and minimizes costs (Vuorinen & Martinsuo, 2019). Inefficient scheduling can lead to delays, cost overruns, and disruptions during project execution, ultimately impacting project success (Sami Ur Rehman et al., 2022). Given the resource availability and the complex interdependencies of project tasks, meticulous planning and scheduling are necessary (Nguyen et al., 2022). However, with the increasing scale and complexity of construction projects, efficient project scheduling has become more challenging (Safapour et al., 2023).

A work breakdown structure (WBS) is an incremental decomposition of a project into a hierarchical tree structure, which has the potential to streamline complex construction projects. The fine-grained breakdown level is the work package, which includes one or more elemental tasks (C.-L. Li & Hall, 2019). In certain research studies, the term "task" is alternatively referred to as "activity." These terms generally have synonymous meanings within project management and can be used interchangeably (Schwindt & Zimmermann, 2015). This paper uses the term "task" to reduce potential confusion. By breaking down a complex project into manageable work packages, project managers can significantly reduce their scheduling workload. Unlike traditional project scheduling, which solely focuses on tasks, scheduling work packages and tasks together can not only enable the timely completion of work packages but also improve overall project scheduling performance (X. Li et al., 2022). Therefore, addressing the multi-work package scheduling problem in construction projects is necessary and crucial.

Priority rule-based heuristics are widely used for solving project scheduling problems

due to their advantages, such as efficient decision-making, reduced complexity, and adaptability (Van Eynde & Vanhoucke, 2020). However, traditional priority rule-based heuristics mainly consider task durations and resource requirements in project scheduling without fully using the work package information. For example, they may view a work package simply as a task without considering the work package size, the required resources in work packages, and the complexity of the precedence relationships within the work package (H.-W. Wang et al., 2020). Moreover, these priority rules cannot be updated adaptively as the project progresses, which may lead to poor performance in solving multi-work package scheduling problems (Pellerin et al., 2020).

To address the aforementioned issues, this study proposes an approach called heuristic rule adaptive selection (HAS). This approach integrates information from both work packages and tasks to optimize the makespan of the multi-work package project scheduling problem (MWPSP). The makespan is measured by two objectives: the Portfolio Delay (PDEL) and the Average Percent Delay (APD). It is worth noting that this problem shares similarities with the existing multi-project scheduling problem (MPSP). Section 2.1 will explain the differences between these two problems (e.g., scope and precedence relations). The HAS approach is inspired by the idea that the performance of heuristics can be enhanced by adaptively selecting heuristic rules based on the scheduling environment (Lin et al., 2019; Guo et al., 2021). Reinforcement learning (RL) is an effective approach for implementing such adaptive selections. Particularly, the Double Deep Q-Network (DDQN) can handle discrete information (i.e., the heuristic rules of tasks and work packages) and has great potential for achieving adaptive selection of heuristic rules (Osband et al., 2016). The objectives of this study

are twofold: 1) To establish the HAS approach with a work package information-enabled scheduling environment and a DDQN-based decision-making agent. 2) To validate the effectiveness of the HAS approach, controlled experiments will be conducted on a real construction project and a newly created MWPSP dataset.

The rest of this paper is organized as follows. Section 2 introduces the existing research on work package-based project scheduling, MPSP, and RL in project scheduling. Section 3 defines and formulates the problem. Section 4 describes the proposed HAS approach, including the establishment of the scheduling environment and the DDQN. Section 5 tests the HAS approach using a real modular construction project and further validates the generalization ability of HAS through a newly created project instances dataset. Meanwhile, the performance of the HAS approach is compared with state-of-the-art single-priority rule-based heuristics and metaheuristics. Section 6 presents the discussion and concludes the study.

2. Literature review

This study involves three relevant topics: construction project scheduling with work packages, the multi-project scheduling problem (MPSP), and reinforcement learning (RL) in project scheduling. Reviewing project scheduling with work packages helps to justify the significance of using work package-based scheduling for construction projects. Investigating MPSP helps clarify its relationship to the multi-work packages project scheduling problem (MWPSP) and provides a reference for MWPSP. The last topic justifies the superiority of RL in achieving adaptive scheduling. The details are summarized as follows.

2.1 Construction project scheduling with work packages

The work breakdown structure (WBS) has been used for various project management applications and is considered an early step in project planning (Salvendy, 2001). It establishes a framework for the project and serves as the foundation for obtaining relevant insights into the time and cost aspects of a project at different management granularity (Demeulemeester & Herroelen, 2006). Scheduling construction projects involves tasks from different specialized areas, which are numerous and do not have a common scheduling framework. This lack of standardization results in difficulty and cost to task-level scheduling (Lee & Hyun, 2019; Liu et al., 2020). However, work package-based scheduling in construction projects simplifies the scheduling process and supports the estimation of time and cost performance (C.-L. Li & Hall, 2019). Many studies have discussed how these advantages can assist with project planning and scheduling. X. Li et al. (2019) proposed a smart work packaging (SWP)-enabled constraint modeling service to explore critical constraints and interrelationships. They also analyzed the effects of these constraints on the schedule performance for prefabrication housing projects. In addition, the ontology and knowledge graph approaches were applied to SWP (X. Li et al., 2022, 2023). Wang et al. (2020) presented an integrated approach that enabled an information model for RCPSP-based scheduling. Within this approach, a work-package-based information model was proposed to capture all the necessary data of the RCPSP. Servranckx & Vanhoucke (2019) proposed a tabu search method for selecting an alternative work package with new content to optimize the scheduling problem's objective. Work package sizing can also significantly affect project performance, and a dynamic programming approach has been proposed to determine the work package sizing that achieves optimal project performance (C.-L. Li & Hall, 2019). Du et al. (2021) considered task splitting and

restructuring to create work packages of different granularities and proposed a metaheuristic solution for project scheduling. Many studies have been conducted on project scheduling using work packages, focusing on the impact of work package planning and sizing on project scheduling. *However, there is still a lack of approaches to comprehensively schedule work packages and their tasks after they are formed.*

2.2 MPSP

As mentioned in Section 1, MWPSP has similarities to MPSP. The research on MPSP is extensive and varied and has evolved over the years. However, to our knowledge, there has still been no systematic research on MWPSP. Therefore, the primary purpose of this review section is to analyze the similarities and differences between these two problems (see details in Table 1).

Table 1. The similarities and differences between MWPSP and MPSP

		MWPSP	MPSP	Reference
Similarities	They both study interrelated tasks with specific precedence relations and resource requirements.			(C.-L. Li & Hall, 2019; Servranckx & Vanhoucke, 2019)
Differences	Scope	Fine granularity	Coarse granularity	(X. Li et al., 2022; Gómez Sánchez et al., 2023)
	Precedence relations	They are necessary between work packages	They are optional between subprojects	(C.-L. Li & Hall, 2019; Van Eynde & Vanhoucke, 2022)
	Completion Necessity	All work packages should be completed	Not all subprojects have to be completed	(Ben Issa et al., 2021; X. Li et al., 2023)

Based on Table 1, the MWPSP can be considered as a special case of the MPSP. However, the most apparent difference is that the work packages have a specific

precedence relationship. As existing solutions mainly focus on MPSP, these solutions could serve as a reference for MWPSP. For example, both exact and approximate algorithms can be used to solve MPSP (Fu & Zhou, 2021; Satic et al., 2022; Gómez Sánchez et al., 2023). However, solving large-scale scheduling problems in a reasonable time is challenging due to their NP-hardness, making the exact algorithm difficult to implement. As for the approximate algorithm, priority rule-based heuristics are quite prevalent for RCMPSP due to their simplicity, efficiency, and flexibility (Ben Issa et al., 2021; Gómez Sánchez et al., 2023; Satic et al., 2022). *However, the priority rules in existing studies are usually unique and predetermined during the project scheduling process (Van Eynde & Vanhoucke, 2020). Limited studies have focused on adaptively selecting multiple different priority rules during the scheduling process.* In addition, some studies have demonstrated the limited improvement of meta-heuristic algorithms (another type of approximate algorithm) for MPSP (Van Eynde & Vanhoucke, 2022). This finding motivates us to use heuristic rules adaptive selection to address the MWPSP.

2.3 RL in project scheduling

Reinforcement learning is a sequential decision-making method that learns the optimal strategy and maximizes returns by interacting with the environment (Sutton & Barto, 2018). In RL, the decision-maker is called an ***agent***, while everything outside the agent is considered the ***environment***. The agents continuously select actions and interact with the environment, which responds by presenting new circumstances. Additionally, the environment rewards agents who seek to maximize rewards over time. Fig. 1 illustrates the interaction between the ***agent*** and the ***environment*** (Sutton & Barto, 2018).

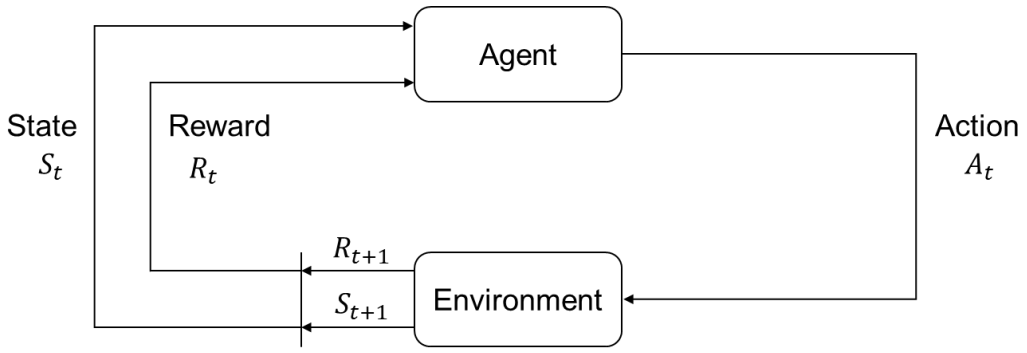


Fig. 1. The interaction between agent and environment in RL

RL has also emerged as an effective scheduling method in recent years (H. Zhang et al., 2021; Lei et al., 2022; Zhao et al., 2023). For example, Wauters et al. (2011) trained the relevant agent for each task in a project to use RL agents to select sequences and execution patterns for subsequent tasks. Chen et al. (2019) proposed an RL-based assignment policy approach to solve the multi-project scheduling problem in a cloud manufacturing system. Sung et al. (2020) formulated the resource allocation of a project schedule as a Markov Decision Process (MDP) and sought the best resource allocation policy using a deep RL algorithm. MDP is a mathematical framework that models decision-making problems in which an agent interacts with an environment. RL uses MDP as a basis for modeling decision-making problems (Ding et al., 2020). Recently, scholars have focused on advanced methods, such as combining metaheuristics with RL. For instance, several agents were used to process a group of solutions in parallel on separate computers. Every agent represents a distinct optimization algorithm, including local search, tabu search, and various specialized heuristics (Jedrzejowicz & Ratajczak-Ropel, 2007); Shahrabi et al. (2017) employed RL to update parameters of the variable neighborhood search during re-scheduling, rather than optimizing the initial scheduling. Z. Li et al. (2021) considered the genetic space as the action strategy space and the fitness function as the reward function for DQN to speed up the convergence of the genetic algorithm. DQN is one of the RL algorithms that combines

Q-Learning with deep neural networks (Mnih et al., 2015). And DDQN is a variant of DQN that addresses the overestimation bias by using two separate networks for selection and evaluation, leading to more accurate and stable action value estimates (Osband et al., 2016). *RL has great potential to realize the adaptive selection of heuristic rules for MWPSP. Particularly, the DDQN can handle discrete information, such as the heuristic rules. However, there is still a lack of research on integrating information from work packages and tasks to adaptively select heuristic rules in project scheduling problems.*

2.4 Research gap

In summary, the MWPSP has received limited research attention, leaving the question of "Can a heuristic rule adaptive selection (HAS) approach improve project scheduling performance by considering information from both work packages and tasks?" unanswered. This paper aims to address the research gap by proposing and testing the HAS approach.

3. Problem statement

The notations used in this section are listed in Table 2. This section defines the multi-work package project scheduling problem (MWPSP). As shown in Fig. 2, the project is represented as an activity-on-node (AON) network $G = (V, E)$ (Since activity-on-node (AON) is a professional term, "activity" is still used here. Its meaning is the same as "task").

Table 2. Notations for the MWPSP model

Notations	Meaning
G	The network graph of the project
V	$V = \{1,2, \dots, \lvert I \rvert\}$ is the node set that represents project work packages

E	$E \subset V \times V$ is the edge set that represents finish-start precedence relations between work packages
G_i	The network graph of the work package i
V_i	$V_i = \{1,2,\dots,\lvert N_i \rvert\}$, is the node set representing project tasks in the work package i
E_i	$E_{Ii} = V_i \times V_i$, is the edge set representing finish-start precedence relations between tasks in the work package i
I	The set of work packages
i	Work package number, $i \in \{1,2,\dots,\lvert i \rvert\}$
N_i	The set of tasks in work package i
j	Task number, $i \in \{1,2,\dots,\lvert N_i \rvert\}$
s_i	The start time of work package i
d_i	The duration of work package i
s_{ij}	The start time of task j in work package i
d_{ij}	The duration of task j in work package i
A_τ	The set of tasks being executed at time τ
r_{ijk}	Consumption of resource type k of task i in work package i
R_k	The availability of resource k for the project
ω	The set of types of resources, $\omega = \{1,2,\dots,K\}$

The first and last work packages are dummy work packages, each consisting of only one dummy task. For any work package i, there is an AON sub-network $G_i = (V_i, E_i)$. The first and last tasks are dummy tasks with durations of 0. The objective is to minimize the makespan of the entire project while ensuring that it aligns with the resource availability and the precedence relationships of work packages/tasks. The MWPSP model is formulated as follows, and the notations are listed in Table 2.

$$minF \tag{1}$$

Subject to:

$$s_i + d_i \leq s_{i'}, \forall (i,i') \in E \tag{2}$$

$$s_{ij} + d_{ij} \leq s_{ij'}, \forall (ij, ij') \in E_i \quad (3)$$

$$\sum_{i \in A_\tau} r_{ijk} \leq R_k, \forall k \in \omega, \tau \in \{1,2, \ldots s_{|N_i|} + d_{|N_i|}\} \quad (4)$$

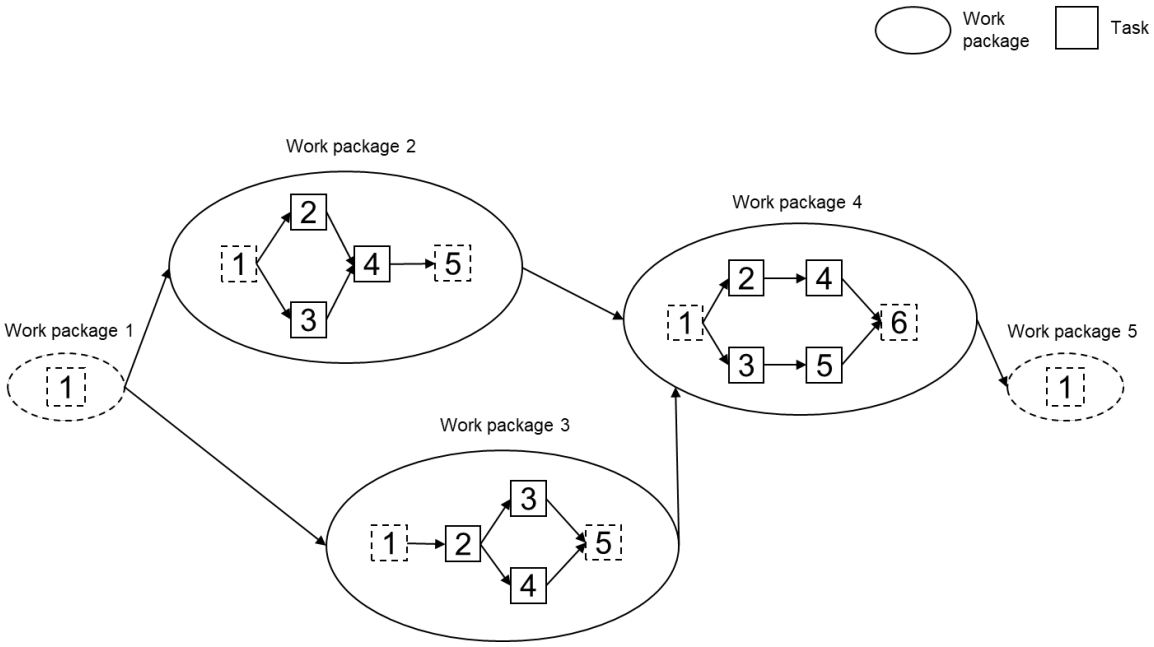


Fig. 2 The multi-work package project network

In Eq. (1), F represents the objective function of MWPSP. This study uses two objective functions: the portfolio delay (PDEL) and the average percent delay (APD) (Van Eynde & Vanhoucke, 2020). The PDEL is defined as follows:

$$PDEL = (M - CP_{max})/CP_{max} \quad (5)$$

This objective function evaluates the entire project's makespan M and normalizes it by the project's critical path CP_{max}.

The APD is defined as follows:

$$APD = (\sum_{i \epsilon I}(M_i - CP_i)/CP_i)/|I| \quad (6)$$

APD is the average of the normalized makespan of all work packages in the project. APD measures the average delay of work packages in a project. As the APD normalizes the average completion time of work packages using the critical path length of the work

package, APD avoids the impact of outliers (i.e., large or small values) on the average makespan. It also facilitates evaluating and comparing the delay with the planned schedule. In addition, the on-time completion of work packages can assist project managers and owners in having concrete visibility into the project's progress. It is also the prerequisite for quality inspection, delivery of outcomes, and financial payments for work packages (C.-L. Li & Hall, 2019; Shalaby & Ezeldin, 2021; X. Li et al., 2022). Thus, the APD is considered as a critical optimization objective.

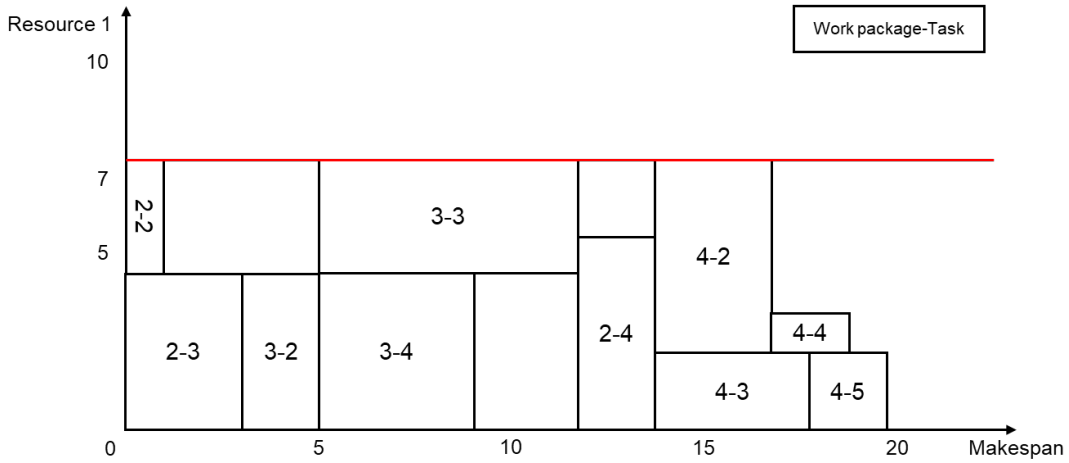
Eq. (2) ensures the existence of precedence relations between work packages. Eq. (3) establishes the existence of precedence relations between tasks. Eq. (4) limits the allocation of resources for the tasks to the available project resources. The formulation of the above MWPSP model is based on the following premises: 1) The resources are renewable. 2) The precedence relations are strict precedence. That means if task j in work package i is a predecessor of task j' in work package i', then all tasks of work package i precede all tasks in work package i' (C.-L. Li & Hall, 2019).

In the rest of this section, an example is presented to demonstrate the scheduling results in two scenarios: 1) considering only task information and 2) integrating information from both work packages and tasks. Here, we take the project in Fig. 2 as an example. The project resource availability, the task resource requirement, and the task duration are provided in Appendix A.

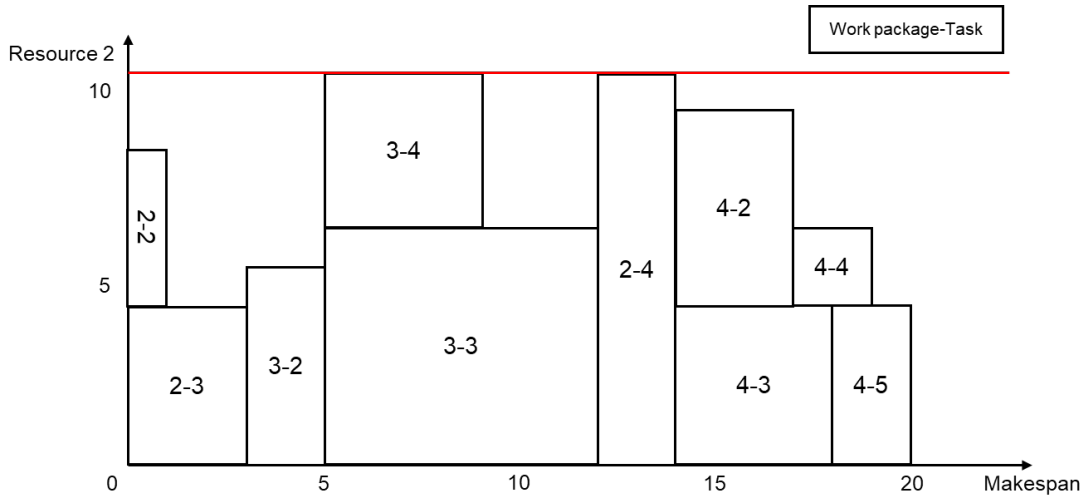
1) Considering only task information.

In this scenario, only the task information is considered during the scheduling process. For example, the earliest start time (EST) heuristic algorithm is used to obtain the

scheduling result. The Gantt chart is shown in Fig. 3. The start time and finish time of the project and work packages are provided in Table 3 (The duration of the dummy task duration is 0, so it is not displayed in the figures).



(a) The Gantt chart for Resource 1



(b) The Gantt chart for Resource 2

Fig.3 The schedule only considers task information

Table. 3 The start time and finish time of the project and work packages only considering task information

Work package/Project	Start time	Finish time	Makespan	CP length

Work package 1	0	0	0	0
Work package 2	0	14	14	5
Work package 3	3	12	9	9
Work package 4	14	20	6	6
Work package 5	20	20	0	0
Project	0	20	20	15

296

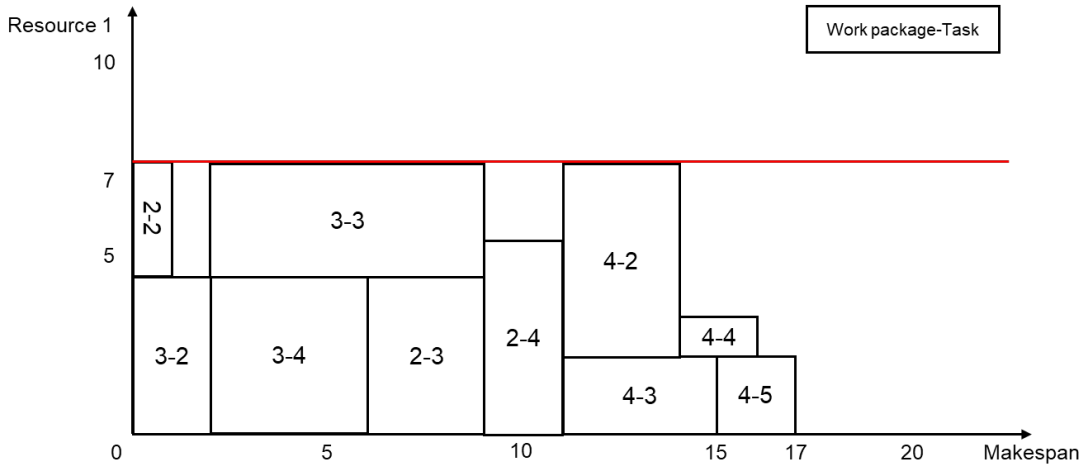
297 In the case of considering only the EST, task (2-2) and task (2-3) are scheduled first.
298 Their start times are both 0. The finish time of the last task (2-4) in work package 2 is
299 14. The makespan of work package 2 is $14 - 0 = 14$. The start time of the first task
300 (3-2) is 3. The finish time of the last task (3-3) in work package 3 is 12. The makespan
301 of work package 3 is $12 - 3 = 9$. The start time of the first task (4-2) in work
302 package 4 is 14. The finish time of the last task (4-5) is 20. The makespan of work
303 package 4 is $20 - 14 = 6$. The makespan of the project is $20 - 0 = 20$. Therefore,
304 the PDEL and APD of this schedule are calculated as follows:

305 $$PDEL = (M - CP_{max})/CP_{max} = (20 - 15)/15 = 0.3333$$

306 $$APD = (\sum_{i \epsilon I} (M_i - CP_i)/CP_i)/|I| = [(14 - 5)/5 + (9 - 9)/9 + (6 - 6)/6]/3 = 0.6$$

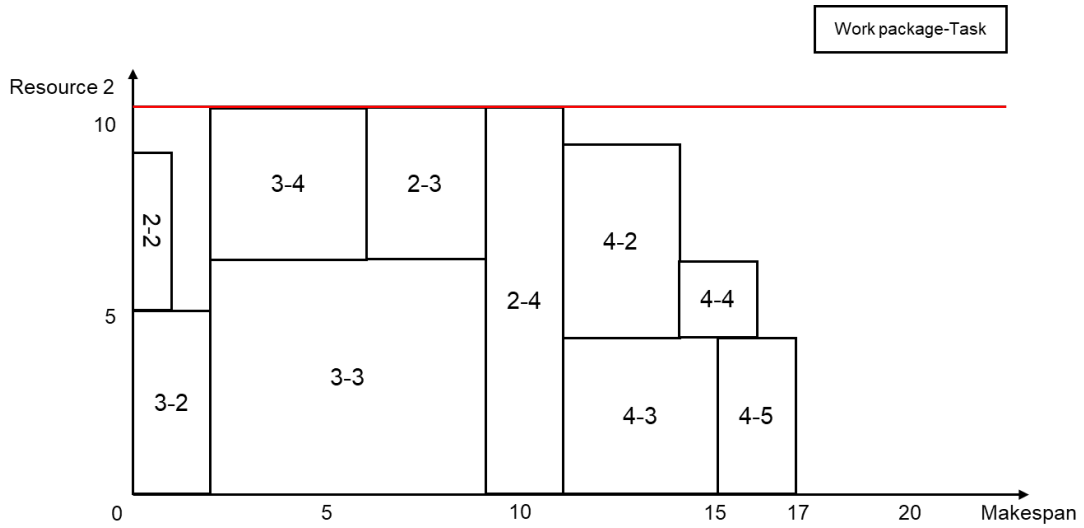
307

308 2) Integrating information from both work packages and tasks.
309 In this scenario, the tasks are integrated with information about the work package they
310 belong to, as well as information about themselves. For example, the total work content
311 of the work package is integrated with the earliest start time of the task. During the
312 scheduling process, priority is given to tasks in the work package with the maximum
313 total work content (MAXTWK), and further prioritization is given to the task with the
314 earliest start time (EST). The Gantt chart is shown in Fig. 4. The start time and finish
315 time of the project and work packages are provided in Table 4.



316

317 (a) The Gantt chart for Resource 1



318

319 (b) The Gantt chart for Resource 2

320 Fig. 4 The schedule integrating information from both work packages and tasks

321 Table 4 The start time and finish time of the project and work packages integrating
322 information from both work packages and tasks

Work package/Project	Start time	Finish time	Makespan	CP length
Work package 1	0	0	0	0
Work package 2	0	11	11	5
Work package 3	0	9	9	9
Work package 4	11	17	6	6
Work package 5	17	17	0	0
Project	0	17	17	15

323

In the case of considering the MAXTWK and EST, task (3-2) is scheduled first because the total work content of work package 3 (113) is larger than that of work package 2 (61). Their start time for work package 3 is 0. The finish time of the last task (3-3) in work package 3 is 9. The makespan of work package 3 is $9 - 0 = 9$. Then, work package 2 is considered according to the total work content. The start time of the first task (2-2) in work package 2 is 0. The finish time of the last task (2-4) is 11. The makespan of work package 2 is $11 - 0 = 11$. The start time of the first task (4-2) in work package 4 is 11. The finish time of the last task (4-5) is 17. The makespan of work package 4 is $17 - 11 = 6$. The makespan of the project is $17 - 0 = 17$. Therefore, the PDEL and APD of this schedule are calculated as follows:

$$PDEL = (M - CP_{max})/CP_{max} = (17 - 15)/15 = 0.1333$$

$$APD = (\sum_{i \epsilon I} (M_i - CP_i)/CP_i)/|I| = [(11 - 5)/5 + (9 - 9)/7 + (6 - 6)/6]/3 = 0.4$$

Integrating work package information in the schedule reduces PDEL from 0.3333 to 0.1333 and APD from 0.6 to 0.4. This finding demonstrates the potential benefits of integrating work package information into scheduling processes, which may improve PDEL and APD. Nevertheless, this particular instance is limited to a specific simple setting, which solely considers the task's earliest start time and work package total work content. Section 5 will delve into discussing the broader scenarios.

4. A heuristic rule adaptive selection (HAS) approach for MWPSP

The notations used in this section and the rest of the paper are shown in Table 5.

Table 5. Notations for Section 4 and the rest of the paper.

Notations	Meaning

$< \boldsymbol{s}, a, r, \boldsymbol{s}' >$	A quadruple consists of the state of the current step, the action, the reward, and the next step.
ES_{ij}	The earliest start time of task j of work package i. It is calculated by the Critical Path Method (CPM).
LS_{ij}	The latest start time of task j of work package i. It is calculated by CPM.
LF_{ij}	The latest start time of task j of work package i. It is calculated by CPM.
W_{ij}	The work content of task j of work package i. $W_{ij} = \sum_{k \in \omega} d_{ij} \cdot r_{ijk}$
CP_i	The critical path length of work package i. It is calculated by the CPM.
SP_i	Serial or parallel indicator of work package i. It is a metric used to measure the degree of serialization or parallelism of the project network.
$Q(\boldsymbol{s}, a)$	The action-value estimate for state $\boldsymbol{s}$ and action a.
α	The learning rate of DDQN
γ	The discount factor of DDQN
θ	The weights of networks.
t	The time step at which the agent interacts with the environment

This section proposes a heuristic rule adaptive selection (HAS) approach for solving the multi-work package project scheduling problem (MWPSP). The approach consists of two parts: 1) integrating information from both work packages and tasks to create a scheduling environment and 2) training a model to select heuristic rules adaptively. The HAS approach is implemented using reinforcement learning (RL), which enables agents to interact with an environment and achieve a specific target. As shown in Fig. 5, the detailed procedures for establishing the HAS approach for MWPSP are as follows: (1) To collect information on the project's work packages and tasks information and

integrate it. Information integration is achieved by combining work package priority rules and task priority rules to form combined heuristic rules.

(2) To provide the agent with the combined heuristic rules.

(3) To extract state features $\boldsymbol{s}$ of work packages and feed them to the agent (i.e., CNN);

(4) To perform forward propagation of the agent and generate a scheduling action a, which is a combined heuristic rule for selecting work packages and tasks.

(5) To calculate the reward r obtained by performing the action a; And extract the updated state feature $\boldsymbol{s}'$.

(6) To store the quaternions $< \boldsymbol{s}, a, r, \boldsymbol{s}' >$ obtained in this round.

(7) To update the network parameters by performing backward propagation using collected quaternions.

In the training process, repeating steps (3) to (6) can train an agent. In the execution process, repeating steps (3) to (5) can achieve project scheduling using a well-trained agent. In the above processes, the establishment of the scheduling environment and the training of the agent are described separately in sections 4.1 & 4.2.

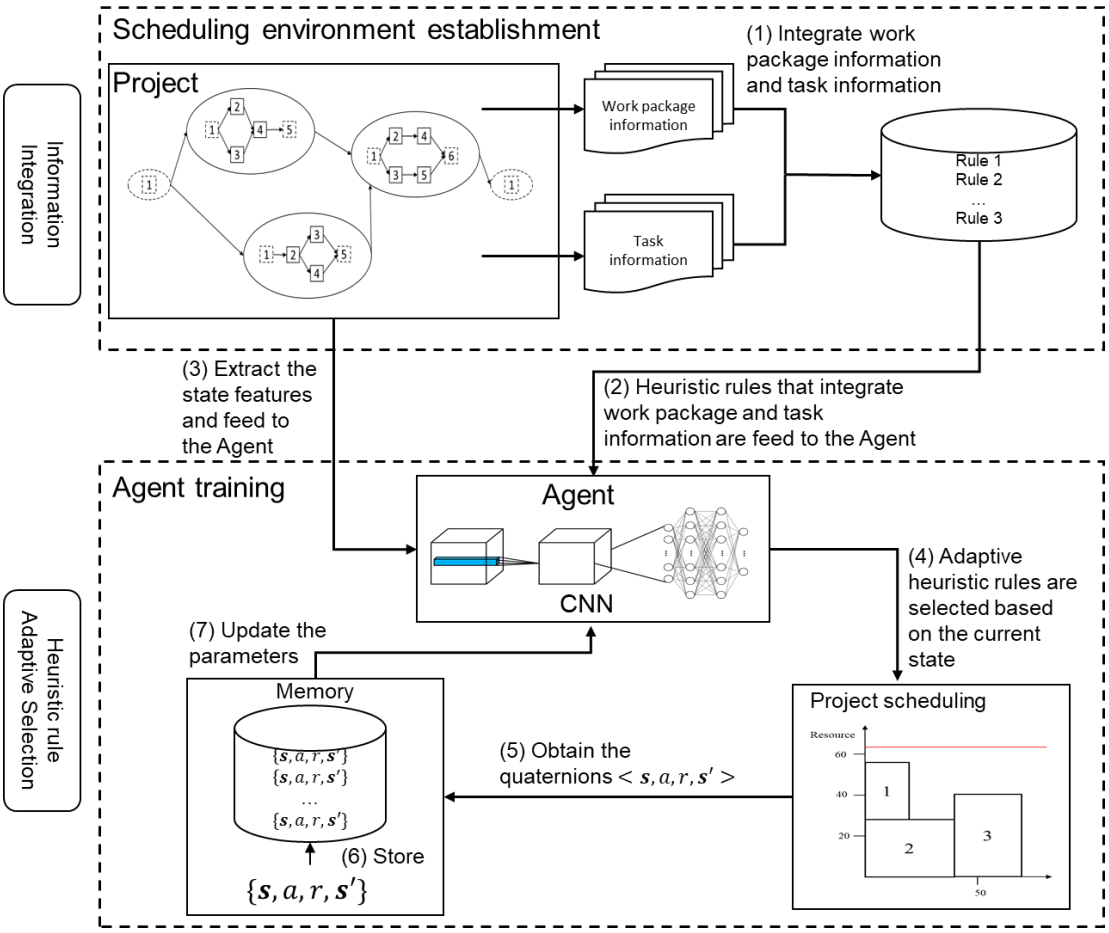


Fig. 5 The architecture of HAS approach

4.1 The scheduling environment establishment for the HAS approach

4.1.1 State features

A state represents the environment's current situation at a specific time step, including the information used for the agent's decision-making. State features refer to selected and relevant attributes or variables derived from the states. The selection of state features is crucial because it can greatly affect the performance and efficiency of the reinforcement learning algorithm (Yang et al., 2022). Previous studies have used various measures as state features in project scheduling, such as network complexity (NC), resource factor (RF), and resource strength (RS) (Pellerin & Perrier, 2019). These state features are typically organized into arrays and used as inputs to the agent (Chen

et al., 2019; Yang et al., 2022).

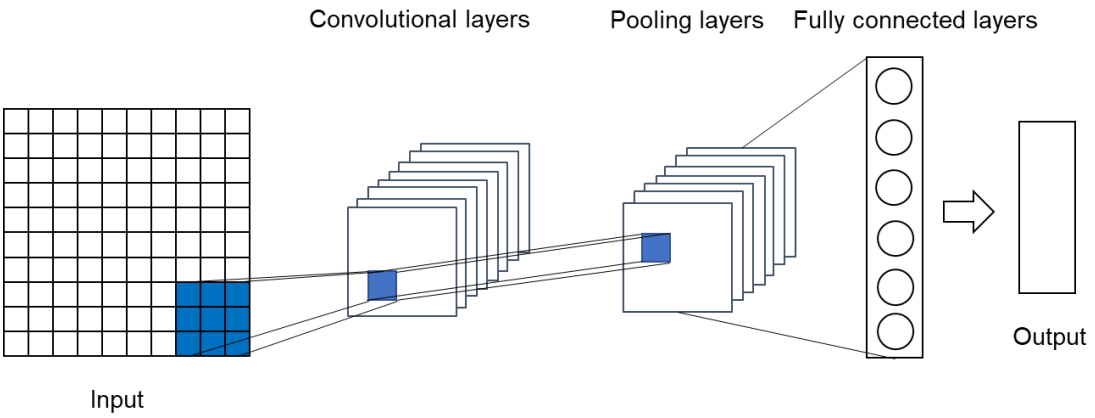


Fig. 6. An example of a CNN architecture

However, the previous measures used as features are arbitrarily selected and may not fully represent the project's states. Inspired by the image channels (Han & Yang, 2020), we construct a multi-channel matrix incorporating various original information, including duration, resource utilization, and scheduling progress. This matrix allows the agent to extract state features using convolutional neural networks (CNNs). As shown in Fig. 6, CNNs are a type of deep learning model specifically designed for analyzing and processing visual data, such as images and videos. They use layers of convolutions and pooling to automatically learn features and patterns from the data, making them highly effective for tasks such as image recognition, object detection, and more (G. Zhang et al., 2022).

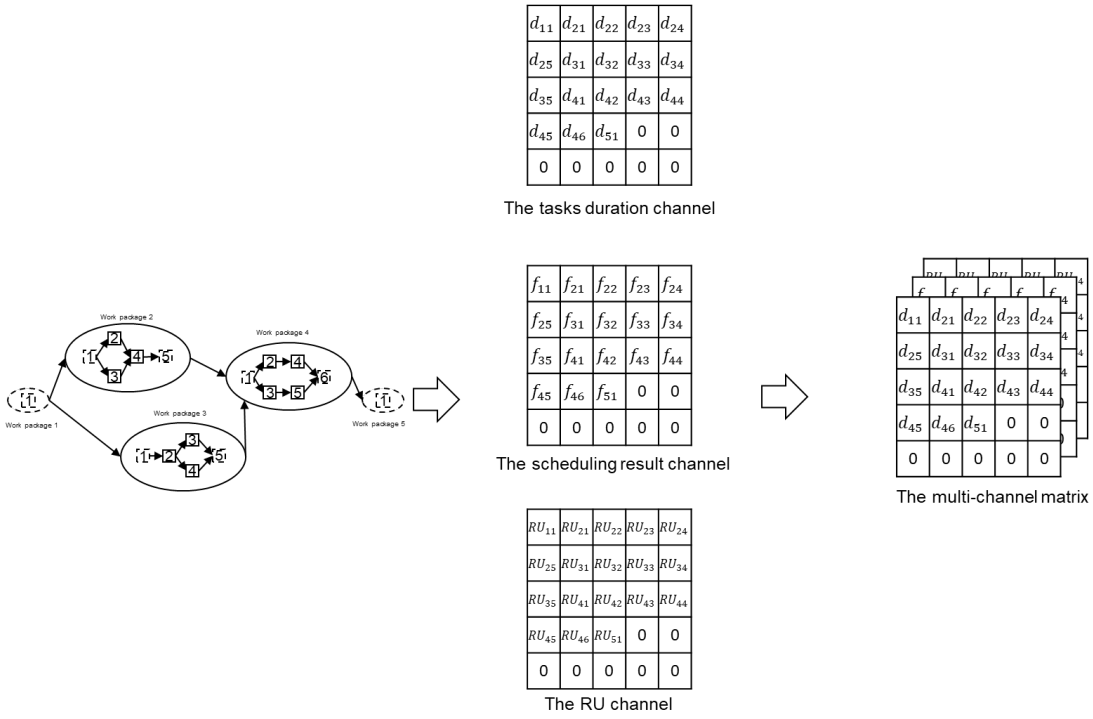


Fig. 7 Multi-channel matrix representing work package states

Fig. 7 presents a visual representation of the multi-channel matrix. In this representation, all channels are padded with zeros to create a matrix with equal height and width. Scheduling information is organized into three matrix channels: task duration d_{ij}, scheduling result f_{ij}, and resource utilization (RU). The task duration channel is a matrix that includes the duration of each task. The duration of the scheduled task and the dummy task is 0. The scheduling result channel is a matrix consisting of the finish time f_{ij} of each task. If the task is not scheduled, its finish time is recorded as 0. The RU channel represents the rate of resource utilization rate during the execution of a work package. RU in period t is calculated using Eq. (7):

$$RU_t = \left(\sum_{k=1}^{K}(x_{ijt} \cdot r_{ijk})\right) / \sum_{k=1}^{K} R_k \tag{7}$$

where x_{ijt} is a binary variable that takes the value 1 only if task j of work package i is being executed at time t. The notations are the same as those in the Table. 2. The generated multi-channel matrix can be used as input for the agent to extract the state features.

4.1.2 Actions

Actions refer to the potential choices or decisions that an agent can make to interact with the environment in a specific state. Scheduling actions are employed to allocate resources and determine the start time by selecting either a task or a work package. The effectiveness of the selected actions under various conditions has a significant impact on the outcome of the training and the quality of the solution. Thus, the action space should include efficient actions for selecting tasks or work packages to improve the agent's learning efficiency. The work package information and task information are integrated into the scheduling action to use the integrated information for making scheduling decisions. To this end, the heuristic rules utilized in RCMPSP are referred (Van Eynde & Vanhoucke, 2022). Five state-of-the-art task priority rules and four work package priority rules have been selected. And they are combined to form heuristic rules (see Table 6) that integrate information from both work packages and tasks. Work package information includes the number of tasks, critical path (CP) length, serial or parallel (SP) indicator, and the work content. The information represents the size, precedence relations, and resource requirements of the work package. Among them, the SP measures the similarity of a network to a serial or parallel graph based on the number of progressive levels (see definition in Appendix B). It takes values in the range of [0,1] (Vanhoucke et al., 2008). The calculation of SP is provided in Appendix B. Task information includes the earliest and latest start time, the latest finish time, slack, and the task work content. This information represents the task duration, precedence relations, and resource requirements. A total of 20 combined heuristic rules have been generated, which make up the scheduling action space. For instance, in "$MINTASK_{MINEST}$", the agent selects the work package with the minimum number of tasks and subsequently chooses the task with the minimum earliest start time among

438 those tasks for scheduling.

439

440 Table 6. Heuristic rules for composing action spaces

Type	Code	Name	Description
Work packages rules	MINTASK	Minimum number of tasks	$min_i(\|N_i\|)$
	MINCP	Minimum CP-length	$min_i(CP_i)$
	MINSP	Minimum SP value	$min_i(SP_i)$
	MAXTWK	Maximum total work content	$max_i(\sum_{j\in\|N_i\|} W_{ij})$
Task rules	MINEST	Minimum earliest start time	$min_j(ES_{ij})$
	MINLST	Minimum latest start time	$min_j(LS_{ij})$
	MINLFT	Minimum latest finish time	$min_j(LF_{ij})$
	MINSLK	Minimum slack	$min_j(LS_{ij} - ES_{ij})$
	MAXWK	Maximum work content	$max_j(W_{ij})$

441

442 *4.1.3 Rewards*

443 The reward is a scalar signal representing the agent's performance in the environment
444 after executing a specific action. The reward function establishes a mapping between
445 the reward and the triple of state, action, and next state. The reward function should
446 align with the objective of project scheduling. During agent training, a reward function
447 is constructed based on the objective function of PDEL. It aims to minimize the overall
448 completion time of the project. This choice is motivated by the fact that project
449 managers typically prioritize the timely delivery of the entire project. The rewards

available to the agent at each step can be defined using Eq. (8), which is calculated as the current makespan minus the next step makespan:

$$r_t^I = max_j(s_{ij} + d_{ij})_t - max_j(s_{ij} + d_{ij})_{t+1} \quad (8)$$

Agent training aims to maximize the cumulative reward, i.e., the sum of rewards per step. It can be calculated by Eq. (9):

$$\begin{aligned} R &= r_1 + r_2 + \cdots + r_t \\ &= max_j\left(s_{ij} + d_{ij}\right)_0 - max_j\left(s_{ij} + d_{ij}\right)_1 + max_j\left(s_{ij} + d_{ij}\right)_1 - max_j\left(s_{ij} + d_{ij}\right)_2 \\ &\quad + \cdots \\ &\quad + max_j\left(s_{ij} + d_{ij}\right)_{t-1} - max_j\left(s_{ij} + d_{ij}\right)_t \\ &= -max_j\left(s_{ij} + d_{ij}\right)_t \end{aligned} \quad (9)$$

4.2 Double Deep Q-network-based agent training for the HAS approach

A well-trained agent that can perform the optimal action based on the current state is crucial for enabling the heuristic rules adaptive selection. In this paper, the Double Deep Q-Network (DDQN) algorithm is chosen to train agents for the HAS approach. The DDQN algorithm is an extension of the Deep Q-Network (DQN) algorithm that addresses the issue of overestimation in Q-learning. In Q-learning, an agent learns to estimate the optimal action-value function (Q-function) by iteratively updating its estimates based on observed rewards and the next states. However, Q-learning is prone to overestimating action values, which can result in suboptimal policies. The DDQN algorithm addresses this problem by incorporating a target network that assists in selecting the optimal action during action-value estimation. The online network is responsible for selecting actions, while the target network is used for evaluating actions. As illustrated in Fig. 8, the states are organized as multi-channel matrices, and a convolutional neural network (CNN) is employed as a deep neural network for the DQN.

The training of agents follows the following steps:

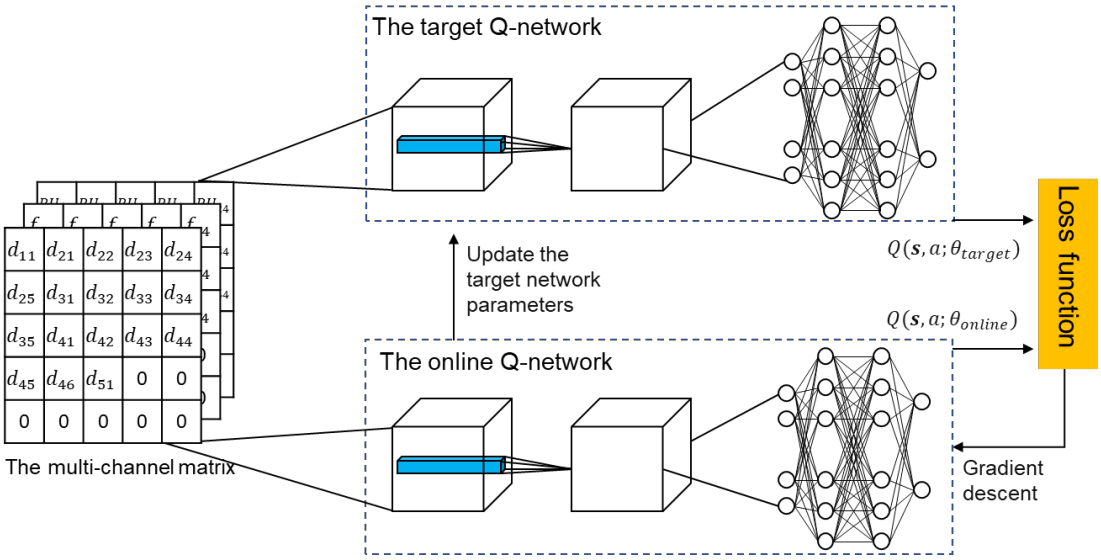


Fig. 8 The structure of DDQN

(1) Initialize the neural network architecture: DDQN utilizes two separate neural networks: the online and target networks. Both networks have the same architecture, typically consisting of multiple layers of neurons, followed by an output layer. The output layer represents the Q-values for each possible action in the given state.

(2) Experience replay: DDQN employs an experience replay buffer to store the agent's experiences during training. Experiences are quaternion $< \boldsymbol{s}, a, r, \boldsymbol{s}' >$ collected during interactions with the environment. The replay buffer allows for random sampling of experiences, which breaks the correlation between consecutive experiences and improves stability.

(3) Target network update: The target network is periodically updated to ensure stable target Q-values for training. Initially, the target network is an exact replica of the online network. After a specific number of iterations or episodes, the weights of the online network are copied to the target network.

(4) Q-value update: The agent selects an action based on an exploration-exploitation strategy during each training iteration. This study used the epsilon-greedy method. The

493 actions can be selected using Eq. (10).

494 $$a = \begin{cases} argmax_a Q(a) & with\ probability\ 1-\varepsilon \\ random & with\ probability\ \varepsilon \end{cases} \quad (10)$$

495 where ε is the probability of randomly selecting an action and is updated using the
496 following Eq. (11):

497 $$\varepsilon = \varepsilon_{min} + (\varepsilon_{max} - \varepsilon_{min}) \times e^{-(n_{iter}/\varepsilon_{decay})} \quad (11)$$

498 The selected action is executed in the environment, and the agent receives the resulting
499 reward and observes the next state. Using the online network, the agent calculates the
500 Q-value $Q(\boldsymbol{s}', a; \theta_{target})$ for the next state. The online network is used to estimate the
501 optimal action a^* for the next state, as shown in Eq. (12):

502 $$a^* = argmax_a Q(\boldsymbol{s}', a; \theta_{online}) \quad (12)$$

503 The Q-value for the current state and action is updated using Eq. (13):

504 $$Q(\boldsymbol{s}, a) = Q(\boldsymbol{s}, a) + \alpha \cdot (r + \gamma \cdot Q'\left(\boldsymbol{s}', argmaxQ(\boldsymbol{s}', a; \theta_{online}); \theta_{target}\right) - Q(\boldsymbol{s}, a))$$

505 (13)

506 (5) Gradient descent: To update the weights of the online network, gradient descent is
507 performed in order to minimize the loss function. The loss function is the mean squared
508 error (MSE) loss between the current Q-value prediction and the target Q-value, as
509 shown in Eq. (14):

510 $$L = (Q(\boldsymbol{s}, a; \theta_{online}) - (r + \gamma(\boldsymbol{s}', a^*; \theta_{online})))^2 \quad (14)$$

511 The gradients are computed by backpropagating the error through the network.
512 (6) Repeat and iterate: The agent continues interacting with the environment, updating
513 the Q-values, and optimizing the network iteratively. This process repeats for a
514 predetermined number of episodes.
515 The training processes (1)-(5) mentioned above are also depicted in Algorithm 1.

Algorithm 1: The training of DDQN

Data: The scheduling information
Result: The trained agent
Initialize Q_{online}-network with random weight θ_{online};
Initialize Q_{online}-network with $\theta_{target} = \theta_{inline}$;
Initialize replay buffer D;
for *episode in range(number_episodes)* **do**
  Initialize state $\boldsymbol{s}$;
  Select a using exploration-exploitation strategy based on $Q_{online}(\boldsymbol{s}, a; \theta_{online})$;
  Execute a in the environment;
  Observe $\boldsymbol{s'}$, r;
  Select a^* using Q: $a^* = argmax_a Q_{online}(\boldsymbol{s'}, a; \theta_{online})$;
  Calculate target Q-value: $Q_{target} = r + \gamma \times Q_{target}(\boldsymbol{s'}, a^*; \theta_{target})$;
  Update Q-value of current state and action: $Q(\boldsymbol{s}, a; \theta) \leftarrow (1 - \alpha) \times Q(\boldsymbol{s}, a; \theta) + \alpha \times Q_{target}$;
  Store experience $(\boldsymbol{s}, a, r, \boldsymbol{s'})$ in D;
  Sample a mini-batch of experiences from D;
  for *each sampled experience* $(\boldsymbol{s_i}, a_i, r_i, \boldsymbol{s'_i})$ **do**
    Select next action a_i^* using Q: $a_i^* = argmax_a Q(\boldsymbol{s'_i}, a; \theta_{online})$;
    Calculate target Q-value: $Q_{target} = r_i + \gamma \times Q_{target}(\boldsymbol{s'_i}, a_i^*; \theta_{target})$;
    Update Q-value of current state and action: $Q(\boldsymbol{s_i}, a_i; \theta_{online}) \leftarrow (1 - \alpha) \times Q(\boldsymbol{s_i}, a_i; \theta_{online}) + \alpha \times Q_{target}$;
  end
  Update the θ_{target};
  Update state $\boldsymbol{s}$ to next state $\boldsymbol{s'}$;
  if $episode\%update\_frequency == 0$ **then**
    $\theta_{target} \leftarrow \theta_{online}$;
  end
  if $episode\%evaluation\_frequency == 0$ **then**
    Evaluate the performance of the current policy;
  end
end

5. Experiments

This section evaluates the proposed HAS on a case project and a project instances dataset. All experiments are implemented using Python 3.8 and run on a PC platform equipped with Windows 10 64-bit operating system, Intel(R) Core(TM) i5-6300HQ CPU @ 2.30GHz.

5.1 Controlled experiment using a real construction project case

5.1.1 Project description

To demonstrate the effectiveness of the proposed HAS in MWPSP, a real-life modular

construction project consisting of five 28-story buildings in the Longhua District of Shenzhen is used as a case study. The project case considers the six typical renewable human resources, which include: R_1 production workers who manufacture modules; R_2 production managers and administrative personnel who manage and supervise production activities; R_3 transportation crews who load, unload, and transport modules; R_4 transportation managers and administrative personnel who manage and supervise transportation; R_5 construction workers who perform on-site construction; R_6 construction managers and administrative personnel who manage and supervise construction activities on site. The overall project networks, resource distributions, task durations, and work package information are provided in the Supplementary data.

5.1.2 Hyperparameters setting

The optimal performance of the HAS approach relies on determining the appropriate hyperparameter values. Finding optimal values for hyperparameters in reinforcement learning is challenging due to the large search space. To address this challenge, experiments are conducted to evaluate the performance of the HAS approach using different CNN structures, learning rates, buffer sizes, batch sizes, and target network update frequencies. The testing of hyperparameters is carried out based on the case projects. The case project involves 339 tasks, 25 work packages, and 6 renewable resources. These provide sufficient data for testing hyperparameters. Fig. 9 displays the experimental results for each hyperparameter.

Fig. 9 (a) compares three different CNN structures. The differences among these three CNN structures are reflected in the setting of a convolutional layer. The convolutional layers are characterized by the number of filters, kernel size, and stride. Structure 1 has three convolutional layers, with 60, 40, and 20 filters in each layer, respectively.

Structure 2 has two layers and fewer filters. Structure 3 has four layers and more filters. Table 7 provides further details of the convolutional layer settings. In addition, all CNN structures do not include pooling layers to preserve scheduling information and avoid any negative impact on training performance (Han & Yang, 2020).

Table 7 The tested CNN structures

CNN structure ID	Layers	The number of filters	Kernel size	Stride
CNN structure 1	Layer 1	60	(1,1)	(1,1)
	Layer 2	40	(2,2)	(1,1)
	Layer 3	20	(3,3)	(1,1)
CNN structure 2	Layer 1	40	(2,2)	(1,1)
	Layer 2	20	(3,3)	(1,1)
CNN structure 3	Layer 1	100	(1,1)	(1,1)
	Layer 2	80	(2,2)	(1,1)
	Layer 3	60	(3,3)	(1,1)
	Layer 4	40	(6,6)	(1,1)

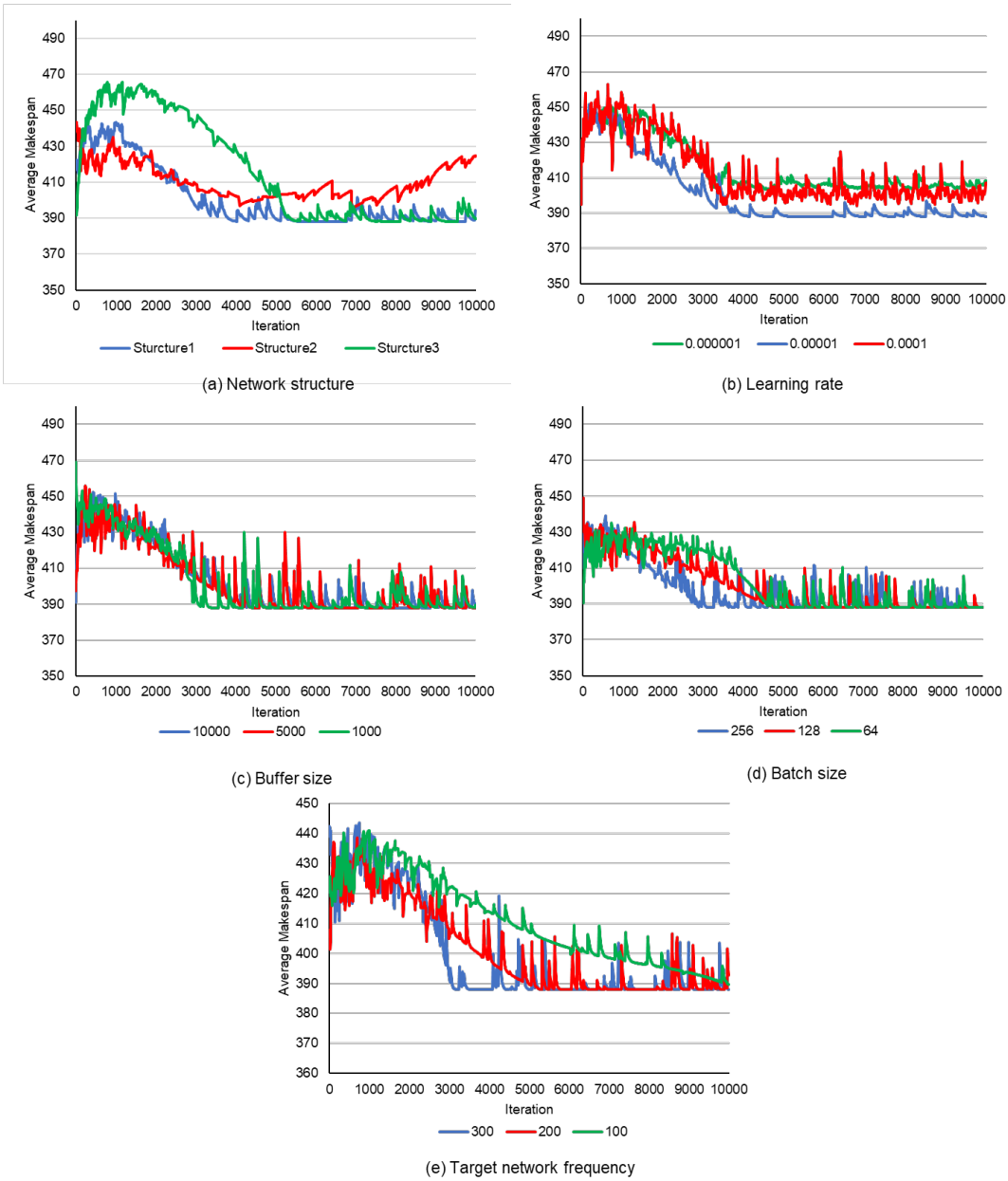


Fig. 9 Verification results of each hyperparameter

CNN structure 1 achieves the optimal project makespan after approximately 3000 training iterations, demonstrating the best training performance. CNN structure 2, a simpler CNN structure, achieves the best project makespan but exhibits unstable early performance. In contrast, employing a more complex structure (i.e., CNN structure 3) results in unstable training performance caused by overfitting. Fig. 9(b) indicates that moderate learning rates produce the best performance, while higher or lower rates prevent the model from converging to the optimal project makespan. Fig. 9(c) illustrates

566 that varying the buffer size does not significantly impact training performance. All three
567 selected buffer sizes lead to convergence to the best project makespan. A smaller buffer
568 size may identify the optimal project makespan earlier, but it lacks stability. Therefore,
569 a larger buffer size is selected to prioritize stability in convergence. Fig. 9(d) indicates
570 that a larger batch size accelerates model convergence. Thus, a batch size of 256 is
571 selected. Furthermore, Fig. 9(e) illustrates that increasing the frequency of target
572 network updates enhances model convergence. The fastest convergence speed is
573 achieved when the update frequency is set to 300. Table 8 provides a summary of the
574 final selected hyperparameter values.

575 Table 8 Hyperparameters

Hyperparameters	Type/Values	References
CNN structure	CNN structure 1	(Han & Yang, 2020)
Learning rate	0.00001	(Wen et al., 2021)
Buffer size	10000	(Fedus et al., 2020)
Batch size	256	(Lin et al., 2019)
Target network update frequency	100	(Z. Wang et al., 2016)

576

577 *5.1.3 Experimental results and analysis*

578 The performance of the HAS approach is evaluated through two sets of control
579 experiments. The first set compares two scheduling strategies: one that includes task
580 information and the other that integrates information from both work packages and
581 tasks. This set of experiments examines the effects of integrating work package
582 information into the scheduling process.

583

584 The second set of experiments compares the performance of HAS with other
585 approaches, including priority rule-based heuristic algorithms and metaheuristic

algorithms (e.g., genetic algorithm (GA)) (Pellerin et al., 2020). The reason for comparing the HAS approach with priority rule-based heuristic algorithms and GA is that they are the most representative approaches for solving MPSP. A recent review study showed that nearly half of the studies used rule-based heuristic algorithms or GA to solve MPSP (Gómez Sánchez et al., 2023). The details of GA are provided in Appendix C. All the approaches involved in the second set of experiments integrate information from both work packages and tasks. This comparison aims to demonstrate the superiority of the HAS approach. The performance of the aforementioned approaches is evaluated using two metrics: PDEL and APD.

(1) "Task" Vs. "Work Package + Task"

The above approaches that only consider task information are labeled as "Approach (TASK)." Similarly, the approaches that integrate information from both work packages and tasks (referred to as "integrating information" in Section 5) are labeled as "Approach (WP+TASK)." It is important to note that the GA500 represents the optimal solution out of 500 solutions. And the mean is used for "priority rule-based heuristics (WP+TASK)," for example,

$$MINEST(WP+TASK) = (\mathrm{MINTASK}_{\mathrm{MINEST}} + \mathrm{MINCP}_{\mathrm{MINEST}} + \mathrm{MINSP}_{\mathrm{MINEST}} + \mathrm{MAXTWK}_{\mathrm{MINEST}})/4 \quad (15)$$

Table 9 presents the PDEL, APD, and running time of the aforementioned approaches. The superior results in PDEL and APD achieved by "Approach (WP+TASK)" are highlighted in bold. Overall, the results of APD indicate that integrating information improves performance in all approaches. The improvement is particularly significant in the priority rule-based heuristic. However, the PDEL results suggest that integrating information may not always lead to improvement, as observed in MINLFT, MINSLK, and MAXWK. Integrating information leads to an increase in PDEL. This can be

attributed to the additional priority introduced by work package information, which leads to delays in tasks belonging to lower-priority work packages and consequently delays the entire project. Furthermore, integrating information does not significantly increase the running time in any approach. This is because integrating work package information not only influences the decision-making process for prioritization but also does not increase the computational complexity of the scheduling problem.

Table 9 Case project's results of "Task" Vs. "Work Package+Task"

	PDEL	APD	Running time (s)
HAS(TASK)	0.213	0.068	5.236
HAS(WP+TASK)	**0.198**	**0.061**	5.987
GA(500)(TASK)	0.213	0.076	193.810
GA(500)(WP+TASK)	**0.191**	**0.069**	229.231
MINEST(TASK)	0.340	0.131	0.106
MINEST(WP+TASK)	**0.324**	**0.074**	0.107
MINLST(TASK)	0.340	0.131	0.098
MINLST(WP+TASK)	**0.316**	**0.066**	0.101
MINLFT(TASK)	0.213	0.075	0.095
MINLFT(WP+TASK)	0.358	**0.070**	0.100
MINSLK(TASK)	0.284	0.198	0.091
MINSLK(WP+TASK)	0.390	**0.074**	0.101
MAXWK(TASK)	0.324	0.127	0.09
MAXWK(WP+TASK)	0.367	**0.085**	0.105

The *improvement degree 1* is defined to evaluate the optimization performance of integrating work package information. The *improvement degree 1* is defined by Eq. (16), where a negative value indicates improvement resulting from the integration of work package information. A minor improvement degree represents a more significant improvement.

$$Improvement\ degree\ 1 = (Approach(WP + TASK) - Approach(TASK))/ Approach(TASK) \times 100\% \quad (16)$$

Fig. 10 illustrates the improvement degree of PDEL and APD achieved by different approaches, highlighting the positive effects of integrating work package information. The improvement in APD is particularly noteworthy, with a maximum improvement of approximately 60%. Additionally, negative effects of integrating work package information are also observed. Specifically, when using *MINLFT* to generate scheduling plans, integrating work package information decreases APD by about 10% but increases PDEL by over 60%. Therefore, integrating work package information when using *MINLFT* to generate scheduling plans is not worthwhile.

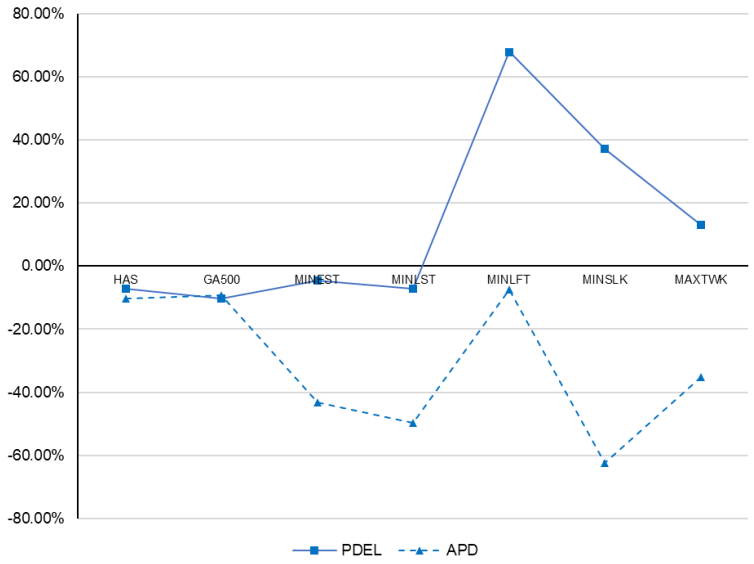


Fig. 10 The improvement degree of "TASK" Vs. "Work Package+Task"

(2) HAS approach Vs. other approaches

Table 9 demonstrates the effectiveness of implementing adaptive heuristic rule selection. It should be noted that: 1) The GA500 outperforms the HAS approach and priority rules-based heuristics in PDEL, while the HAS approach closely trails behind GA500. Both of them significantly outperform priority rules-based heuristics. 2) The HAS approach outperforms GA500 and priority rule-based heuristics in APD. These have demonstrated the superior optimization performance of the HAS approach. This

644 superiority is attributed to the adaptive selection of heuristic rules. Specifically, during
645 the project scheduling process, the makespan and availability of resources are subject
646 to constant change. The HAS can select the optimal heuristic scheduling rules based on
647 real-time makespan and resource availability. Moreover, the HAS approach exhibits
648 additional superiority by reducing the running time. The running time required by the
649 HAS (5.987s) is significantly less than that of GA500 (229.231s), accounting for only
650 0.5% of the latter's running time. Despite the considerable time required for pre-training
651 in the HAS, this process can be conducted offline. Once well-trained, HAS can
652 efficiently solve the MWPSP within a short timeframe. Conversely, reducing
653 computation time through pre-training is challenging for the GA500.
654
655 The *improvement degree 2* is defined to evaluate the effectiveness of HAS compared to
656 other approaches, as illustrated in Eq. (17). The "Other approaches" include GA500 and
657 five priority rule-based heuristic algorithms.
658 $$Improvement\ degree\ 2 = (HAS(WP + TASK) - Other\ approaches(WP + TASK))/$$
659 $$Other\ approaches(WP + TASK) \quad (17)$$
660 Fig. 11 illustrates the *improvement degree 2* of HAS in PDEL and APD compared to
661 other approaches. The results demonstrate that the performance of HAS is comparable
662 to that of GA500. Notably, HAS's PDEL is only about 3% worse than GA, while its
663 APD outperforms GA by around 12%. HAS demonstrates a significant improvement
664 compared to rule-based heuristics, with a maximum of approximately 49% in PDEL
665 and 25% in APD.

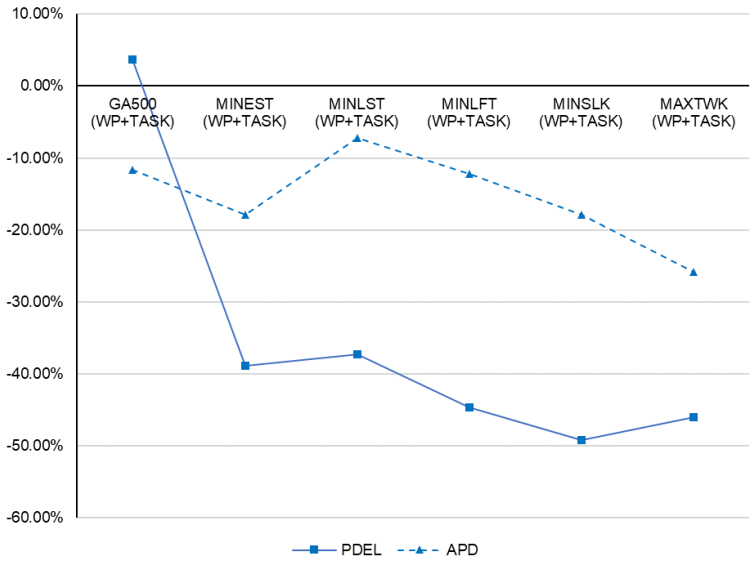


Fig. 11 The improvement degree of HAS Vs. other approaches

5.2 Controlled experiment using project dataset

5.2.1 Dataset description

This section evaluates the superiority of the HAS approach in more project instances. Typically, this is accomplished by using simulated project datasets that represent various project scheduling problems. The basic, multi-mode, and multi-project scheduling problems are frequently tested using corresponding project datasets (Kolisch & Sprecher, 1997; Van Eynde & Vanhoucke, 2022). However, no project dataset is completely suitable for the MWPSP. Thus, the project network generator RanGen2 (Vanhoucke et al., 2008) is used to create a new dataset for the MWPSP. When using Rangen2, we specify the following measures:

- TN: The number of tasks in each work package. TN is uniformly distributed integers from $\{25, 26, 27, \dots, 35\}$ in the dataset.
- WN: The number of work packages in a project. $\mathrm{WN} = \{8, 10, 12\}$ in the dataset;
- $\mathrm{SP_T}$: The serial or parallel indicator of tasks in each work package, $0 \leq \mathrm{SP_T} \leq 1$. The network is considered close to being serial if $\mathrm{SP_T}$ is close to 1, and the

network is considered close to being parallel if SP_T is close to 0. In our dataset SP_T is uniformly distributed from $[0{,}2, 0.8]$.

- SP_W: The serial or parallel indicator of work packages in the project. We set $SP_W = 0.7$ and $SP_W = 0.3$ to represent more serial work package networks and more parallel work package networks, respectively.
- RC: Resource constraint (RC) reflects the degree of limitation of the resources available to the project, where $0 \leq RC \leq 1$. We set $RC = 0.7$ and $RC = 0.3$ to represent lower and higher resource availability, respectively.

Accordingly, twelve sets of project instances are generated, each containing ten project instances, and their measures are shown in Table 10. All instances of the dataset are provided in the Supplementary data.

Table. 10 The value of the dataset measures

	SP_W	RC	WN	TN	SP_T
Set 1	0.3	0.3	8	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 2	0.3	0.7	8	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 3	0.7	0.3	8	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 4	0.7	0.7	8	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 5	0.3	0.3	10	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 6	0.3	0.7	10	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 7	0.7	0.3	10	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 8	0.7	0.7	10	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 9	0.3	0.3	12	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 10	0.3	0.7	12	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 11	0.7	0.3	12	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$
Set 12	0.7	0.7	12	$\{25,26,27,\ldots,35\}$	$[0{,}2, 0.8]$

5.2.2 *Experimental results and analysis*

In this section, two sets of control experiments are conducted to evaluate the performance of the HAS on the dataset. Experiments conducted on the dataset can confirm the superiority of HAS in more general situations.

(1) "Task" Vs. "Work package + Task"

Table 11 presents the mean results of ten instances in each set, computed by the HAS, GA, and priority rule heuristic (denoted as "RULE"). In the first set of experiments, the mean result obtained by the RULE represents its performance, which is denoted as "RULE(M)." Because the mean result can reflect the overall performance of the RULE. The superior results in PDEL and APD achieved by "Approach (WP+TASK)" are highlighted in bold. Integrating information consistently significantly improves APD across all approaches. This finding reinforces the notion that integrating work package information into the scheduling process facilitates the timely completion of individual work packages, thereby enhancing project efficiency. At the same time, integrating information has a slight negative impact on PDEL. However, the negative impact is insignificant in both HAS and GA. This is because the reward function of HAS and the fitness of GA are both designed based on the entire project's makespan. Integrating information does not significantly impede the project's progress. Moreover, running time indicates that integrating information does not increase computational complexity across all tested approaches.

To enhance comprehension of the performance and evaluate the impact of WN, SP_W and RC, Fig. 12 illustrates the *improvement degree 1* of the four sets of instances using Eq. (16). There are a few interesting findings: 1) For all approaches, integrating

information of both work packages and tasks demonstrates a significant improvement in APD for the majority of instances. 2) When $WN = 12$, integrating information from both work packages and tasks exhibits a significant negative impact on PDEL. 3) When $RC = 0.3$, indicating higher resource availability, the integration of information from both work packages and tasks also has a negative impact on PDEL. These findings can be attributed to the fact that projects with larger scale and higher resource availability tend to have more schedules. Integrating information increases the constraints of the problem, thereby eliminating partially feasible schedules. In contrast, considering only task information allows for exploring more schedules, resulting in an increased number of schedules with improved PDEL.

Table. 11 Results on MWPSP datasets: "Task" Vs. "Work Package + Task"

	RULE(M) (WP+TASK)	RULE(M) (TASK)	GA500 (WP+TASK)	GA500 (TASK)	HAS (WP+TASK)	HAS (TASK)
			PDEL			
Set 1	**1.707**	1.866	1.485	1.302	1.476	1.294
Set 2	**4.930**	6.502	**4.911**	4.923	**4.907**	4.912
Set 3	**1.000**	1.435	0.875	0.855	0.868	0.845
Set 4	**2.560**	3.558	**2.550**	2.558	**2.547**	2.548
Set 5	2.583	2.449	2.319	2.160	2.311	2.148
Set 5	**5.753**	5.757	**5.728**	5.736	5.726	5.726
Set 6	0.462	0.413	0.375	0.333	0.373	0.327
Set 7	**2.184**	2.185	**2.176**	2.178	**2.174**	2.176
Set 8	1.337	1.121	1.134	0.988	1.129	0.977
Set 9	4.415	4.405	**4.381**	4.392	**4.376**	4.387
Set 10	0.294	0.125	0.192	0.097	0.190	0.093
Set 11	1.842	1.836	**1.827**	1.832	**1.826**	1.830
Set 12	**1.707**	1.866	1.485	1.302	1.476	1.294
MEAN	**2.422**	2.638	2.329	2.280	2.325	2.272
			APD			
Set 1	**1.529**	2.195	**1.493**	1.952	**1.445**	1.815
Set 2	**4.508**	5.571	**4.538**	5.235	**4.567**	4.944
Set 3	1.279	0.915	**1.287**	1.532	**1.250**	1.387
Set 4	**3.038**	3.289	**2.856**	3.675	**2.834**	3.416
Set 5	**1.961**	2.195	**1.934**	2.211	**1.911**	2.167
Set 5	**4.733**	5.571	**4.422**	5.326	**4.679**	4.925
Set 6	**0.879**	0.915	**0.871**	0.966	**0.853**	0.942
Set 7	**2.968**	3.289	**3.048**	3.169	**2.985**	3.170
Set 8	**1.220**	2.393	**1.150**	2.520	**1.083**	2.343
Set 9	**4.192**	7.172	**3.961**	7.131	**3.934**	6.919
Set 10	**0.666**	1.490	**0.624**	1.517	**0.703**	1.473

Set 11	**2.871**	5.791	**2.855**	5.689	**2.887**	5.548
Set 12	**1.529**	2.195	**1.493**	1.952	**1.445**	1.815
MEAN	**2.487**	3.399	**2.420**	3.410	**2.427**	2.487
			Running time(s)			
Set 1	0.077	0.112	126.949	126.154	9.012	9.331
Set 2	0.074	0.108	122.178	124.662	9.863	10.042
Set 3	0.072	0.093	120.382	125.421	9.074	9.328
Set 4	0.072	0.099	120.528	127.339	10.206	10.113
Set 5	0.099	0.112	148.562	172.442	9.929	9.388
Set 5	0.100	0.108	163.592	154.259	10.216	9.366
Set 6	0.092	0.093	149.940	149.477	10.396	10.244
Set 7	0.099	0.099	159.730	151.816	10.062	10.361
Set 8	0.154	0.124	189.863	222.221	10.300	9.441
Set 9	0.153	0.124	193.995	221.917	9.043	9.068
Set 10	0.147	0.113	183.177	220.580	10.131	10.019
Set 11	0.151	0.119	190.145	209.287	9.223	9.477
Set 12	0.077	0.112	126.949	126.154	9.012	9.331
MEAN	0.107	0.109	155.753	167.131	9.788	9.6815

733

Fig. 12 The improvement degree of “Task” Vs. “Work Package + Task”

(2) HAS approach Vs. “Other approaches”

A second set of controlled experiments is conducted to evaluate the generalizability of HAS’s performance beyond the specific case project. The performance of the HAS, GA, and RULE is evaluated using the MWPSP dataset. In the second set of experiments, the optimal result obtained by the RULE represents its performance, which is noted as “RULE(O).” Because comparing the HAS with the optimal results can further demonstrate its superiority. GA produces two sets of results. One is the optimal solution out of 300 solutions (GA300), and the other is the optimal solution out of 500 solutions (GA500). Table 12 summarizes the averages of PDEL, APD, and running time. The superior results in PDEL and APD achieved by the HAS are highlighted in bold.

For the PDEL, the HAS outperforms the priority rule-based heuristic algorithm by incorporating an adaptive selection of heuristic rules during the scheduling process. The adaptive rules, tailored to the real-time scheduling state, effectively reduce the project makespan. The results demonstrate that the HAS exhibits robust optimization and generalization capabilities on the MWPSP dataset, outperforming the GA300 and achieving comparable performance to the GA500. For the APD, the HAS demonstrates optimal results in partial instances. However, the HAS’s performance is comparable to that of the GA500 in other instances. This is because the reward function of the HAS primarily focuses on minimizing project makespan, which leads to suboptimal performance in optimizing APD. For running time, experimental results confirm the high efficiency of the priority rule-based heuristics, as they can generate scheduling plans within a mere 0.2 seconds. Although the HAS requires more computational time, the running time of approximately 10 seconds is acceptable. Moreover, the HAS outperforms GA300 and exhibits similar performance to GA500 while consuming only

761 17.8% and 6.3% of running time, respectively.

762

763 To provide a more intuitive representation of the performance and analyze the effects
764 of WN, SP_W and RC, Fig. 13 illustrates the *improvement degree 2* of the four sets of
765 instances based on Eq. (17). Fig. 13 provides valuable insights into the performance of
766 HAS. 1) In most instances, the HAS demonstrates significantly superior performance
767 compared to priority rule-based heuristics in PDEL and APD. Furthermore, the HAS
768 outperforms the GA300 and shows similar results to the GA500. 2) The robustness and
769 generalizability of the HAS performance across projects of varying scales are
770 demonstrated by its insensitivity to the influence of WN. 3) When $RC = 0.3$ (indicating
771 higher resource availability), HAS demonstrates a significant improvement in PDEL.
772 4) When $SP_W = 0.7$ (indicating a more serial work package network), HAS
773 demonstrates superior improvement in APD. During such scenarios, improvements in
774 both PDEL and APD are observed. The aforementioned superiority can be attributed to
775 the intelligent decision-making abilities of its agent, which enable it to explore a broader
776 range of scheduling plans under conditions of high resource availability. In summary,
777 the above analyses effectively support the superiority of the HAS in terms of optimality
778 and generalization.

779 Table 12. The PDEL, APD, and Running time of HAS, GA, and RULE

	RULE(O) (WP+TASK)	GA300 (WP+TASK)	GA500 (WP+TASK)	HAS (WP+TASK)
		PDEL		
Set 1	1.579	1.514	1.485	**1.476**
Set 2	4.920	4.916	4.911	**4.907**
Set 3	1.000	0.892	0.875	**0.868**
Set 4	2.560	2.550	2.550	**2.547**
Set 5	2.400	2.357	2.319	**2.311**
Set 6	5.753	5.737	5.728	**5.726**
Set 7	0.396	0.381	0.375	**0.373**
Set 8	2.179	2.177	2.176	**2.174**
Set 9	1.223	1.145	1.134	**1.129**
Set 10	4.395	4.390	4.381	**4.376**

Set 11	0.218	0.195	0.192	**0.190**
Set 12	1.835	1.830	1.827	**1.826**
MEAN	2.372	2.340	2.329	**2.325**
		APD		
Set 1	1.485	1.516	1.493	**1.445**
Set 2	4.292	4.640	4.538	4.567
Set 3	1.123	1.268	1.287	1.250
Set 4	2.927	2.844	2.856	**2.834**
Set 5	1.788	1.902	1.934	1.911
Set 6	4.604	4.632	4.422	4.679
Set 7	0.872	0.861	0.871	**0.853**
Set 8	2.679	3.037	3.048	2.985
Set 9	1.107	1.167	1.150	**1.083**
Set 10	3.976	4.145	3.961	**3.934**
Set 11	0.525	0.665	0.624	0.703
Set 12	2.332	2.467	2.855	2.887
MEAN	22.309	2.429	2.420	2.427
		Running time(s)		
Set 1	0.077	45.097	126.949	9.012
Set 2	0.074	43.188	122.178	9.863
Set 3	0.072	41.616	120.382	9.074
Set 4	0.072	42.225	120.528	10.206
Set 5	0.099	54.323	148.562	9.929
Set 6	0.100	57.314	163.592	10.216
Set 7	0.092	52.964	149.940	10.396
Set 8	0.099	56.583	159.730	10.062
Set 9	0.154	68.854	189.863	10.300
Set 10	0.153	68.034	193.995	9.043
Set 11	0.147	64.605	183.177	10.131
Set 12	0.151	66.602	190.145	9.223
MEAN	0.107	55.117	155.753	9.788

780

781

782 Fig. 13 The improvement degree of HAS approach compared to GA and RULE

6. Conclusions

This paper investigates the multi-work package scheduling problem (MWPSP) and proposes the heuristic rule adaptive selection (HAS) approach for intelligent decision-making. The primary objective of this study is to improve adaptive project scheduling by integrating work package information. Controlled experiments on case projects demonstrate that HAS significantly reduces the portfolio delay (PDEL) and the average percent delay (APD). Additionally, experiments on a newly established MWPSP dataset indicate the following findings: 1) The proposed HAS approach outperforms traditional single priority rule-based heuristics. HAS achieved comparable performance to metaheuristic algorithms while maintaining higher solution efficiency. 2) In all approaches, integrating information from both work packages and tasks demonstrates a significant improvement in APD in the general case. 3) The HAS demonstrates distinct advantages compared to priority rule-based heuristics and meta-heuristics when the work package network is serial and resource availability is high.

The study makes three contributions: 1) It defines the multi-work package scheduling problem (MWPSP) with with the optimization objective of minimizing the PDEL and the APD and creates a dataset for MWPSP to enhance controlled experiments and provide a reference for testing future scheduling approaches. 2) The DDQN-based HAS approach is proposed for MWPSP, which integrates information from both work packages and tasks and facilitates adaptive decision-making in project scheduling. The experiment has demonstrated that integrating work package information into the scheduling process can improve scheduling efficiency. The HAS showed superior optimality and generalization compared to priority rule-based heuristics and meta-heuristics, highlighting its effectiveness in solving MWPSP. 3) Experimental results

offer practical implications for project managers. For example, project managers typically prefer to complete the entire project as quickly as possible. However, considering the requirements of each work package is also essential. For example, integrating work package information by HAS can significantly improve the average percent delay.

Future studies can investigate stochastic factors and explore other optimization objectives in the MWPSP. For instance, studying stochastic task durations instead of deterministic ones could enhance the practicality of the HAS approach. In addition to minimizing the makespan, other objectives, such as net present value and resource leveling, can also be considered. Furthermore, alternative reinforcement learning algorithms, such as Actor-Critic (A2C), could be further investigated to enhance intelligent decision-making capabilities in project scheduling.

Supplementary data

The data covered in the paper are provided at https://github.com/dzzyn/A-heuristic-rule-adaptive-selection-appraoch-for-multi-work-package-project-scheduling-problem.

CRediT authorship contribution statement

Yaning Zhang: Methodology; Data Curation; Writing - Original Draft; **Xiao Li**: Conceptualization; Supervision; Writing - Review & Editing; **Yue Teng:** Data Curation; Investigation ; **Geoffrey Qiping Shen**: Resources; Supervision; **Sijun Bai**: Supervision

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal

relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by grants from the Research Grants Council of the Hong Kong SAR of China and the National Natural Science Foundation of China, The University of Hong Kong (RGC Project No.15219422 & G-HKU502/22, NSFC Project No. 72201228, HKU Project No. 2201100548, 2023A1515011162, 260910193).

Data availability

Data has been provided in “Supplementary data”.

Appendix A

T The project information in Fig. 2

Work package/Task No.	Duration	Resource 1 requirement	Resource 2 requirement	
Work package 1				CP length = 0
Task 1-1	0	0	0	
Work package 2				CP length = 5
Task 2-1	0	0	0	
Task 2-2	1	3	4	
Task 2-3	3	4	4	
Task 2-4	2	5	10	
Task 2-5	0	0	0	
Work package 3				CP length = 9
Task 3-1	0	0	0	
Task 3-2	2	4	5	
Task 3-3	7	3	6	
Task 3-4	4	4	4	
Task 3-5	0	0	0	
Work package 4				CP length = 6
Task 4-1	0	0	0	
Task 4-2	3	5	5	
Task 4-3	4	2	4	
Task 4-4	2	1	2	
Task 4-5	2	2	4	
Task 4-6	0	0	0	
Work package 5				CP length = 0
Task 5-1	0	0	0	

	Resource availability = 7	Resource 1 availability = 10 2 CP length = 15

Appendix B

For an Activity-On-Node (AON) graph $G = (V, E)$, For any of these nodes $i \in V$, define P_i as the set of its predecessor nodes. For SP_W, the nodes represent work packages in a project. For SP_T, the nodes represent tasks in a work package.

Define $Progressive\ level\ PL_i$ as follows,

$$PL_i = \begin{cases} 1 & if\ P_i = \emptyset \\ max_{j \in P_i} PL_j + 1 & if\ P_I \neq \emptyset \end{cases}$$

And define m is the maximum value for PL_i.

Therefore, The Serial or parallel indicator SP can be defined as follows,

$$SP = \begin{cases} 1 & if\ |V| = 1 \\ (m-1)/(|V|-1) & if\ |V| > 1 \end{cases}$$

SP takes values in the range of [0,1]. When SP=0, all tasks are parallel; when SP=1, all tasks are serial.

Appendix C

The genetic algorithm (GA) is implemented based on heuristic rules for encoding. When only tasks are considered, heuristic rules based on task information are used for encoding individuals. When both work packages and tasks are considered, heuristic rules based on combined work package and task information are used for encoding individuals.

Take the project in Fig.2 as an example. When only tasks are considered, an individual can be encoded as:

$\{MINEST, MINLST, MINLFT, MINSLK, MAXWK, MINEST, MINLST, MINLFT, MINSLK, MAXWK, MINLFT, MINSLK, MAXWK, MINEST, MINLST, MINLFT, MINSLK, MAXWK\}$

The following uses two chromosomes as an example to illustrate the process of genetic iteration. 1) Initialization: Randomly encode the chromosomes of the initial population (Fig. 14).

Fig, 14 Initialization

2) Crossover: When the crossover conditions are met, the father's and mother's chromosomes are crossed over (Fig. 15).

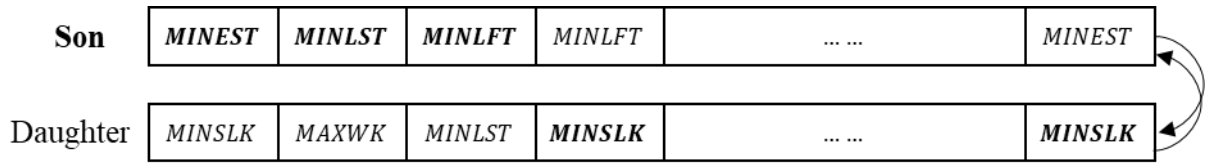


Fig. 15 Cross over

3) Mutation: When the mutation conditions are met, the genotype at a random position in the chromosome mutates into any other available genotype (Fig. 16). Finally, the offspring are selected for the next iteration.

Fig 16 Mutation

In the above process, the genotype encoding adopts the priority rules of the task, so

only the task information is considered.

When both work packages and tasks are considered, an individual can be encoded as:

$$\{MINTASK + MINEST, MINCP + MINLST, MINSP + MINLFT,$$
$$MAXTWK + MINSLK, MINCP + MINLST, MINSP + MINLFT,$$
$$MINTASK + MINEST, MINCP + MINLST, MINCP + MINLST,$$
$$MINSP + MINLFT, MAXTWK + MINSLK, MINCP + MINLST,$$
$$MINCP + MAXWK, MINSP + MINLFT, MINTASK + MINSLK,$$
$$MINTASK + MINEST, MINCP + MAXWK, MINSP + MINLFT\}$$

The integration of task information and work package information is realized by changing the encoding method. And the selection, crossover, and mutation processes of GA are the same. Fig.17 uses two chromosomes as an example to illustrate the process of genetic iteration.

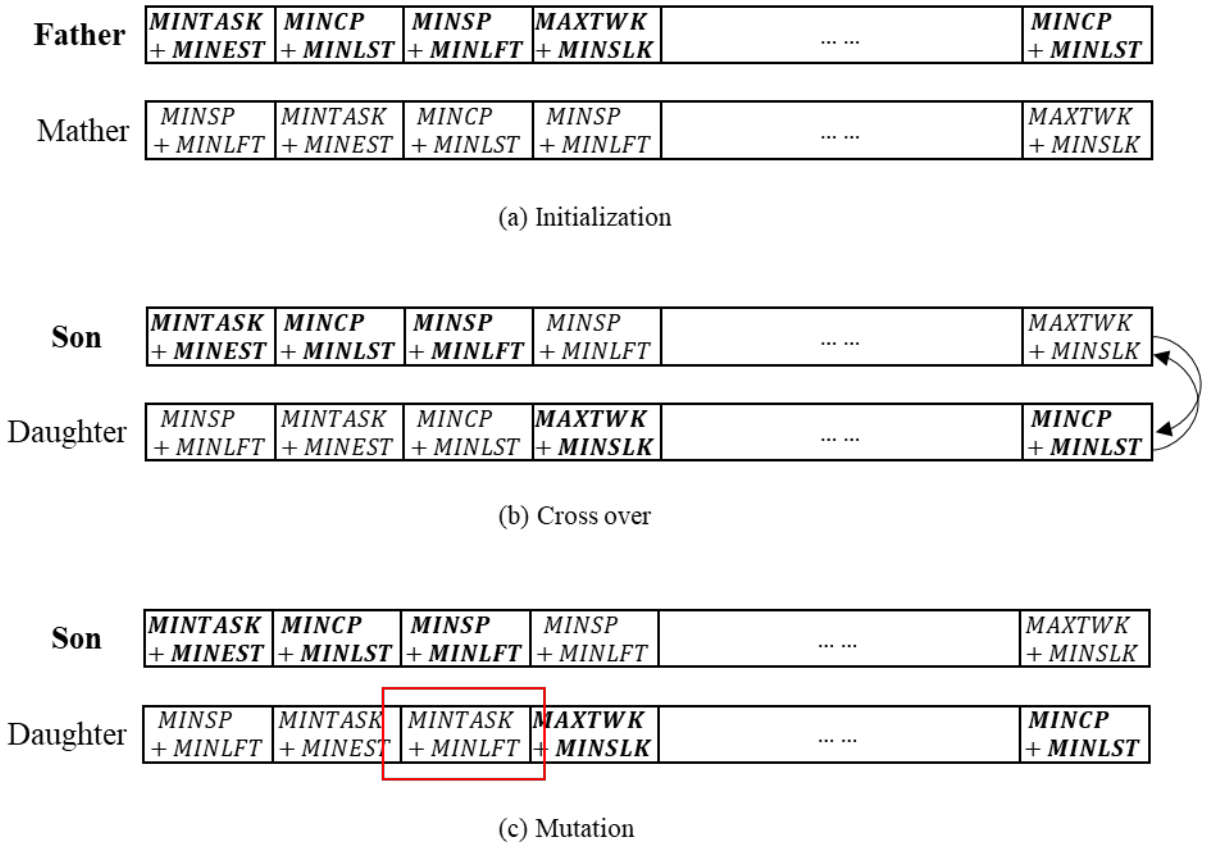


Fig. 17 The process of genetic iteration

In the above process, the genotype encoding adopts the priority rules of the "task+WP," so the task and work package information are both considered.

The pseudocode of the GA is shown in Algorithm 2.

Algorithm 2: The Genetic Algorithm

```
Data: Project tasks, heuristic rules, population size, max generations,
      crossover rate, mutation rate
Result: Best individual representing a feasible schedule
Initialization:
Initialize an empty population list;
for individual in population size do
    Initialize an empty heuristic rule list;
    while random probability < add rule probability do
        Randomly select a heuristic rule from options;
        Add the selected rule to the heuristic rule list;
    end
    Add the heuristic rule list to the population;
end
for generation = 1 to max generations do
    Evaluate fitness of each individual;
    Select parents for crossover;
    for each pair of parents do
        if random number < crossover rate then
            Perform crossover to create offspring;
            for each offspring do
                if random number < mutation rate then
                    Apply mutation to the offspring;
                end
            end
            Evaluate fitness of offspring;
            Select individuals for next generation;
        end
    end
end
Select best individual as final result;
```

Reference

Ben Issa, S., Patterson, R. A., & Tu, Y. (2021). Solving resource-constrained multi-project environment under different activity assumptions. *International Journal of Production Economics*, *232*, 107936.

https://doi.org/10.1016/j.ijpe.2020.107936

Chen, S., Fang, S., & Tang, R. (2019). A reinforcement learning based approach for multi-projects scheduling in cloud manufacturing. *International Journal of Production Research*, *57*(10), 3080–3098. https://doi.org/10.1080/00207543.2018.1535205

Demeulemeester, E. L., & Herroelen, W. S. (2006). *Project Scheduling: A Research Handbook*. Springer Science & Business Media.

Ding, T., Zeng, Z., Bai, J., Qin, B., Yang, Y., & Shahidehpour, M. (2020). Optimal Electric Vehicle Charging Strategy With Markov Decision Process and Reinforcement Learning Technique. *IEEE Transactions on Industry Applications*, *56*(5), 5811–5823. https://doi.org/10.1109/TIA.2020.2990096

Du, B., Tan, T., Guo, J., Li, Y., & Guo, S. (2021). Energy-cost-aware resource-constrained project scheduling for complex product system with activity splitting and recombining. *Expert Systems with Applications*, *173*, 114754. https://doi.org/10.1016/j.eswa.2021.114754

Fedus, W., Ramachandran, P., Agarwal, R., Bengio, Y., Larochelle, H., Rowland, M., & Dabney, W. (2020). Revisiting Fundamentals of Experience Replay. *International Conference on Machine Learning*, 3061–3071. https://proceedings.mlr.press/v119/fedus20a.html

Fu, F., & Zhou, H. (2021). A combined multi-agent system for distributed multi-project scheduling problems. *Applied Soft Computing*, *107*, 107402. https://doi.org/10.1016/j.asoc.2021.107402

Gómez Sánchez, M., Lalla-Ruiz, E., Fernández Gil, A., Castro, C., & Voß, S. (2023). Resource-constrained multi-project scheduling problem: A survey. *European Journal of Operational Research*, *309*(3), 958–976.

https://doi.org/10.1016/j.ejor.2022.09.033

Guo, W., Vanhoucke, M., Coelho, J., & Luo, J. (2021). Automatic detection of the best performing priority rule for the resource-constrained project scheduling problem. *Expert Systems with Applications*, *167*, 114116. https://doi.org/10.1016/j.eswa.2020.114116

Han, B.-A., & Yang, J.-J. (2020). Research on Adaptive Job Shop Scheduling Problems Based on Dueling Double DQN. *IEEE Access*, *8*, 186474–186495. https://doi.org/10.1109/ACCESS.2020.3029868

Hua, Z., Liu, Z., Yang, L., & Yang, L. (2022). Improved genetic algorithm based on time windows decomposition for solving resource-constrained project scheduling problem. *Automation in Construction*, *142*, 104503. https://doi.org/10.1016/j.autcon.2022.104503

Jedrzejowicz, P., & Ratajczak-Ropel, E. (2007). Agent-Based Approach to Solving the Resource Constrained Project Scheduling Problem. In B. Beliczynski, A. Dzielinski, M. Iwanowski, & B. Ribeiro (Eds.), *Adaptive and Natural Computing Algorithms* (pp. 480–487). Springer. https://doi.org/10.1007/978-3-540-71618-1_53

Kolisch, R., & Sprecher, A. (1997). PSPLIB - A project scheduling problem library: OR Software - ORSEP Operations Research Software Exchange Program. *European Journal of Operational Research*, *96*(1), 205–216. https://doi.org/10.1016/S0377-2217(96)00170-1

Lee, J., & Hyun, H. (2019). Multiple Modular Building Construction Project Scheduling Using Genetic Algorithms. *Journal of Construction Engineering and Management*, *145*(1), 04018116. https://doi.org/10.1061/(ASCE)CO.1943-7862.0001585

Lei, K., Guo, P., Zhao, W., Wang, Y., Qian, L., Meng, X., & Tang, L. (2022). A multi-action deep reinforcement learning framework for flexible Job-shop scheduling problem. *Expert Systems with Applications*, *205*, 117796. https://doi.org/10.1016/j.eswa.2022.117796

Li, C.-L., & Hall, N. G. (2019). Work Package Sizing and Project Performance. *Operations Research*, *67*(1), 123–142. https://doi.org/10.1287/opre.2018.1767

Li, X., Wu, C., Wu, P., Xiang, L., Shen, G. Q., Vick, S., & Li, C. Z. (2019). SWP-enabled constraints modeling for on-site assembly process of prefabrication housing production. *Journal of Cleaner Production*, *239*, 117991. https://doi.org/10.1016/j.jclepro.2019.117991

Li, X., Wu, C., Xue, F., Yang, Z., Lou, J., & Lu, W. (2022). Ontology-based mapping approach for automatic work packaging in modular construction. *Automation in Construction*, *134*, 104083. https://doi.org/10.1016/j.autcon.2021.104083

Li, X., Wu, C., Yang, Z., Guo, Y., & Jiang, R. (2023). Knowledge graph-enabled adaptive work packaging approach in modular construction. *Knowledge-Based Systems*, *260*, 110115. https://doi.org/10.1016/j.knosys.2022.110115

Li, Z., Wei, X., Jiang, X., & Pang, Y. (2021). A Kind of Reinforcement Learning to Improve Genetic Algorithm for Multiagent Task Scheduling. *Mathematical Problems in Engineering*, *2021*, e1796296. https://doi.org/10.1155/2021/1796296

Lin, C.-C., Deng, D.-J., Chih, Y.-L., & Chiu, H.-T. (2019). Smart Manufacturing Scheduling With Edge Computing Using Multiclass Deep Q Network. *IEEE Transactions on Industrial Informatics*, *15*(7), 4276–4284. https://doi.org/10.1109/TII.2019.2908210

Liu, D., Li, H., Wang, H., Qi, C., & Rose, T. (2020). Discrete symbiotic organisms

search method for solving large-scale time-cost trade-off problem in construction scheduling. *Expert Systems with Applications*, *148*, 113230. https://doi.org/10.1016/j.eswa.2020.113230

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), Article 7540. https://doi.org/10.1038/nature14236

Nguyen, D.-T., Chou, J.-S., & Tran, D.-H. (2022). Integrating a novel multiple-objective FBI with BIM to determine tradeoff among resources in project scheduling. *Knowledge-Based Systems*, *235*, 107640. https://doi.org/10.1016/j.knosys.2021.107640

Osband, I., Blundell, C., Pritzel, A., & Van Roy, B. (2016). Deep Exploration via Bootstrapped DQN. *Advances in Neural Information Processing Systems*, *29*. https://proceedings.neurips.cc/paper/2016/hash/8d8818c8e140c64c743113f563cf750f-Abstract.html

Pellerin, R., & Perrier, N. (2019). A review of methods, techniques and tools for project planning and control. *International Journal of Production Research*, *57*(7), 2160–2178. https://doi.org/10.1080/00207543.2018.1524168

Pellerin, R., Perrier, N., & Berthaut, F. (2020). A survey of hybrid metaheuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research*, *280*(2), 395–416. https://doi.org/10.1016/j.ejor.2019.01.063

Safapour, E., Kermanshachi, S., & Ramaji, I. (2023). Selection of Best Practices that Enhance Phase-Based Cost and Schedule Performances in Complex

Construction Projects. *Engineering Management Journal*, *35*(1), 84–99. https://doi.org/10.1080/10429247.2022.2036068

Salvendy, G. (2001). *Handbook of Industrial Engineering: Technology and Operations Management*. John Wiley & Sons.

Sami Ur Rehman, M., Thaheem, M. J., Nasir, A. R., & Khan, K. I. A. (2022). Project schedule risk management through building information modelling. *International Journal of Construction Management*, *22*(8), 1489–1499. https://doi.org/10.1080/15623599.2020.1728606

Satic, U., Jacko, P., & Kirkbride, C. (2022). Performance evaluation of scheduling policies for the dynamic and stochastic resource-constrained multi-project scheduling problem. *International Journal of Production Research*, *60*(4), 1411–1423. https://doi.org/10.1080/00207543.2020.1857450

Schwindt, C., & Zimmermann, J. (2015). *Handbook on project management and scheduling*. Springer International Publishing.

Servranckx, T., & Vanhoucke, M. (2019). A tabu search procedure for the resource-constrained project scheduling problem with alternative subgraphs. *European Journal of Operational Research*, *273*(3), 841–860. https://doi.org/10.1016/j.ejor.2018.09.005

Shahrabi, J., Adibi, M. A., & Mahootchi, M. (2017). A reinforcement learning approach to parameter estimation in dynamic job shop scheduling. *Computers & Industrial Engineering*, *110*, 75–82. https://doi.org/10.1016/j.cie.2017.05.026

Shalaby, A., & Ezeldin, A. S. (2021). A model for work packages optimization in results-based-finance projects. *Engineering, Construction and Architectural Management*, *29*(7), 2810–2835. https://doi.org/10.1108/ECAM-10-2019-0556

Sung, I., Choi, B., & Nielsen, P. (2020). Reinforcement Learning for Resource

Constrained Project Scheduling Problem with Activity Iterations and Crashing. *IFAC-PapersOnLine*, *53*(2), 10493–10497. https://doi.org/10.1016/j.ifacol.2020.12.2794

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning, second edition: An Introduction*. MIT Press.

Van Eynde, R., & Vanhoucke, M. (2020). Resource-constrained multi-project scheduling: Benchmark datasets and decoupled scheduling. *Journal of Scheduling*, *23*(3), 301–325. https://doi.org/10.1007/s10951-020-00651-w

Van Eynde, R., & Vanhoucke, M. (2022). New summary measures and datasets for the multi-project scheduling problem. *European Journal of Operational Research*, *299*(3), 853–868. https://doi.org/10.1016/j.ejor.2021.10.006

Vanhoucke, M., Coelho, J., Debels, D., Maenhout, B., & Tavares, L. V. (2008). An evaluation of the adequacy of project network generators with systematically sampled networks. *European Journal of Operational Research*, *187*(2), 511–524. https://doi.org/10.1016/j.ejor.2007.03.032

Vuorinen, L., & Martinsuo, M. (2019). Value-oriented stakeholder influence on infrastructure projects. *International Journal of Project Management*, *37*(5), 750–766. https://doi.org/10.1016/j.ijproman.2018.10.003

Wang, H.-W., Lin, J.-R., & Zhang, J.-P. (2020). Work package-based information modeling for resource-constrained scheduling of construction projects. *Automation in Construction*, *109*, 102958. https://doi.org/10.1016/j.autcon.2019.102958

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016). Dueling Network Architectures for Deep Reinforcement Learning. *Proceedings of The 33rd International Conference on Machine Learning*, 1995–2003.

https://proceedings.mlr.press/v48/wangf16.html

Wauters, T., Verbeeck, K., Berghe, G. V., & De Causmaecker, P. (2011). Learning agents for the multi-mode project scheduling problem. *Journal of the Operational Research Society*, *62*(2), 281–290. https://doi.org/10.1057/jors.2010.101

Wen, L., Li, X., & Gao, L. (2021). A New Reinforcement Learning Based Learning Rate Scheduler for Convolutional Neural Network in Fault Classification. *IEEE Transactions on Industrial Electronics*, *68*(12), 12890–12900. https://doi.org/10.1109/TIE.2020.3044808

Yang, S., Wang, J., & Xu, Z. (2022). Real-time scheduling for distributed permutation flowshops with dynamic job arrivals using deep reinforcement learning. *Advanced Engineering Informatics*, *54*, 101776. https://doi.org/10.1016/j.aei.2022.101776

Zhang, G., Lu, X., Liu, X., Zhang, L., Wei, S., & Zhang, W. (2022). An effective two-stage algorithm based on convolutional neural network for the bi-objective flexible job shop scheduling problem with machine breakdown. *Expert Systems with Applications*, *203*, 117460. https://doi.org/10.1016/j.eswa.2022.117460

Zhang, H., Peng, Q., Zhang, J., & Gu, P. (2021). Planning for automatic product assembly using reinforcement learning. *Computers in Industry*, *130*, 103471. https://doi.org/10.1016/j.compind.2021.103471

Zhao, F., Zhou, G., Xu, T., Zhu, N., & Jonrinaldi. (2023). A knowledge-driven cooperative scatter search algorithm with reinforcement learning for the distributed blocking flow shop scheduling problem. *Expert Systems with Applications*, *230*, 120571. https://doi.org/10.1016/j.eswa.2023.120571